

УТВЕРЖДЕН
643.72410666.00067-07 98 01-ЛУ

ЗАЩИЩЕННАЯ СИСТЕМА УПРАВЛЕНИЯ
БАЗАМИ ДАННЫХ «JАТОВА»

Руководство по настройке. Часть 3.
Краткое руководство по настройке.
Компонент «jaDog»

643.72410666.00067-07 98 02-03

Листов 79

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

АННОТАЦИЯ

Во третьей части документа приведены основные сведения по проектированию и настройке отказоустойчивого кластера jaDog защищенной системы управления базами данных «Jatoba» (далее — СУБД «Jatoba»).

Настоящее руководство предназначено для администратора СУБД «Jatoba», включая требования к архитектуре, варианты развёртывания и детальные инструкции по конфигурации.

Представленные в документе снимки экрана могут отличаться для различных версий настраиваемой СУБД и предназначены для демонстрации хода настройки отказоустойчивого кластера СУБД «Jatoba».

Администратор СУБД «Jatoba» должен иметь навыки по работе с системами управления базами данных (СУБД) PostgreSQL или защищенной СУБД «Jatoba» (ООО «Газинформсервис»).



Примеры в данном документе приведены для СУБД «Jatoba» версии ядра 5, а также для СУБД «Jatoba» версии ядра 6.

Для СУБД «Jatoba» версий ядра 5 и 6 используется версия компонента — 3.4.

На приведенных в руководстве иллюстрациях версия компонента «jaDog» может отличаться от фактической.

Степени важности примечаний, применяемые в документе:



Важная информация – указания, требующие особого внимания



Дополнительная информация – указания, позволяющие упростить работу с изделием



Важная информация

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

Настоящий документ оснащён навигацией, гиперссылками и перекрестными ссылками. В приложении Adobe Acrobat навигация вызывается кнопкой «Закладки», как показано на рисунке 1.1.

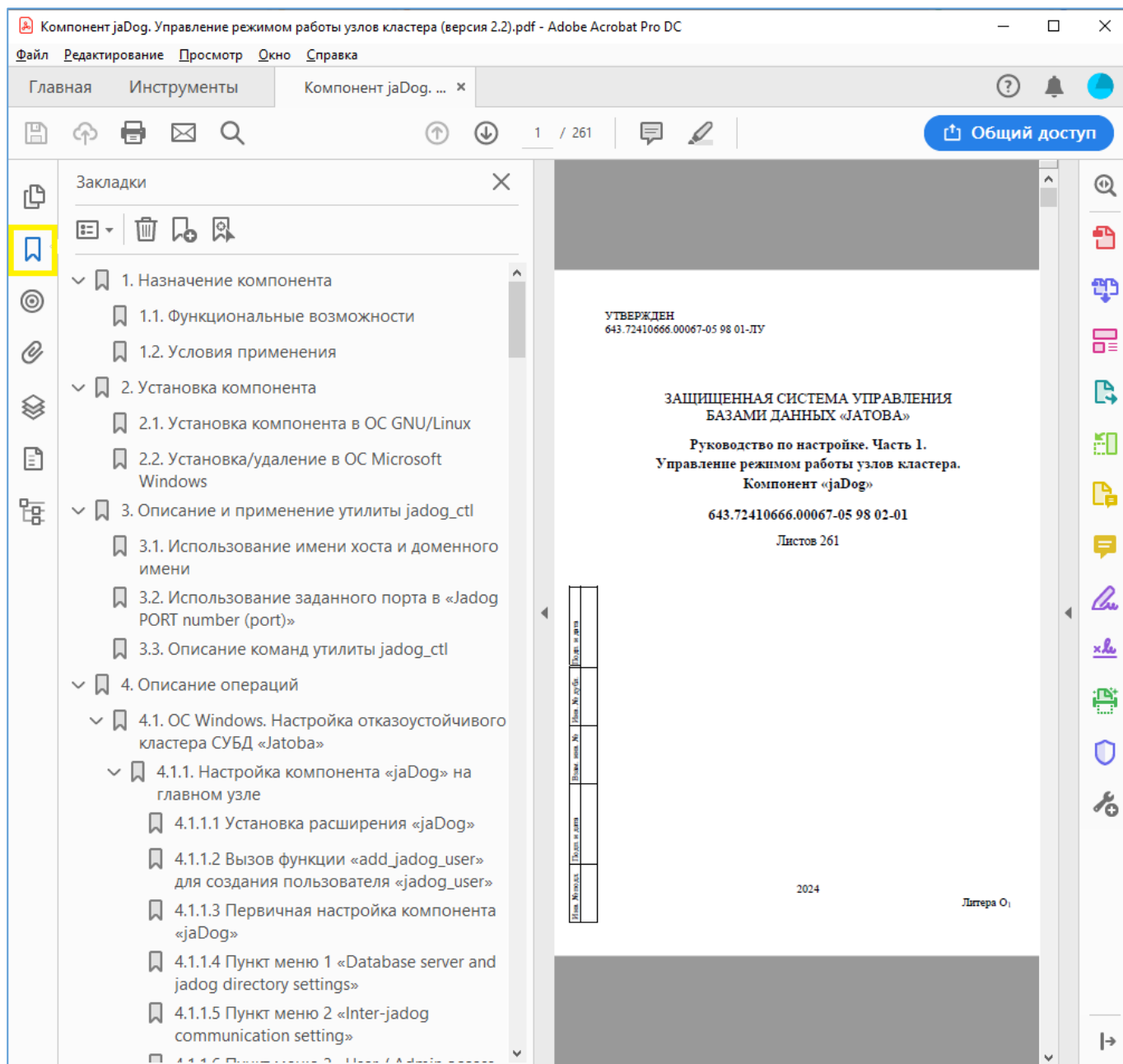


Рисунок 1.1 - Структура документа

СОДЕРЖАНИЕ

1. Проектирование кластера	7
1.1. Выбор архитектуры.....	7
1.2. Устойчивая архитектура кластера	7
1.2.1. Устойчивый вариант кластера с хранилищем WAL-файлов	8
1.2.2. Устойчивый вариант кластера без хранилища WAL-файлов	9
1.2.3. Устойчивая архитектура при использовании нескольких дата-центров	10
1.3. Минимальная архитектура кластера.....	11
1.3.1. Кластер минимальной конфигурации с узлом-арбитром и хранилищем WAL-файлов	11
1.3.2. Кластер минимальной конфигурации с узлом-арбитром без хранилища WAL-файлов.....	12
1.3.3. Кластер минимальной конфигурации с использованием Trusted Address.....	13
1.3.4. Минимальная архитектура при использовании нескольких дата-центров.....	14
1.4. Размещение хранилища WAL-файлов.....	15
2. Варианты построения отказоустойчивого кластера	16
2.1. Построение нового кластера и перенос данных СУБД.....	16
2.1.1. Планирование на основании новой инфраструктуры	16
2.2. Использование существующей инфраструктуры.....	17
2.2.1. Планирование на основании существующей инфраструктуры	17
2.3. Организация дискового пространства	17
2.4. Определение конфигурации кластера	18
2.4.1. Минимальная конфигурация кластера.....	18
2.4.2. Рекомендуемая конфигурация кластера	18
2.5. Шаблоны файлов ответов для проектирования кластеров	18
2.6. Подготовка информации для проекта кластера	19
2.6.1. Название кластера	19
2.6.2. Роли узлов и их IP-адреса.....	20
2.6.3. Виртуальный IP-адрес кластера (Public Address)	20
2.6.4. Название сетевого интерфейса.....	20
2.6.5. Название сервиса СУБД.....	20
2.6.6. Учетные записи пользователей.....	20
2.6.7. Название БД.....	21
3. Подготовка самоподписанных сертификатов TLS (SSL)	22
3.1. Создание СА ключей	22
3.2. Создание сертификатов сервера и администратора	22
4. Шаблоны файлов ответов	24
4.1. Заголовок файла ответов.....	25
4.2. Общие параметры узлов кластера	25
4.3. Секция инициализации СУБД	25

4.4. Запуск скриптов при формировании кластера	26
4.5. Секция настроек кластера	27
4.6. Группа настроек узлов кластера по умолчанию	29
4.7. Группа настроек доступа к WAL-архивам	30
4.8. Секция настроек дата-центров.....	31
4.8.1. Группа настроек узлов кластера	32
4.8.2. Группа настроек узла-арбитра	33
5. Автоматическое формирование кластера с использованием готовой конфигурации	34
5.1. Запуск специального режима jadog0	34
5.2. Запуск развертывания кластера	34
6. Ручное формирование кластера.....	36
6.1. Установка экземпляра компонента «jaDog» с использованием утилиты jadog_ctl.....	36
6.2. Установка экземпляра компонента «jaDog» с файлом ответов	36
6.3. Запуск настройки экземпляра компонента «jaDog» с подготовленным файлом ответов.....	36
7. Использование jaDog REST API.....	38
8. Устранение неполадок.....	39
8.1. Ошибка при подключении к компоненту с использованием jadog_ctl или REST API	39
8.1.1. Не создан пользователь компонента «jaDog».....	39
8.1.2. Некорректные настройки в конфигурационном файле jadog_hba.yml	39
8.1.3. При подключении используется сертификат SSL не того пользователя или не действительный.	40
8.2. Рассинхронизация кластера	41
8.2.1. Малые и средние объемы данных СУБД.....	41
8.2.2. Большие объемы данных СУБД.....	42
8.3. Некорректная перезагрузка кластера и/или СУБД.....	44
8.4. Диагностика работоспособности кластера	45
Приложение 1	48
Расположение служебных файлов, скриптов и шаблонов jaDog	48
Значения по умолчанию, используемые в компоненте «jaDog»	48
Приложение 2	50
Работа с консольной утилитой jadog_ctl из командной строки.....	50
Длительное ожидание выполнения консольной утилиты jadog_ctl	50
Приложение 3	51
Ключи запуска консольной утилиты jadog_ctl	51
Приложение 4	52
Команды консольной утилиты jadog_ctl.....	52
Приложение 5	60
Шаблон файла ответов автоматизированной настройки устойчивой конфигурации кластера без хранилища WAL-файлов	60
Приложение 6	66

Шаблон файла ответов автоматизированной настройки устойчивой конфигурации кластера с хранилищем WAL-файлов	66
Приложение 7	72
Шаблон файла ответов автоматизированной настройки устойчивой конфигурации кластера с использованием Trusted Address	72
Термины и определения	77
Перечень сокращений.....	78

1. ПРОЕКТИРОВАНИЕ КЛАСТЕРА

При проектировании отказоустойчивого кластера необходимо учитывать требования к доступности Recovery Time Objective (RTO), допустимой потере данных Recovery Point Objective (RPO), а также бюджетные и инфраструктурные ограничения.

В данном разделе рассматриваются рекомендуемые архитектурные решения, их преимущества, недостатки и сферы применения.

Описание применяемых терминов дано в разделе «Термины и определения».

1.1. Выбор архитектуры

В качестве рассматриваемых в данном руководстве примеров проектируемого отказоустойчивого кластера (далее – кластера) предлагаются следующие варианты архитектуры:

- устойчивая (рекомендуемая);
- минимально допустимая.

1.2. Устойчивая архитектура кластера

Устойчивая архитектура кластера, для поддержания его работоспособности, предполагает наличие в его составе нечетного количества узлов. К данному числу относятся главный и резервные узлы, узлы-арбитры (Referee).



Применение в кластере узла-арбитра совместно с использованием параметра `trusted_address` (см. первую часть документа «Компонент jaDog. Управление режимом работы узлов кластера» 643.72410666.00067-07 98 02-01) не рекомендуется. Совместное использование, в некоторых случаях, может привести к сетевому разделению узлов (split brain) или полной остановке кластера.



Рекомендуется размещать узел-арбитр во внешнем, по отношению к кластеру СУБД, ресурсе (датацентре, площадке и пр.).

Хорошей практикой устойчивой архитектуры кластера также является использование хранилища WAL-файлов для обеспечения восстановления данных при возникновении сбоев.

1.2.1. Устойчивый вариант кластера с хранилищем WAL-файлов

Устойчивый вариант кластера с хранилищем WAL-файлов применяется для конфигураций критически важных систем с требованиями к RPO и RTO, а также при высокой нагрузке на чтение данных.

Пример данного вида кластера имеет следующий состав узлов:

- главный узел;
- резервный узел №1;
- резервный узел №2;
- хранилище WAL-файлов.

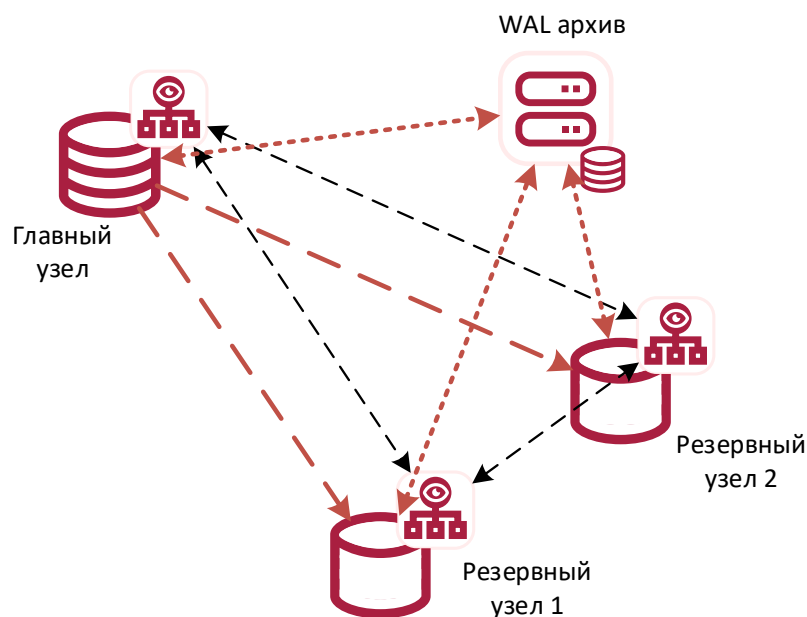


Рисунок 1.1 – Архитектура кластера с хранилищем WAL-файлов

Преимущества данной архитектуры кластера с хранилищем WAL-файлов:

- Кластер устойчивого варианта архитектуры из трех узлов:
 - остается отказоустойчивым при временном или полном выходе из строя одного из узлов;

- если приложение одновременно работает с главным и резервным узлами для чтения данных - оно продолжит работать без помех;
 - различные варианты построения типов репликации данных (синхронная, асинхронная, каскадная);
 - меньшее время восстановления RTO в случае возникновения аварии.
- Использование хранилища WAL-файлов:
- меньший объем безвозвратных потерь информации RPO;
 - допустимые длительные остановки с восстановлением из журналов без полного бэкапирования.

Шаблон файла ответов в формате YML для конфигурации «главный узел + резервный + резервный узел + WAL» приведен в Приложение 6.

