**RT.WIDESTORE**

**ИНСТРУКЦИЯ ПО УСТАНОВКЕ**

2025

# 1. Введение

**Примечание:** Непосредственно операции по подготовке к созданию и созданию кластера описаны в документации к RT.ClusterManager. Данные о порядке создания компонент и по распределению сервисов компонент по нодам, описанные  в данном документе, относятся к операциям описанным в [п. 8.4.1 и п. 8.4.2 документа “RT.ClusterManager. Руководство администратора”](https://docs.data.rt.ru/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F/RT_ClusterManager/Vers_2/RT_ClusterManager_Admin_Guide_ver2#:~:text=8.4.1%20%D0%A1%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%BE%D0%B2).

**Важно: На данный момент установка кластеров RT.WideStore через Wizard не поддерживается - используйте ручной режим**

Предлагаемая конфигурация кластеров, создаваемых на основании плагина «RT.WideStore» может состоять либо из одной, либо из нескольких нод принадлежащих одному провайдеру. Ноды равнозначны и среди них не выделяются  master-node или slave-node.

В данном описании ноды – сервера (хосты) на базе операционных систем семейства Linux, которые будут включены в кластер.

Режим высокой доступности  – это метод проектирования системы, позволяющий достигать высокий уровень доступности кластера в течение определённого промежутка времени. Для данного приложения реализованного с помощью плагина «RT.WideStore» переключатель “Высокая доступность” на форме создания кластера ни на что не влияет.

**Важно: При развёртывании стандартного кластера необходимо строго придерживаться порядка установки компонентов, указанного в таблице ниже. В иных случаях существует возможность частично откорректировать распределение.**

**Примечание**: В RT.ClusterManager версии 2.Х-ХХ был изменен дизайн и терминология, поменялись местами термины “Компонент” и “Сервис”. Так в RT.ClusterManager версии 1.Х-ХХ: Zookeeper, HDFS … были сервисами, и входящие в них элементы программного обеспечения которые привязывались к нодам  назывались компонентами, а в RT.ClusterManager версии 2.Х-ХХ: Zookeeper, HDFS … стали компонентами, и входящие в них элементы программного обеспечения которые привязываются к нодам  называются сервисами. В данном документе используется терминология RT.ClusterManager версии 2.Х-ХХ.

**Примечание**: clickhouse-server и clickhouse-keeper могут работать как на IPv4 так и IPv6 протоколе. При установке RT.WideStore если включена галочка listen\_host\_wildcard в true - автоматически в listen\_address прописывается маска для IPv6, если он включен на хостовых машинах. Однако, если IPv6 включен на хостовых машинах, но DNS адресация для него не настроена - могут быть ошибки маршрутизации между нодами кластера. Поэтому в таком случае рекомендуется перевести listen\_host\_wildcard в false и вручную прописать маску подсети listen\_host для IPv4 (например “127.0.0.1”)

# ****2. Распределение сервисов по нодам - плагин «RT.WideStore»****

**RT.WideStore** поддерживает два режима установки:

Установка в режиме одного узла.
Установки сервиса **Zookeeper**или  **keeper**не требуется, установка поддерживается только на один хост, отсутствует репликация и шардирование.

Установка в режиме кластера.
Обязательно установите сервис **Zookeeper** или  **keeper**перед установкой **RT.WideStore**, все хосты из группы single-node становятся репликами внутри одного шарда.

## 2.1 Установка в режиме одного узла

|  |  |  |  |
| --- | --- | --- | --- |
| ***№*** | ***Компонент*** | ***Назначение*** | ***Распределение сервисов по нодам*** |
| ***Без режима “Высокой доступности”*** | ***В режиме “Высокая доступность”*** |
| 1 | **ch** | Столбцовая система управления базами данных.Проверка статуса - см. [п. 3.1](https://docs.data.rt.ru/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F/RT_WideStore/Vers_2/RT_WideStore_Installation_of_services_based_of_the_RT_WideStore_plugin#:~:text=3.1%20%D0%9F%D1%80%D0%BE%D0%B2%D0%B5%D1%80%D0%BA%D0%B0%20%D1%81%D1%82%D0%B0%D1%82%D1%83%D1%81%D0%B0%20ch). | **single-node–** Хост-1. | – |

