СИСТЕМА
УПРАВЛЕНИЯ
БАЗАМИ
ДАННЫХ

ЛИНТЕР®

ЛИНТЕР БАСТИОН

ЛИНТЕР СТАНДАРТ

Система резервирования

НАУЧНО-ПРОИЗВОДСТВЕННОЕ ПРЕДПРИЯТИЕ

РЕЛЭКС

Товарные знаки

РЕЛЭКС™, ЛИНТЕР® являются товарными знаками, принадлежащими АО НПП «Реляционные экспертные системы» (далее по тексту – компания РЕЛЭКС). Прочие названия и обозначения продуктов в документе являются товарными знаками их производителей, продавцов или разработчиков.

Интеллектуальная собственность

Правообладателем продуктов ЛИНТЕР® является компания РЕЛЭКС (1990-2026). Все права защищены.

Данный документ является результатом интеллектуальной деятельности, права на который принадлежат компании РЕЛЭКС.

Все материалы данного документа, а также его части/разделы могут свободно размещаться на любых сетевых ресурсах при условии указания на них источника документа и активных ссылок на сайты компании РЕЛЭКС: relex.ru и linter.ru.

При использовании любого материала из данного документа несетевым/печатным изданием обязательно указание в этом издании источника материала и ссылок на сайты компании РЕЛЭКС: relex.ru и linter.ru.

Цитирование информации из данного документа в средствах массовой информации допускается при обязательном упоминании первоисточника информации и компании РЕЛЭКС.

Любое использование в коммерческих целях информации из данного документа, включая (но не ограничиваясь этим) воспроизведение, передачу, преобразование, сохранение в системе поиска информации, перевод на другой (в том числе компьютерный) язык в какой-либо форме, какими-либо средствами, электронными, механическими, магнитными, оптическими, химическими, ручными или иными, запрещено без предварительного письменного разрешения компании РЕЛЭКС.

О документе

Материал, содержащийся в данном документе, прошел доскональную проверку, но компания РЕЛЭКС не гарантирует, что документ не содержит ошибок и пропусков, поэтому оставляет за собой право в любое время вносить в документ исправления и изменения, пересматривать и обновлять содержащуюся в нем информацию.

Контактные данные

394006, Россия, г. Воронеж, ул. Бахметьева, 2Б.

Тел./факс: (473) 2-711-711, 2-778-333.

e-mail: info@linter.ru.

Техническая поддержка

С целью повышения качества программного продукта ЛИНТЕР и предоставляемых услуг в компании РЕЛЭКС действует автоматизированная система учёта и обработки пользовательских рекламаций. Обо всех обнаруженных недостатках и ошибках в программном продукте и/или документации на него просим сообщать нам в раздел Поддержка на сайте ЛИНТЕР.

Содержание

Предисловие 6
Назначение документа 6
Для кого предназначен документ 6
Необходимые предварительные знания 6
Дополнительные документы 6
Общие сведения 7
Назначение системы резервирования 7
Условия применения 7
Состав программных средств 8
Архитектура системы резервирования 9
Основные понятия 9
Принципы построения системы резервирования 15
Характеристики системы резервирования 16
Программные компоненты 17
Администрирование системы резервирования 19
Установка системы резервирования 19
Запуск системы резервирования 19
Ключи запуска управляющей программы системы резервирования 20
Управление режимом запуска 22
Продолжительность прослушивания сети (wait) 22
Порт сервера резервирования (port) 23
Автоматический запуск системы резервирования (nq) 23
Разрешение на запуск системы резервирования по сигналу (wp) 24
Периодичность запросов интерактивного запуска (input) 25
Минимально допустимое количество серверов системы резервирования (nservers) 25
Запрет функционирования в фоновом режиме (nodaemon) 26
Запрет запуска сервисом (local) 26
Допустимый интервал простоя системы резервирования (down) 27
Приоритет сервера резервирования (prior) 27
Выбор рабочей БД на основе системного журнала (cmpsyslog) 28
Файл регистрационных данных администратора БД (pf) 29
Предотвращение потери данных при останове системы резервирования (testslave) 29
Отключение запуска ядра СУБД в безопасном режиме (unsafe_mode) 30
Ключ работы с защищенной базой (cryptadd) 30
Управление выдачей справочной информацией 30
Справочная информация о ключах (h) 30
Справочная информация о версии (version) 31
Предоставление информации о состоянии сервера и БД (show) 31
Пути к объектам 32
Путь к файлу сетевой конфигурации (ntab) 32
Путь к каталогу рабочей БД (pathtodb) 32
Путь к каталогу архивной БД (pathtoarc) 33
Управление состоянием 33
Активация проверки полномочий пользователя для управления сервером (cu) 33
Перевод рабочей БД в заданное состояние (setstate) 34
Длительность ожидания завершения операций (forceshut) 34
Управление обработкой событий 35
Имя обработчика событий (stproc) 35
Генерирование альтернативных кодов событий (altstproc) 36
Генерирование событий, относящихся к контролю времени (tclog) 36
Контроль за сетевыми событиями (watchnet) 37