Актуальная версия шаблона файлов располагается в каталоге /usr/jatoba-6/share/doc/jadog/clusters_kits/jadog_referee_wa/init_jadog_referee_wa.yml.

1.2.2. Устойчивый вариант кластера без хранилища WAL-файлов

Устойчивый вариант кластера без использования хранилища WAL-файлов обеспечивает баланс между расходами на инфраструктуру и надежностью.

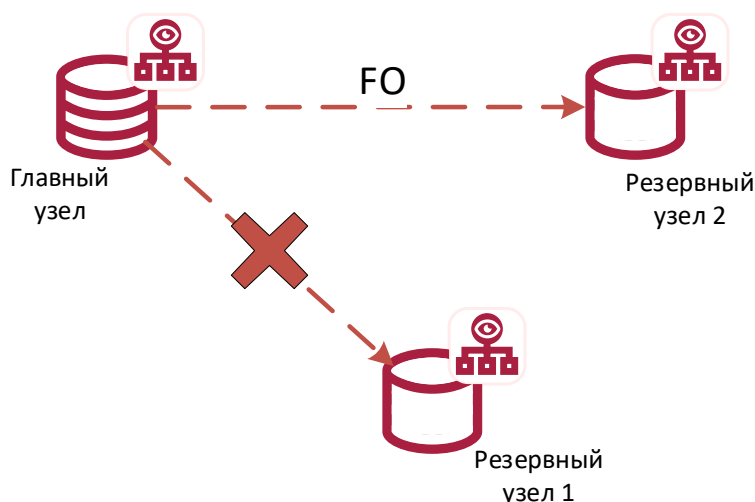


Рисунок 1.2 – Архитектура кластера без хранилища WAL-файлов

Преимущества архитектуры кластера без хранилища WAL-файлов:

- Кластер из трех узлов:

- остается отказоустойчивым при временном или полном выходе из строя одного из узлов;
- если приложение одновременно работает с главным и резервным узлами для чтения данных - оно продолжит работать без помех;
- различные варианты построения типов репликации (синхронная, асинхронная, каскадная).

Недостатком архитектуры без хранилища WAL-файлов является ограниченное время восстановления резервного узла без полного бэкапирования.

Шаблон файла ответов в формате YML для конфигурации «главный узел + резервный + резервный узел» приведен в Приложение 5.

Актуальная версия шаблона файлов располагается в каталоге /usr/jatoba-6/share/doc/jadog/clusters_kits/jadog_referee/init_jadog_referee.yml.

1.2.3. Устойчивая архитектура при использовании нескольких дата-центров

Устойчивый вариант при использовании нескольких дата-центров, в которых располагаются узлы геораспределенного кластера, состоит из следующих элементов:

1) Дата-центр №1:

- главный узел;
- резервный узел;
- узел арбитр.

2) Дата-центр №2:

- главный узел;
- резервный узел;
- узел арбитр.

3) Дата-центр №3:

- узел-арбитр;
- хранилище WAL.

Актуальная версия шаблона файлов для архитектуры при использовании нескольких дата-центров располагается в каталоге /usr/jatoba-6/share/doc/jadog/clusters_kits/ja_dtc_as/init_ja_dtc_as.yml.

1.3. Минимальная архитектура кластера

Минимально допустимая архитектура кластера содержит только три узла: главный и два резервных (реплики) с использованием дополнительных средств.

К дополнительным средствам относится использование:

- узла-арбитра (Referee);
- хранилища WAL-файлов.

1.3.1. Кластер минимальной конфигурации с узлом-арбитром и хранилищем WAL-файлов

Кластер минимально допустимой конфигурации с узлом-арбитром и хранилищем WAL-файлов применяется для систем с умеренными требованиями к RPO, а также при ограниченном бюджете на инфраструктуру.

Узел-арбитр в рассматриваемом кластере может одновременно выполнять роль хранилища WAL-файлов.



Применение в кластере узла-арбитра совместно с использованием параметра `trusted_address` (см. первую часть документа «Компонент jaDog. Управление режимом работы узлов кластера» 643.72410666.00067-07 98 02-01) не рекомендуется. Совместное использование, в некоторых случаях, может привести к сетевому разделению узлов (split brain) или полной остановке кластера.

Преимущества данной архитектуры с точки зрения отказоустойчивости:

- узел-рефери для интеллектуального определения состояния кластера при отработке аварийной ситуации;
- обеспечение сетевой связности узлов кластера;
- наличие хранилища WAL-файлов.

Недостатки: при выходе из строя главного узла отказоустойчивость кластера нарушается, поскольку в работе остаётся единственный резервный узел, к которому переходит роль главного.

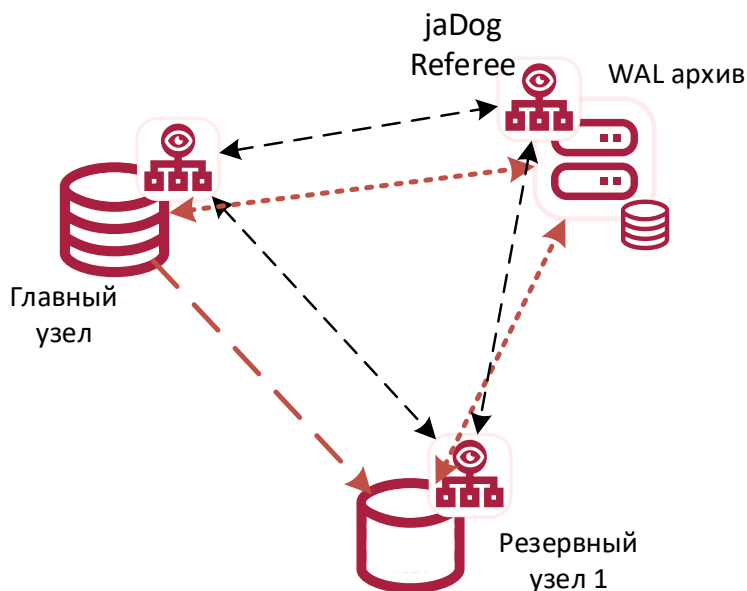


Рисунок 1.3 – Минимальная архитектура кластера с узлом-рефери и хранилищем WAL-файлов

1.3.2. Кластер минимальной конфигурации с узлом-арбитром без хранилища WAL-файлов

Кластер минимально допустимой конфигурации с узлом-арбитром без хранилища WAL-файлов обеспечивает баланс между расходами на инфраструктуру и надежностью.



Применение в кластере узла-арбитра совместно с использованием параметра `trusted_address` (см. первую часть документа «Компонент jaDog. Управление режимом работы узлов кластера» 643.72410666.00067-07 98 02-01) не рекомендуется. Совместное использование, в некоторых случаях, может привести к сетевому разделению узлов (split brain) или полной остановке кластера.

Преимущества данной архитектуры с точки зрения отказоустойчивости:

- узел-рефери для интеллектуального определения состояния кластера при отработке аварийной ситуации;
- обеспечение сетевой связности узлов кластера.

Недостатки:

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

– при выходе из строя главного узла отказоустойчивость кластера нарушается, поскольку в работе остаётся единственный резервный узел, к которому переходит роль главного;

– отсутствие хранилища WAL-файлов.

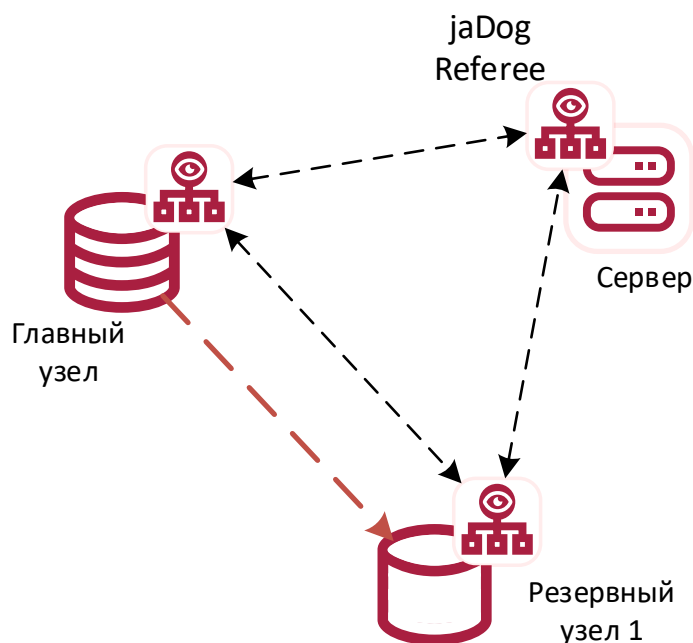


Рисунок 1.4 – Минимальная архитектура кластера с узлом-рефери без хранилища WAL-файлов

1.3.3. Кластер минимальной конфигурации с использованием Trusted Address

Минимальная архитектура кластера с использованием Trusted Address обеспечивает минимальные требования к инфраструктуре и простоту развёртывания.

! Применение в кластере узла-арбитра совместно с использованием параметра `trusted_address` (см. первую часть документа «Компонент jaDog. Управление режимом работы узлов кластера» 643.72410666.00067-07 98 02-01) не рекомендуется. Совместное использование, в некоторых случаях, может привести к сетевому разделению узлов (split brain) или полной остановке кластера.

Преимущества данной архитектуры с точки зрения отказоустойчивости:

– Trusted Address для определения состояния кластера при отработке аварийной ситуации;

Недостатки:

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

– риск возникновения сетевого разделения узлов кластера (Split Brain) при одновременном применении в кластере узла-арбитра;

– увеличение объема безвозвратных потерь информации (RPO);

– увеличение времени восстановления (RTO) в случае возникновения аварии.

Шаблон файла ответов в формате YML для конфигурации «главный узел + резервный + Trusted Address» приведен в Приложение .

Актуальная версия шаблона файлов располагается в каталоге /usr/jatoba-6/share/doc/jadog/clusters_kits/jadog_trusted_ip/init_jadog_trusted_ip.yml.

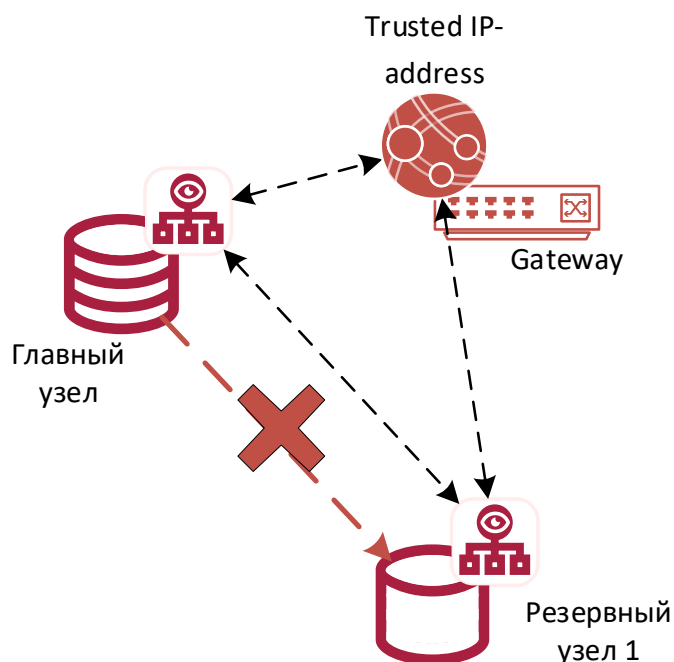


Рисунок 1.5 – Кластер минимально допустимой конфигурации с использованием Trusted Address

1.3.4. Минимальная архитектура при использовании нескольких дата-центров

Вариант минимальной архитектуры при использовании нескольких дата-центров, в которых располагаются узлы геораспределенного кластера, состоит из следующих элементов:

1) Дата-центр №1:

– главный узел;

– резервный узел.

2) Дата-центр №2:

– главный узел;

– резервный узел;

– узел арбитр.

1.4. Размещение хранилища WAL-файлов

WAL-архив рекомендуется располагать за пределами дата-центра, в котором расположен кластер.

В случае возникновения проблем в дата-центре данные из WAL-архива позволят сохранить больше данных и, соответственно, уменьшить RPO.

При отсутствии возможности размещения WAL-архива за пределами дата-центра, допустимо расположить WAL-архив внутри ДЦ, к примеру на узле-арбитре (см. пример из п.п. 1.3.1).

2. ВАРИАНТЫ ПОСТРОЕНИЯ ОТКАЗОУСТОЙЧИВОГО КЛАСТЕРА

В данном разделе приводятся два основных подхода к созданию отказоустойчивого кластера.

- развертывание нового кластера с последующей миграцией данных;
- преобразование существующей (Stand-Alone, SA) конфигурации в отказоустойчивый кластер.

Оба варианта включают общие этапы, такие как проектирование архитектуры, подготовка серверов и настройка SSL, но отличаются деталями выполнения.

2.1. Построение нового кластера и перенос данных СУБД

Развертывание нового кластера с миграцией данных применяется в случаях когда:

- требуется создать новый кластер без зависимостей от существующей инфраструктуры;
- данные можно перенести без длительного простоя.

2.1.1. Планирование на основании новой инфраструктуры

В общем случае план создания подобного кластера состоит в выполнении следующих действий.

- создание проекта архитектуры кластера (см. раздел 1):
 - определение роли узлов (Master, Slave, Referee, WAL-архив);
 - определение количества узлов (нечетное);
 - выбор типа репликации данных (синхронная/асинхронная);
 - планирование сетевой адресации и интерфейсов.
- подготовка инфраструктуры: дисковых пространств и сетевых служб (firewall, DNS).
- выпуск сертификатов SSL;
- заполнение YML-файла ответов с параметрами для автоматизированного формирования кластера;
- запуск служебной утилиты jalog0 на всех узлах формируемого кластера;

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

- запустить автоматизированное формирование кластера с узла (будущий главный узел);
- перенос данных из существующей SA-конфигурации в новый кластер с помощью инструментов репликации или дампа.

2.2. Использование существующей инфраструктуры

Развертывание кластера с использованием существующей инфраструктуры применяется в случаях, когда:

- требуется модернизировать работающую SA-систему без полного перестроения;
- необходимо минимизировать простой.

При использовании существующей архитектуры необходимо довести количество узлов до нечетного добавив или удалив.

2.2.1. Планирование на основании существующей инфраструктуры

В общем случае план преобразования существующей инфраструктуры в кластер состоит в выполнении следующих действий:

- создание проекта архитектуры кластера (см. раздел 1) аналогично первому варианту, но с учетом интеграции существующего узла SA;
- подготовка инфраструктуры: дисковых пространств и сетевых служб (firewall, DNS).
- выпуск сертификатов SSL;
- заполнение YML-файла ответов с параметрами для автоматизированного формирования кластера;
- запуск служебной утилиты jadog0 на всех узлах формируемого кластера;
- запустить автоматизированное формирование кластера с узла (будущий главный узел).

