

ЗАЩИЩЕННАЯ СИСТЕМА УПРАВЛЕНИЯ  
БАЗАМИ ДАННЫХ «JAТОВА»

Руководство по настройке. Часть 12.  
Централизованный сбор записей событий в СУБД.  
Компонент «ja\_Log»

643.72410666.00067-07 98 01-12

Листов 55

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

## АННОТАЦИЯ

В документе приведены сведения необходимые для установки и эксплуатации компонента централизованного сбора записей событий СУБД «ja\_Log» (далее по тексту – компонент либо ja\_Log).

Настоящее руководство предназначено для администраторов СУБД «Jatoba».



Все примеры в данном документе приведены для СУБД «Jatoba» версии ядра 6.x, для других версий все шаги выполняются аналогично, разница состоит в именах директорий.

Например, СУБД «Jatoba» версии 5.x по умолчанию устанавливается в директорию:

- ОС Windows – «C:\Program Files\GIS\Jatoba\5\bin»;
- ОС Linux – «/usr/jatoba-5/bin».

Используется версия компонента — 2.1



### **Важная информация**

Для сертифицированной версии СУБД «Jatoba» поддерживается работа только на ОС, указанных в формуляре на поставку!

Степени важности примечаний, применяемые в документе:



**Важная информация** – указания, требующие особого внимания



**Дополнительная информация** – указания, позволяющие упростить работу с изделием

## СОДЕРЖАНИЕ

1. Назначение компонента.....	5
1.1. Условия применения.....	7
2. Установка и настройка.....	8
2.1. Установка компонента «ja_Log» в ОС GNU/Linux .....	8
2.2. Установка в ОС Microsoft Windows.....	9
3. Настройка архитектуры агент-сервер на ОС GNU/Linux .....	11
3.1. Службная СУБД. Конфигурирование серверной части компонента.....	11
3.1.1. Создание БД и установка расширения.....	11
3.1.2. Создание технологической учетной записи .....	13
3.1.3. Создание табличного пространства для хранения архивных секций .....	14
3.1.4. Установка параметров jalog_server.yml .....	15
3.1.5. Установка службы jalog_server .....	16
3.1.6. Добавление нового агента в служебной СУБД .....	17
3.1.7. Запуск сервера на служебной СУБД.....	20
3.1.8. Удаление агента в служебной СУБД.....	21
3.1.9. Функционал ротации журналов служебной СУБД .....	22
3.2. Целевая СУБД. Конфигурирование клиентской части компонента.....	24
3.2.1. Добавление в список логирования на целевой СУБД .....	24
3.2.2. Настройка конфигурационного файла jalog_agent.yml.....	25
3.2.3. Установка службы jalog_agent .....	26
3.2.4. Запуск агента на целевой СУБД .....	26
4. Структура конфигурационных файлов .....	28
4.1. Структура конфигурационного файла сервера jalog_server.yml.....	28
4.1.1. Собственные параметры сервера .....	28
4.1.2. Параметры TLS.....	29
4.1.3. Параметры подключения к базе данных.....	30
4.2. Структура конфигурационного файла агента jalog_agent.yml.....	31
5. Настройка TLS в ОС GNU/Linux.....	33
6. Обновление компонента .....	34
6.1. Обновление компонента в ОС GNU/Linux с версии 1.2 до версии 2.0.....	34
6.2. Обновление компонента в ОС GNU/Linux с версии 2.0 до версии 2.x.....	36
6.3. Обновление компонента в ОС Windows.....	36
7. Удаление компонента .....	38
7.1. Удаление компонента в ОС GNU/Linux .....	38
7.2. Удаление компонента в ОС Windows.....	39
8. Ошибки .....	40
8.1. Дублирование сообщений при рассылке уведомлений .....	40

8.2. Ошибка при выполнении подключения агента к серверу .....	40
8.3. Ошибка при подключении компонента «jaLog» к СУБД «Jatoba».....	41
Приложение 1 .....	42
Пример установки СУБД «Jatoba» из локального репозитория для ОС Ubuntu.....	42
Приложение 2 .....	49
Структура конфигурационного файла сервера jalog_server.yml .....	49
Структура конфигурационного файла сервера jalog_agent.yml .....	50
Термины и определения .....	53
Перечень сокращений.....	54

## 1. НАЗНАЧЕНИЕ КОМПОНЕНТА

Компонент «ja\_Log» предназначен для централизованного сбора записей событий с целевых СУБД «Jatoba» в служебную СУБД «Jatoba data safe» (далее по тексту – JDS).

Компонент работает по клиент-серверной технологии. Возможна клиент-серверная и локальная установка.

При клиент-серверной установке на серверах целевых СУБД устанавливается агент компонента, а на сервере служебной СУБД серверная часть компонента. Передача данных осуществляется по протоколу Libpq или TLS.

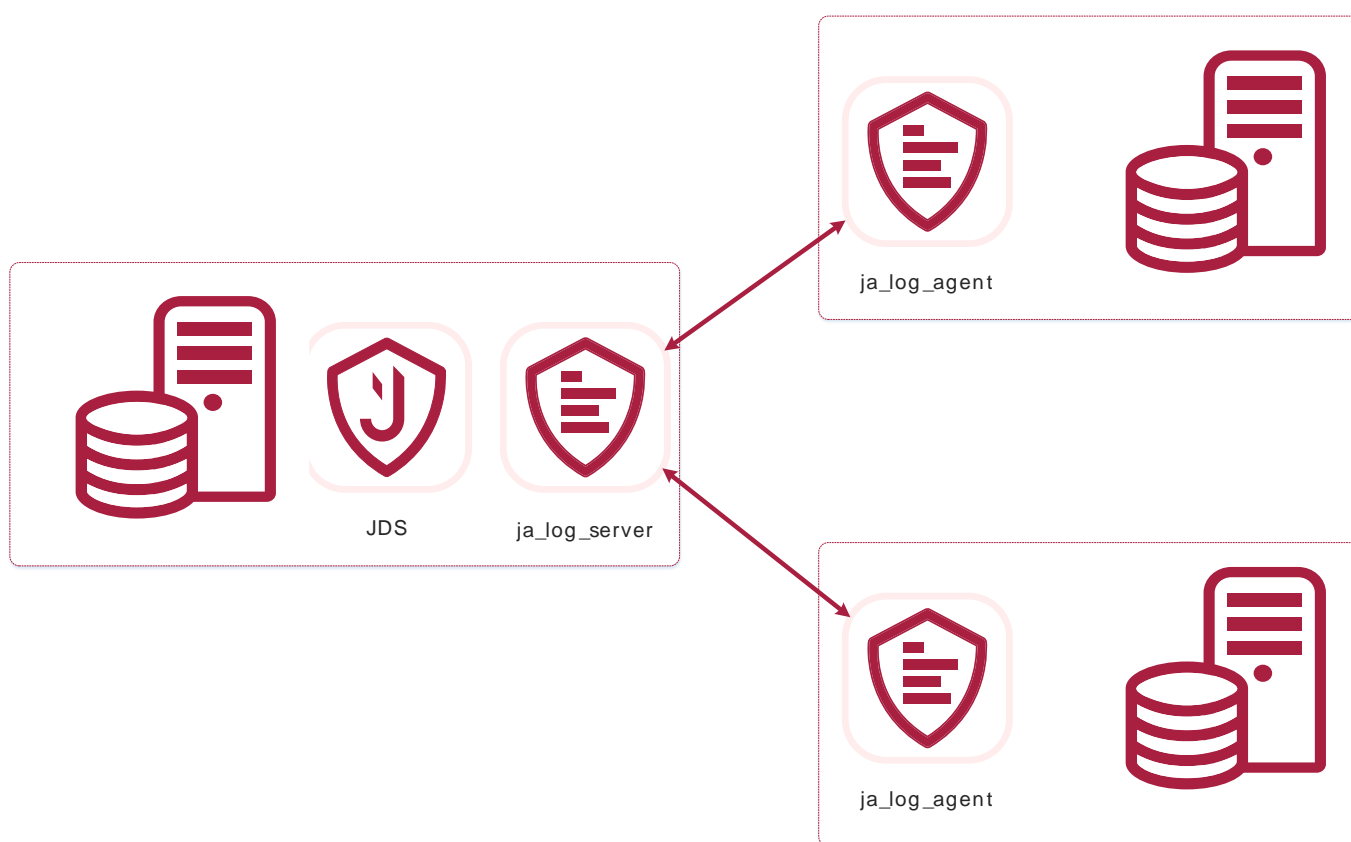


Рисунок 1.1 – Схема работы компонента при клиент-серверной установке

При локальной установке на сервере СУБД устанавливается агент и сервер компонента. Передача данных осуществляется по внутреннему интерфейсу.

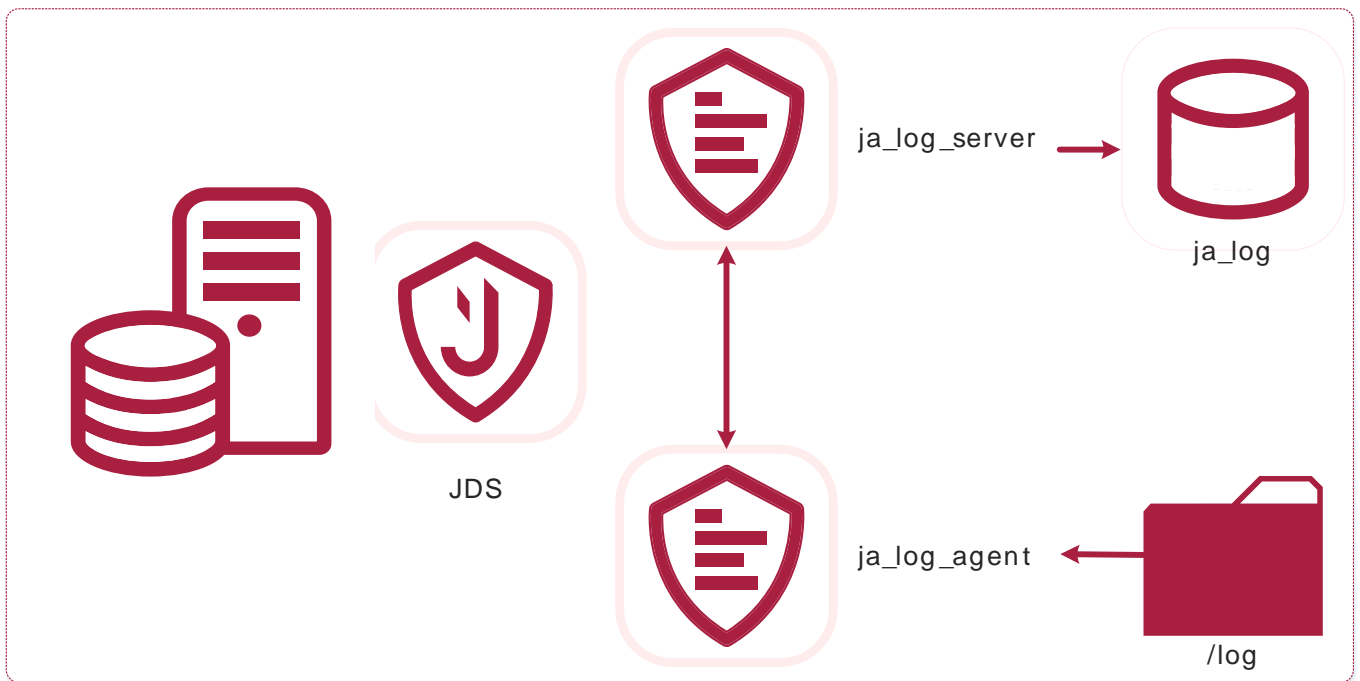


Рисунок 1.2 – Схема работы компонента при локальной установке

Агент из каталога СУБД /LOG передает события безопасности серверу компонента, который их складывает в служебную БД.



При установке сервера и агента на одном экземпляре БД необходимо избегать полного логирования всех операций `log_statement = 'all'`, т.к. в таком случае файлы журналов будут накапливать как записи самих запросов, так и записи записей этих запросов в служебную таблицу компонента, что приведет к разрастанию этих файлов.

Также необходимо избегать использования `log_statement = 'mod'`, т.к. в этом случае будет выполняться журналирование всех INSERT-запросов, что также приведет к прогрессивному увеличению файлов журналов.

Компонент организует секционирование таблицы журнала событий с секцией равной одним суткам. В расширении jalog применяется функция, которая на вход принимает дату и создает секцию для таблицы журнала событий соответствующего календарного дня. Секции таблицы журнала событий при помощи функции могут создаваться администратором СУБД как заранее, так и автоматически при вставке новых записей событий. При миграции с предыдущих версий компонента будет создана отдельная таблица, содержащая все предыдущие журналы событий, а также записи, по которым по каким-то причинам не определена секция в основной таблице журнала событий.