## 2.2 Установка в режиме кластера

В следующей таблице представлен порядок распределения по нодам компонентов и их сервисов на основании плагина «RT.WideStore»:

|  |  |  |  |
| --- | --- | --- | --- |
| ***№*** | ***Компонент*** | ***Назначение*** | ***Распределение сервисов по нодам*** |
| ***Без режима “Высокой доступности”*** | ***В режиме “Высокая доступность”*** |
| 1 | **zookeeper** | Управляет распределением данных по хостам ch.Проверка статуса - см. [п. 3.3](https://docs.data.rt.ru/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F/RT_WideStore/Vers_2/RT_WideStore_Installation_of_services_based_of_the_RT_WideStore_plugin#:~:text=3.3%20%D0%9F%D1%80%D0%BE%D0%B2%D0%B5%D1%80%D0%BA%D0%B0%20%D1%81%D1%82%D0%B0%D1%82%D1%83%D1%81%D0%B0%20zookeeper).  | **–** | **server –** Хост-1, Хост-2, Хост-3. |
| 2 | **keeper** | Управляет распределением данных по хостам ch (является альтернативой **zookeeper**уменьшающей затраты оперативной памяти**)**.Проверка статуса - см. [п. 3.4](https://docs.data.rt.ru/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F/RT_WideStore/Vers_2/RT_WideStore_Installation_of_services_based_of_the_RT_WideStore_plugin#:~:text=3.4%20%D0%9F%D1%80%D0%BE%D0%B2%D0%B5%D1%80%D0%BA%D0%B0%20%D1%81%D1%82%D0%B0%D1%82%D1%83%D1%81%D0%B0%20keeper).  | **–** | **server –** Хост-1, Хост-2, Хост-3. |
| 3 | **ch** | Столбцовая система управления базами данных.Проверка статуса - см. [п. 3.1](https://docs.data.rt.ru/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F/RT_WideStore/Vers_2/RT_WideStore_Installation_of_services_based_of_the_RT_WideStore_plugin#:~:text=3.1%20%D0%9F%D1%80%D0%BE%D0%B2%D0%B5%D1%80%D0%BA%D0%B0%20%D1%81%D1%82%D0%B0%D1%82%D1%83%D1%81%D0%B0%20ch) и проверка работы кластерного режима  [п. 3.2](https://docs.data.rt.ru/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F/RT_WideStore/Vers_2/RT_WideStore_Installation_of_services_based_of_the_RT_WideStore_plugin#:~:text=3.2%20%D0%9F%D1%80%D0%BE%D0%B2%D0%B5%D1%80%D0%BA%D0%B0%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%8B%20%D0%BA%D0%BB%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%BD%D0%BE%D0%B3%D0%BE%20%D1%80%D0%B5%D0%B6%D0%B8%D0%BC%D0%B0). | **–** | **single-node** **–** Хост-4,  Хост-5 ( + …, Хост-n). |

# 3. Проверка работы компонентов

## 3.1 Проверка статуса ****ch****

Успешность установки можно проверить, зайдя на хост и выполнив команду под любым пользователем:

clickhouse-client

Copy

## 3.2 Проверка работы кластерного режима

Для проверки работы кластерного режима нужно использовать консольную утилиту clickhouse-client на хостах из группы sigle-node.

На любом хосте смотрим статус кластера:

select \* from system.clusters;

Copy

Затем на всех хостах этой группы создадим таблицу:

CREATE TABLE test (id int) ENGINE = ReplicatedReplacingMergeTree('/clickhouse/tables/{shard}/test', '{replica}', id) order by id;

Copy

На одном из хостов добавим данные:

insert into test values (1);

Copy

А на любом другом проверим, что они видны:

select \* from test;

Copy

## 3.3 Проверка статуса zookeeper

Для проверки статуса zookeeper на каждом хосте, на котором установлен сервис **zookeeper server** можно воспользоваться командой:

/usr/lib/zookeeper/bin/zkServer.sh status

Copy

При корректно работающем сервисе в выводе должен быть упомянут путь до файла конфигурации, используемый сетевой порт, сетевой адрес/имя, статус SSL и режим работы сервиса (Mode: **leader**или **follower**. На одном хосте должен быть режим – **leader**, на остальных – **follower**) :

## 3.4 Проверка статуса keeper

Для проверки статуса **keeper**на каждом хосте, на котором установлен сервис **keeper server** можно воспользоваться командой:

systemctl status clickhouse-keeper

Copy

На корректно работающем сервисе статус: *active (running)*.

# 4. Настройка компонентов

## 4.1 Настройка keeper

### 4.1.1 Назначение keeper

**Keeper**— это готовая замена **ZooKeeper**с полностью совместимым клиентским протоколом и той же моделью данных. Помимо этого, он предлагает следующие преимущества:

Более простая настройка и эксплуатация: **keeper**реализован на C++, а не на Java.

Снимки и журналы занимают гораздо меньше места на диске благодаря лучшему сжатию.

Нет ограничений на размер пакета и данных узла по умолчанию ( в **ZooKeeper**он [составляет 1 МБ).](https://zookeeper.apache.org/doc/r3.4.11/zookeeperAdmin.html#Unsafe+Options)

Нет проблем [с переполнением ZXID](https://issues.apache.org/jira/browse/ZOOKEEPER-1277) (принудительный перезапуск после каждых 2B транзакций в **ZooKeeper**).