2.3. Организация дискового пространства

При планировании нового кластера или использовании существующей инфраструктуры необходимо убедиться в том, что дисковое пространство на каждом узле составляет не менее $X*3$, где X — размер каталога данных (pg_data) СУБД.

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



Допускается использовать $X*2$, но в этом случае при возникновении первого сбоя и последующей попытке восстановить работоспособность кластера, свободного места может не хватить, что приведёт к вынужденной остановке кластера.

2.4. Определение конфигурации кластера

Здесь и далее условные обозначения:

- M – главный узел с ролью «Master»;
- S(r) – резервный узел с ролью «Slave», а (r) – выбранный тип репликации данных: а – асинхронный (Async), s – синхронный (Sync).
- Referee – узел-арбитр.

2.4.1. Минимальная конфигурация кластера

Минимальная конфигурация кластера, состоящая из трех узлов, имеет следующую формулу:

$$my_cluster = M + S(a) + S(a)$$

2.4.2. Рекомендуемая конфигурация кластера

Рекомендуемая конфигурация кластера имеет следующую формулу:

$$my_cluster = M + S(a)*x + Referee + WALarchive$$

Где x – четное количество резервных узлов.

2.5. Шаблоны файлов ответов для проектирования кластеров

Параметры проектируемого кластера указываются в шаблоне файла ответов.

Шаблоны файлов ответов входят в поставку СУБД «Jatoba» и располагаются в каталогах /usr/jatoba-6/share/doc/jadog/clusters_kits.

Названия шаблонов файлов ответов и формула проектируемого кластера:

- init_jadog_referee_wa.yml - $M + S(a)*2 + Referee + WALarchive$;
- init_jadog_referee.yml - $M + S(a)*2 + Referee$;
- init_jadog_trusted_ip.yml - $M + S(a)*2 + Trusted Address$



Применение в кластере узла-арбитра совместно с использованием параметра `trusted_address` (см. первую часть документа «Компонент jaDog. Управление режимом работы узлов кластера» 643.72410666.00067-07 98 02-01) не рекомендуется. Совместное использование, в некоторых случаях, может привести к сетевому разделению узлов (split brain) или полной остановке кластера.

Описание процедуры заполнения шаблона файла ответов при проектировании кластера приводится в разделе 4 данного руководства.

2.6. Подготовка информации для проекта кластера

В соответствии с ранее приведенными рекомендациями из данного документа подготавливается информация для внесения в шаблон файла ответов, примеры которой приводятся далее в этом документе.

Курсивом приводится название параметра в шаблоне файла ответов.

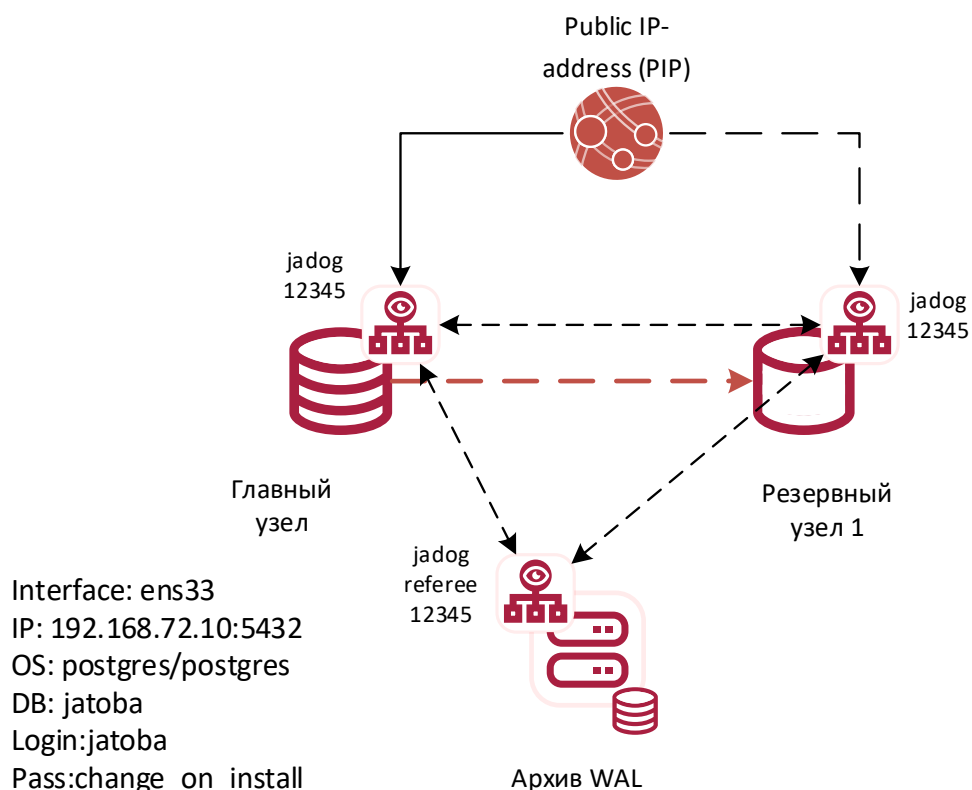


Рисунок 2.1 – Проект конфигурации кластера

2.6.1. Название кластера

Уникальное название кластера (например, `my_cluster`), которое будет использоваться для идентификации в системе.

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

cluster_name: my_cluster

2.6.2. Роли узлов и их IP-адреса

Роли узлов и их IP-адреса (здесь и далее IP адреса приводятся как примеры):

- Master: главный узел, 192.168.72.11. DNS-запись: jatoba-11.
- Slave (асинхронная реплика): резервный узел, 192.168.72.12. DNS-запись: jatoba-12
- Slave (асинхронная реплика): резервный узел, 192.168.72.13. DNS-запись: jatoba-13
- Referee (опционально): узел-арбитр для разрешения конфликтов, 192.168.72.14. DNS-запись: jatoba-14
- WAL-архив (опционально): сервер для хранения журналов транзакций, 192.168.72.15.

2.6.3. Виртуальный IP-адрес кластера (Public Address)

Адрес, через который будет осуществляться доступ к кластеру (например, 192.168.72.10). DNS-запись: jatoba-10.

public_address: 192.168.72.10

2.6.4. Название сетевого интерфейса

Название сетевого интерфейса на узлах, используемого для работы кластера (например, ens33). В ОС Linux название сетевого интерфейса отображается при использовании команды «ip a».

2.6.5. Название сервиса СУБД

Название сервиса СУБД, например, jatoba-6.

db_service_name: jatoba-6

2.6.6. Учетные записи пользователей

Для администрирования и взаимодействия между узлами потребуются следующие технологические учетные записи:

- УЗ администратора jaDog (admin);
- УЗ для взаимодействия узлов (interdog);
- УЗ для репликации данных в СУБД (jadog_user).

2.6.6.1 Технологическая УЗ администратора кластера

Название технологической УЗ администратора кластера:

name: admin

Пароль технологической УЗ администратора кластера:

pass: change_on_install

Разрешенные адреса или подсети для доступа к кластеру. Значения: 'all', имя хоста ('myhost.local.net') или IP адрес с маской ('192.168.100.1/24'):

address: all

Метод доступа. Значения: "sha-256", "ssl". "sha-256" по умолчанию:

method: sha-256

2.6.6.2 Технологическая УЗ для взаимодействия с узлами кластера

Название технологической УЗ:

name: interdog

Пароль технологической УЗ:

pass: change_on_install

Разрешенные адреса или подсети для доступа к кластеру. Значения: 'all', имя хоста ('myhost.local.net') или IP адрес с маской ('192.168.100.1/24'):

address: all

Метод доступа. Значения: "sha-256", "ssl". "sha-256" по умолчанию:

method: sha-256

2.6.6.3 Технологическая УЗ для репликации данных

jadog_user

2.6.7. Название БД

Название базы данных (БД), где будет установлена серверная часть кластера ("postgres" по умолчанию):

database: jatoba

3. ПОДГОТОВКА САМОПОДПИСАННЫХ СЕРТИФИКАТОВ TLS (SSL)



Использование самоподписанных сертификатов TLS (SSL) допустимо только при проектировании и отладке кластера. В целях обеспечения информационной безопасности в промышленной эксплуатации допускается использование только сертификатов, выданных удостоверяющим центром (CA).



extendedKeyUsage для сертификата сервера должен в обязательном порядке serverAuth.

Для клиентского сертификата параметр должен быть clientAuth.

Допускается указание обоих значений вместе через запятую.

3.1. Создание CA ключей

```
openssl req -x509 -days 365 -newkey rsa:4096 -sha256 -nodes -  
keyout ca-key.crt -out ca-cert.crt -subj "/C=RU/ST=SPB/L=Saint-  
Petersburg /CN=Self-signed CA"
```

3.2. Создание сертификатов сервера и администратора

В качестве значений «CN=» указывается название сервера (hostname) или имя (логин) администратора, в зависимости от назначения сертификата, например:

```
openssl req -newkey rsa:4096 -nodes -keyout server1-key.crt -  
out server1-req.crt -subj  
"/extendedKeyUsage=serverAuth,clientAuth  
/subjectAltName=DNS:jatoba-10,DNS:jatoba-  
11,DNS:localhost,IP:127.0.0.1,IP:192.168.72.10,IP:192.168.72.11  
/CN=jatoba-11"
```

Аналогично сертификаты создаются для всех остальных узлов кластера:

```
openssl req -newkey rsa:4096 -nodes -keyout server3-key.crt -  
out server3-req.crt -subj  
"/extendedKeyUsage=serverAuth,clientAuth /subjectAltName=  
DNS:jatoba-10,DNS:DNS_name_server,DNS:localhost,IP:127.0.0.1,  
IP:192.168.72.10,IP:IP_address_server /CN=server_name_N"
```

Создание сертификата для администратора кластера:

```
openssl req -newkey rsa:4096 -nodes -keyout user-key.crt -out  
user-req.crt -subj "/extendedKeyUsage=clientAuth /CN=user_name"
```

Подписание выпущенного сертификата для узла кластера:

```
openssl x509 -req -in server1-req.crt -days 365 -CA ca-cert.crt  
-CAkey ca-key.crt -CAcreateserial -out server1-cert.crt
```

Аналогично сертификаты подписываются для всех остальных узлов кластера:

```
openssl x509 -req -in server3-req.crt -days 365 -CA ca-cert.crt  
-CAkey ca-key.crt -CAcreateserial -out serverN-cert.crt
```

Подписание выпущенного сертификата для администратора кластера:

```
openssl x509 -req -in user-req.crt -days 365 -CA ca-cert.crt -  
CAkey ca-key.crt -CAcreateserial -out user-cert.crt
```

Проверка корректности выпущенного сертификата узла кластера:

```
openssl x509 -in server1-cert.crt -noout -text
```

Проверка корректности выпущенного сертификата администратора (пользователя) кластера:

```
openssl x509 -in user-cert.crt -noout -text
```

4. ШАБЛОНЫ ФАЙЛОВ ОТВЕТОВ

Шаблоны файлов ответов, с помощью которых выполняется разворачивание кластеров из данного руководства, располагаются в каталогах `/usr/jatoba-6/share/doc/jadog/clusters_kits`. В данном каталоге также размещен файл `read-me_first.md`, в котором приводится краткое описание назначения файлов ответов.

В каталоге `/usr/jatoba-6/share/doc/jadog/clusters_kits` размещены каталоги с файлами ответов для автоматизированного разворачивания кластеров, указанные в таблице 4.1.

Таблица 4.1– Каталоги с файлами ответов в `/usr/jatoba-6/share/doc/jadog/clusters_kits`

Каталог	Описание	Схема кластера
<code>jadog_referee</code>	Двухузловой кластер с узлом-арбитром	M+S(a)+R
<code>jadog_referee_wa</code>	Двухузловой кластер с узлом-арбитром и WAL-архивом, расположенном на узле-арбитре	M+S(a)+R+Wa
<code>jadog_trusted_ip</code>	Двухузловой кластер с Trusted IP	M+S(a)
<code>jadog_trusted_ip_ssl</code>	Двухузловой кластер с <code>trusted_ip</code> с SSL аутентификацией всех соединений	M+S(a)
<code>ja_dtc_as</code>	Отказоустойчивый, географически распределенный кластер Active-StandBy	DC1(M+S(a)) + DC2(S(a)+S(a)+R)
<code>ja_hipe_cluster</code>	Отказоустойчивый, высокопроизводительный кластер - <code>ja_Hipe_Cluster</code> и создание группы кластеров (bundle)	-
<code>ja_hipe_cluster_fqdn_ssl</code>	Отказоустойчивый, высокопроизводительный кластер - <code>ja_Hipe_Cluster</code> на FQDN и SSL, Cluster и создание группы кластеров (bundle)	-
<code>standalone</code>	Настройки отдельного сервиса «jaDog» для standalone сервера СУБД «Jatoba»	-

В качестве примера при проектировании кластера будет рассмотрен шаблон файла ответов `init_jadog_referee_wa.yml`.

Шаблон файла ответов разделен на секции, подсекции и блоки параметров.

Шаблон файла ответов содержит перечень параметров, а также комментарии (отмеченные символом #) с описанием назначения параметра.

4.1. Заголовок файла ответов

Заголовок файла ответов не требует внесения изменений и содержит служебную информацию о версиях компонента «jaDog» и назначении:

```
apiVersion: jaDog v4          # Версия jaDog
kind: jadog_referee_wal      # Назначение скрипта
```

4.2. Общие параметры узлов кластера

Секция общих параметров узлов кластера предназначена для описания универсальных параметров кластера или кластеров (если они создаются одновременно).

Значения из этой секции параметров применяются к каждому узлу кластера при его формировании. Например, параметр `db_init_conn_string` предназначается для параметров инициализации СУБД.

Для формирования кластера на основе существующей СУБД параметры доступа (`user`, `password` и так далее) к ней должны совпадать с теми, что уже установлены в настройках СУБД. Для кластера на основе новой СУБД параметры доступа создаются при инициализации.

```
default_cluster_params:      # Блок атрибутов используемый для
                             # каждого кластера

                             db_init_conn_string: host=127.0.0.1 port=5432
user=postgres dbname=postgres password='change_on_install'
                             # Строка коннекта к СУБД для установки расширений и
                             # выполнения скриптов. Требует прав доступа администратора
```

4.3. Секция инициализации СУБД

Блок `initdb` содержит атрибуты для инициализации СУБД

```
initdb:                      # Блок инициализации СУБД. Содержит параметры
                             # инициализации СУБД

                             initdb_options: "--locale=ru_RU.utf8 --encoding=UTF-8" #
                             Строка параметров инициализации СУБД Jatoba.
```

Секция `postgresql.conf` используется для установки параметров в СУБД сразу после инициализации. В секции `postgresql.conf` можно вносить необходимые для настройки СУБД изменения:

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

```
postgresql.conf:  # Параметры будут установлены при
формировании кластера в конец файла postgresql.conf.

listen_addresses: '*'
log_destination: 'stderr'
logging_collector: on
log_directory: 'log'
log_filename: 'jatoba-%Y-%m-%d_%H%M%S.log'
log_rotation_age: 1d
log_rotation_size: 0
log_truncate_on_rotation: off
log_line_prefix: '%m [%p] '
log_statement: ddl
```

Секция pg_hba.conf используется для установки параметров доступа к СУБД.