### **1.1. Условия применения**

В текущей реализации не поддерживается управление с помощью пользовательского веб-интерфейса для администраторов «Jatoba Data Safe». Управление осуществляется через корректировку управляющих параметров и конфигурационных файлов.

При совместном использовании компонента управления режимом работы узлов кластера «ja\_Dog» и компонента Централизованный сбор записей событий СУБД «ja\_Log», рекомендуется хранить журналы аудита вне каталога данных.

## 2. УСТАНОВКА И НАСТРОЙКА

Установка модуля должна производиться от имени пользователя, обладающего административными привилегиями в системе. Компонент штатным образом может быть установлен только с СУБД «Jatoba» (см. документ «Защищенная система управления базами данных «Jatoba». Руководство по установке).

### 2.1. Установка компонента «ja\_Log» в ОС GNU/Linux

Установка компонента осуществляется в процессе установки СУБД «Jatoba», также компонент можно установить опционально после основной инсталляции СУБД.

Установку компонента возможно провести двумя способами:

- 1) установка из локального репозитория (CDROM) – производится из файлов, записанных на компакт-диск или скопированных с него;
- 2) установка непосредственно из deb/rpm-файлов – производится опционально, по усмотрению пользователя.

Компонент выполнен в виде отдельного deb или rpm-пакета. Установка компонента осуществляется средствами пакетного менеджера ОС. Для разных типов пакетных менеджеров команда установки немного отличается. Ниже приведены основные типы:

– для систем на основе пакетного менеджера APT (к таким системам относятся все ОС семейства Debian, использующие deb-пакеты) команда установки следующая:

```
apt-get install jatoba<ver>-ja-log
```

– для систем на основе пакетных менеджеров YUM/DNF (к таким системам относятся все ОС семейства RedHat и вышедшие из нее, использующие rpm-пакеты) команда установки следующая:

```
yum install jatoba<ver>-ja-log
```

Отдельного уточнения требуют операционные системы ALT Linux и openSUSE.

– ALT Linux использует пакетный менеджер APT, но распространяется в виде rpm-пакетов и для нее команда установки выглядит аналогично Debian:



```
apt-get install jatoba<ver>-ja-log
```

– openSUSE также распространяется в виде rpm-пакетов, но использует собственный пакетный менеджер zypper, для нее команда установки выглядит следующим образом:

```
zypper install jatoba<ver>-ja-log
```

Установка компонента в составе других версий СУБД «Jatoba» осуществляется аналогично. Отличие будет только в номере версии СУБД, в составе которой он распространяется. Например, jatoba5-ja-log и т.п.

Для получения детальной информации по пакетному менеджеру рекомендуется обратиться к документации по ОС.

## 2.2. Установка в ОС Microsoft Windows

Компонент распространяется в составе инсталляционного msi-файла СУБД «Jatoba». Для установки компонента «ja\_Log» при выборе режима «Полный», модуль установится автоматически.

При режиме «Выборочный» потребуется в списке компонент дополнительно выбрать «Централизованный сбор событий безопасности (jalog)» для последующей его установки.

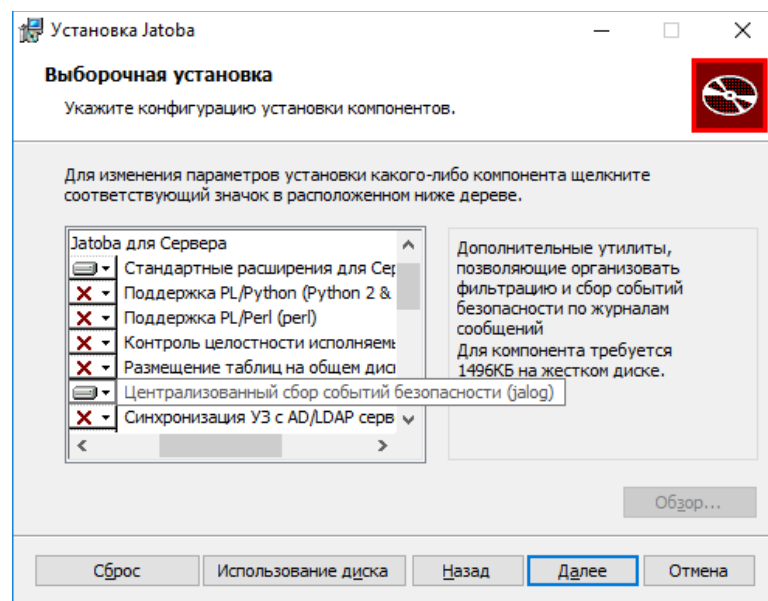


Рисунок 2.1 – Выбор компонента при установке

Удаление «ja\_Log» происходит также через инсталлятор СУБД «Jatoba». Компонент удалится либо вместе с удалением СУБД «Jatoba», либо отдельно его можно выключить из состава установленных компонент СУБД (через управление приложениями в панели управления Microsoft Windows).

Для получения детальной информации по установке/удалению программ в Microsoft Windows рекомендуется обратиться к документации на ОС.

### 3. НАСТРОЙКА АРХИТЕКТУРЫ АГЕНТ-СЕРВЕР НА ОС GNU/LINUX

В разделе описывается настройка компонента ja\_Log на ОС GNU Linux в архитектуре агент-сервер. В качестве примера используются 2 ЭВМ с ОС Ubuntu. Параметры, которых приведены в таблице 3.1.

Таблица 3.1 – Параметры ЭВМ

Имя ЭВМ	IP	Роль
u602doc-jds01	10.116.102.41	сервер
u602doc-jdog03	10.116.102.56	агент

Целесообразнее в качестве сервера использовать хост с установленным компонентом «Jatoba Data Safe» (JDS). Используемая СУБД считается служебной и в ней же будет располагаться БД с собранными событиями безопасности компонентом ja\_Log.

#### 3.1. Служебная СУБД. Конфигурирование серверной части компонента

Конфигурирование серверной части компонента потребует выполнения нижеописанных действий.

##### 3.1.1. Создание БД и установка расширения

Авторизоваться в psql от имени привилегированного пользователя:

```
psql -U postgres
```

Создать отдельную БД для компонента:

```
CREATE DATABASE ja_log;
```

Подключиться к созданной БД и установить расширение «jalog» в служебную БД:

```
\c ja_log  
ja_log=# CREATE EXTENSION jalog;
```



**ВАЖНО!** В случае совместной работы компонента «ja\_Log» с SecurityProfile в одной СУБД, расширение «jalog» должно устанавливаться в ту же БД, в которую установлено расширение «securityprofile».

Справочная информация по настройке SecurityProfile приведена в документе «Руководство администратора СУБД Jatoba» 643.72410666.00067-07 95 01

```

root@node1: /usr/jatoba-6/bin

postgres@node1:~$ psql
Password for user postgres:
psql (16.9)
Type "help" for help.

postgres=# CREATE DATABASE ja_log;
CREATE DATABASE
postgres=# \c ja_log
You are now connected to database "ja_log" as user "postgres".
ja_log=# CREATE EXTENSION jalog;
CREATE EXTENSION
ja_log=#

```

Рисунок 3.1 – Создание служебной БД и установка расширения

Проверить установку расширения в служебной СУБД при помощи команды:

```
\dx+ jalog
```

```

ja_log=# \dx
               List of installed extensions
  Name | Version | Schema | Description
-----+-----+-----+-----
 jalog | 2.1     | jalog  | Adds jalog integration features
 plpgsql | 1.0     | pg_catalog | PL/pgSQL procedural language
(2 rows)

ja_log=# \dx+ jalog
               Objects in extension "ja_log"
               Object description
-----+-----+-----+-----
function ja_log.add_agent_task(text,text)
function ja_log.add_jalog_role(text)
function ja_log.add_jalog_user(text,text)
function ja_log.create_partition(date)
function ja_log.delete_agent_task(integer)
function ja_log.log_save(jsonb)
function ja_log.rotation_hot_logs_days(integer,date,integer,text)
function ja_log.split_archive_partition()
function ja_log.truncate_max_logs_days(integer,date,integer)
function ja_log.turn_off_task(integer)
function ja_log.turn_on_task(integer)
sequence ja_log.key_key_id_seq
sequence ja_log.logcsv_default_id_seq
table ja_log.category
table ja_log.category_reference
table ja_log.component
table ja_log.key
table ja_log.logcsv
table ja_log.logcsv_default
table ja_log.meta
table ja_log.severity_translation
type ja_log.severity
(22 rows)

ja_log=#

```

Рисунок 3.2 – Проверка установленного расширения



Расширение может быть установлено в разделе «Ландшафт» компонента JDS.

### 3.1.2. Создание технологической учетной записи

После установки расширения компонента администратор создает технологическую учетную запись (УЗ) `jalog_user`, при помощи которой компонент выполняет свои функции в СУБД.

Технологическая УЗ создается в рамках групповой роли `jalog_role`, у которой имеются права на доступ к схемам, процедурам и табличным пространствам компонента «ja\_Log».

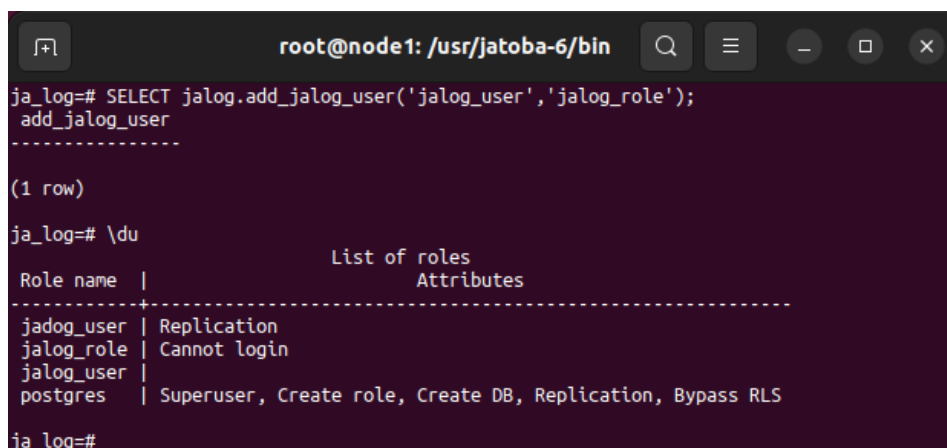
Групповая роль `jalog_role` создается автоматически в процессе установки расширения компонента в БД. Для данной групповой роли автоматически назначаются права доступа владельца БД, схем, функций и процедур компонента «ja\_Log».

Создание технологической УЗ `jalog_user` осуществляется при помощи SQL-команды:

```
SELECT jalog.add_jalog_user('jalog_user','jalog_role');
```

Проверить успешность создания в СУБД технологической учетной записи `jalog_user` при помощи команды:

```
\du
```



```
root@node1: /usr/jatoba-6/bin
ja_log=# SELECT jalog.add_jalog_user('jalog_user','jalog_role');
add_jalog_user
-----
(1 row)
ja_log=# \du
List of roles
Role name | Attributes
-----|-----
jalog_user | Replication
jalog_role | Cannot login
jalog_user |
postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS
ja_log=#
```

Рисунок 3.3 – Список ролей БД

В перечне пользователей должна быть созданная технологическая УЗ.

Для установки пароля технологической УЗ для подключения к СУБД необходимо выполнить SQL-команду следующего синтаксиса:

```
ALTER ROLE jalog_user WITH LOGIN PASSWORD
['password_jalog_user'];
```

```
root@node1: /usr/jatoba-6/bin
ja_log=# ALTER ROLE jalog_user WITH LOGIN PASSWORD 'password';
ALTER ROLE
ja_log=#
```

Рисунок 3.4 – Создание пароля технологической УЗ

После создания пароля технологической УЗ необходимо убедиться в том, что технологическая УЗ корректно наследует атрибуты групповой роли jalog\_role с помощью следующей команды:

```
\drg
```

```
root@node1: /usr/jatoba-6/bin
ja_log=# \drg
List of role grants
Role name | Member of | Options | Grantor
-----+-----+-----+-----
jadog_user | pg_read_all_stats | INHERIT, SET | postgres
jalog_user | jalog_role | INHERIT, SET | postgres
(2 rows)
ja_log=#
```

Рисунок 3.5 – Проверка наследования атрибутов групповой роли

### 3.1.3. Создание табличного пространства для хранения архивных секций

Для хранения архивных секций необходимо создать отдельное табличное пространство.

Для создания такого табличного пространства необходимо

- 1) Создать в ОС каталог для хранения данных табличного пространства:

```
mkdir /var/lib/jatoba/6/ts_jalog_archive
```

Пользователь postgres должен иметь права доступа на созданный каталог.