Отмена очередности обработки событий (nostprocqueue) 37
Управление ресурсами 37
Размер пула памяти ядра СУБД ЛИНТЕР (pool) 37
Размер пула сортировки СУБД ЛИНТЕР (spool) 38
Управление отладочной информацией 39
Трассировка запросов к СУБД ЛИНТЕР (lintlog) 39
Протоколирование работы сервера резервирования (debug) 39
Стандартное протоколирование (log) 43
Протоколирование pid (pid) 43
Управление запуском программ 44
Дополнительные ключи запуска СУБД ЛИНТЕР (lintadd) 44
Ограничение повторных запусков ядра СУБД ЛИНТЕР (lretry) 44
Количество повторных запусков программ системы резервирования (pretry) 45
Задержка запуска сетевых драйверов (dbsctout) 45
Файл конфигурации ключей запуска программы (cf) 46
Запрет использования файла конфигурации запуска программы (nocf) 48
Управление архивами 48
Создание архивного файла (archivate) 48
Местоположение архивного файла (barc) 48
Дублирование архивируемых данных (mklhbarc) 49
Работа с базами данных на многих устройствах (devenv) 49
Разрешение на восстановление БД из архивного файла (crash) 51
Периодичность копирования рабочей БД в архивный каталог (arcint) 51
Автоматический обмен статуса каталогов БД резервного сервера (exchdir) 54
Копирование только отличий БД (diff) 54
Управление размером системного журнала (syslogcount) 55
Шаблон командной строки для сохранения БД в архивном файле (cmdarcadd) 56
Шаблон командной строки для восстановления БД из архивного файла (cmdarcext) 58
Сетевые настройки 58
Посылка тестовых пакетов (testint) 58
Критерий разрыва соединения (tstlimit) 59
Периодичность сохранения состояния баз данных серверов (dbtestint) 59
Управление тестированием 60
Регулярность тестирования БД резервного сервера (testdb) 60
Дополнительные указания по тестированию БД (tdbadd) 62
Управление слежением за процессами 62
Тайм-аут и интервал проверки ядра СУБД ЛИНТЕР (treq) 62
Облегченная проверка активности ядра СУБД ЛИНТЕР (st) 63
Слежение за родительским процессом (parent) 63
Слежение за управляющей программой системы резервирования (wd) 64
Скорость подстройки внутреннего времени системы резервирования (tcorrect) 65
Управление запуском компонентов 66
Создание сервиса 66
Удаление сервиса 67
Переменные окружения, влияющие на работу системы резервирования 67
PATH 67
SY00 67
ARJPT 68
SERVER_HOME 68
HOTRES_LOG 68
LINTER_MBX 68
NET_MBX 69
SRVCMD_LOG 69
Завершение работы системы резервирования 69