Более быстрое восстановление после разделения сети за счет использования более распределенного протокола консенсуса.

Дополнительные гарантии [согласованности](https://en.wikipedia.org/wiki/Consistency_model) : **keeper**предоставляет те же гарантии согласованности, что и **ZooKeeper** — [линеаризуемые](https://en.wikipedia.org/wiki/Linearizability) записи плюс строгий порядок операций внутри одного [сеанса](https://zookeeper.apache.org/doc/r3.5.10/zookeeperProgrammers.html#ch_zkSessions) . Кроме того, и опционально (через настройку*quorum\_reads* ), **keeper**предоставляет линеаризуемые чтения.

**Keeper** более эффективен в плане использования ресурсов и использует меньше памяти для того же объема данных.

### 4.1.2 Требования для работы

Минимально необходимые версии ПО для работы с **кeeper**:

Плагин RT.WideStore: 1.25.7.

### 4.1.3 Настройка keeper в RT.ClusterManager

Если при использовании режима высокой доступности в кластер созданном с помощью плагина **RT.WideStore** вы используете компонент **кeeper**вместо **ZooKeeper**, то для настройки **кeeper**необходимо выполнить следующее:

Авторизуйтесь в**RT.ClusterManager**.

На форме “Кластеры” выберите нужный кластер, нажав на его наименование (см. Рис.1).

*Рис.1. Выбор кластера*

3. Откроется форма описания выбранного кластера. На вкладке “Общие” выберите компонент “кeeper” нажав на его (См. Рис 2).

*Рис.2. Выбор компонента “кeeper*

3. Откроется форма настройки компонента “кeeper”. Для настройки конфигурации компонента выберите вкладку “Конфигурация”, нажав на неё (см. Рис.3).

*Рис.3. Настройка конфигурации компонента “кeeper".*

4.Фома “Конфигурация” компонента “кeeper” содержит 2 вкладки, для настройки соответствующих конфигурационных файлов:

main,

кeeper\_server.

Для изменения параметров конфигурации внесите изменения в текущие значения  на соответствующей вкладке и нажмите кнопку “Сохранить” в нижней части формы.

Чтобы внесенные изменения вступили в действие запустите операцию “Переконфигурировать”, для чего выберите соответствующий пункт меню нажав на “…” в правом верхнем углу формы (см. Рис. 4).

*Рис.4. Запуск операции “Переконфигурировать".*

Проверить результаты завершения операции можно в логах на форме описания выбранного кластера, вкладка “События” (см. Рис. 5) .

*Рис.5. Вкладка “События" Выбранного кластера.*

### 4.1.4 Настройка параметров main

Вкладка “main” предназначена для настройки основных параметров конфигурации компонента **кeeper**.

При инсталляции компонента параметры полей во вкладках на панели “Конфигурация” установлены по умолчанию.

**Параметры конфигурации**:

**max\_connections** – Максимальное количество входящих соединений.

Значение по умолчанию: 2048

2) **listen\_host\_wildcard** – При значении true keeper будет слушать на всех адресах, в том числе ipv6 если он доступен.

Значение по умолчанию: true

3) **listen\_host –** По умолчанию keeper будет слушать на loopback адресах, если не используется listen\_host\_wildcard.

Значение по умолчанию: (2 строки)

::1

127.0.0.1

4) **logger\_level** – Уровень логирования keeper

Значение по умолчанию: notice

5) **logger\_size** – Максимальный размер файла с логами

Значение по умолчанию: 100M

6) **logger\_count** – Количество файлов логов в ротации.

Значение по умолчанию: 10.

### 4.1.5 Настройка параметров кeeper\_server

Вкладка “кeeper\_server” содержит настройки сервиса “server” компонента  **кeeper**.

**Параметры конфигурации**:

**tcp\_port –**Порт для подключения клиента.

Значение по умолчанию: 9181

2) **enable\_reconfiguration –**Включить динамическую реконфигурацию кластера.

Значение по умолчанию: False

3) **max\_memory\_usage\_soft\_limit\_ratio –**Если значение **max\_memory\_usage\_soft\_limit** не задано или равно нулю, мы используем это значение для определения мягкого ограничения по умолчанию.

Значение по умолчанию: 0.9

4) **cgroups\_memory\_observer\_wait\_time –**Если значение **max\_memory\_usage\_soft\_limit** не задано или равно 0, мы используем этот интервал для определения объема физической памяти. Как только объем памяти изменится, мы пересчитаем допустимый лимит памяти **keeper**на значение **max\_memory\_usage\_soft\_limit\_ratio**.

Значение по умолчанию: 15

5) **digest\_enabled –**Включить проверку согласованности данных в режиме реального времени.