- 2) Авторизоваться в psql от имени привилегированного пользователя:

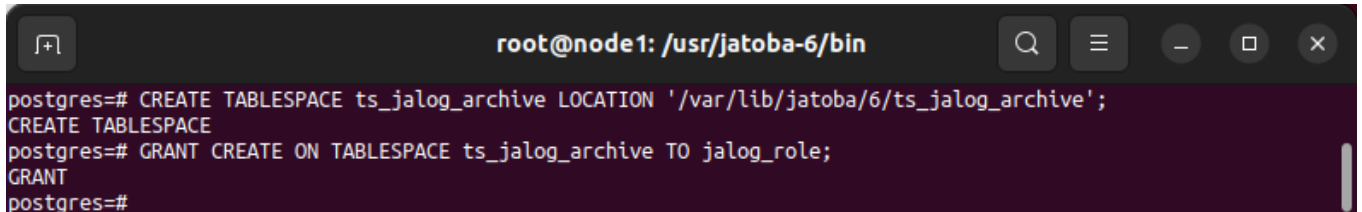
```
psql -U postgres
```

- 3) Создать в БД табличное пространство для хранения архивных секций

```
CREATE TABLESPACE ts_jalog_archive LOCATION
'/var/lib/jatoba/6/ts_jalog_archive';
```

4) Предоставить права доступа для групповой роли `jalog_role` к созданному табличному пространству:

```
GRANT CREATE ON TABLESPACE ts_jalog_archive TO jalog_role;
```



```
root@node1: /usr/jatoba-6/bin
postgres=# CREATE TABLESPACE ts_jalog_archive LOCATION '/var/lib/jatoba/6/ts_jalog_archive';
CREATE TABLESPACE
postgres=# GRANT CREATE ON TABLESPACE ts_jalog_archive TO jalog_role;
GRANT
postgres=#
```

Рисунок 3.6 – Предоставление групповой роли прав доступа к табличному пространству

### 3.1.4. Установка параметров `jalog_server.yml`

Запуск агента сервера сбора событий безопасности требует установки параметров в конфигурационном файле «`jalog_server.yml`».

Для внесения изменений необходимо открыть конфигурационный файл «`jalog_server.yml`» в терминале ОС:

```
nano /usr/jatoba-<ver>/etc/jalog/jalog_server.yml
```

и указать следующие значения:

```
# Собственные параметры сервера
server:
    listen_ip: 0.0.0.0          # IP-адрес, который прослушивает сервер
    listen_port: 10051         # Порт, который прослушивает сервер
# Параметры TLS
tls:
    # cert_file:                # Путь до сертификата
    # key_file:                 # Путь до файла ключа (только Linux)
    # ca_file:                  # Путь до файла ca (только Linux)
    # crl_file:                 # Путь до файла crl (только Linux)
# Параметры подключения к базе данных
database_params:
    conn_string: host=127.0.0.1 user=jalog_user dbname=ja_log
    password=[пароль УЗ jalog_user] # Строка подключения к СУБД
# Параметры логирования
```

**log:**

```
    path: "/usr/jatoba-<ver>/var/log/jalog/"    # Путь к каталогу логов
(Linux)

    # path: "C:\\Program files\\GIS\\Jatoba\\<ver>\\var\\log\\jalog\\"    #
Путь к каталогу логов (Windows)

    filename: jalog_server                      # Шаблон для имени файлов
логов

    level: info                                # Уровень логирования

    type: txt                                  # Форматы файлов логов (txt,
csv, json)

    filemode: 0600                             # Параметр доступа к файлам
логов (только Linux)

    rotation_age: 1d                           # Интервал ротации файлов
логов по времени

    rotation_size: 10MB                        # Интервал ротации по объему
файла логов

    truncate_on_rotation: false                # Признак перезаписи файла
логов
```



Компонент jalog при установке в конфигурационном файле параметра «type: json» обеспечивает поддержку журналов событий PostgreSQL – PG-json.

После установки параметров в файле jalog\_server.yml можно устанавливать службы агентов компонента на серверах с целевыми СУБД (см. п.п. 3.2.3).

### 3.1.5. Установка службы jalog\_server

На целевой СУБД устанавливается служба «jalog\_server» от имени и с правами привилегированного пользователя.

Перейти в директорию bin:

```
# cd /usr/jatoba-<ver>/bin/
```

- Для GNU/Linux:

```
# ./jalog_setup.sh
```

- Для ОС Windows



```
jalog_agent.exe install "C:\Program  
Files\GIS\Jatoba\<ver>\etc\jalog\jalog_server.yml" <полный  
домен>\postgres <пароль пользователя ОС postgres>
```

Убедиться, что сервис создан командой в терминале ОС:

```
CMD > services.msc
```

Перейти в свойства службы и выбрать тип запуска: Автоматический и применить изменения. Выполнить отключение брандмауэра.

Только в GNU/Linux добавить сервис в автозапуск:

```
# systemctl enable jalog_server
```

### 3.1.6. Добавление нового агента в служебной СУБД

Авторизоваться в psql от имени привилегированного пользователя:

```
# su - postgres  
psql -U postgres
```

Подключиться к базе данных ja\_log:

```
\c ja_log
```

Добавить нового агента при помощи запросов:

- Для GNU/Linux:

```
SELECT jalog.add_agent_task('<имя агента>',  
'/var/lib/jatoba/<ver>/data/log/jatoba-[0-9]{4}-[0-9]{2}-[0-9]{2}_[0-9]{6}.csv');
```

- Для ОС Windows:

```
SELECT jalog.add_agent_task('<имя агента>', 'C:\Program  
Files\GIS\Jatoba\<ver>\data\log\jatoba-[0-9]{4}-[0-9]{2}-[0-9]{2}_[0-9]{6}.csv');
```

Имя файла журнала событий формируется по параметру log\_filename в конфигурационном файле «postgresql.conf».

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

```
log_filename = 'jatoba-%a.log'
```

Имя журнала файла событий безопасности указывается по маске.

```
jatoba-....csv
```

В представленном примере в имени файла журнала событий указано наименование СУБД и день недели, когда был сформирован файл, по параметру «%a». Соответственно от разделительного знака до разделительного знака перед расширением файла в маске имени файла SQL-команды устанавливается 3 точки. Специальный символ «.» обозначает ровно один символ в имени файла.

Вид сформированного файла событий безопасности представлен на рисунке 3.7.

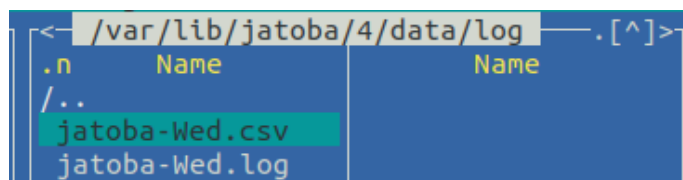


Рисунок 3.7 – Сформированный файл событий безопасности СУБД

Соотношение параметров и символов в имени файла регистрации событий представлено в таблице 3.2.

Таблица 3.2 – Соотношение параметров и символов в имени файла регистрации событий

Параметры		Имя СУБД	Разделительный знак	День недели	Разделительный знак перед расширением	Расширение файла
postgresql.conf	log_filename	jatoba	-	%a		log
Имя файла в директории LOG СУБД		jatoba	-	Wed	.	csv
Параметр в SQL-команде	key_text	jatoba	-		.	csv

Маска зависит от настроенного формата префикса для записей журнала. Например, для параметра log\_filename = 'jatoba-%Y-%m-%d\_%H%M%S.log' при формировании задания стоит указать следующее значение параметра:

```
jatoba-[0-9]{4}-[0-9]{2}-[0-9]{2}_{[0-9]{6}}.csv
```

В этом случае имя файла журнала событий безопасности будет следующим:

```
jatoba-2023-05-18_053315.csv
```

Такое имя файла журнала событий образуется при использовании аббревиатуры элементов интервалов по стандарту ISO 8601. Расчет количества символов для маски в SQL-команде приведен в таблице 3.3.

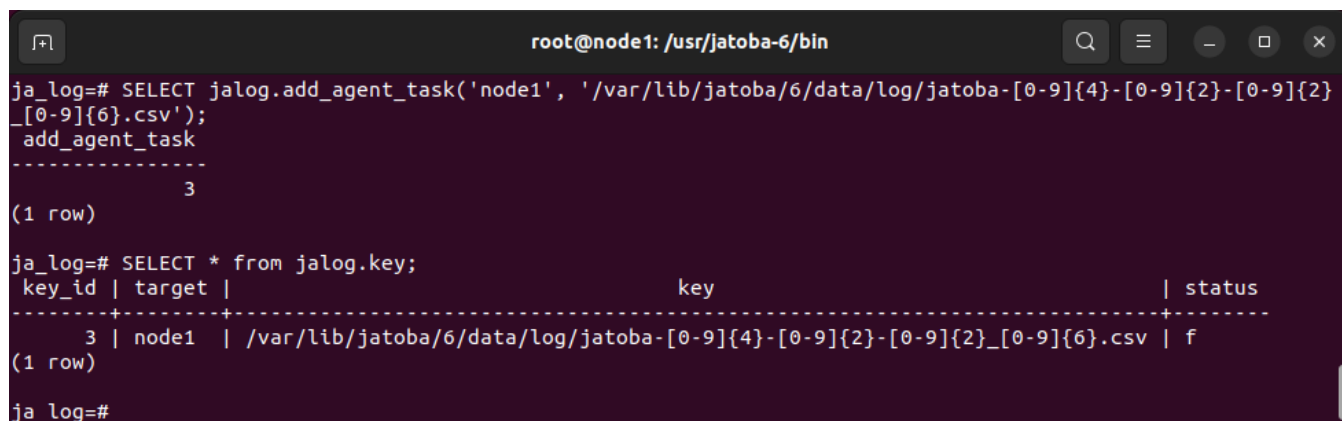
Таблица 3.3 – Количество символов в маске файла

	Имя СУБД		Годы		Месяцы		Дни		Часы	Минуты	Секунды		Расширение файла
Аббревиатуры элементов интервалов по стандарту ISO 8601	jatoba	-	%Y	-	%m	-	%d	_	%H	%M	%S	.	csv
Количество символов	0	0	4	1	2	1	2	1	2	2	2	1	0
Итого символов	18												

После добавления нового агента установленные параметры, внесенные в таблицу «jalog.key», возможно проверить командой:

```
SELECT * from jalog.key;
```

В колонке «status» установлен параметр «f» – (false), что означает, что параметр выключен.



```

root@node1: /usr/jatoba-6/bin

ja_log=# SELECT jalog.add_agent_task('node1', '/var/lib/jatoba/6/data/log/jatoba-[0-9]{4}-[0-9]{2}-[0-9]{2}_{[0-9]{6}}.csv');
add_agent_task
-----
          3
(1 row)

ja_log=# SELECT * from jalog.key;
 key_id | target | key | status
-----+-----+-----+-----
      3 | node1  | /var/lib/jatoba/6/data/log/jatoba-[0-9]{4}-[0-9]{2}-[0-9]{2}_{[0-9]{6}}.csv | f
(1 row)

ja_log=#
  
```

Рисунок 3.8 – Добавление нового агента и проверка установленных параметров

Активировать агента SQL-командой:

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

```
SELECT jalog.turn_ON_task (<key_id добавленного агента>);
```

при помощи которой устанавливается статус директивы в параметр «t» – (true).