Завершение работы отдельной управляющей программы системы резервирования 69
Завершение работы всех управляющих программ 70
Конфигурирование и настройка системы резервирования 70
Синхронизация времени 70
Настройка переменных окружения 71
Создание и настройка файла сетевой конфигурации 71
Настройка клиентских компьютеров 72
Работа с единственной системой резервирования 72
Одновременная работа с несколькими системами резервирования 73
Выбор временных интервалов 74
Создание файла регистрационных данных администратора 75
Создание файла ключей запуска 76
Запуск системы резервирования 77
Первоначальный запуск 77
Синхронизация времени 77
Установка переменных окружения 77
Создание БД 77
Создание файла сетевой конфигурации 78
Запуск управляющей программы системы резервирования на главном сервере 78
Запуск управляющей программы системы резервирования на резервных серверах 79
Повторный запуск 79
Просмотр профиля сервера резервирования 80
Примеры запуска системы резервирования 82
Первоначальный запуск на главном сервере 82
Первоначальный запуск на резервном сервере 83
Пример командного файла запуска системы резервирования 83
Внешнее управление системой резервирования 85
Управление с помощью программ hresctl и srvcmd 85
Обмен статусами серверов резервирования 87
Остановить работу сервера резервирования 88
Остановить работу системы резервирования 88
Получить состояние сервера резервирования 88
Перейти в режим главного сервера 88
Перейти в режим резервного сервера 89
Протестировать БД резервного сервера 89
Скопировать рабочую БД в архивный каталог 89
Создать архивный файл рабочей БД 89
Отсоединить узел системы резервирования 90
Отсоединить сервер системы резервирования 90
Сообщения утилиты srvcmd 91
Linter error <NNNN> 91
Unknown user name 91
Invalid password 91
Inproper system state 91
Not enough user privilegies 91
Unknown error 92
Управление с помощью сигналов 92
Структура системы резервирования 94
Горячее архивирование – основа системы резервирования 94
Управляющая программа – средство автоматизации резервирования 96
Старт управляющей программы 97
Определение предыдущего состояния сервера 98

Анализ файла настройки системы резервирования 98
Инициализация сетевой связи с другими серверами 98
Информирование других серверов о своей конфигурации и сбор аналогичных сведений от других серверов 99
Выбор по заданным критериям сервера на роль главного сервера 99
Ожидание условий запуска 99
Вывод подтверждения запуска 100
Принятие решения о статусе 100
Алгоритм выбора главного сервера 101
Принципы сравнения баз данных по времени 101
Принципы сравнения баз данных по системному журналу 101
Запуск ядра и утилит СУБД в режиме главного сервера 101
Алгоритм выбора базы данных при переходе сервера в режим главного 102
Запуск утилит в режиме резервного сервера 103
Стационарный режим работы 105
Мониторинг состояния удаленных серверов и линий связи 106
Мониторинг линий связи 106
Мониторинг серверов 107
Мониторинг состояния серверов 107
Запуск обработчика событий системы резервирования 107
Конкурс на замещение главного сервера 108
Подключение резервных серверов к системе резервирования 108
Синхронизация внутреннего и системного времени 109
Рестарт процессов 109
Управление очередностью запуска 109
Задержка рестарта 109
Анализ кода завершения процессов 110
Ограничение попыток рестарта 110
Мониторинг работы утилит и ядра СУБД ЛИНТЕР 110
Способы мониторинга 110
Действия при обнаружении сбоя 110
Мониторинг состояния ядра СУБД ЛИНТЕР на главном сервере 110
Мониторинг сетевого драйвера сервера dbs_tcp 111
Мониторинг работы утилиты lhb 111
Мониторинг сетевого драйвера клиента 111
Самоконтроль управляющей программы 111
Автоматический рестарт 111
Рестарт пользовательской программой 111
Пользовательский мониторинг состояния сервера резервирования 112
Отслеживание контрольных точек 112
Управление рабочей БД 113
Очистка резервного каталога 113
Останов ядра СУБД ЛИНТЕР в специальном режиме 113
Копирование файлов рабочей БД в резервный каталог 114
Запуск ядра СУБД ЛИНТЕР в специальном режиме для рабочей БД 114
Создание файла архива резервной БД 114
Запуск ядра СУБД ЛИНТЕР для резервной БД 114
Тестирование БД 115
Обработка команд оператора 115
Удаленное управление 115
Проверка санкционированности команды 115
Выполнение сетевых команд управляющей программой 115
Причины отказа выполнения команды 116
Управление сигналами 117
Завершение работы запущенных процессов при переходе в новое состояние ... 117