Параметры секции pg_hba.conf полностью заменяют значения из файла pg_hba.conf.

```
pg_hba.conf:  # Параметры будут установлены в файл pg_hba.conf
при формировании кластера

- local all postgres scram-sha-256
- local jatoba jalog_user scram-sha-256
- host all postgres 192.168.72.0/24 trust
- host all postgres 127.0.0.1/32 scram-sha-256
- host jatoba jalog_user 127.0.0.1/32 scram-sha-256
- host replication jalog_user 127.0.0.1/32 scram-sha-256
- host replication jalog_user 192.168.72.0/24 scram-sha-256
```

4.4. Запуск скриптов при формировании кластера

Скрипты из секции sql_scripts выполняются во время формирования каждого кластера также сразу после инициализации СУБД.

```
sql_scripts:      # Секция пользовательских скриптов, выполняемых
на главном узле каждого создаваемого кластера. Скрипты
определяются администратором кластера.

    - /home/user/cluster_default_script.sql      #
Пользовательский SQL скрипт выполняемый с аутентификационными
данными cluster_settings.db_init_conn_string после создания
всех расширений
```

Ключевые особенности:

– скрипты выполняются с учётными данными, указанными в параметре `db_init_conn_string`;

– целевой базой данных для выполнения скриптов является БД, заданная в `cluster_settings:db_connection_settings:database` ;

– поддерживается выполнение нескольких скриптов последовательно.

Типичные сценарии использования скриптов:

– наполнение базы начальными данными.

– установка дополнительных расширений (при наличии соответствующих пакетов).

– выполнение специальных настроек.



Примечание: администратор кластера самостоятельно определяет состав и содержание выполняемых скриптов в соответствии с требованиями конкретного проекта.

4.5. Секция настроек кластера

Следующий блок `cluster_settings` и входящие в него подгруппы предназначены для описания характеристик конкретного кластера.

Данные атрибуты требуют переназначения, если есть необходимость индивидуализировать конкретный кластер. Описание атрибутов даны в комментариях к ним.

```
cluster_settings:

    - cluster_name: my_cluster      # Наименование кластера.
Обязательный параметр. Кластер может быть один или несколько.
```

```
cluster_master_node: node_11    # Название главного узла.
Обязательный параметр.

cluster_referee_nodes:          # Блок узлов, являющимися
referee. Их может быть несколько.

    - node_referee              # Название узла, выступающей в роли
referee.

activated: true                 # Флаг активации кластера после
формирования. true - активировать (назначать public_address),
false - не активировать.
```

Внутри секции `cluster_settings` может располагаться подсекция `sql_scripts`. Скрипты в данной подсекции будут выполняться только для данного конкретного кластера с указанным `cluster_name`.

```
sql_scripts:

    - /home/user/cl_coord_inside_script.sql    #
Пользовательский SQL скрипт выполняемый с аутентификационными
данными cluster_settings.db_init_conn_string после создания
всех расширений
```



Примечание: администратор кластера самостоятельно определяет состав и содержание выполняемых скриптов в соответствии с требованиями конкретного проекта.

Подсекция `jadog_users` описывает учетные записи для компонента «jaDog». Это персонифицированные УЗ для администраторов кластера и технологические для взаимодействия служб компонента с другими экземплярами в кластере или работы с jaDog REST API.

Описание УЗ позволяет указать логин (`name`), пароль (`pass`) (если он необходим), данные для внесения в файл `jadog_hba.cfg` – маска адреса доступа (`address`) и метод аутентификации в `jadog` (`method`).

```
jadog_users:                  # Учетные записи для администрирования jaDog.
Создаются УЗ и запись в jadog_hba.cfg.

    - name: admin              # Имя учетной записи. Обязательный параметр.

    pass: change_on_install    # Пароль учетной записи. Обязательный
параметр. Для обеспечения безопасности смените пароль УЗ после установки.
```

```

        address: all      # Разрешенные адреса или подсети для доступа к
jaDog. Значения: 'all', имя хоста( 'myhost.local.net' ) или IP адрес с
маской ( '192.168.100.1/24' ).

        method: sha-256  # Метод доступа. Значения: "sha-256", "ssl".
"sha-256" по умолчанию.

        - name: interdog  # УЗ можно создать несколько. Для обеспечения
безопасности смените пароль УЗ после установки.

        pass: change_on_install  # Для обеспечения безопасности смените
пароль УЗ после установки.

        address: all

#        - name: restuser  # УЗ можно создать несколько. Для обеспечения
безопасности смените пароль УЗ после установки.

#        address: all

#        method: ssl

```

4.6. Группа настроек узлов кластера по умолчанию

Группа атрибутов `default_node_params` служит для определения параметров, которые будут применены к каждому узлу создаваемого кластера.

```

default_node_params:  # Параметры по-умолчанию, которые будут
использовать для каждого узла

    param_jadog:      # Параметры конфигурирования экземпляра jaDog.

#        service_name: jadog  # Имя сервиса jaDog для ноды. По умолчанию
"jadog"

        public_address: 192.168.72.10  # Виртуальный IP адрес кластера.
Public address. Обязательный параметр.

#        trusted_address: 192.168.72.2  # Доверенный IP адрес. !!!Если
кластер собирается с использование узла Referee, параметр не
устанавливается!!!

        network_interface: ens33  # Наименование сетевого интерфейса для
виртуального IP кластера. Network interface name. Обязательный параметр.

        interconnect_user: interdog  # Имя технологической УЗ для
взаимодействия с другими jaDog сервисами. Обязательный параметр.

    param_log:        # Параметры журналирования

        logs_filename: jadog-%Y-%m-%d_%H%M%S  # Шаблон для файлов
функциональных журналов

        logs_screen: true  # Включение вывода журналов на экран. По
умолчанию "false"

        logs_type: csv, security.csv  # Форматы файлов функционального
журнала и журнала событий ИБ. На первом месте формат функционального файла
(csv, json) на втором формат файла журнала событий ИБ (security.json,
security.csv)

```

```
logs_level: debug2 # Уровень функционального журналирования
param_security_log: # Параметры журнала событий информационной
безопасности (ИБ)
security_logs_filename: jalog-%Y-%m-%d_%H%M%S # Шаблон для файлов
журнал событий ИБ
param_postgres: # Блок параметров для работы с СУБД
db_service_name: jatoba-@JATOBA_MAJ_VER@ # Название сервиса
экземпляра СУБД в операционной системе
db_connection_settings: # Параметры для подключения к СУБД
database: jatoba # Имя базы данных для установки расширения
jaDog. По умолчанию "postgres"
passfile: /var/lib/jatoba/.pgpass # Jatoba passfile
user: jalog_user # Имя УЗ для работы с СУБД. Обязательный
параметр.
user_pass: change_on_install # Пароль УЗ для работы с СУБД.
Обязательный параметр. Сменить пароль УЗ в СУБД после формирования
кластера.
param_rest_api: # Блок параметров для REST-сервера
rest_api_use: true # Признак использования REST-сервера
rest_api_cert_file: /var/lib/jatoba/ssl/jalog_cert/jalog_server.crt
# Путь до файла сертификата REST-сервера
rest_api_key_file: /var/lib/jatoba/ssl/jalog_cert/jalog_server.key
# Путь до файла ключа REST-сервера
rest_api_ca_file: /var/lib/jatoba/ssl/jatoba_ca.crt # Путь до
файла CA REST-сервера
# rest_api_crl_file: /var/lib/jatoba/ssl/jalog_cert/ca-cert.crt
# Путь до файла CRL REST-сервера
rest_api_listen_address: 0.0.0.0 # Прослушиваемый IP-адрес REST-
сервера
rest_api_listen_port: 54443 # Порт REST-сервера
```

4.7. Группа настроек доступа к WAL-архивам

В группе `param_archive` необходимо определить пути к каталогам и команды работы с WAL-архивом.



Примечание: администратору системы необходимо заранее создать и смонтировать указанные каталоги на каждом узле кластера, где требуется хранение WAL-файлов.

При настройке путей для WAL-архивов необходимо учитывать следующие требования:

- должны указываться локальные пути файловой системы.
- соответствующие каталоги должны быть предварительно смонтированы.
- использование сетевых адресов (UNC-путей) в конфигурационных файлах не поддерживается.

```
param_archive:
    wal_archive_directory: "/mnt/wals/" # Путь до
сохранения архивов
    wal_archive_command: "cp %p /mnt/wals/%f" # Команды
архивирования
    wal_archive_cleanup_command: "/usr/jatoba-
@JATOBA_MAJ_VER@/bin/pg_archivecleanup /mnt/wals/ %s" # Команды
очистки
    wal_archive_cleanup_needed: true      # Признак
необходимости очистки
    wal_archive_cleanup_timeout: 500000 # Временной
промежуток очистки(ms)
param_restore:
    wal_restore_command: "cp /mnt/wals/%f %p" # Команда
восстановления
```

4.8. Секция настроек дата-центров

Секция `datacenters` описывает конфигурацию дата-центров и распределение по ним узлов. Необходим, как минимум, один дата-центр. В примере название дата-центра: `DC_Piter`

```
datacenters:
    - datacenter: DC_Piter      # Уникальное наименование
дата-центра. Дата-центр может быть один или несколько.
    nodes:                    # Блок описания узлов кластера
```

4.8.1. Группа настроек узлов кластера

Уникальные параметры для каждого узла указываются в группе nodes, входящей в секцию datacenters.

```
- node_name: node_11      # Название узла в кластере.  
    param_jadog:          # Параметры конфигурирования экземпляра  
    jadog.  
    ip: 192.168.72.11     # IP-адрес узла  
  
#    network_interface: ens33      # Наименование сетевого  
# интерфейса для виртуального IP кластера. Network interface  
# name. Обязательный параметр
```

Для использования синхронной репликации узлов необходимо применять следующие настройки узла:

```
param_synchronous:      # Раскомментировать блок параметров, если  
# потребуется синхронная реплика  
    synchronous_commit: on      # Тип синхронного коммита согласно  
# документации postgresql: on, remote_apply, remote_write, local,  
# off  
    synchronous_type: ANY      # Как будет работать синхронная  
# репликация по принципам: FIRST или ANY  
    synchronous_commit_max_nodes: 1 # Количество узлов,  
# требующих подтверждения синхронного коммита  
param_replication:      # Параметры репликации узла  
    replication_slot_name: rs_node_11 # Уникальное имя слота  
# репликации в рамках кластера
```

В результате настройки узла с синхронной репликацией будут выглядеть следующим образом:

```
- node_name: node_12  
    param_jadog:  
    ip: 192.168.72.12  
    param_replication:  
    replication_slot_name: rs_node_12
```



```

param_synchronous:  # Раскомментировать блок
параметров, если потребуется синхронная реплика

    synchronous_commit: on          # Тип синхронного коммита
согласно документации postgresql: on, remote_apply,
remote_write, local, off

    synchronous_type: ANY           # Как будет работать
синхронная репликация по принципам: FIRST или ANY

    synchronous_commit_max_nodes: 1 # Количество узлов,
требующих подтверждения синхронного коммита

```

4.8.2. Группа настроек узла-арбитра

Информация по параметрам узла jadog-referee также описывается в соответствующем блоке

```

- node_name: node_referee          # Название узла с ролью
referee

    param_jadog:
        ip: 192.168.72.23

    param_replication:
        replication_slot_name: rs_referee

```

5. АВТОМАТИЧЕСКОЕ ФОРМИРОВАНИЕ КЛАСТЕРА С ИСПОЛЬЗОВАНИЕМ ГОТОВОЙ КОНФИГУРАЦИИ

5.1. Запуск специального режима jalog0

Автоматическое формирование кластера на узлах, описанных в файле ответов, производится с помощью запуска компонента «jaDog» в специальном режиме jalog0 на каждом узле кластера.

```
sudo /usr/jatoba-6/bin/jalog jalog0 --sslcafile  
/var/lib/jatoba/ssl/jatoba_ca.crt --sslcertfile  
/var/lib/jatoba/ssl/jalog_cert/jalog_server.crt --sslkeyfile  
/var/lib/jatoba/ssl/jalog_cert/jalog_server.key
```

Специальный режим jalog0 запускается с указанием путей к расположению SSL сертификатов. Сертификаты SSL должны быть сформированы для каждого узла адресно (раздел 3).



Запуск специального режима jalog0 без указания сертификатов SSL возможен с использованием режима --basic. Описание режима приведено в первой части «Компонент jaDog. Управление режимом работы узлов кластера» 643.72410666.00067-07 98 02-01.

После запуска специального режима jalog0 компонент «jaDog» находится в режиме ожидания запуска развертывания кластера с использованием файла ответов.

В процессе формирования кластера в консоли ОС отображается процесс настройки компонента «jaDog» на данном узле кластера.

5.2. Запуск развертывания кластера



Команду развертывания кластера можно запустить с любого узла, на котором запущен jalog0. Рекомендуется выполнять команду с узла, который согласно проекту будет главным узлом будущего кластера.

Развертывание кластера происходит при помощи выполнения следующей команды:

```
/usr/jatoba-6/bin/jalog_ctl create_cluster -T 9999999 -c  
/<dir1>/ jalog_cross_cluster.yml --ssl --sslcafile  
/var/lib/jatoba/ssl/jatoba_ca.crt --sslcertfile  
/var/lib/jatoba/ssl/jalog_cert/jalog_server.crt --sslcrlfile
```

```
/var/lib/jatoba/ssl/jadog_cert/jadog_server.crl --sslkeyfile  
/var/lib/jatoba/ssl/jadog_cert/jadog_server.key
```

6. РУЧНОЕ ФОРМИРОВАНИЕ КЛАСТЕРА

6.1. Установка экземпляра компонента «jaDog» с использованием утилиты `jadog_ctl`

Подробное описание использования утилиты `jadog_ctl` при установке экземпляра компонента «jaDog» приведено в первой части документа «Компонент jaDog. Управление режимом работы узлов кластера» 643.72410666.00067-07 98 02-03.

Ручное формирование кластера с использованием `jadog_ctl` выполняется при помощи следующей команды:

```
cd /usr/jatoba-6/bin  
./jadog setup
```

6.2. Установка экземпляра компонента «jaDog» с файлом ответов

Шаблоны файлов ответов установки компонента «jaDog» располагаются в каталоге `/usr/jatoba-6/share/doc/jadog/clusters_kits/standalone`:

– `jadog_setup_faq.yml` – только основные обязательные параметры, все дополнительные параметры установлены в значения по умолчанию;

– `jadog_setup_faq_full.yml` – с полным набором конфигурационных параметров.