Далее необходимо убедиться, что агент активирован:

```
SELECT * from jalog.key;
```

В результате выполнения команды активации агента значение в колонке «status» сменится с «f» – (false) на «t» – (true).



```
root@node1: /usr/jatoba-6/bin

ja_log=# SELECT * from jalog.key;
key_id | target | key | status
-----+-----+-----+-----
      3 | node1  | /var/lib/jatoba/6/data/log/jatoba-[0-9]{4}-[0-9]{2}-[0-9]{2}_{0-9}{6}.csv | f
(1 row)

ja_log=# SELECT jalog.turn_ON_task (3);
turn_on_task
-----
t
(1 row)

ja_log=# SELECT * from jalog.key;
key_id | target | key | status
-----+-----+-----+-----
      3 | node1  | /var/lib/jatoba/6/data/log/jatoba-[0-9]{4}-[0-9]{2}-[0-9]{2}_{0-9}{6}.csv | t
(1 row)

ja_log=#
```

Рисунок 3.9 Активация и проверка агента

### 3.1.7. Запуск сервера на служебной СУБД

- Для GNU/Linux:

```
# systemctl start jalog_server
```

- Для ОС Windows:

```
net start JalogServerService
```

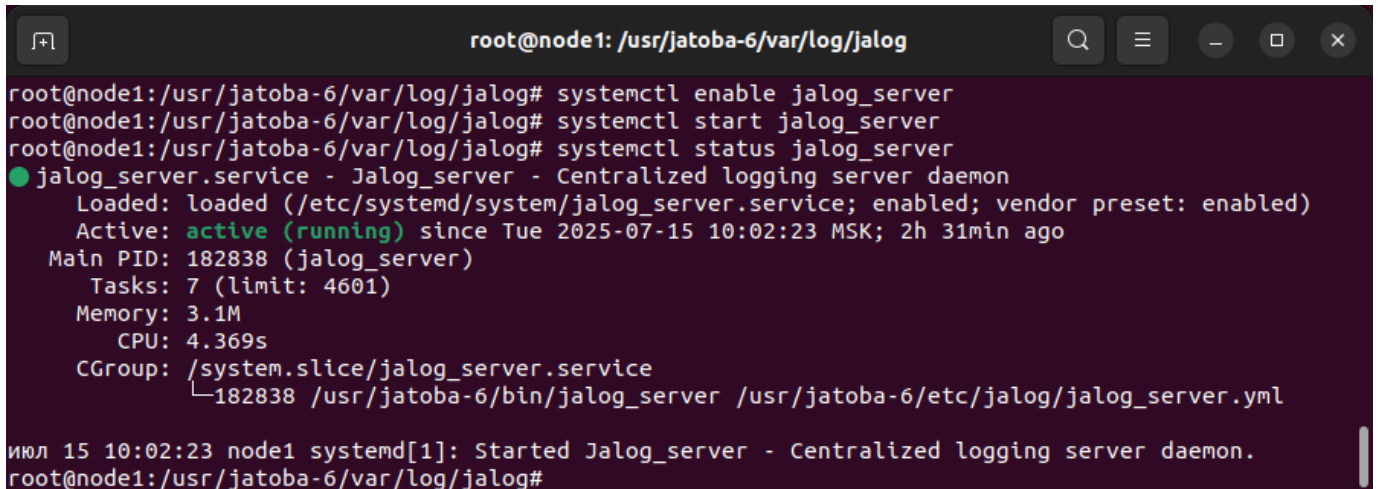
Убедиться, что служба успешно запущена:

- Для GNU/Linux:

```
# systemctl status jalog_server
```

- Для ОС Windows:

```
sc query JalogServerService
```



```
root@node1: /usr/jatoba-6/var/log/jalog
root@node1:/usr/jatoba-6/var/log/jalog# systemctl enable jalog_server
root@node1:/usr/jatoba-6/var/log/jalog# systemctl start jalog_server
root@node1:/usr/jatoba-6/var/log/jalog# systemctl status jalog_server
● jalog_server.service - Jalog_server - Centralized logging server daemon
   Loaded: loaded (/etc/systemd/system/jalog_server.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2025-07-15 10:02:23 MSK; 2h 31min ago
 Main PID: 182838 (jalog_server)
    Tasks: 7 (limit: 4601)
   Memory: 3.1M
      CPU: 4.369s
   CGroup: /system.slice/jalog_server.service
           └─182838 /usr/jatoba-6/bin/jalog_server /usr/jatoba-6/etc/jalog/jalog_server.yml

июл 15 10:02:23 node1 systemd[1]: Started Jalog_server - Centralized logging server daemon.
root@node1:/usr/jatoba-6/var/log/jalog#
```

Рисунок 3.10 – Установка службы сервера компонента в ОС GNU/Linux

В случае если служба сервера компонента не запускается необходимо проверить системный журнал на наличие ошибок при помощи команды:

```
# cat /var/log/syslog | grep jalog
```

На данном шаге настройка службы сервера компонента в ОС GNU/Linux завершена.

### 3.1.8. Удаление агента в служебной СУБД

Авторизоваться в psql от имени привилегированного пользователя:

```
psql -U postgres
```

Подключиться к базе данных ja\_log:

```
\c ja_log
```

Удаление агента производится по его идентификатору. Для получения идентификатора необходимо выполнить запрос:

```
SELECT * from jalog.key;
```

В столбце «key\_id» определить номер агента, который требуется удалить.

Удалить агента при помощи запросов:

```
SELECT jalog.delete_agent_task(<key_id выбранного агента>);
```

### 3.1.9. Функционал ротации журналов служебной СУБД

Для оптимизации ресурсов по хранению журналов администратор БД использует систему их ротации.

Компонент «ja\_Log» позволяет выполнять следующие процедуры:

- Контролировать глубину журнала на служебной СУБД;
- Перенос архивных журналов служебной СУБД в другое табличное пространство, например на более медленные носители информации;
- Удалять архивированные записи журнала служебной СУБД.

Для того, чтобы перенести журнал служебной СУБД в другое табличное пространство (например созданное в п. 3.1.3) необходимо воспользоваться процедурой следующего синтаксиса:

```
call jalog.rotation_hot_logs_days(0, ['date_point']::date, [-hot_logs_days], ['target_tablespace'])
```

где 0 – виртуальный идентификатор для журналирования; date\_point – дата, начиная с которой необходимо переместить журналы в другое ТП. В качестве date\_point можно указывать дату в формате YYYYMMDD. Значение null указывает на текущую дату; hot\_logs\_days – глубина перемещения в днях (значение всегда отрицательное); target\_tablespace – табличное пространство для выполнения перемещения журналов служебной СУБД.



Для ротации архивных журналов служебной СУБД может быть использовано существующее табличное пространство. При установке расширения компоненту выдаются все необходимые права доступа ко всем существующим на тот момент табличным пространствам служебной СУБД. Назначаемые права доступа необходимы для корректного перемещения архивных журналов, выполнения функций и процедур.

Если после установки расширения создается новое табличное пространство, которое будет использовано для ротации архивных журналов, то в этом случае для групповой роли `jalog_role` определяются права доступа к нему (см. пример в п. 3.1.3).

### Пример:

```
\c ja_log  
call jalog.rotation_hot_logs_days(0, null, -7,  
'ts_jalog_archive');
```

Данный пример процедуры выполняет перемещение журналов служебной СУБД в табличное пространство `ts_jalog_archive` за последние 7 дней начиная с предыдущего дня.



Администратор при выполнении переноса журналов в другое табличное пространство должен оценивать объем журналов. В случае, если журнал содержит значительное количество информации, рекомендуется выполнять его поэтапный перенос используя параметры процедуры `ja_log.rotation_hot_logs_days`.

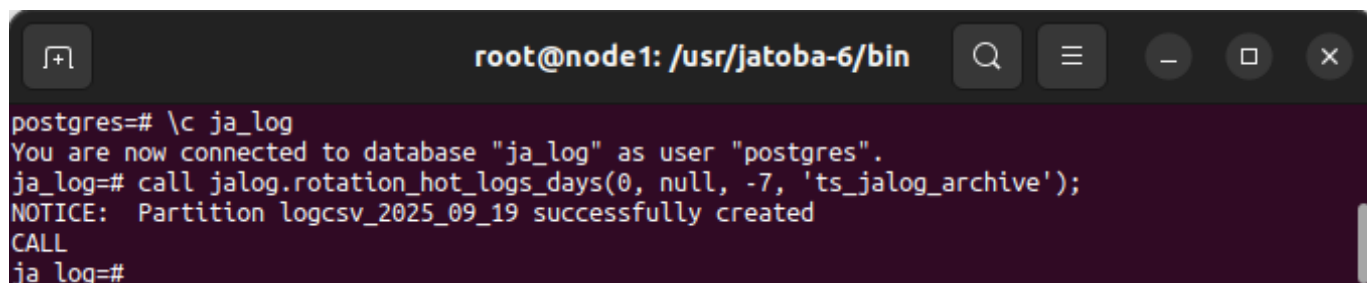


Рисунок 3.11 – Перенос журналов служебной СУБД в другое табличное пространство

Удаление архивированных журналов необходимо воспользоваться процедурой следующего синтаксиса:

```
call jalog.truncate_max_logs_days(0, ['date_point']::date, [-  
max_logs_days]);
```

где 0 – виртуальный идентификатор для журналирования; `date_point` – дата, начиная с которой необходимо удалить архивированные журналы. В качестве `date_point` можно указывать дату в формате `YYYYMMDD`. Значение `null` указывает на текущую дату;

max\_logs\_days – глубина удаления архивных журналов в днях (значение всегда отрицательное).

**Пример:**

```
\c ja_log  
call jalog.truncate_max_logs_days(0, null, -89);
```

Данный пример процедуры выполняет удаление архивированных журналов, например при помощи процедуры jalog.rotation\_hot\_logs\_days (см. выше), за последние 90 дней начиная с текущей даты.

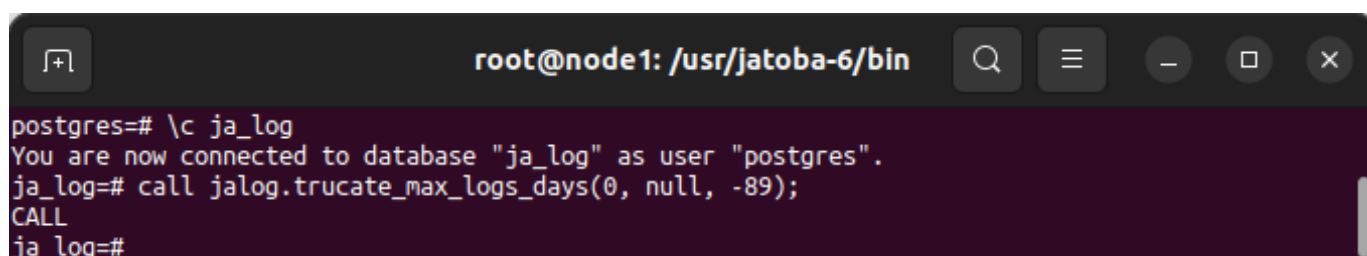


Рисунок 3.12 – Удаление архивированных журналов служебной СУБД

С целью автоматизации выполнения процедур ротации журналов служебной СУБД администратор может добавить процедуры в системную утилиту cron или в планировщик заданий СУБД (см. документа «Руководство по настройке. Часть 5. Планирование заданий СУБД. Компонент «pg\_Task»» 643.72410666.00067-07 98 01-05).

### **3.2. Целевая СУБД. Конфигурирование клиентской части компонента**

#### **3.2.1. Добавление в список логирования на целевой СУБД**

Добавление целевой СУБД в список логирования требует выполнения нижеописанных действий.

Открыть конфигурационный файл СУБД postgresql.conf:

```
# nano /var/lib/jatoba/<ver>/data/postgresql.conf
```

Изменить формат логов (значения менять в конце файла, а не в середине):

```
log_destination = 'csvlog'  
logging_collector = on
```

Выполнить перезагрузку СУБД в терминале ОС:

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------



```
# systemctl restart jatoba-<ver>
```

### 3.2.2. Настройка конфигурационного файла jalog\_agent.yml

Запуск агента сервера сбора событий безопасности требует установки параметров в конфигурационном файле.

Требуется открыть файл конфигурационный файл jalog\_agent.yml в терминале ОС:

```
# nano /usr/jatoba-<ver>/etc/jalog/jalog_agent.yml
```

и указать следующие значения:

```
# Собственные параметры агента
jalog_agent:
  hostname: u602doc-jdog03      # Уникальное имя агента
  ip: 10.116.102.56             # Ip-адрес агента
  # port: 22345                  # Порт агента
  # task_puller_frequency: 15    # Частота запроса задач у сервера, в
секундах
  # task_execution_frequency: 5  # Частота проверки лог-файлов, в
секундах
  # max_logs_per_iteration: 1000 # Максимальное количество логов,
которые обрабатываются за итерацию (10-1000)
# Параметры сервера, с которым работает агент
jalog_server:
  ip: 10.116.102.41             # Ip-адрес сервера
  port: 10051                   # Порт сервера
# Параметры TLS
tls:
  # cert_file:                   # Путь до сертификата
  # key_file:                    # Путь до файла ключа (только Linux)
  # ca_file:                     # Путь до файла ca (только Linux)
  # crt_file:                    # Путь до файла crt (только Linux)
# Параметры логирования
log:
  path: "/usr/jatoba-<ver>/var/log/jalog/"      # Путь к каталогу логов
(Linux)
  # path: "C:\\Program files\\GIS\\Jatoba\\<ver>\\var\\log\\jalog\\"  #
Путь к каталогу логов (Windows)
  filename: jalog_agent          # Шаблон для имени файлов
логов
  level: info                    # Уровень логирования
  type: txt                      # Форматы файлов логов (txt,
csv, json)
  filemode: 0600                 # Параметр доступа к файлам
логов (только Linux)
  rotation_age: 1d               # Интервал ротации файлов
логов по времени
  rotation_size: 10MB           # Интервал ротации по объему
файла логов
```

```
truncate_on_rotation: false  
ЛОГОВ
```

```
# Признак перезаписи файла
```

### 3.2.3. Установка службы jalog\_agent

На целевой СУБД устанавливается служба «jalog\_agent» от имени и с правами привилегированного пользователя.

Установка службы jalog\_agent выполняется после выполнения п.п. 3.1.3 на сервере служебной СУБД.

Перейти в директорию bin:

```
# cd /usr/jatoba-<ver>/bin/
```

- Для GNU/Linux:

```
# ./jalog_setup.sh
```

- Для ОС Windows

```
jalog_agent.exe install "C:\Program  
Files\GIS\Jatoba\<ver>\etc\jalog\jalog_agent.yml" <полный  
домен>\postgres <пароль пользователя ОС postgres>
```

Убедиться, что сервис создан командой в терминале ОС:

```
CMD > services.msc
```

Перейти в свойства службы и выбрать тип запуска: Автоматический и применить изменения. Выполнить отключение брандмауэра.