Значение по умолчанию: True

6) **hostname\_checks\_enabled –**Включить проверку имени хоста на работоспособность для конфигурации кластера (например, если localhost используется с удаленными конечными точками).

Значение по умолчанию: True

7)**operation\_timeout\_ms –**Время ожидания для одной клиентской операции (мс).

Значение по умолчанию:10000

8) **min\_session\_timeout\_ms –**Минимальное время ожидания клиентского сеанса (мс).

Значение по умолчанию: 10000

9) **session\_timeout\_ms –**Максимальное время ожидания клиентского сеанса (мс).

Значение по умолчанию: 100000

10) **dead\_session\_check\_period\_ms –**Как часто **keeper**проверяет наличие неработающих сеансов и удаляет их (мс)

Значение по умолчанию: 500

11) **heart\_beat\_interval\_ms –**Как часто лидер **keeper** будет посылать сообщения подписчикам (мс)

Значение по умолчанию: 500

12)**election\_timeout\_lower\_bound\_ms –**Если последователь не получит сообщение от лидера в течение этого интервала, то он может инициировать выборы лидера. Значение параметра должно быть меньше или равно значению параметра **election\_timeout\_upper\_bound\_ms**. В идеале они не должны быть равны.

Значение по умолчанию: 1000

13) **election\_timeout\_upper\_bound\_ms –**Если последователь не получает сигнала от лидера в течение этого промежутка времени, то он должен инициировать избрание лидера.

Значение по умолчанию: 2000

14) **rotate\_log\_storage\_interval –**Сколько записей журнала следует хранить в одном файле.

Значение по умолчанию: 100000

15) **reserved\_log\_items –**Сколько записей журнала координации необходимо сохранить перед уплотнением.

Значение по умолчанию: 100000

16) **snapshot\_distance –**Как часто **keeper** будет создавать новые снимки (по количеству записей в журналах).

Значение по умолчанию: 100000

17)**snapshots\_to\_keep –** Сколько снимков нужно сохранить.

Значение по умолчанию:  3

18) **stale\_log\_gap –**Порог, при котором лидер считает подписчика устаревшим и отправляет ему снимок вместо журналов.

Значение по умолчанию: 10000

19) **fresh\_log\_gap –**Когда узел стал свежим.

Значение по умолчанию:  200

20)**max\_requests\_batch\_size –**Максимальный размер пакета в запросах учитывается до того, как он будет отправлен в RAFT.

Значение по умолчанию:  100

21)**force\_sync –**Вызывать fsync при каждой записи в журнал координации.

Значение по умолчанию: True

22)**quorum\_reads –** Выполнять запросы на чтение по мере записи через весь консенсус RAFT с одинаковой скоростью.

Значение по умолчанию: False

23**) raft\_logs\_level –**Уровень ведения текстового журнала о координации (трассировка, отладка и так далее).

Значение по умолчанию: information

24) **auto\_forwarding –**Позволять перенаправлять запросы на запись от подписчиков лидеру.

Значение по умолчанию: True

25)**shutdown\_timeout –**Дожидаться завершения внутренних подключений и выключения (мс).

Значение по умолчанию: 5000

26) **startup\_timeout –**Если сервер не подключится к другим участникам кворума в течение указанного времени ожидания, он завершит работу (мс).

Значение по умолчанию: 30000

27)**four\_letter\_word\_white\_list –**Белый список команд 4lw.

Значение по умолчанию: conf, cons, crst, envi, ruok, srst, srvr, stat, wchs, dirs, mntr, isro, rcvr, apiv, csnp, lgif, rqld, ydld

28) **async\_replication –**Включить асинхронную репликацию. Все гарантии на запись и чтение сохраняются ,а производительность повышается. По умолчанию параметр настройки отключен, чтобы не нарушать обратную совместимость.

Значение по умолчанию: False

29**) latest\_logs\_cache\_size\_threshold –**Максимальный общий размер кэша в памяти для последних записей журнала.

Значение по умолчанию: 1073741824

30)**commit\_logs\_cache\_size\_threshold –**Максимальный общий размер кэша записей журнала в памяти, необходимых в следующий раз для фиксации.

Значение по умолчанию: 524288000

31)**disk\_move\_retries\_wait\_ms –**Как долго ждать между повторными попытками после сбоя, произошедшего во время перемещения файла между дисками (мс).

Значение по умолчанию: 1000

32) **disk\_move\_retries\_during\_init –**Количество повторных попыток после сбоя, произошедшего при перемещении файла между дисками во время инициализации.

Значение по умолчанию: 100

33) **compress\_logs –**Сжимать логи.

Значение по умолчанию: False

34) **raft\_port –**Порт для подключения RAFT**.**

Значение по умолчанию: 9234

Powered by [Wiki.js](https://wiki.js.org/)