Подготовку шаблона файлов ответов требуется выполнять для настройки каждого отдельного узла проектируемого кластера.

6.3. Запуск настройки экземпляра компонента «jaDog» с подготовленным файлом ответов

Ручное формирование кластера с использованием файла выполняется при помощи следующей команды:

```
/usr/jatoba-6/bin# ./jadog setup -c jadog_setup_faq.yml
```

Процесс настройки экземпляра компонента «jaDog» с файлом ответов подробно описан во второй части документации «Компонент jaDog. Управление режимом работы узлов кластера».

При ручном формировании кластера могут не понадобиться TLS (SSL) сертификаты, если нет необходимости включать режим использования jaDog REST API.

Режим REST API можно будет включить после формирования кластера.

Необходимо добавить, что кроме прямого диалогового режима формирования кластера в утилите управления - `jadog_ctl`, можно использовать возможность запуска команд не в диалоговом режиме.

Приведенный в подразделе 2.6 кластер возможно сформировать, при условии предварительной настройки компонента «jaDog» на каждом узле, следующим набором команд:

```
/usr/jatoba-6/bin/jadog_ctl -h 192.168.72.11 -U admin -W  
'change_on_install' -q -f json -C "add cluster 'my_cluster'"  
  
/usr/jatoba-6/bin/jadog_ctl -h 192.168.72.11 -U admin -W  
'change_on_install' -q -f json -C "cluster add master  
192.168.72.11 54321 as 'node_11'"  
  
/usr/jatoba-6/bin/jadog_ctl -h 192.168.72.11 -U admin -W  
'change_on_install' -q -f json -C "cluster add slave  
192.168.72.12 54321 as 'node_12'"  
  
/usr/jatoba-6/bin/jadog_ctl -h 192.168.72.11 -U admin -W  
'change_on_install' -q -f json -C "cluster add slave  
192.168.72.13 54321 as 'node_13'"  
  
/usr/jatoba-6/bin/jadog_ctl -h 192.168.72.11 -U admin -W  
'change_on_install' -q -f json -C "cluster add referee  
192.168.72.23 54321 as 'node_referee'"
```

После выполнения каждой команды будет формироваться ответ в формате JSON, который может быть использован для автоматической обработки.

Получить статус сформированного кластера можно при помощи команды:

```
/usr/jatoba-6/bin/jadog_ctl -U admin -W 'admin' -q -f json -C  
"cluster status"
```

7. ИСПОЛЬЗОВАНИЕ JADOG REST API

Компонент «jaDog», помимо утилиты управления jadog_ctl, имеет встроенный функционал REST API.

YML-файл с описанием команд REST API (в нотации OpenAPI V3) расположен в каталоге /usr/jatoba-6/share/doc/jadog/api.yml

Функционал использования jaDog REST API работает только по защищенному протоколу HTTPS. Для использования jaDog REST API потребуются выпущенные сертификаты SSL (см. раздел 3), а также указание пути к ним в параметрах файла ответов при развертывании кластера или в последствии при включении данного функционала в работу.

Описание команд jaDog REST API приведено во второй части документа «Компонент jaDog. Управление режимом работы узлов кластера».

Если выполнена настройка REST API, проверка работоспособности возможна с использованием следующей команды:

```
curl --cacert /<dir>/root.crt --cert /<dir>/client.crt --key  
/<dir>/client.key -k https://host_name(ip):54443/metrics
```

где <dir> - каталог с сертификатами SSL

8. УСТРАНЕНИЕ НЕПОЛАДОК

При поиске и устранении неполадок, возникающих при работе с компонентом «jaDog», необходимо в первую очередь просмотреть журналы событий на предмет сообщений, указывающих на причину возникновения проблемы.

8.1. Ошибка при подключении к компоненту с использованием `jadog_ctl` или REST API

8.1.1. Не создан пользователь компонента «jaDog»

Первой вероятной причиной данной проблемы является отсутствие пользователя в компоненте «jaDog». Для того чтобы добавить пользователя, необходимо на главном узле (роль Master) выполнить следующую команду с использованием утилиты `jadog_ctl`:

```
add user 'new_user_name' 'user_password'
```

Если у создаваемого пользователя будет определен режим доступа с использованием сертификатов SSL, то в указанной команде пароль задавать не обязательно.

После создания необходимо внести правила доступа для данного пользователя в конфигурационный файл `jadog_hba.yml`.

8.1.2. Некорректные настройки в конфигурационном файле `jadog_hba.yml`

Следующей вероятной причиной упомянутой проблемы являются некорректные правила доступа пользователя в конфигурационном файле `jadog_hba.yml`.

При отсутствии правил внести в конфигурационный файл `jadog_hba.yml`:

# USER	ADDRESS	METHOD
new_user_name	all	ssl

После этого синхронизировать конфигурационный файл `jadog_hba.yml` с другими узлами кластера при помощи:

– копированием файла вручную;

– при помощи механизма синхронизации файлов компонента «jaDog» (описание данного функционала приводится в первой части документа «Компонент jaDog. Управление режимом работы узлов кластера» 643.72410666.00067-07 98 02-01).

Для того чтобы измененные настройки вступили в силу, необходимо перезагрузить конфигурацию на главном узле одним из следующих способов:

– с использованием утилиты `jadog_ctl`:

```
/usr/jatoba-6/bin/jadog_ctl -U admin -W '[password]' -q -C  
"reload jadog" -T 20000
```

– с использованием REST API с любого узла кластера:

```
curl -X POST --cacert /<dir>/root.crt --cert /<dir>/client.crt  
--key /<dir>/client.key -k  
https://host_name(ip):54443/cluster/[cluster_name]/node/[node_  
ame]/jadog/reload
```

или

```
curl -X POST --cacert /<dir>/root.crt --cert /<dir>/client.crt  
--key /<dir>/client.key -k  
https://host_name(ip):54443/cluster/[cluster_name]/node/[ip]/[p  
ort]/jadog/reload
```

где `<dir>` – каталог с сертификатами SSL.

– с использованием REST API с главного узла кластера:

```
curl -X POST --cacert /<dir>/root.crt --cert /<dir>/client.crt  
--key /<dir>/client.key -k https://host_name(ip):54443/reload
```

8.1.3. При подключении используется сертификат SSL не того пользователя или не действительный

Синтаксис запроса через REST:

```
curl --cacert /var/lib/jatoba/ssl/jatoba_ca.crt --cert  
/home/new_user/clusters/cert-gen/certs/users/new_user_pers.crt  
--key  
/home/new_user/clusters/certgen/certs/users/new_user_pers.key -  
k https://host_name(ip):54443/metrics
```

Где `<dir>` - путь к каталогу с сертификатами SSL.

В этом случае необходимо:

– проверить, что используется сертификат SSL нужного пользователя;

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

– проверить корректность сертификата и его внутреннюю информацию при помощи команды:

```
openssl x509 -in new_user_pers.crt -noout -text
```

8.2. Рассинхронизация кластера

Степень критичности возникшей рассинхронизации кластера зависит от имеющегося объема данных в СУБД. В зависимости от этого необходимо применять разные подходы к восстановлению штатной работы узлов кластера.

8.2.1. Малые и средние объемы данных СУБД

Симптом проблемы: кластер «развалился», нет возможности восстановить и узлы находятся в постоянном статусе «recovery».

Решение заключается в возможности разборки кластера и повторной его сборки, и зависит от объема данных СУБД. Приемлемым объемом для применения предлагаемого способа восстановления считается такой объем данных, который реплицируется на резервный узел за приемлемое время.

- 1) Ознакомиться с содержанием журналов работы СУБД «Jatoba» и компонента «jaDog».
- 2) Проконсультироваться с техподдержкой.
- 3) На резервных узла остановить компонент «jaDog» и СУБД «Jatoba»:

```
systemctl stop jadog | systemctl stop jatoba-6
```

- 4) Остановить службу компонент «jaDog» на главном узле:

```
systemctl stop jadog
```

- 5) Удалить файлы состояний компонента «jaDog» на каждом узле кластера:

```
rm -f /usr/jatoba-6/etc/jadog/jadog_state.yml
```

- 6) Отчистить каталог data только на резервных узлах:

```
rm -rf /var/lib/jatoba/6/data/*
```

7) Подключиться на главном узле кластера к утилите psql под правами superuser и удалить слоты репликации:

```
SELECT * FROM pg_replication_slots;  
SELECT pg_drop_replication_slot('название_слота_репликации');
```

8) Запустить компонент «jaDog» на всех узлах кластера:

```
systemctl start jadog
```

9) На главном узле авторизоваться в утилите jadog_ctl и собрать кластер с использованием инструкций из документа «Компонент jaDog. Управление режимом работы узлов кластера».



Для добавления в кластер узлов с методом репликации sync рекомендуется использовать команду с следующим синтаксисом:

```
cluster add sync slave [host_ip/host_name] [Jadog PORT  
number (port)]
```

8.2.2. Большие объемы данных СУБД

Описание: проблемы с реплицированием данных (на хватило дискового пространства, не запустился процесс восстановления).

Данный аварийно-восстановительный сценарий выполнять только под руководством специалистов технической поддержки.

1) Проконсультироваться с техподдержкой.
2) Снять нагрузку (входящие, исходящие подключения) с кластера;
3) На главном узле рекомендуется увеличить параметр max_wal_size = 16GB (до 32GB) на время аварийно-восстановительных работ. После изменения параметра выполнить:

```
select pg_reload_conf();
```

4) На резервных узлах остановить компонент «jaDog» и СУБД «Jatoba»:

```
systemctl stop jadog | systemctl stop jatoba-6
```

- 5) Отчистить каталог data только на резервных узлах:

```
rm -rf /var/lib/jatoba/6/data/*
```

- 6) На резервных узлах запустить компонент «jaDog»:

```
systemctl start jadog
```

7) На главном узле авторизоваться в утилите jadog_ctl и собрать кластер с использованием инструкций из документа «Компонент jaDog. Управление режимом работы узлов кластера».

8) Дождаться копирования БД с главного узла на резервные узлы. Завершение процесса контролировать с использованием информации из журналов работы компонента «jaDog»:

```
2025-05-16 21:39:45 Jadog 10.250.4.69():12345 INFO Script
output: pg_basebackup: начинается базовое резервное
копирование, ожидается завершение контрольной точки

...
2025-05-16 21:39:45 Jadog 10.250.4.69():12345 INFO Script
finished successfully: /usr/jatoba-
6/share/jadog/scripts/backup.sh /u01/postgres /usr/jatoba-6/bin
/usr/jatoba-6/bin/db_passfile postgres node1 "host=10.250.4.70
port=5433 user=jadog_user" DefaultCluster.10.250.4.69():12345
Jadog ver. 3.3.0

2025-05-16 21:39:45 Jadog 10.250.4.69():12345 INFO
Postgresql.conf: port = 5433 now set to new value = 5433
DefaultCluster.10.250.4.69():12345 Jadog ver. 3.3.0

2025-05-16 21:39:45 Jadog 10.250.4.69():12345 INFO
Postgresql.conf: cluster_name = 'node1' now set to new value =
'node1'
```

- 9) Остановить компонент «jaDog» на резервных узлах:

```
systemctl stop jadog
```

- 10) Аналогично остановить компонент «jaDog» на главном узле.

11) Заменить файл /usr/jatoba-6/etc/jadog/jadog_state.yml из приложенного на обоих узлах. В случае, если на резервных узлах данный файл отсутствует, просто добавить в указанный каталог.

12) На главном узле запустить компонент «jaDog»:

```
systemctl start jadog
```

13) Аналогично запустить компонент «jaDog» на резервных узлах.

14) На главном узле авторизоваться в утилите jadog_ctl и выполнить команду:

```
cluster activate
```

15) Подключить нагрузку (входящие, исходящие подключения) и проверить при помощи журнала событий работу кластера.

8.3. Некорректная перезагрузка кластера и/или СУБД

Не корректная перезагрузка кластера и/или СУБД, в большинстве случаев, заключается в не правильном порядке перезагрузки узлов кластера или СУБД. Следствием чего часто является ФО и нежелательная смена конфигурации (смена ролей узлов кластера).

1) В документации описан порядок перезагрузки узлов кластера. Основным принципом является то, что главный узел перезагружается последним. Утверждение справедливо как для СУБД, так и для компонента «jaDog».

2) Если СУБД функционирует в кластере под управлением компонента «jaDog», то рекомендуется воспользоваться командами, которые обязательно следует выполнять на главном узле):

— остановка СУБД кластера в правильной последовательности:

```
stop dbs on cluster
```

— запуск СУБД кластера в правильной последовательности:

```
start dbs on cluster
```

— перезагрузка СУБД кластера в правильной последовательности:

```
restart dbs on cluster
```

— проверка параметров СУБД (reload()) кластера в правильной последовательности:

```
reload dbs on cluster
```

— проверка всех компонента «jaDog» кластера в правильной последовательности:

```
reload jadog
```

3) Выполнять изменение параметров и перезагрузку (см. предыдущий пункт) в режиме технического обслуживания (maintenance):

```
set maintenance
```

4) Отключение режима технического обслуживания после завершения работ:

```
reset maintenance
```

8.4. Диагностика работоспособности кластера

В целях получения информации о состоянии работы кластера и/или его узлов в компоненте «jaDog» реализованы сервисные команды:

1) Статус кластера и входящих в него узлов.

Пример с использованием интерфейса утилиты jadog_ctl

```
cluster status
```

Пример с внешним использованием утилиты jadog_ctl

```
/usr/jatoba-6/bin/jadog_ctl -U admin -W '[password_admin]' -q -f json -T 30000 -C "cluster status"
```

Синтаксис с использованием jaDog REST API:

```
curl --cacert /<dir>/root.crt --cert /<dir>/client.crt --key  
/<dir>/client.key -k  
https://host_name(ip):54443/cluster/[cluster_name]
```

Где <dir> - каталог с сертификатами SSL.

Штатное значение параметра "State" для узлов: "Master(ACTIVE)", "Slave(ACTIVE)", "Referee(ACTIVE)".

2) Статус бандла:

```
bundle status
```

3) Статус отдельного узла:

```
node is  
node is {'checked_role'}  
node is {'checked_status'}
```

Синтаксис с использованием jaDog REST API:

```
curl --cacert /<dir>/root.crt --cert /<dir>/client.crt --key  
/<dir>/client.key -k https://host_name(ip):54443/node_is
```

Где <dir> - каталог с сертификатами SSL.

4) Значение встроенных метрик компонента:

Синтаксис с использованием jaDog REST API:

```
curl --cacert /<dir>/root.crt --cert /<dir>/client.crt --key  
/<dir>/client.key -k https://host_name(ip):54443/metrics
```

Где <dir> - каталог с сертификатами SSL.