Только в GNU/Linux добавить сервис в автозапуск:

```
# systemctl enable jalog_agent
```

### 3.2.4. Запуск агента на целевой СУБД

- Для GNU/Linux:

```
# systemctl start jalog_agent
```

- Для ОС Windows:

```
net start JalogAgentService
```

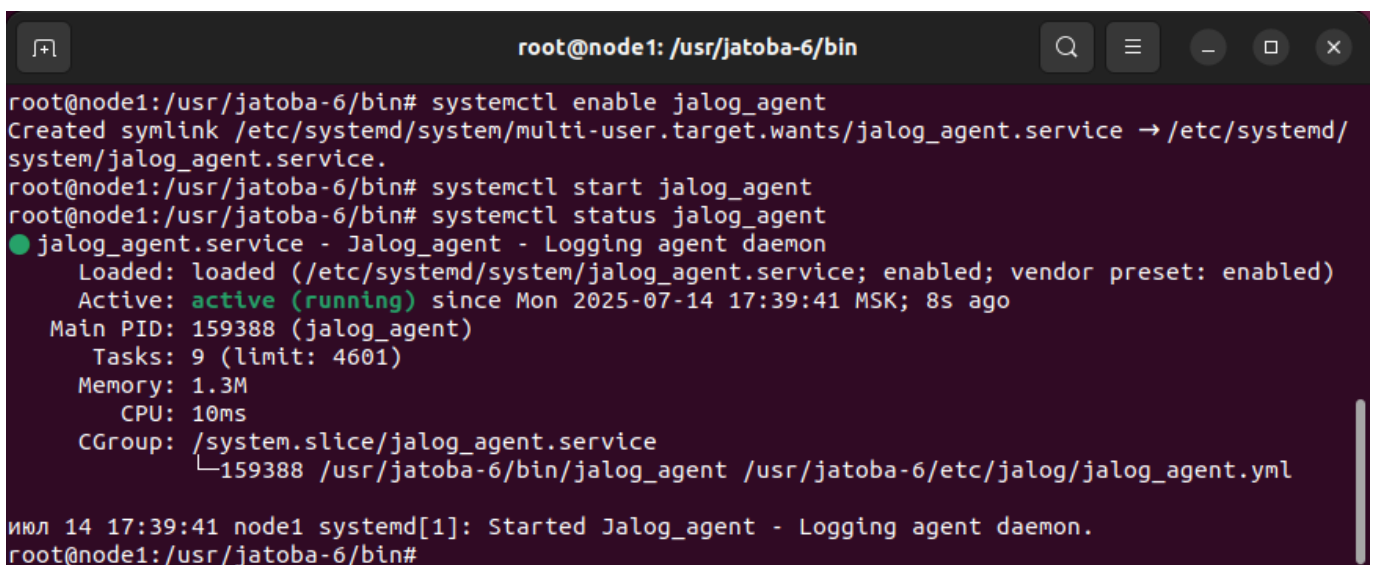
Убедиться, что служба успешно запущена:

- Для GNU/Linux:

```
# systemctl status jalog_agent
```

- Для ОС Windows:

```
sc query JalogAgentService
```



```
root@node1: /usr/jatoba-6/bin
root@node1:/usr/jatoba-6/bin# systemctl enable jalog_agent
Created symlink /etc/systemd/system/multi-user.target.wants/jalog_agent.service → /etc/systemd/system/jalog_agent.service.
root@node1:/usr/jatoba-6/bin# systemctl start jalog_agent
root@node1:/usr/jatoba-6/bin# systemctl status jalog_agent
● jalog_agent.service - Jalog_agent - Logging agent daemon
   Loaded: loaded (/etc/systemd/system/jalog_agent.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2025-07-14 17:39:41 MSK; 8s ago
     Main PID: 159388 (jalog_agent)
        Tasks: 9 (limit: 4601)
       Memory: 1.3M
          CPU: 10ms
      CGroup: /system.slice/jalog_agent.service
              └─159388 /usr/jatoba-6/bin/jalog_agent /usr/jatoba-6/etc/jalog/jalog_agent.yml

июл 14 17:39:41 node1 systemd[1]: Started Jalog_agent - Logging agent daemon.
root@node1:/usr/jatoba-6/bin#
```

Рисунок 3.13 – Установка службы агента компонента в ОС GNU/Linux

В случае если служба сервера компонента не запускается необходимо проверить системный журнал на наличие ошибок при помощи команды:

```
# cat /var/log/syslog | grep jalog
```

На данном шаге настройка службы агента компонента в ОС GNU/Linux завершена.

Далее становится возможным подключение к служебной БД «ja\_log» в компоненте JDS.

## 4. СТРУКТУРА КОНФИГУРАЦИОННЫХ ФАЙЛОВ

### 4.1. Структура конфигурационного файла сервера jalog\_server.yml

В файле jalog\_server.yml присутствует следующая структура:

```
# Собственные параметры сервера
server:
  # listen_ip: 0.0.0.0      # IP-адрес, который прослушивает
сервер
  # listen_port: 10051     # Порт, который прослушивает сервер
# Параметры TLS
tls:
  # cert_file:             # Путь до сертификата
  # key_file:              # Путь до файла ключа
  # ca_file:               # Путь до файла ca
  # crl_file:              # Путь до файла crl
# Параметры подключения к базе данных
database_params:
  # conn_string: host=127.0.0.1 user=jalog_user dbname=ja_log
password=[пароль УЗ jalog_user] # Строка подключения к СУБД
# Параметры логирования
log:
  # path: "/usr/jatoba-<ver>/var/log/jalog/"
# Путь к каталогу логов (Linux)
  # path: "C:\\Program
files\\GIS\\Jatoba\\<ver>\\var\\log\\jalog\\" # Путь к
каталогу логов (Windows)
  # filename: jalog_server
# Шаблон для имени файлов логов
  # level: info
# Уровень логирования
  # type: txt
# Форматы файлов логов (txt, csv, json)
  # filemode: 0600
# Параметр доступа к файлам логов (только Linux)
  # rotation_age: 1d
# Интервал ротации файлов логов по времени
  # rotation_size: 10MB
# Интервал ротации по объему файла логов
  # truncate_on_rotation: false
# Признак перезаписи файла логов
```

#### 4.1.1. Собственные параметры сервера

Данный раздел конфигурационного файла jalog\_server.yml определяет сетевые параметры сервера.

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

#### **4.1.1.1 listen\_ip**

Числовой IP-адрес сервера для подключения агентов. Он должен быть представлен в стандартном формате адресов IPv4, например, 172.28.40.9.

Значение по умолчанию – 0.0.0.0

#### **4.1.1.2 listen\_port**

Номер порта, по которому агенты подключаются к серверу.

Значение по умолчанию – 10051

### **4.1.2. Параметры TLS**

Данный раздел конфигурационного файла `jalog_server.yml` определяет пути к каталогам, содержащим сертификаты и ключи SSL/TLS.

#### **4.1.2.1 cert\_file**

Путь до файла сертификата, который агент `ja_Log` использует для подключения к СУБД.

Значение по умолчанию не задано.

#### **4.1.2.2 key\_file**

Путь до файла с ключом, который агент `ja_Log` использует для подключения к серверу.

Значение по умолчанию не задано.

#### **4.1.2.3 ca\_file**

Путь до сертификата центра сертификации, который агент `ja_Log` использует для подключения к серверу.

Значение по умолчанию не задано.

#### **4.1.2.4 crl\_file**

Путь до CRL файла, который агент `ja_Log` использует для подключения к серверу.

Значение по умолчанию не задано.

### **4.1.3. Параметры подключения к базе данных**

Раздел `database_params` конфигурационного файла `jalog_server.yml` определяет параметры подключения к базе данных компонента. Параметры подключения определяются в строке подключения `conn_string`.

#### **4.1.3.1 host**

IP-адрес БД, по которому выполняется подключение компонента `ja_Log`. Он должен быть представлен в стандартном формате адресов IPv4, например, 172.28.40.9.

Значение по умолчанию – 127.0.0.1

#### **4.1.3.2 user**

Имя пользователя СУБД `Jatoba`, используемое для подключения компонента `ja_Log`.

Значение по умолчанию – `ja_log`.

#### **4.1.3.3 dbname**

Название БД, к которой подключается компонента `ja_Log`. Название БД определяется на этапе настройки компонента (см. 3.1.1).

Значение по умолчанию – `ja_log`.

#### **4.1.3.4 password**

Пароль для доступа к БД, используемый в случае, когда компонент `ja_Log` настроен на аутентификацию по паролю.

#### **4.1.3.5 passfile**

Путь к расположению файла, в котором хранятся пароли пользователей СУБД. По умолчанию это `~/pgpass` в домашнем каталоге СУБД.

#### **4.1.3.6 require\_auth**

Параметр определяет метод аутентификации, который клиент требует от сервера. Если сервер не использует требуемый метод для аутентификации клиента или если обмен сообщениями аутентификации со стороны сервера не завершён, соединение не будет установлено. Доступные методы аутентификации – `password`, `md5`, `none`.

#### 4.1.3.7 connect\_timeout

Максимальное время ожидания подключения в секундах (задаётся десятичным целым числом, например: 10). При нуле, отрицательном или не заданном значении ожидание будет бесконечным.

#### 4.2. Структура конфигурационного файла агента jalog\_agent.yml

В файле jalog\_agent.yml присутствует следующая структура:

```
# Параметры агента
jalog_agent:
  hostname:                # Уникальное имя агента
  ip:                      # Ip-адрес агента
  # port: 22345             # Порт агента
  # task_puller_frequency: 15 # Частота запроса задач у
сервера, в секундах
  # task_execution_frequency: 5 # Частота проверки лог-
файлов, в секундах
  # max_logs_per_iteration: 1000 # Максимальное
количество логов, которые обрабатываются за итерацию (10-1000)

# Параметры сервера, с которым работает агент
jalog_server:
  # ip: 127.0.0.1           # Ip-адрес сервера
  # port: 10051            # Порт сервера

# Параметры TLS
tls:
  # cert_file:             # Путь до сертификата
  # key_file:              # Путь до файла ключа
  # ca_file:               # Путь до файла ca
  # crl_file:              # Путь до файла crl

# Параметры логирования
log:
  # path: "/usr/jatoba-<ver>/var/log/jalog/" # Путь к
каталогу логов (Linux)
  # path: "C:\\Program
files\\GIS\\Jatoba\\<ver>\\var\\log\\jalog\\" # Путь к
каталогу логов (Windows)
  # filename: jalog_agent # Шаблон для имени файлов логов
  # level: info           # Уровень логирования
  # type: txt             # Форматы файлов логов (txt, csv,
json)
  # filemode: 0600        # Параметр доступа к файлам логов
(только Linux)
  # rotation_age: 1d      # Интервал ротации файлов логов по
времени
```

```
# rotation_size: 10MB    # Интервал ротации по объему файла
ЛОГОВ
# truncate_on_rotation: false  # Признак перезаписи файла
ЛОГОВ
```



## **5. НАСТРОЙКА TLS В ОС GNU/LINUX**

Настройка TLS/SSL соединений описана в документе «Руководство по безопасности».