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Завершение процессов главным сервером 117
Завершение процессов резервным сервером 118
Самостоятельный останов управляющей программы 118
Разрешение конфликтов главных серверов 118
Регистрация событий системы резервирования 119
Обработчик событий 119
Формат вызова обработчика событий 119
Наборы событий 120
События системы резервирования 120
Значения дополнительных аргументов 122
Событие SLAVE_OK 122
Событие SHUT_DOWN 123
Событие NOT_SHUT_DOWN 123
Событие SLAVEFAILER 123
Событие SLAVECRASH 124
Событие MAINFAILER 124
Событие MAINCRASH 124
Событие A_SLAVE_OK 125
Событие A_STOPPED 125
Событие E_TESTDB 125
Событие MONO 126
Событие MAIN 127
Событие SLAVE 127
Событие SLAVE_WAIT 127
Событие SW_TO_MONO 127
Событие STARTMONO 127
Событие UNDEFINED 127
Событие E_PROCESS_START 127
Событие E_PROCESS_EXIT 128
Событие E_SHUT_COMMAND 128
Событие E_STOP_COMMAND 128
Событие E_NET_INFO 128
Событие E_TIME_CHANGE 129
Событие E_SERVER_TIME_DIFF 129
Событие W_DEADLOCK 129
Примеры вызова обработчика событий 130
Пример программы обработчика событий 130
Сообщения системы резервирования 132
Проверка конфигурирования системы резервирования 133
Обращение к системе резервирования из клиентского приложения 135
Коды завершения при работе с системой резервирования 136
Приложение 1. Пример фильтра трассировки 138
Приложение 2. Коды событий системы резервирования 140
Приложение 3. Пример работы с системой резервирования 141
Указатель ключей 147

Предисловие

Назначение документа

Документ является руководством по использованию системы резервирования СУБД ЛИНТЕР. В нем описывается архитектура системы резервирования, особенности разработки клиентских приложений для СУБД ЛИНТЕР с поддержкой системы резервирования, процедуры запуска и управления работой системы резервирования.

Документ предназначен для СУБД ЛИНТЕР БАСТИОН 6.0 сборка 20.8, далее по тексту СУБД ЛИНТЕР.

Для кого предназначен документ

Документ предназначен для разработчиков информационных и автоматизированных систем управления на базе СУБД ЛИНТЕР.

Необходимые предварительные знания

Для работы с системой резервирования необходимо:

- знать основы реляционных баз данных;
- владеть языком баз данных SQL для СУБД ЛИНТЕР;
- уметь работать в соответствующей операционной системе на уровне простого пользователя.

Дополнительные документы

- Установка СУБД ЛИНТЕР в среде ОС Linux
- Установка СУБД ЛИНТЕР в среде ОС Windows
- Сетевые средства
- Запуск и останов СУБД ЛИНТЕР в среде ОС Windows
- Запуск и останов СУБД ЛИНТЕР в среде ОС Linux
- Администрирование комплекса средств защиты данных
- Тестирование базы данных
- Создание и конфигурирование базы данных
- Справочник кодов завершения

© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Общие сведения

Назначение системы резервирования

Система резервирования СУБД ЛИНТЕР предназначена для обеспечения непрерывной круглосуточной работы программно-аппаратного комплекса повышенной надежности на основе СУБД ЛИНТЕР.

Условия применения

Система резервирования функционирует в комплексе с СУБД ЛИНТЕР на компьютерах с операционными системами типа Linux, Windows, ЗОСРВ Нейтрино. Все компьютеры комплекса должны иметь одинаковую программно-техническую архитектуру.

На всех компьютерах системы резервирования должен быть установлен и настроен протокол TCP/IP. Все компьютеры системы должны быть объединены в локальную сеть. Для повышения надежности можно использовать несколько линий связи между компьютерами.

Компьютеры системы резервирования должны иметь достаточное количество свободного дискового пространства для одновременного размещения двух БД с учетом возможного роста размера файлов системного журнала из-за временного рассогласования данных между главным и резервным компьютером.

Ядро СУБД при работе в системе горячего резервирования запускается в безопасном режиме (с ключом /SAFE_MODE). В данном режиме запрещено выполнение ряда команд. Список запрещенных команд приведен в Запуск и останов СУБД ЛИНТЕР в среде ОС Linux или в Запуск и останов СУБД ЛИНТЕР в среде ОС Windows в разделе "Ключи управления функционированием".

Не допускается использование БД, файлы которых расположены на нескольких устройствах. Все файлы БД должны быть размещены на устройстве, прописанном в переменной окружения SY00, или в текущем каталоге.