ПРИЛОЖЕНИЕ 1

Расположение служебных файлов, скриптов и шаблонов jaDog

Таблица П.1.1 – Расположение служебных файлов, скриптов и шаблонов компонента «jaDog» для ОС Linux

Описание	Значение
Журналы	
Журналы работы компонента: /usr/jatoba-6/var/log	
Конфигурация кластера и узлов	
Основные конфигурационные файлы	/usr/jatoba-6/etc/jadog
bundle_state.yml	Состояние бандла ja_hipe_bundle (citus)
jadog.yml	Параметры компонента «jaDog»
jadog_hba.cfg	Правила доступа к компоненту «jaDog»
jadog_state.yml	Состояние кластера и его узлов
users.yml	Перечень УЗ компонента «jaDog»
Сертификаты	
Расположение сертификатов TLS (SSL):	/var/lib/jatoba/ssl (рекомендуется)
Шаблоны файлов ответов установки компонента «jaDog» и формирования кластеров	
Расположение шаблонов	/usr/jatoba-6/share/doc/jadog/clusters_kits
jadog_setup_faq.yml.in	Настройка одного узла (сокращенная)
jadog_setup_faq_full.yml.in	Настройка одного узла (расширенная)
jadog_setup_faq_win.yml.in	Настройка одного узла (сокращенная, Windows)
jadog_setup_faq_full_win.yml.in	Настройка одного узла (расширенная, Windows)
init_ja_dtc_as.yml.in	Развертывание географически распределенного кластера
init_ja_hipe_cluster.yml.in	Развертывание высокопроизводительного кластера с группой кластеров
init_ja_hipe_cluster_fqdn_ssl.yml.in	Развертывание высокопроизводительного кластера с FQDN и SSL, и с группой кластеров
init_jadog_referee.yml.in	Развертывание отказоустойчивого кластера с узлом-арбитром
init_jadog_referee_wa.yml.in	Развертывание отказоустойчивого кластера с узлом-арбитром с архивом WAL
init_jadog_trusted_ip.yml.in	Развертывание отказоустойчивого кластера с trusted address
init_jadog_trusted_ip_ssl.yml.in	Развертывание отказоустойчивого кластера с trusted address и SSL

Значения по умолчанию, используемые в компоненте «jaDog»

- база данных: postgres;
- учетные записи:
 - admin – УЗ администратора «jaDog»;
 - jadog_user – УЗ для репликации данных;
 - Пароли по умолчанию: admin, change_on_install.

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

– TCP порты:

- 12345 – компонент «jaDog»;
- 54321 – интерфейс при подключении в jadog_ctl;
- 64321 – служебный режим jadog0;
- 54443 – REST API.

ПРИЛОЖЕНИЕ 2

Работа с консольной утилитой **jadog_ctl** из командной строки

Для того чтобы получить быстрый ответ от компонента «jaDog» команды следует выполнять с ключом «-C»:

```
/usr/jatoba-6/bin/jadog_ctl -U ivanoff -W 'change_on_install' -q -C "\h"
```

Длительное ожидание выполнения консольной утилиты **jadog_ctl**

Некоторые команды компонента «jaDog», например, switchover или формирование кластеров и бандлов, требуют длительного ожидания результата.

Для того чтобы дождаться окончания выполнения команды и получения результата необходимо воспользоваться ключом «-T» с указанием времени ожидания.

```
/usr/jatoba-6/bin/jadog_ctl -U ivanoff -W 'change_on_install' -q -f json -C "cluster status" -T 30000
```

ПРИЛОЖЕНИЕ 3

Ключи запуска консольной утилиты `jadog_ctl`

При управлении кластером при помощи консольной утилиты `jadog_ctl` применяются следующие ключи:

```
/usr/jatoba-6/bin/jadog_ctl --help
```

```
Usage: jadog_ctl [OPTION]...
```

```
Jadog_Ctl - Jadog console utility.
```

List of OPTIONS:

```
-h, --host=host Jadog master node to connect
```

```
-p, --port=port Port to connect to jadog node
```

```
-U, --username=username Username to connect to jadog node
```

```
-W, --password=password password to connect to jadog node
```

```
-C, --command=command command to fast execute and exit
```

```
-q, --quiet command execution without explanatory messages
```

```
-f, --format output message format: table (default) or json
```

```
-T, --timeout=timeout timeout(in milliseconds) to wait  
execution command with -C
```

```
--ssl ssl connection enabled
```

```
--sslcafile ssl CA file
```

```
--sslcertfile ssl certificate file
```

```
--sslcrlfile ssl obsolete credential list file
```

```
--sslkeyfile ssl private key file
```

List of COMMANDS:

```
create_cluster Installed jadog and create clusters
```

```
cluster_update Update existing cluster structure
```

ПРИЛОЖЕНИЕ 4

Команды консольной утилиты **jadog_ctl**

При управлении кластером при помощи консольной утилиты **jadog_ctl** применяются следующие команды:

```
/usr/jatoba-6/bin/jadog_ctl -U admin -W 'admin' -q -C "\h"
Reply: Supported commands:
Commands for jaDog:
\h - this help.
\password - change current user password.
\q - terminate console.
protocol version - show jadog protocol version.
clear - clear console.
connect {ip} {port} {'login'} - connect to jadog node.
disconnect - disconnect from jadog node.
reload jadog - reload the configuration parameters of this
jadog if it is located in the Master-node or it is not in the
cluster.

Commands for work with parameters of jadog:
set parameter param_name = param_value - single parameter
setting.
set parameters - setting a set of parameters in a transaction.
show parameter param_name - show parameter value.
save - save parameter set transaction.
cancel - cancel parameter set transaction.

Commands for working with scheduled jobs:
schedule - show the list of scheduled jobs.
schedule completed - show the list of result completed
scheduled jobs.
schedule set [job_name] "{job_time}" "{job_command}" - set the
schedule job by timestamp.
```

```
example: schedule set JobSwitchover "2025-05-28
14:21:52+00:00" "switchover 'node1'".
```

```
schedule set [job_name] {job_delay_interval} "{job_command}" -
set the schedule job by delay interval.
```

```
example: schedule set testJobMaintenance 15m "set
maintenance".
```

```
schedule reset {job_name} - Remove job from schedule.
```

Commands for work over API:

```
set async mode - setup asynchronous command processing.
```

```
set sync mode - setup synchronous command processing.
```

```
get last response - print last message from server excluding
status.
```

```
get result id - print last message from server by the given id
excluding status.
```

Commands for bundle:

```
bundle create {'bundle_name'} - create new cluster bundle.
```

```
bundle delete {'bundle_name'} - delete empty bundle.
```

```
bundle attach cluster {ip} {port} - attach cluster to bundle
with specified ip, port.
```

```
bundle attach cluster {ip} {port} {'interconnect_user'} -
attach cluster to bundle with specified ip, port,
interconnect_user.
```

```
bundle detach cluster {ip} {port} - detach cluster from bundle
with specified ip, port.
```

```
bundle detach cluster {cluster_name} - detach cluster from
bundle with specified cluster name.
```

```
bundle status - show list of clusters in bundle.
```

Commands for cluster:

```
add cluster {'cluster_name'} - create new cluster.
```

```
delete cluster - delete empty cluster, reverse command to add
cluster.
```

```
cluster add master {ip} {port} as {'node_name'} - add jadog
with specified ip, port and node name as master.

cluster add master {ip} {port} - add jadog with specified ip
and port as master.

cluster add cascade slave [nodata] {ip} {port} primary
{primary_ip} {primary_port} as {'node_name'} - add jadog with
specified ip, port and node name as cascade slave with primary.
[nodata] - without backup.

cluster add cascade slave [nodata] {ip} {port} primary
{primary_ip} {primary_port} - add jadog with specified ip and
port as cascade slave with primary. [nodata] - without backup.

cluster add cascade slave [nodata] {ip} {port} primary
{'primary_node_name'} as {'node_name'} - add jadog with
specified ip, port and node name as cascade slave with primary.
[nodata] - without backup.

cluster add cascade slave [nodata] {ip} {port} primary
{'primary_node_name'} - add jadog with specified ip and port as
cascade slave with primary. [nodata] - without backup.

cluster add slave [nodata] {ip} {port} as {'node_name'} - add
jadog with specified ip, port and node name as slave. [nodata]
- without backup.

cluster add slave [nodata] {ip} {port} - add jadog with
specified ip and port as slave. [nodata] - without backup.

cluster add sync slave [nodata] {ip} {port} as {'node_name'} -
add jadog with specified ip, port and node name as sync slave.
[nodata] - without backup.

cluster add sync slave [nodata] {ip} {port} - add jadog with
specified ip, port as sync slave. [nodata] - without backup.

cluster add async slave [nodata] {ip} {port} as {'node_name'} -
add jadog with specified ip, port and node name as async slave.
[nodata] - without backup.

cluster add async slave [nodata] {ip} {port} - add jadog with
specified ip, port as async slave. [nodata] - without backup.

cluster add referee {ip} {port} as {'node_name'} - add jadog
with specified ip, port and node name as referee.

cluster add referee {ip} {port} - add jadog with specified ip
and port as referee.

cluster delete node {'node_name'} - delete jadog with specified
node name from cluster.
```

cluster delete node {ip} {port} - delete jadog with specified ip from cluster.

cluster status - show list of nodes in cluster.

cluster activate - provide external access to cluster by setting up public address.

cluster deactivate - forbid external access to cluster by removing public address.

reload jadog - reload jadog on master.

reload jadog {ip} {port} - reload jadog on node with ip and port.

reload jadog {'node_name'} - reload jadog on node with name.

reload jadog on cluster - reload jadog on all active nodes.

set maintenance - set the maintenance mode on the cluster.

reset maintenance - reset the maintenance mode on the cluster. All nodes will switch to recovery mode.

set maintenance on node {'node_name'} - set the maintenance mode on the node with node_name and its slave nodes.

reset maintenance on node {'node_name'} - reset the maintenance mode on the node with node_name. The node will switch to recovery mode.

set maintenance on node {ip} {port} - set the maintenance mode on the node with ip and port and its slave nodes.

reset maintenance on node {ip} {port} - reset the maintenance mode on the node with ip and port. The node will switch to recovery mode.

cluster_update {'yaml_file_path'} - updating the structure of an existing cluster using yaml-configuration.

cluster get structure {'short'} - getting a short version of the yaml template of the current cluster.

cluster get structure {'full'} - getting a full version of the yaml template of the current cluster.

cluster reinit node {'node_name'} - reinitialize jadog with specified node name to the cluster.

cluster reinit node nodata {'node_name'} - reinitialize jadog with specified node name to the cluster.

cluster reinit node {ip} {port} - reinitialize jadog with specified ip to the cluster.

cluster reinit node nodata {ip} {port} - reinitialize jadog with specified ip to the cluster.

node reinit - reinitialize jadog to the cluster.

node reinit nodata - reinitialize jadog to the cluster.

Commands for datacenters:

datacenter create {'dc_name'} - create new data center.

datacenter delete {'dc_name'} - delete data center.

datacenter {'dc_name'} attach node {'node_name'} - attach node to data center with specified node name from cluster.

datacenter {'dc_name'} attach node {ip} {port} - attach node to data center with specified ip from cluster.

datacenter {'dc_name'} detach node {'node_name'} - detach node from data center with specified node name from cluster.

datacenter {'dc_name'} detach node {ip} {port} - detach node from data center with specified ip from cluster.

datacenter promote - forced switchover to the current datacenter.

Commands for work with accounts of jadog:

create user {'user_name'} - create account in cluster with password in console.

create user {'user_name'} with password {'password'} - create account with password.

drop user {'user_name'} - drop account.

alter user {'user_name'} with password {'password'} - change account password.

alter user {'user_name'} rename to {'new name'} - rename account.

alter user {'user_name'} account lock - block account.

alter user {'user_name'} account unlock - unlock account.

show users - show accounts list.

Commands for work with nodes:


```
alter node {'old_node_name'} rename to {'new_node_name'} -
rename jadog node.

alter node {ip} {port} rename to {'new_node_name'} - rename
jadog node with specified ip from cluster.

alter node {'node_name'} set replication type =
{'replication_type'} - set replication type (sync, async).

alter node {ip} {port} set replication type =
{'replication_type'} - set replication type (sync, async).

alter node {ip} {port} set primary = {'primary_node'} - set
primary node with node name.

alter node {ip} {port} set primary {primary_ip} {primary_port}
- set primary node with ip and port.

alter node {'node_name'} set primary = {'primary_node'} - set
primary node with node name.

alter node {'node_name'} set primary {primary_ip}
{primary_port} - set primary node with ip and port.

node show - show node info with current ip from cluster.

node show {ip} {port} - show node info with specified ip from
cluster.

node show {'node_name'} - show node info with node_name from
cluster.

node show {'cluster_name'} {ip} {port} - show node info with
specified ip from cluster.

switchover {ip} {port} - switch slave jadog with specified ip
to master.

switchover {'node_name'} - switch slave jadog with specified
node name to master.

clusters list - show list of cluster names.

node ls - show list of node names.

node {'cluster_name'} - show list of node names.

node {'cluster_name'} ls - show list of node names.

node set node_name = {'node_name'} - rename jadog master node.

node set node_name {'cluster_name'} {ip} {port} = {'node_name'}
- confirm action rename jadog master node.

node is - show current node info about role and state in the
cluster.
```

node is {'checked_role'} - check current node role in the cluster.

node is {'checked_status'} - check current node replication type or maintenance mode.

Commands for work with RDBMS:

stop dbs on cluster - stop database on all cluster.

stop dbs on node - stop database on master.

stop dbs on node {ip} {port} - stop database on node with ip and port.

stop dbs on node {'node_name'} - stop database on node with node name.

start dbs on cluster - start database on all cluster.

start dbs on node - start database on master.

start dbs on node {ip} {port} - start database on node with ip and port.

start dbs on node {'node_name'} - start database on node with node name.

restart dbs on cluster - restart database on all cluster.

restart dbs on node - restart database on master.

restart dbs on node {ip} {port} - restart database on node with ip and port.

restart dbs on node {'node_name'} - restart database on node with node name.

reload dbs on cluster - reload database on all cluster.

reload dbs on node - reload database on master.

reload dbs on node {ip} {port} - reload database on jadog with ip and port.

reload dbs on node {'node_name'} - reload database on jadog with node name.

Commands for work with files:

fm - Show list of sync files.

fm delete - Removing all paths from the sync list.

fm delete {file id} - Removing path with id from the sync list.

fm set {'file path'} - Add path with file path to the sync list;.

fm copy - Copy files to cluster nodes, without referee.

fm copy --referee - Copy files to cluster nodes, with referee.

fm copy -n {node name array} - Copy files to specified nodes.

fm copy {file id} - Copy file with file id to cluster nodes, without referee.

fm copy {file id} --referee - Copy file with file id to cluster nodes, with referee.

fm copy {file id} -n {node name array} - Copy file with file id to specified nodes.