## 6. ОБНОВЛЕНИЕ КОМПОНЕНТА

### 6.1. Обновление компонента в ОС GNU/Linux с версии 1.2 до версии 2.0

В случае обновления компонента с версии 1.2 до версии 2.0 в ОС GNU/Linux требуется:

- 1) Остановить СУБД и сервер компонента «ja\_log»:

```
# systemctl stop jatoba-6 && systemctl stop jalog_server
```

- 2) Обновить локальный репозиторий;

- 3) Обновить пакет компонента;

```
# apt-get install jatoba<ver>-ja-log
```

- 4) Запустить скрипт migration.sql из служебной СУБД, который находится в каталоге /usr/jatoba-6/etc/jalog:

```
psql -U postgres
\c ja_log
\i /usr/jatoba-<ver>/etc/jalog/migration.sql
```

- 5) Проверить версию расширения компонента (должна быть установлена версия 2.1):

```
\dx
```

- 6) Перейти в /usr/jatoba-<ver>/etc/jalog/ и проверить наличие конфигурационных файлов версии 2.0: jalog\_server.yml и jalog\_agent.yml;

- 7) Выполнить настройку конфигурационных файлов согласно п.п. 3.1.3 и п.п. 3.2.2;

- 8) На сервере служебной СУБД указать новый конфигурационный файл jalog\_server.yml

```
# systemctl stop jalog_server
# systemctl edit --full jalog_server.service
### Параметры запуска Jalog_server
```

```
ExecStart=/usr/jatoba-<ver>/bin/jalog_server /usr/jatoba-  
6/etc/jalog/jalog_server.yml  
# systemctl daemon-reload  
# systemctl start jalog_server  
# systemctl status jalog_server
```

9) На сервере целевой СУБД указать новый конфигурационный файл jalog\_agent.yml

```
# systemctl stop jalog_agent  
# systemctl edit --full jalog_agent.service  
### Параметры запуска Jalog_agent  
ExecStart=/usr/jatoba-<ver>/bin/jalog_agent /usr/jatoba-  
6/etc/jalog/jalog_agent.yml  
# systemctl daemon-reload  
# systemctl start jalog_agent  
# systemctl status jalog_agent
```

10) На сервере служебной СУБД убедиться, что журналы событий предыдущей версии компонента будут присоединены в качестве секции в таблице jalog.logcsv\_archive

```
psql -U postgres  
\c ja_log  
SELECT key_id, message FROM jalog.logcsv_archive ORDER BY  
key_id desc;
```

11) Проверить наличие секции в таблице jalog.logcsv

```
\d+ jalog.logcsv
```

12) На служебной СУБД проверить корректность передачи журналов событий с целевой СУБД в основную таблицу jalog.logcsv

```
SELECT message FROM jalog.logcsv;
```

## 6.2. Обновление компонента в ОС GNU/Linux с версии 2.0 до версии 2.x

В случае обновления компонента с версии 2.0 до версии 2.x в ОС GNU/Linux порядок действий аналогичен рассмотренному в п. 6.1 кроме шага 4, вместо которого необходимо выполнить следующую команду:

```
ALTER EXTENSION jalog UPDATE;
```

## 6.3. Обновление компонента в ОС Windows

На сервере с служебной СУБД, в случае обновления компонента с версии 1.2 до версии 2.x, в командной строке Windows выполнить остановку службы (если она запущена):

```
net stop JalogServerService
```

При помощи инсталлятора выполнить обновление компонента до версии 2.x

Выполнить миграцию при помощи специального скрипта:

```
cd "C:\Program Files\GIS\Jatoba\<версия>\bin"  
psql -U postgres -f "C:\Program  
Files\GIS\Jatoba\6\etc\jalog\migration.sql"
```

Необходимо убедиться в том, что журналы событий из версии 1.2 присоединены в качестве секции к таблице jalog.logcsv\_archive:

```
psql -U postgres  
\c ja_log  
select key_id, message from jalog.logcsv_archive order by  
key_id desc;
```

Проверка наличия секций выполняется в СУБД при помощи команды:

```
\d+ jalog.logcsv
```

После успешного выполнения всех проверок необходимо запустить на сервере с служебной СУБД:

```
net start JalogServerService
```

После успешного запуска СУБД процедура обновления компонента до версии 2.x считается выполненной.

## 7. УДАЛЕНИЕ КОМПОНЕНТА

### 7.1. Удаление компонента в ОС GNU/Linux

На сервере с служебной СУБД авторизоваться в psql от имени привилегированного пользователя:

```
psql -U postgres
```

Подключиться к БД ja\_log и выполнить удаление целевой СУБД из списка журналирования:

```
#\c ja_log  
SELECT jalog.delete_agent_task(<key_id агента>);
```

Убедиться в том, что агент удален:

```
SELECT * from jalog.key;
```

На сервере с целевой СУБД остановить службу компонента:

```
# systemctl stop jalog_agent
```

Открыть конфигурационный файл postgresql.conf и задать формирование лога в формате, указанном по умолчанию:

```
log_destination = 'stderr'
```

Сохранить изменения и выполнить перезагрузку целевой СУБД:

```
# systemctl stop jatoba-<ver>  
# systemctl start jatoba-<ver>
```

Подключиться к серверу со служебной СУБД и остановить службу:

```
# systemctl stop jalog_server
```

Подключиться к служебной СУБД и удалить БД jalog со всем содержимым:

```
drop database ja_log;
```

Удаление пакета компонента осуществляется средствами пакетного менеджера ОС. Для этого необходимо использовать соответствующую пакетному менеджеру ОС команду удаления (remove, purge, erase и т.п.).

## 7.2. Удаление компонента в ОС Windows

На сервере с служебной СУБД в командной строке Windows выполнить остановку службы (если она запущена):

```
net stop JalogServerService
```

Перейти в директорию bin и запустить удаление службы:

```
cd "C:\Program Files\GIS\Jatoba\<версия>\bin"  
jalog_server.exe uninstall
```

После завершения удаления необходимо убедиться в том, что службы не существует:

```
sc query JalogServerService
```

Переключиться на сервер с целевой СУБД и выполнить остановку службы агента (если она запущена):

```
net stop JalogAgentService
```

Перейти в директорию bin и запустить удаление службы:

```
cd "C:\Program Files\GIS\Jatoba\<версия>\bin"  
jalog_agent.exe uninstall
```

После завершения удаления необходимо убедиться в том, что службы не существует:

```
sc query JalogAgentService
```

Подключиться к служебной СУБД и удалить БД jalog со всем содержимым:

```
drop database ja_log;
```

После этого удаление компонента на служебной и целевой СУБД считается успешно выполненным.

## 8. ОШИБКИ

### 8.1. Дублирование сообщений при рассылке уведомлений

Ошибка возникает при некорректной настройке клиента синхронизации времени.

Ошибка устраняется на уровне ОС командами от имени привилегированного пользователя:

```
# apt purge ntp
# apt purge chrony
# timedatectl set-ntp true
# systemctl start systemd-timesyncd
```

### 8.2. Ошибка при выполнении подключения агента к серверу

В случае ошибки с настройками подключения агента к серверу компонента «jaLog» в журнале работы компонента будет отображаться следующая ошибка:

```
2025-07-14 18:20:48 ERROR Cannot fetch tasks from jalog_server:
Connection to address 10.116.102.54():10051 is failed. Reason:
TCPSocket::connect::connection to 10.116.102.54:10051 failed.
Error: 111. The next attempt will be in 15 seconds, Jalog ver.
2.0.0
```

Для решения данной ошибки необходимо на узле, выполняющем роль сервера компонента «jaLog», убедиться что служба «jalog\_server» запущена и корректно работает:

```
# systemctl status jalog_server

jalog_server.service - Jalog_server - Centralized logging
server daemon

    Loaded: loaded (/etc/systemd/system/jalog_server.service;
    enabled; vendor preset: enabled)

    Active: active (running) since Tue 2025-07-15 10:02:23
    MSK; 55s ago

    Main PID: 182838 (jalog_server)

    Tasks: 7 (limit: 4601)

    Memory: 2.6M

    CPU: 51ms

    CGroup: /system.slice/jalog_server.service
```



```
└─182838 /usr/jatoba-6/bin/jalog_server
/usr/jatoba-6/etc/jalog/jalog_server.yml
```

```
июл 15 10:02:23 node1 systemd[1]: Started Jalog_server -
Centralized logging server daemon.
```

### 8.3. Ошибка при подключении компонента «jaLog» к СУБД «Jatoba»

В случае ошибки с настройками подключения компонента «jaLog» к СУБД «Jatoba» в журнале работы компонента будет отображаться следующая ошибка:

```
2025-07-14 18:27:17 ERROR Cannot load category reference:
connection is failed: connection to server at "127.0.0.1", port
5432 failed: fe_sendauth: no password supplied

ConnInfo :host=127.0.0.1 user=postgres dbname=postgres, Jalog
ver. 2.0.0
```

Данная ошибка связана с тем, что для подключения компонента «jaLog» к СУБД «Jatoba» используется пароль пользователя postgres. Для выполнения успешного подключения необходимо указать пароль (password) в секции database\_params:conn\_string в конфигурационном файле jalog\_server.yml, который располагается в каталоге /usr/jatoba-<версия>/etc/jalog/.

Также необходимо проверить что в строке подключения к СУБД указан параметр dbname=ja\_log.

## ПРИЛОЖЕНИЕ 1

### Пример установки СУБД «Jatoba» из локального репозитория для ОС Ubuntu

Установка СУБД «Jatoba» из локального репозитория для ОС Ubuntu проводится в следующем порядке:

- 1) В терминале войти в режим суперпользователя, выполнив команду:

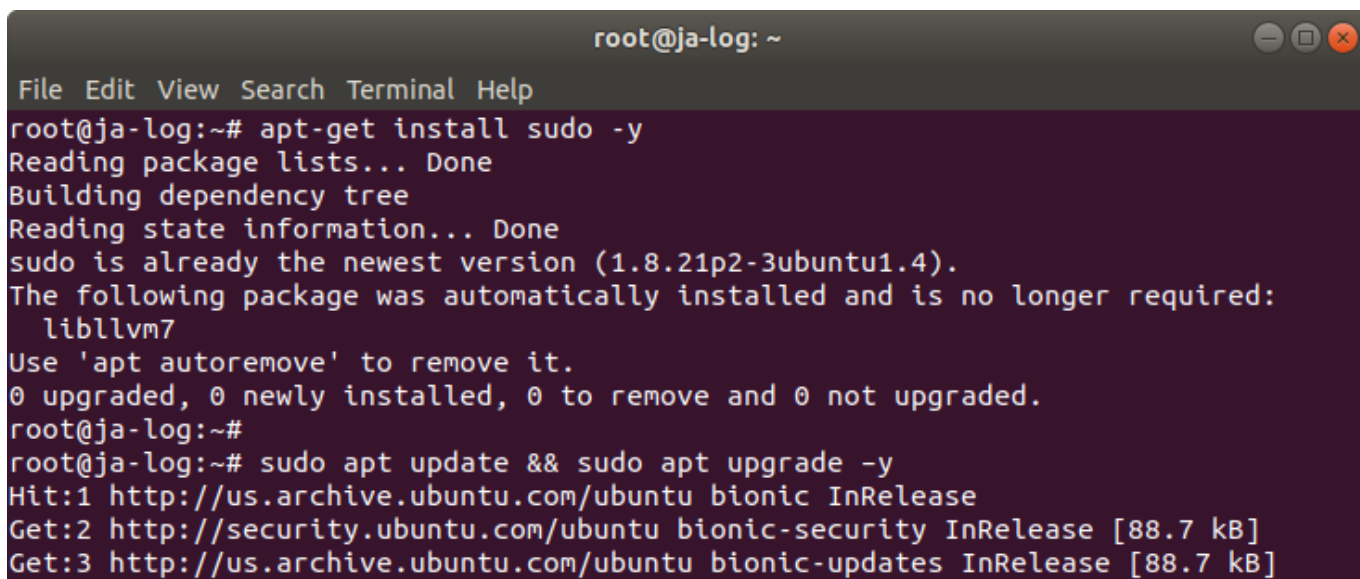
```
sudo su
```

- 2) Если команды sudo не существует – установить:

```
su -l,  
# apt-get install sudo -y
```

- 3) Выполнить обновление системы:

```
# apt update && sudo apt upgrade -y  
# apt -s dist-upgrade  
# apt dist-upgrade
```



The screenshot shows a terminal window titled 'root@ja-log: ~'. The terminal output is as follows:

```
File Edit View Search Terminal Help  
root@ja-log:~# apt-get install sudo -y  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
sudo is already the newest version (1.8.21p2-3ubuntu1.4).  
The following package was automatically installed and is no longer required:  
  libllvm7  
Use 'apt autoremove' to remove it.  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
root@ja-log:~#  
root@ja-log:~# sudo apt update && sudo apt upgrade -y  
Hit:1 http://us.archive.ubuntu.com/ubuntu bionic InRelease  
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]  
Get:3 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
```

Рисунок 8.1 – Обновление системы

- 4) Создать папку localrepo в корневом каталоге:

```
# mkdir /localrepo
```

- 5) В созданную папку скопировать:

```
каталог <pool>  
каталог <dist>  
файл <DEB-GPG-KEY-Jatoba>
```



Рисунок 8.2 – Структура каталога «localrepo»

- 6) Установить открытый ключ репозитория:

```
# apt-key add /localrepo/DEB-GPG-KEY-Jatoba
```

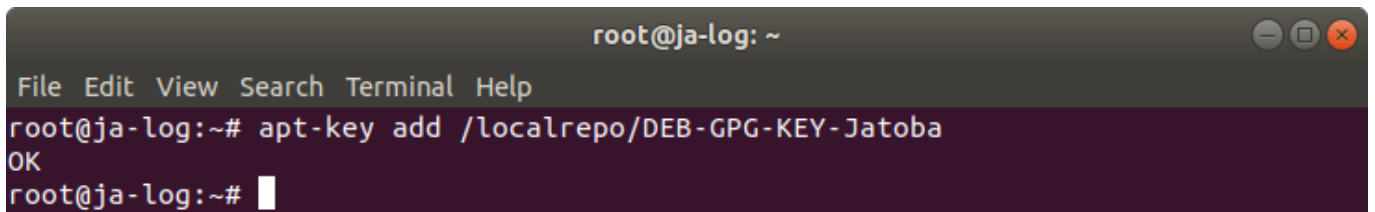


Рисунок 8.3 – Установка открытого ключа репозитория

- 7) Добавить описание локального репозитория в систему:

```
# nano /etc/apt/sources.list.d/jatoba-4.list
```