Примечания

1. В случае использования нескольких сетевых адаптеров в каждой машине необходимо, чтобы адаптеры в машинах были соединены попарно и их маршруты не пересекались. То есть, чтобы первому адаптеру одной машины всегда приходили пакеты с первого адаптера другой, а второму всегда со второго. Обычно это осуществляется разделением адаптеров в разные подсети. Обычно в каждой подсети устанавливается свой маршрутизатор, такое разделение защищает также и от выхода из строя маршрутизатора.
2. В случае наличия только одной линии связи и выхода ее из строя оба сервера становятся главными (main). При восстановлении сети в общем случае не определено кто из них останется главным (main). Поэтому в случае выхода из строя сети, администратор перед восстановлением соединения должен выбрать сервер, который останется главным (main), а на втором остановить систему резервирования. Для предотвращения таких ситуаций рекомендуется использовать дублирование линий связи между серверами и клиентами.
3. При работе с системой горячего резервирования рекомендуется использовать только запросы типа DML, для выполнения других типов запросов рекомендуется перевести систему в MONO-режим (одномашинный режим).

Состав программных средств

На компьютерах системы резервирования, в общем случае, должно быть установлено следующее программное обеспечение:

1) для ОС Linux, ЗОСРВ Нейтрино:
   - программа управления сервером резервирования `server`;
   - ядро СУБД ЛИНТЕР с компонентами (`linter`, `sql`, `intsrt`, `tsp`);
   - сетевой драйвер сервера (`dbs_tcp`);
   - сетевой драйвер клиента (`dbc_tcp`);
   - утилита архивирования и восстановления БД (`lhb`);
   - утилита тестирования БД (`testdb`);
   - системная утилита копирования файлов (`cp`);
   - системная утилита переноса файлов (`mv`);
   - системная утилита удаления файлов (`rm`);
   - системная утилита архивации (по выбору);
   - пользовательские приложения (при необходимости).

   На каждом клиентском компьютере системы резервирования должны быть установлены:
   - сетевой драйвер клиента (`dbc_tcp`);
   - пользовательские приложения.

2) для ОС типа Windows:
   - программа управления сервером резервирования `server.exe` (далее при использовании `server` для Windows подразумевается `server.exe`);
   - ядро СУБД ЛИНТЕР (`linternt.exe`/`linter64.exe` – в зависимости от разрядности установленной СУБД, далее при использовании `linter` для Windows подразумевается `linternt.exe`/`linter64.exe`);
   - сетевой драйвер сервера (`dbs_tcp.exe`);
   - сетевой драйвер клиента (`dbc_tcp.exe`);
   - утилита архивирования и восстановления БД (`lhb.exe`, далее при использовании `lhb` для Windows подразумевается `lhb.exe`);
   - утилита тестирования БД (`testdb.exe`, далее при использовании `testdb` для Windows подразумевается `testdb.exe`);
   - системная утилита командной строки (`cmd.exe`);
   - пользовательские приложения (при необходимости).

   На каждом клиентском компьютере системы резервирования должны быть установлены:
   - сетевой драйвер клиента (`dbc_tcp`);
   - пользовательские приложения.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Архитектура системы резервирования

Основные понятия

Система резервирования

Система резервирования – совокупность серверов резервирования, функционирующих совместно и обеспечивающих повышенную степень надежности функционирования СУБД ЛИНТЕР в случае отказа вычислительного оборудования и программных средств.

Сервер резервирования

Сервер резервирования – компьютер системы резервирования, предназначенный для работы в составе системы резервирования, на котором функционирует программа управления `server`.

Компьютер системы резервирования

Компьютер системы резервирования – компьютер, предназначенный для выполнения функций главного/резервного сервера системы резервирования.

Узел системы резервирования

Узел системы резервирования – точка подключения к локальной сети, имеющая уникальный сетевой адрес и указанная в файле сетевой конфигурации системы резервирования `nodetab`. Физически один компьютер может быть подключен к нескольким линиям связи через разные разъемы с разными сетевыми адресами. В данном случае система резервирования на этом компьютере будет иметь несколько узлов сети. В файле сетевой конфигурации `nodetab`, используемом системой резервирования, должны указываться **узлы** системы резервирования.

Программа управления

Программа управления – программа `server`, запускаемая на **каждом** компьютере системы резервирования. Предназначена для выполнения главным/резервным сервером функций сервера системы резервирования при его взаимодействии с программами управления на других серверах системы резервирования.

Главный сервер

Главный сервер – сервер резервирования, на котором запущено ядро СУБД ЛИНТЕР, работающее с клиентскими приложениями.

Резервный сервер

Резервный сервер – сервер резервирования, на котором поддерживается копия рабочей БД главного сервера.