Supported shortcuts:

<TAB> - autocomplete of the current command.

Double <TAB> - show autocomplete candidates for the current input.

ПРИЛОЖЕНИЕ 5

Шаблон файла ответов автоматизированной настройки устойчивой конфигурации кластера без хранилища WAL-файлов

```
# Файл ответов для автоматического формирования кластера jaDog.

# Значения не закоментированные - обязательные. Рекомендуется установить
собственные значения.

# Значения закоментированные - будут использоваться значения по-умолчанию.
Их можно раскомментировать и установить собственные значения.

# Любые другие дополнительные параметры, которых нет в шаблоне, возможно
размещать в следующих группах:

# - cluster_settings.cluster_name.default_node_params (будет применено к
каждой(!) ноде кластера, включая мастер) или

# - cluster_settings.cluster_name.datacenters.nodes.node_name (будет
применено к конкретной ноде)

# Добавляя параметры необходимо соблюдать нотацию конфигурационного файла
jadog.yml (Обычно он тут: /usr/jatoba-5/etc/jadog/jadog.yml )

apiVersion: jaDog v4.0 # Версия jaDog

kind: jadog_simple # Назначение скрипта

default_cluster_params: # Блок атрибутов
используемый для каждого кластера в бандле.

  db_init_conn_string: host=127.0.0.1 port=5432 user=postgres
dbname=postgres password='change_on_install' # Строка коннекта к СУБД для
установки расширений и выполнения скриптов. Требуется SU прав.

  initdb: # Блок инициализации СУБД.
Содержит параметры инициализации СУБД

  initdb_options: "--locale=ru_RU.utf8 --encoding=UTF-8" # Строка
параметров инициализации СУБД Jatoba.

  postgresql.conf: # Параметры будут
установлены при формировании кластера в конец файла postgresql.conf.

    listen_addresses: "*"

    log_destination: "stderr"

    logging_collector: on

    log_directory: "log"

    log_filename: "jatoba-%Y-%m-%d_%H%M%S.log"

    log_rotation_age: 1d

    log_rotation_size: 0

    log_truncate_on_rotation: off

    log_line_prefix: "%m [%p] "

    log_statement: all
```

```
shared_preload_libraries: "'pg_stat_statements'"
max_connections: 128
max_prepared_transactions: 256
data_sync_retry: off

pg_hba.conf:                                     # Параметры будут
установлены при формировании кластера в файл pg_hba.conf

- local all postgres scram-sha-256
- local jatoba jalog_user scram-sha-256
- host all postgres 192.168.72.0/24 trust
- host all postgres 127.0.0.1/32 scram-sha-256
- host jatoba jalog_user 127.0.0.1/32 scram-sha-256
- host replication jalog_user 127.0.0.1/32 scram-sha-256
- host replication jalog_user 192.168.72.0/24 scram-sha-256

# sql_scripts:                                   # Раздел пользовательских
скриптов, выполняемых на мастере каждого создаваемого кластера. Скрипты
определяются администратором кластера.

# - /home/<user>/clusters/ja_hipe_cluster/cluster_default_script.sql
# Пользовательский SQL скрипт выполняемый с аутентификационными данными
cluster_settings.db_init_conn_string после создания всех расширений.

cluster_settings:

- cluster_name: my_cluster                       # Наименование кластера.
Обязательный параметр. Кластер может быть один или несколько.

cluster_master_node: node_11                     # Нода, являющаяся Мастером.
Обязательный параметр.

# cluster_referee_nodes:                         # Блок узлов, являющимися
referee. Их может быть несколько.

# - node_referee                                № Имя ноды, выступающей в
роли рефери.

activated: true                                   # Флаг активации кластера после
формирования. true - активировать (назначать public_address), false - не
активировать.

jalog_users:                                     # Учетные записи для
администрирования jaDog. Создаются УЗ и запись в jalog_hba.cfg.

- name: admin                                    # Имя учетной записи.
Обязательный параметр.

pass: change_on_install                          # Пароль учетной записи.
Обязательный параметр. Для обеспечения безопасности смените пароль УЗ после
установки.

address: all                                     # Разрешенные адреса или
подсети для доступа к jaDog. Значения: 'all', имя хоста( 'myhost.local.net'
) или IP адрес с маской ('192.168.100.1/24').
```

```

method: sha-256 # Метод доступа. Значения:
"sha-256", "ssl". "sha-256" по умолчанию.

- name: interdog # УЗ можно создать несколько.
Для обеспечения безопасности смените пароль УЗ после установки.

pass: change_on_install # Для обеспечения безопасности
смените пароль УЗ после установки.

address: all

default_node_params: # Параметры по-умолчанию,
которые будут использовать для каждой ноды. Записи включаются в нотации
конфигурационного файла jadog.yml.

param_jadog: # Параметры конфигурирования
экземпляра jaDog.

# service_name: jadog # Имя сервиса jaDog для ноды.
"jadog" по умолчанию

public_address: 192.168.72.10 # Виртуальный IP адрес
кластера. Public address. Обязательный параметр.

trusted_address: 192.168.72.2 # Доверенный IP адрес. Trusted
IP address. Обязательный параметр для кластеров с одним slave.

network_interface: ens33 # Наименование сетевого
интерфейса для виртуального IP кластера. Network interface name.
Обязательный параметр.

interconnect_user: interdog # Имя технологической учетной
записи для взаимодействия с другими jaDog сервисами. Обязательный
параметр.

param_log: # Параметры логирования

logs_filename: jadog-%Y-%m-%d_%N%M%S # Шаблон для файлов
функциональных логов

logs_screen: true # Включение вывода логов на
экран. "false" по умолчанию

logs_type: csv, security.csv # Форматы файлов
функционального и логов событий ИБ. На первом месте формат функционального
файла логов (csv, json) на втором формат файла логов событий ИБ
(security.json, security.csv)

logs_level: debug2 # Уровень функционального
логирования

param_security_log: # Параметры журнала событий
информационной безопасности (ИБ)

security_logs_filename: jadog-%Y-%m-%d_%N%M%S # Шаблон для файлов
журнал событий ИБ

param_postgres: # Блок параметров для работы с
СУБД

db_service_name: jatoba-6 # Название сервиса экземпляра
СУБД в операционной системе

```

```

db_connection_settings:                                # Параметры для соединения с
СУБД

    database: jatoba                                    # Имя базы данных для установки
расширения jaDog. "postgres" по умолчанию

    passfile: /var/lib/jatoba/.pgpass                  # Jatoba passfile

    user: jadog_user                                    # Имя УЗ для работы с СУБД.
Обязательный параметр.

    user_pass: change_on_install                        # Пароль УЗ для работы с СУБД.
Обязательный параметр. Сменить пароль УЗ в СУБД после формирования
кластера.

    param_rest_api:                                     # Блок параметров для REST-
сервера

        rest_api_use: true                             # Признак использования
REST-сервера

        rest_api_cert_file: /var/lib/jatoba/ssl/jadog_cert/jadog_server.crt
# Путь до файла сертификата REST-сервера

        rest_api_key_file: /var/lib/jatoba/ssl/jadog_cert/jadog_server.key
# Путь до файла ключа REST-сервера

        rest_api_ca_file: /var/lib/jatoba/ssl/jatoba_ca.crt
# Путь до файла CA REST-сервера

#         rest_api_crl_file: /var/lib/jatoba/ssl/jadog_cert/ca-cert.crt
# Путь до файла CRL REST-сервера

        rest_api_listen_address: 0.0.0.0               # Прослушиваемый адрес
REST-сервера

        rest_api_listen_port: 54443                    # Порт REST-сервера

datacenters: #

    - datacenter: DC_Piter                             # Уникальное наименование
датацентра. Датацентр может быть один или несколько.

    nodes:                                              # Блок описания нод кластера

        - node_name: node_11                          # Кластерное имя ноды. Все
записи конфигурирования включаются в нотации конфигурационного файла
jadog.yml.

        param_jadog:                                    # Параметры конфигурирования
экземпляра jaDog.

            ip: 192.168.72.11                           # IP адрес ноды

#             network_interface: ens33                  # Наименование сетевого
интерфейса для виртуального IP кластера. Network interface name.
Обязательный параметр.

#             param_synchronous:                        # Раскомментировать блок
параметров, если потребуется синхронная реплика

#             synchronous_commit: on                    # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off

```

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

```

#               synchronous_type: ANY           # Как будет работать синхронная
репликация по принципам: FIRST или ANY

#               synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита

                param_replication:              # Параметры репликации ноды

                replication_slot_name: rs_node_11 # Уникальное в рамках
кластера имя слота репликации.
- node_name: node_12
                param_jadog:
                ip: 192.168.72.12
                param_replication:
                replication_slot_name: rs_node_12

#               param_synchronous:              # Раскомментировать блок
параметров, если потребуется синхронная реплика

#               synchronous_commit: on          # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off

#               synchronous_type: ANY           # Как будет работать синхронная
репликация по принципам: FIRST или ANY

#               synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита

- node_name: node_13
                param_jadog:
                ip: 192.168.72.13
                param_replication:
                replication_slot_name: rs_node_13

#               param_synchronous:              # Раскомментировать блок
параметров, если потребуется синхронная реплика

#               synchronous_commit: on          # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off

#               synchronous_type: ANY           # Как будет работать синхронная
репликация по принципам: FIRST или ANY

#               synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита

#               - node_name: node_referee      # Заготовка для новой ноды в
кластере, например, с ролью referee

#               param_jadog:
#               ip: 192.168.72.23
- node_name: node_referee

```



```
param_jadog:
    ip: 192.168.72.14
param_replication:
    replication_slot_name: rs_node_referee
#           param_synchronous:           # Раскомментировать блок
параметров, если потребуется синхронная реплика
#           synchronous_commit: on       # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off
#           synchronous_type: ANY        # Как будет работать синхронная
репликация по принципам: FIRST или ANY
#           synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита
```

ПРИЛОЖЕНИЕ 6

Шаблон файла ответов автоматизированной настройки устойчивой конфигурации кластера с хранилищем WAL-файлов

```
# Файл ответов для автоматического формирования кластера jaDog.
# Значения не закомментированные - обязательные. Рекомендуется установить
# собственные значения.
# Значения закомментированные - будут использоваться значения по-умолчанию.
# Их можно раскомментировать и установить собственные значения.
# Любые другие дополнительные параметры, которых нет в шаблоне, возможно
# размещать в следующих группах:
# - cluster_settings.cluster_name.default_node_params (будет применено к
# каждой(!) ноде кластера, включая мастер) или
# - cluster_settings.cluster_name.datacenters.nodes.node_name (будет
# применено к конкретной ноде)
# Добавляя параметры необходимо соблюдать нотацию конфигурационного файла
# jadog.yml (Обычно он тут: /usr/jatoba-5/etc/jadog/jadog.yml )
apiVersion: jaDog v4.0 # Версия jaDog
kind: jadog_referee # Назначение скрипта
default_cluster_params: # Блок атрибутов
используемый для каждого кластера в бандле.
  db_init_conn_string: host=127.0.0.1 port=5432 user=postgres
  dbname=postgres password='change_on_install' # Строка коннекта к СУБД для
  установки расширений и выполнения скриптов. Требуется SU прав.
  initdb: # Блок инициализации СУБД.
  Содержит параметры инициализации СУБД
  initdb_options: "--locale=ru_RU.utf8 --encoding=UTF-8" # Строка
  параметров инициализации СУБД Jatoba.
  postgresql.conf: # Параметры будут
  установлены при формировании кластера в конец файла postgresql.conf.
  listen_addresses: '*'
  log_destination: 'stderr'
  logging_collector: on
  log_directory: 'log'
  log_filename: 'jatoba-%Y-%m-%d_%H%M%S.log'
  log_rotation_age: 1d
  log_rotation_size: 0
  log_truncate_on_rotation: off
  log_line_prefix: '%m [%p] '
  log_statement: all
```

```

shared_preload_libraries: "'pg_stat_statements'"
max_connections: 128
max_prepared_transactions: 256
data_sync_retry: off

pg_hba.conf:                                     # Параметры будут
установлены при формировании кластера в файл pg_hba.conf

- local all postgres scram-sha-256
- local jatoba jalog_user scram-sha-256
- host all postgres 192.168.72.0/24 trust
- host all postgres 127.0.0.1/32 scram-sha-256
- host jatoba jalog_user 127.0.0.1/32 scram-sha-256
- host replication jalog_user 127.0.0.1/32 scram-sha-256
- host replication jalog_user 192.168.72.0/24 scram-sha-256

# sql_scripts:                                   # Раздел пользовательских
скриптов, выполняемых на мастере каждого создаваемого кластера. Скрипты
определяются администратором кластера.

# - /home/<user>/clusters/ja_hipe_cluster/cluster_default_script.sql
# Пользовательский SQL скрипт выполняемый с аутентификационными данными
cluster_settings.db_init_conn_string после создания всех расширений.

cluster_settings:

- cluster_name: my_cluster                       # Наименование кластера.
Обязательный параметр. Кластер может быть один или несколько.

cluster_master_node: node_11                     # Нода, являющаяся Мастером.
Обязательный параметр.

cluster_referee_nodes:                           # Блок узлов, являющимися
referee. Их может быть несколько.

- node_referee                                   # Имя ноды, выступающей в роли
рефери.

activated: true                                  # Флаг активации кластера после
формирования. true - активировать (назначать public_address), false - не
активировать.

jalog_users:                                     # Учетные записи для
администрирования jaDog. Создаются УЗ и запись в jalog_hba.cfg.

- name: admin                                    # Имя учетной записи.
Обязательный параметр.

pass: change_on_install                          # Пароль учетной записи.
Обязательный параметр. Для обеспечения безопасности смените пароль УЗ после
установки.

address: all                                     # Разрешенные адреса или
подсети для доступа к jaDog. Значения: 'all', имя хоста( 'myhost.local.net'
) или IP адрес с маской ('192.168.100.1/24').