- 8) Вставить в файл следующее содержимое и сохранить:

```
deb file:///localrepo stable non-free
```

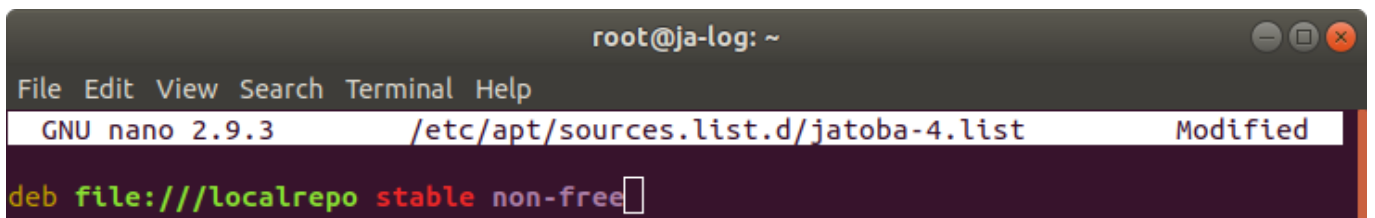
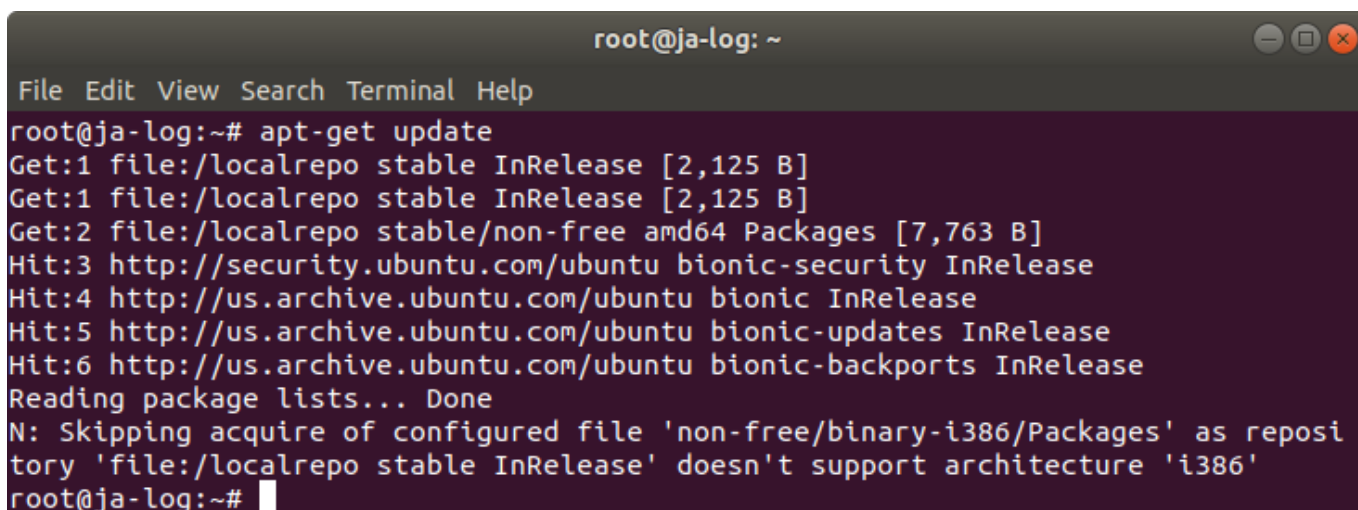


Рисунок 8.4 – Содержание файла «jatoba-4.list»

- 9) Проиндексировать обновленное состояние репозитория:

```
# apt-get update
```

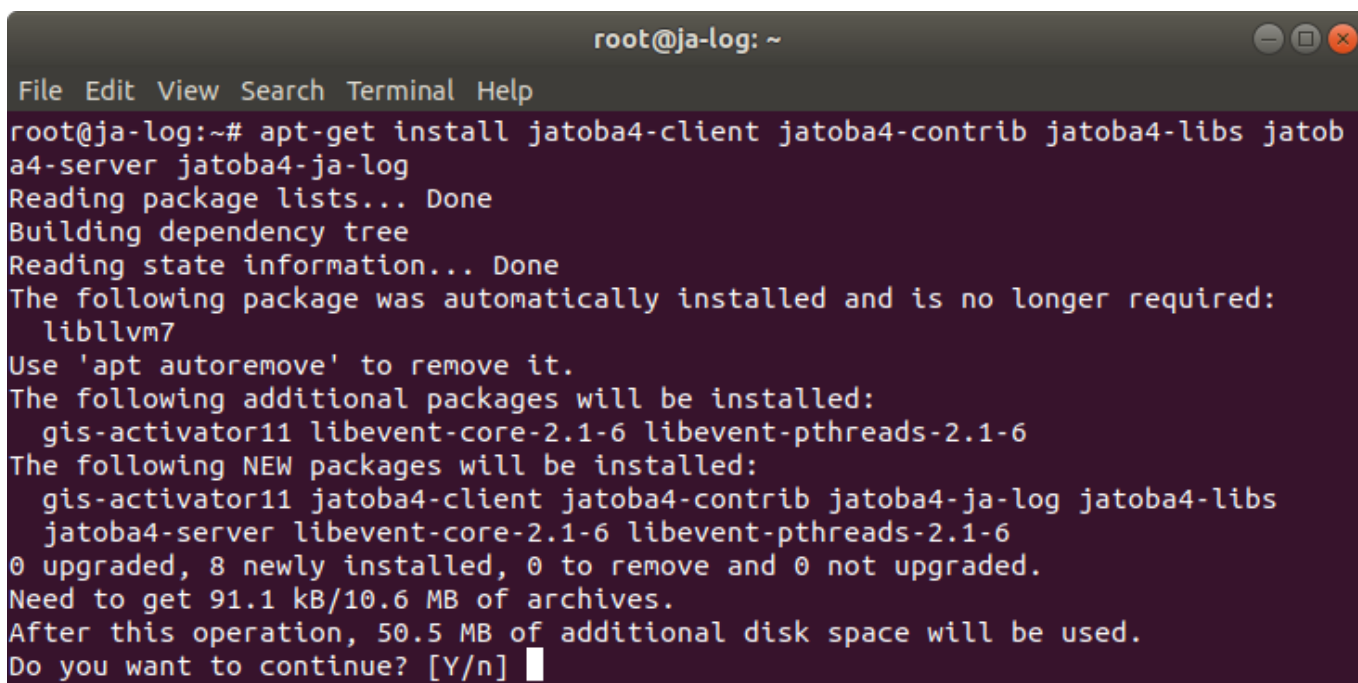


```
root@ja-log: ~
File Edit View Search Terminal Help
root@ja-log:~# apt-get update
Get:1 file:/localrepo stable InRelease [2,125 B]
Get:1 file:/localrepo stable InRelease [2,125 B]
Get:2 file:/localrepo stable/non-free amd64 Packages [7,763 B]
Hit:3 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Hit:5 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:6 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
N: Skipping acquire of configured file 'non-free/binary-i386/Packages' as repository 'file:/localrepo stable InRelease' doesn't support architecture 'i386'
root@ja-log:~#
```

Рисунок 8.5 – Индексация репозитория

- 10) Установить СУБД Jatoba при помощи команды:

```
# apt-get install jatoba4-client jatoba4-contrib jatoba4-libs
jatoba4-server jatoba4-ja-log
```

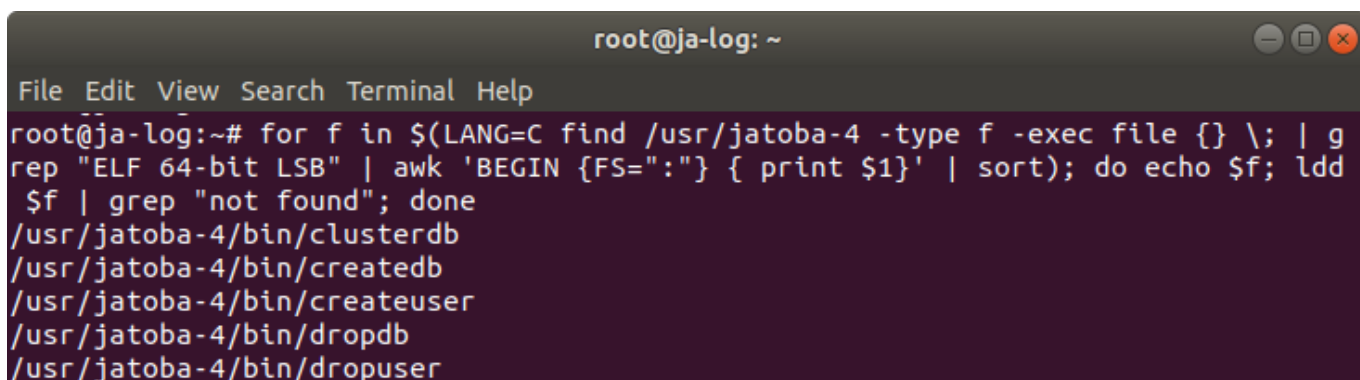


```
root@ja-log: ~
File Edit View Search Terminal Help
root@ja-log:~# apt-get install jatoba4-client jatoba4-contrib jatoba4-libs jatoba4-server jatoba4-ja-log
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm7
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
  gis-activator11 libevent-core-2.1-6 libevent-pthreads-2.1-6
The following NEW packages will be installed:
  gis-activator11 jatoba4-client jatoba4-contrib jatoba4-ja-log jatoba4-libs
  jatoba4-server libevent-core-2.1-6 libevent-pthreads-2.1-6
0 upgraded, 8 newly installed, 0 to remove and 0 not upgraded.
Need to get 91.1 kB/10.6 MB of archives.
After this operation, 50.5 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Рисунок 8.6 – Установка пакетов

- 11) Убедиться, что отсутствуют ошибки зависимостей:

```
for f in $(LANG=C find /usr/jatoba-4 -type f -exec file {} \; |
grep "ELF 64-bit LSB" | awk 'BEGIN {FS=":"} { print $1}' |
sort); do echo $f; ldd $f | grep "not found"; done
```

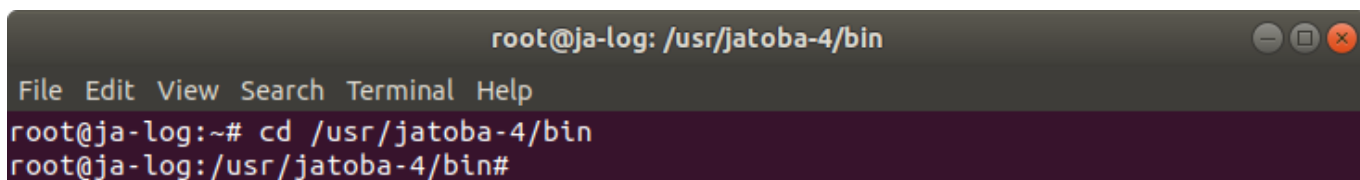


```
root@ja-log: ~  
File Edit View Search Terminal Help  
root@ja-log:~# for f in $(LANG=C find /usr/jatoba-4 -type f -exec file {} \; | grep "ELF 64-bit LSB" | awk 'BEGIN {FS=":"} { print $1}' | sort); do echo $f; ldd $f | grep "not found"; done  
/usr/jatoba-4/bin/clusterdb  
/usr/jatoba-4/bin/createdb  
/usr/jatoba-4/bin/createuser  
/usr/jatoba-4/bin/dropdb  
/usr/jatoba-4/bin/dropuser
```

Рисунок 8.7 – Проверка отсутствия ошибок зависимостей

- 12) Перейти в директорию исполняемых файлов СУБД:

```
# cd /usr/jatoba-4/bin
```

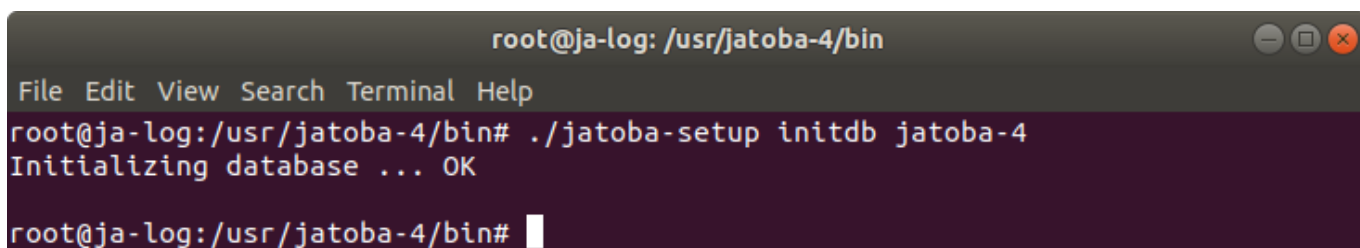


```
root@ja-log: /usr/jatoba-4/bin  
File Edit View Search Terminal Help  
root@ja-log:~# cd /usr/jatoba-4/bin  
root@ja-log:/usr/jatoba-4/bin#
```

Рисунок 8.8 – Переход в каталог

- 13) Инициализировать каталог данных СУБД при помощи команды:

```
# ./jatoba-setup initdb jatoba-4
```

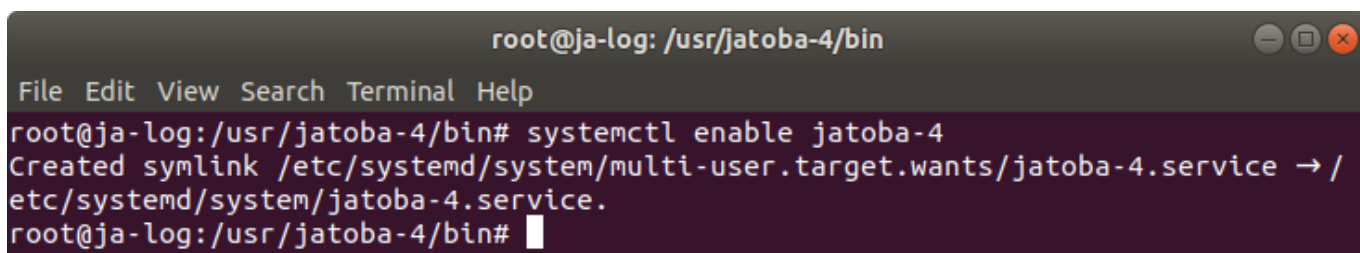


```
root@ja-log: /usr/jatoba-4/bin  
File Edit View Search Terminal Help  
root@ja-log:/usr/jatoba-4/bin# ./jatoba-setup initdb jatoba-4  
Initializing database ... OK  
root@ja-log:/usr/jatoba-4/bin#
```

Рисунок 8.9 – Инициализация СУБД

- 14) Добавить сервис в список автозапуска:

```
# systemctl enable jatoba-4
```

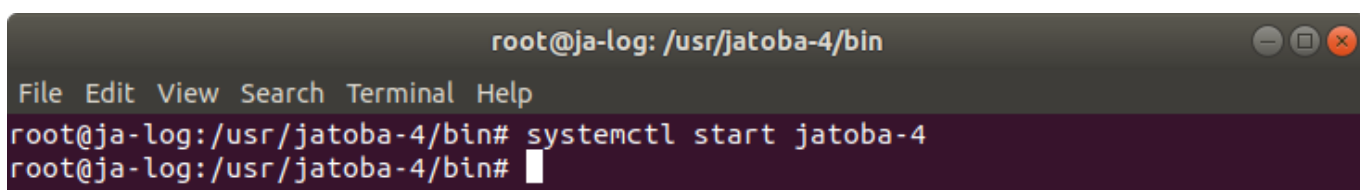


```
root@ja-log: /usr/jatoba-4/bin
File Edit View Search Terminal Help
root@ja-log:/usr/jatoba-4/bin# systemctl enable jatoba-4
Created symlink /etc/systemd/system/multi-user.target.wants/jatoba-4.service → /etc/systemd/system/jatoba-4.service.
root@ja-log:/usr/jatoba-4/bin#
```

Рисунок 8.10 – Добавление сервиса jatoba-4 а автозагрузку ОС

15) Запустить службу:

```
# systemctl start jatoba-4
```



```
root@ja-log: /usr/jatoba-4/bin
File Edit View Search Terminal Help
root@ja-log:/usr/jatoba-4/bin# systemctl start jatoba-4
root@ja-log:/usr/jatoba-4/bin#
```

Рисунок 8.11 – Запуск службы jatoba-4

16) Проверить статус службы:

```
# systemctl status jatoba-4
```

```
root@ja-log: /usr/jatoba-4/bin
File Edit View Search Terminal Help
root@ja-log:/usr/jatoba-4/bin# systemctl status jatoba-4
● jatoba-4.service - Jatoba 4 database server
   Loaded: loaded (/etc/systemd/system/jatoba-4.service; enabled; vendor preset:
   Active: active (running) since Wed 2022-09-07 03:47:40 PDT; 1min 2s ago
     Docs: https://www.gaz-is.ru/Jatoba/doc
  Process: 8518 ExecStartPre=/usr/jatoba-4/bin/jatoba-check-db-dir ${PGDATA} (co
 Main PID: 8526 (postmaster)
    Tasks: 9 (limit: 4622)
   CGroup: /system.slice/jatoba-4.service
           └─8526 /usr/jatoba-4/bin/postmaster -D /var/lib/jatoba/4/data/
             └─8527 postgres: logger
               └─8529 postgres: check licenser
                 └─8530 postgres: checkpointer
                   └─8531 postgres: background writer
                     └─8532 postgres: walwriter
                       └─8533 postgres: autovacuum launcher
                         └─8534 postgres: stats collector
                           └─8535 postgres: logical replication launcher

Sep 07 03:47:40 ja-log systemd[1]: Starting Jatoba 4 database server...
Sep 07 03:47:40 ja-log postmaster[8526]: 2022-09-07 03:47:40.876 PDT [8526] LOG:
Sep 07 03:47:40 ja-log postmaster[8526]: 2022-09-07 03:47:40.876 PDT [8526] HINT
Sep 07 03:47:40 ja-log systemd[1]: Started Jatoba 4 database server.
lines 1-22/22 (END)
```

Рисунок 8.12 – Проверка статуса службы jatoba-4

17) Авторизоваться в psql:

```
# su - postgres
psql
```

18) Установить пароль для пользователя СУБД «postgres»:

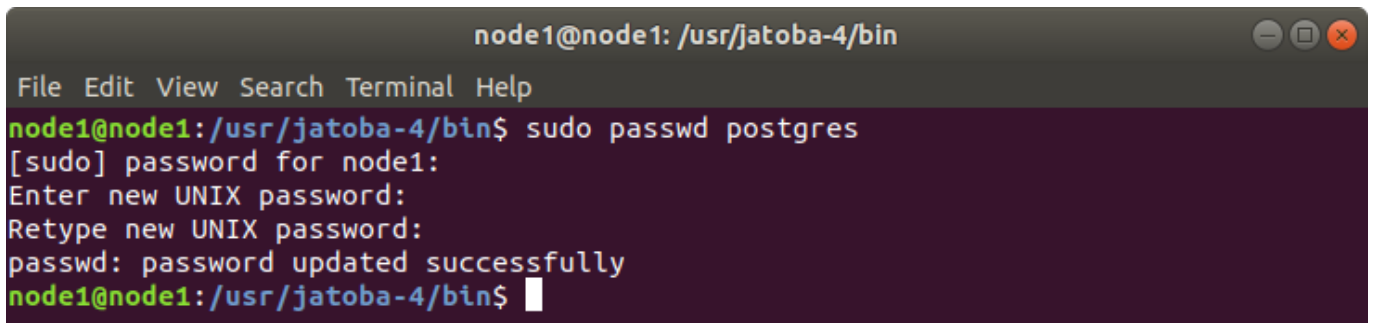
```
\password
```

```
postgres=# \password
Enter new password for user "postgres":
Enter it again:
```

Рисунок 8.13 – Установка пароля для пользователя ОС

19) Установить пароль для системного пользователя ОС «postgres»:

```
# passwd postgres
```



```
node1@node1: /usr/jatoba-4/bin
File Edit View Search Terminal Help
node1@node1:/usr/jatoba-4/bin$ sudo passwd postgres
[sudo] password for node1:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
node1@node1:/usr/jatoba-4/bin$
```

Рисунок 8.14 – Установка пароля для пользователя СУБД

На этом этапе установку СУБД с компонентом «ja\_Log» можно считать оконченной.



## ПРИЛОЖЕНИЕ 2

### Структура конфигурационного файла сервера jalog\_server.yml

Таблица П.8.1 – Структура конфигурационного файла jalog\_server.yml компонента «jaLog» для ОС Windows/Linux

Наименование параметра	Значение по умолчанию Linux/Windows	Описание параметра
<b>server</b>		<b>Собственные параметры сервера ja_Log</b>
listen_ip	0.0.0.0	IP-адрес, который прослушивает сервер
listen_port	10051	Номер порта, по которому агенты подключаются к серверу
<b>tls</b>		<b>Параметры TLS</b>
cert_file	—	Путь до файла сертификата, который агент ja_Log использует для подключения к СУБД.
key_file	—	Путь до файла с ключом, который агент ja_Log использует для подключения к серверу.
ca_file	—	Путь до сертификата центра сертификации, который агент ja_Log использует для подключения к серверу
crl_file	—	Путь до CRL файла, который агент ja_Log использует для подключения к серверу
<b>database_params</b>		<b>Параметры подключения к базе данных</b>
conn_string	—	Строка с параметрами подключения компонента к БД
<b>log</b>		<b>Параметры ведения журналов работы</b>
path	/usr/jatoba- <ver>/var/log/jalog/ "C:\\Program files\\GIS\\Jatoba\\	Каталог хранения журналов работы

Наименование параметра	Значение по умолчанию Linux/Windows	Описание параметра
	<ver>\\var\\log\\jalog\\	
filename	jalog_server	Шаблон названия файлов журналов
level	info	Уровень детализации журналов
type	txt	Тип файлов журналов (txt, csv, json)
filemode	0600 (Linux)	Параметры доступа к файлам журналов
rotation_age	1d	Интервал ротации файлов журналов по времени
rotation_size	10MB	Интервал ротации файлов журналов по размеру
truncate_on_rotation	false	Признак перезаписи файла журнала

### Структура конфигурационного файла сервера jalog\_agent.yml

Таблица П.8.2 – Структура конфигурационного файла jalog\_agent.yml компонента «jaLog» для ОС Windows/Linux

Наименование параметра	Значение по умолчанию Linux/Windows	Описание параметра
<b>jalog_agent</b>		<b>Параметры агента ja_Log</b>
hostname	—	Название узла, на котором установлен агент
ip	—	IP-адрес узла, на котором установлен агент
port	22345	Номер сетевого порта агента
task_puller_frequency	15	Частота запроса задач у сервера, в секундах
max_logs_per_iteration	1000	Максимальное количество журналов, которые обрабатываются за итерацию (10-1000)
<b>jalog_server</b>		<b>Параметры сервера ja_Log</b>
ip	—	IP-адрес сервера ja_Log

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Наименование параметра	Значение по умолчанию Linux/Windows	Описание параметра
port	10051	Номер порта, по которому агенты подключаются к серверу
<b>tls</b>		<b>Параметры TLS</b>
cert_file	—	Путь до файла сертификата, который агент ja_Log использует для подключения к СУБД.
key_file	—	Путь до файла с ключом, который агент ja_Log использует для подключения к серверу.
ca_file	—	Путь до сертификата центра сертификации, который агент ja_Log использует для подключения к серверу
crl_file	—	Путь до CRL файла, который агент ja_Log использует для подключения к серверу
<b>database_params</b>		<b>Параметры подключения к базе данных</b>
conn_string	—	Строка с параметрами подключения компонента к БД
<b>log</b>		<b>Параметры ведения журналов работы</b>
path	/usr/jatoba- <ver>/var/log/jalog/ "C:\\Program files\\GIS\\Jatoba\\ \\<ver>\\var\\log\\jalog\\	Каталог хранения журналов работы
filename	jalog_server	Шаблон названия файлов журналов
level	info	Уровень детализации журналов
type	txt	Тип файлов журналов (txt, csv, json)
filemode	0600 (Linux)	Параметры доступа к файлам журналов
rotation_age	1d	Интервал ротации файлов журналов по времени

Наименование параметра	Значение по умолчанию Linux/Windows	Описание параметра
rotation_size	10MB	Интервал ротации файлов журналов по размеру
truncate_on_rotation	false	Признак перезаписи файла журнала

## ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

**Целевая СУБД** – СУБД являющаяся целью мониторинга.

При использовании компонента пользовательского веб-интерфейса для администраторов «Jatoba data safe», компонент ведет мониторинг, обслуживание и прочие действия отдельно установленных СУБД «Jatoba». Такие СУБД для компонента «Jatoba data safe» являются целевыми.

**Служебная СУБД** – СУБД, обслуживающая компонент «Jatoba data safe», и выполняющая служебные функции, такие как:

- обеспечение собственного функционирования;
- сбор событий безопасности с отдельно установленных СУБД «Jatoba».

Таким образом СУБД обслуживающая компонент «Jatoba data safe» выполняет служебные функции и к ней применяется термин – «служебная СУБД».

**ПЕРЕЧЕНЬ СОКРАЩЕНИЙ**

SQL	–	Structured Query Language — язык структурированных запросов
БД	–	База данных
ОС	–	Операционная система
СУБД	–	Система управления базами данных
ЭВМ	–	Электронно-вычислительная машина
ДСЧ	–	Датчик случайных чисел

[illegible]