```

```

method: sha-256 # Метод доступа. Значения:
"sha-256", "ssl". "sha-256" по умолчанию.

- name: interdog # УЗ можно создать несколько.
Для обеспечения безопасности смените пароль УЗ после установки.

pass: change_on_install # Для обеспечения безопасности
смените пароль УЗ после установки.

address: all

default_node_params: # Параметры по-умолчанию,
которые будут использовать для каждой ноды. Записи включаются в нотации
конфигурационного файла jadog.yml.

param_jadog: # Параметры конфигурирования
экземпляра jaDog.

# service_name: jadog # Имя сервиса jaDog для ноды.
"jadog" по умолчанию

public_address: 192.168.72.10 # Виртуальный IP адрес
кластера. Public address. Обязательный параметр.

# trusted_address: 192.168.72.2 # Доверенный IP адрес. !!!Если
кластер собирается с referee, параметр не устанавливается!!!

network_interface: ens33 # Наименование сетевого
интерфейса для виртуального IP кластера. Network interface name.
Обязательный параметр.

interconnect_user: interdog # Имя технологической учетной
записи для взаимодействия с другими jaDog сервисами. Обязательный
параметр.

param_log: # Параметры логирования

logs_filename: jadog-%Y-%m-%d_%H%M%S # Шаблон для файлов
функциональных логов

logs_screen: true # Включение вывода логов на
экран. "false" по умолчанию

logs_type: csv, security.csv # Форматы файлов
функционального и логов событий ИБ. На первом месте формат функционального
файла логов (csv, json) на втором формат файла логов событий ИБ
(security.json, security.csv)

logs_level: debug2 # Уровень функционального
логирования

param_security_log: # Параметры журнала событий
информационной безопасности (ИБ)

security_logs_filename: jadog-%Y-%m-%d_%H%M%S # Шаблон для файлов
журнал событий ИБ

param_postgres: # Блок параметров для работы с
СУБД

db_service_name: jatoba-6 # Название сервиса экземпляра
СУБД в операционной системе

```

```

db_connection_settings: # Параметры для соединения с
СУБД

    database: jatoba # Имя базы данных для установки
расширения jaDog. "postgres" по умолчанию

    passfile: /var/lib/jatoba/.pgpass # Jatoba passfile

    user: jadog_user # Имя УЗ для работы с СУБД.
Обязательный параметр.

    user_pass: change_on_install # Пароль УЗ для работы с СУБД.
Обязательный параметр. Сменить пароль УЗ в СУБД после формирования
кластера.

    param_rest_api: # Блок параметров для REST-
сервера

        rest_api_use: true # Признак использования
REST-сервера

        rest_api_cert_file: /var/lib/jatoba/ssl/jadog_cert/jadog_server.crt
# Путь до файла сертификата REST-сервера

        rest_api_key_file: /var/lib/jatoba/ssl/jadog_cert/jadog_server.key
# Путь до файла ключа REST-сервера

        rest_api_ca_file: /var/lib/jatoba/ssl/jatoba_ca.crt
# Путь до файла CA REST-сервера

#        rest_api_crl_file: /var/lib/jatoba/ssl/jadog_cert/ca-cert.crt
# Путь до файла CRL REST-сервера

        rest_api_listen_address: 0.0.0.0 # Прослушиваемый адрес
REST-сервера

        rest_api_listen_port: 54443 # Порт REST-сервера

#Параметры архивации

    param_archive:

        wal_archive_directory: "/mnt/wals/" # Путь до
сохранения архивов

        wal_archive_command: "cp %p /mnt/wals/%f" # Команды
архивирования

        wal_archive_cleanup_command: "/usr/jatoba-6/bin/pg_archivecleanup
/mnt/wals/ %s" # Команды очистки

        wal_archive_cleanup_needed: true # Признак
необходимости очистки

        wal_archive_cleanup_timeout: 500000 #
Временной промежуток очистки(ms)

#Параметры восстановления

    param_restore:

        wal_restore_command: "cp /mnt/wals/%f %p" #
Команда восстановления

    datacenters: #

```

```
- datacenter: DC_Piter                # Уникальное наименование
датацентра. Датацентр может быть один или несколько.

    nodes:                            # Блок описания нод кластера

        - node_name: node_11          # Кластерное имя ноды. Все
записи конфигурирования включаются в нотации конфигурационного файла
jadog.yml.

            param_jadog:              # Параметры конфигурирования
экземпляра jaDog.

                ip: 192.168.72.11      # IP адрес ноды

#            network_interface: ens33  # Наименование сетевого
интерфейса для виртуального IP кластера. Network interface name.
Обязательный параметр.

#            param_synchronous:       # Раскомментировать блок
параметров, если потребуется синхронная реплика

#            synchronous_commit: on   # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off

#            synchronous_type: ANY    # Как будет работать синхронная
репликация по принципам: FIRST или ANY

#            synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита

            param_replication:        # Параметры репликации ноды

                replication_slot_name: rs_node_11 # Уникальное в рамках
кластера имя слота репликации.

        - node_name: node_12

            param_jadog:

                ip: 192.168.72.12

            param_replication:

                replication_slot_name: rs_node_12

#            param_synchronous:       # Раскомментировать блок
параметров, если потребуется синхронная реплика

#            synchronous_commit: on   # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off

#            synchronous_type: ANY    # Как будет работать синхронная
репликация по принципам: FIRST или ANY

#            synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита

        - node_name: node_13

            param_jadog:

                ip: 192.168.72.13
```

```
param_replication:
    replication_slot_name: rs_node_13
#      param_synchronous:          # Раскомментировать блок
параметров, если потребуется синхронная реплика
#      synchronous_commit: on      # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off
#      synchronous_type: ANY       # Как будет работать синхронная
репликация по принципам: FIRST или ANY
#      synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита
```

ПРИЛОЖЕНИЕ 7

Шаблон файла ответов автоматизированной настройки устойчивой конфигурации кластера с использованием Trusted Address

```
# Файл ответов для автоматического формирования кластера jaDog.
# Значения не закомментированные - обязательные. Рекомендуется установить
# собственные значения.
# Значения закомментированные - будут использоваться значения по-умолчанию.
# Их можно раскомментировать и установить собственные значения.
# Любые другие дополнительные параметры, которых нет в шаблоне, возможно
# размещать в следующих группах:
# - cluster_settings.cluster_name.default_node_params (будет применено к
# каждой(!) ноде кластера, включая мастер) или
# - cluster_settings.cluster_name.datacenters.nodes.node_name (будет
# применено к конкретной ноде)
# Добавляя параметры необходимо соблюдать нотацию конфигурационного файла
# jadog.yml (обычно он тут: /usr/jatoba-5/etc/jadog/jadog.yml )
apiVersion: jaDog v4.0 # Версия jaDog
kind: jadog_simple # Назначение скрипта
default_cluster_params: # Блок атрибутов
используемый для каждого кластера в бандле.
  db_init_conn_string: host=127.0.0.1 port=5432 user=postgres
  dbname=postgres password='change_on_install' # Строка коннекта к СУБД для
  установки расширений и выполнения скриптов. Требуется SU прав.
  initdb: # Блок инициализации СУБД.
  Содержит параметры инициализации СУБД
  initdb_options: "--locale=ru_RU.utf8 --encoding=UTF-8" # Строка
  параметров инициализации СУБД Jatoba.
  postgresql.conf: # Параметры будут
  установлены при формировании кластера в конец файла postgresql.conf.
  listen_addresses: "*"
  log_destination: "stderr"
  logging_collector: on
  log_directory: "log"
  log_filename: "jatoba-%Y-%m-%d_%H%M%S.log"
  log_rotation_age: 1d
  log_rotation_size: 0
  log_truncate_on_rotation: off
  log_line_prefix: "%m [%p] "
  log_statement: all
```

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


```

shared_preload_libraries: "'pg_stat_statements'"
max_connections: 128
max_prepared_transactions: 256
data_sync_retry: off

pg_hba.conf:                                     # Параметры будут
установлены при формировании кластера в файл pg_hba.conf

- local all postgres scram-sha-256
- local jatoba jalog_user scram-sha-256
- host all postgres 192.168.72.0/24 trust
- host all postgres 127.0.0.1/32 scram-sha-256
- host jatoba jalog_user 127.0.0.1/32 scram-sha-256
- host replication jalog_user 127.0.0.1/32 scram-sha-256
- host replication jalog_user 192.168.72.0/24 scram-sha-256

# sql_scripts:                                   # Раздел пользовательских
скриптов, выполняемых на мастере каждого создаваемого кластера. Скрипты
определяются администратором кластера.

# - /home/<user>/clusters/ja_hipe_cluster/cluster_default_script.sql
# Пользовательский SQL скрипт выполняемый с аутентификационными данными
cluster_settings.db_init_conn_string после создания всех расширений.

cluster_settings:

- cluster_name: my_cluster                       # Наименование кластера.
Обязательный параметр. Кластер может быть один или несколько.

cluster_master_node: node_11                     # Нода, являющаяся Мастером.
Обязательный параметр.

# cluster_referee_nodes:                         # Блок узлов, являющимися
referee. Их может быть несколько.

# - node_referee                                № Имя ноды, выступающей в
роли рефери.

activated: true                                   # Флаг активации кластера после
формирования. true - активировать (назначать public_address), false - не
активировать.

jalog_users:                                     # Учетные записи для
администрирования jaDog. Создаются УЗ и запись в jalog_hba.cfg.

- name: admin                                    # Имя учетной записи.
Обязательный параметр.

pass: change_on_install                          # Пароль учетной записи.
Обязательный параметр. Для обеспечения безопасности смените пароль УЗ после
установки.

address: all                                     # Разрешенные адреса или
подсети для доступа к jaDog. Значения: 'all', имя хоста( 'myhost.local.net'
) или IP адрес с маской ('192.168.100.1/24').

```

```

method: sha-256 # Метод доступа. Значения:
"sha-256", "ssl". "sha-256" по умолчанию.

- name: interdog # УЗ можно создать несколько.
Для обеспечения безопасности смените пароль УЗ после установки.

pass: change_on_install # Для обеспечения безопасности
смените пароль УЗ после установки.

address: all

default_node_params: # Параметры по-умолчанию,
которые будут использовать для каждой ноды. Записи включаются в нотации
конфигурационного файла jadog.yml.

param_jadog: # Параметры конфигурирования
экземпляра jaDog.

# service_name: jadog # Имя сервиса jaDog для ноды.
"jadog" по умолчанию

public_address: 192.168.72.10 # Виртуальный IP адрес
кластера. Public address. Обязательный параметр.

trusted_address: 192.168.72.2 # Доверенный IP адрес. Trusted
IP address. Обязательный параметр для кластеров с одним slave.

network_interface: ens33 # Наименование сетевого
интерфейса для виртуального IP кластера. Network interface name.
Обязательный параметр.

interconnect_user: interdog # Имя технологической учетной
записи для взаимодействия с другими jaDog сервисами. Обязательный
параметр.

param_log: # Параметры логирования

logs_filename: jadog-%Y-%m-%d_%H%M%S # Шаблон для файлов
функциональных логов

logs_screen: true # Включение вывода логов на
экран. "false" по умолчанию

logs_type: csv, security.csv # Форматы файлов
функционального и логов событий ИБ. На первом месте формат функционального
файла логов (csv, json) на втором формат файла логов событий ИБ
(security.json, security.csv)

logs_level: debug2 # Уровень функционального
логирования

param_security_log: # Параметры журнала событий
информационной безопасности (ИБ)

security_logs_filename: jadog-%Y-%m-%d_%H%M%S # Шаблон для файлов
журнал событий ИБ

param_postgres: # Блок параметров для работы с
СУБД

db_service_name: jatoba-6 # Название сервиса экземпляра
СУБД в операционной системе

```

```

db_connection_settings:                                # Параметры для соединения с
СУБД

    database: jatoba                                    # Имя базы данных для установки
расширения jaDog. "postgres" по умолчанию

    passfile: /var/lib/jatoba/.pgpass                  # Jatoba passfile

    user: jadog_user                                    # Имя УЗ для работы с СУБД.
Обязательный параметр.

    user_pass: change_on_install                       # Пароль УЗ для работы с СУБД.
Обязательный параметр. Сменить пароль УЗ в СУБД после формирования
кластера.

    param_rest_api:                                     # Блок параметров для REST-
сервера

        rest_api_use: true                             # Признак использования
REST-сервера

        rest_api_cert_file: /var/lib/jatoba/ssl/jadog_cert/jadog_server.crt
# Путь до файла сертификата REST-сервера

        rest_api_key_file: /var/lib/jatoba/ssl/jadog_cert/jadog_server.key
# Путь до файла ключа REST-сервера

        rest_api_ca_file: /var/lib/jatoba/ssl/jatoba_ca.crt
# Путь до файла CA REST-сервера

#         rest_api_crl_file: /var/lib/jatoba/ssl/jadog_cert/ca-cert.crt
# Путь до файла CRL REST-сервера

        rest_api_listen_address: 0.0.0.0              # Прослушиваемый адрес
REST-сервера

        rest_api_listen_port: 54443                   # Порт REST-сервера

datacenters: #

    - datacenter: DC_Piter                             # Уникальное наименование
датацентра. Датацентр может быть один или несколько.

    nodes:                                              # Блок описания нод кластера

        - node_name: node_11                          # Кластерное имя ноды. Все
записи конфигурирования включаются в нотации конфигурационного файла
jadog.yml.

        param_jadog:                                    # Параметры конфигурирования
экземпляра jaDog.

            ip: 192.168.72.11                          # IP адрес ноды

#             network_interface: ens33                  # Наименование сетевого
интерфейса для виртуального IP кластера. Network interface name.
Обязательный параметр.

#             param_synchronous:                       # Раскомментировать блок
параметров, если потребуется синхронная реплика

#             synchronous_commit: on                   # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off

```

```

#           synchronous_type: ANY           # Как будет работать синхронная
репликация по принципам: FIRST или ANY

#           synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита

           param_replication:                # Параметры репликации ноды

           replication_slot_name: rs_node_11 # Уникальное в рамках
кластера имя слота репликации.
- node_name: node_12
           param_jadog:
           ip: 192.168.72.12
           param_replication:
           replication_slot_name: rs_node_12

#           param_synchronous:              # Раскомментировать блок
параметров, если потребуется синхронная реплика

#           synchronous_commit: on          # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off

#           synchronous_type: ANY           # Как будет работать синхронная
репликация по принципам: FIRST или ANY

#           synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита

- node_name: node_13
           param_jadog:
           ip: 192.168.72.13
           param_replication:
           replication_slot_name: rs_node_13

#           param_synchronous:              # Раскомментировать блок
параметров, если потребуется синхронная реплика

#           synchronous_commit: on          # Тип синхронного коммита
согласно документации postgresql: on, remote_apply, remote_write, local,
off

#           synchronous_type: ANY           # Как будет работать синхронная
репликация по принципам: FIRST или ANY

#           synchronous_commit_max_nodes: 1 # Количество нод, требующих
подтверждения синхронного коммита

#           - node_name: node_referee      # Заготовка для новой ноды в
клстере, например, с ролью referee

#           param_jadog:

#           ip: 192.168.72.23

```

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

Master – главный узел кластера, обрабатывающий запросы на запись.

Slave – синхронная и асинхронная реплики для чтения и резервирования.

Referee – узел-арбитр для выбора нового главного узла кластера при возникновении сбоев.

Trusted Address – IP-адрес для аварийного переключения.

WAL-архив – отдельное хранилище журналов транзакций для аварийного восстановления данных.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

SQL	–	Structured Query Language – язык структурированных запросов
SSL	–	Secure Sockets Layer – уровень защищенных сокетов. Криптографический протокол, который подразумевает безопасную связь
БД	–	База данных
ОС	–	Операционная система
СУБД	–	Система управления базами данных

[illegible]

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