Актуальная БД

Актуальная БД – БД, с которой работали клиентские приложения (в режиме главного сервера), или полученная с главного сервера на резервный в данном сеансе работы управляющей программы, то есть актуальная БД – это база данных на главном или резервном сервере, которая изменялась или могла быть изменена после запуска системы резервирования. Резервная БД может стать актуальной только в результате копирования актуальной рабочей БД.

Рабочая БД

Для главного сервера – это БД, с которой работает ядро СУБД ЛИНТЕР (и, соответственно, клиентские приложения).

Для резервного сервера – это БД, в которой сохраняются данные, получаемые с главного сервера. Ядро СУБД ЛИНТЕР работает с этой БД в специальном режиме. Рабочая БД хранится в рабочем каталоге сервера.

Рабочий каталог сервера устанавливается либо переменной SY00, либо ключом /pathtodb, либо будет использован текущий каталог. Для задания пути к БД необходимо использовать абсолютные пути.

Архивная БД

БД, которая не является рабочей. Служит для хранения копии рабочей БД на главном/резервном сервере резервирования. Архивная БД располагается в архивном каталоге.

Местоположение архивной БД определяется при старте системы резервирования (см. описание ключа /pathtoarc).

Для задания пути к БД необходимо использовать абсолютные пути.

Архивный каталог

Каталог для хранения архивной БД.

Рабочий каталог

Каталог для хранения рабочей БД.

Архивный файл БД

Упакованная в один файл копия архивной БД, созданная с помощью внешней программы архивирования файлов (`zip`, `tar`, `gz`, `bzip2`).

Состояние сервера резервирования

Состояние сервера резервирования – возможное (с точки зрения системы резервирования) логическое состояние сервера резервирования.

Состояние сервера резервирования	Характеристика состояния
UNDEFINED	Состояние после запуска на компьютере сервера резервирования (до ранжирования серверов). В этом состоянии происходит чтение предыдущих состояний рабочей и архивной БД данного сервера резервирования, передача состояния БД другим серверам системы резервирования и подготовка к выбору MONO-сервера.
MONO	Главный сервер в одномашинном режиме. В MONO-состояние сервер переходит в следующих случаях: • при старте не было обнаружено других серверов за период опроса сети; • при старте группы серверов данный сервер оказался наиболее подходящим по времени или по состоянию системного журнала (в этом случае MONO будет лишь промежуточным состоянием перед переходом в MAIN). При переходе в MONO-состояние сервер проходит через промежуточные состояния: • STARTMONO – при переходе из UNDEFINED-состояния (после старта сервера резервирования);

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Состояние сервера резервирования	Характеристика состояния
	• SW_TO_MONO – при переходе из SLAVE-состояния.
MAIN	Главный сервер в многомашинном режиме. В MAIN-состояние сервер переходит из MONO-состояния при подключении хотя бы одного резервного сервера.
SLAVE	Резервный сервер в состоянии первоначальной загрузки (копирования) БД с главного сервера.
SLAVE_OK	Первоначальная загрузка (копирование) БД на резервном сервере завершена. Резервный сервер вышел в режим горячего резерва.
SLAVE_WAIT	Резервный сервер в ожидании готовности главного сервера.
STARTMONO	Выполняется переход из UNDEFINED-состояния в MONO-состояние.
SW_TO_MONO	Выполняется переход из SLAVE-состояния в MONO-состояние.
SHUT_DOWN	Сервер завершает работу.

Во время нахождения сервера в промежуточных состояниях `STARTMONO`, `SW_TO_MONO` происходит запуск ядра СУБД ЛИНТЕР и сетевого драйвера сервера `dbs_tcp`. После успешного запуска этих программ сервер переходит в состояние `MONO`, в котором остается до тех пор, пока хотя бы один резервный сервер не закончит первоначальное копирование рабочей БД с главного сервера. Как только это произойдет, сервер перейдет из состояния `MONO` в состояние `MAIN`. В этом состоянии сервер будет находиться до тех пор, пока активен хотя бы один резервный сервер, выполнивший первоначальное копирование БД.

В это время остальные серверы резервирования должны работать в режиме резервных (`SLAVE_OK`). До перехода в это состояние сервер сначала попадает в состояние `SLAVE_WAIT` – состояние ожидания готовности главного сервера. В этом состоянии сервер ждет подтверждения готовности от главного сервера (главный может еще находиться в процессе запуска). Как только главный сервер будет готов, он пришлет разрешение на продолжение работы резервного сервера, и тот перейдет в состояние копирования БД (`SLAVE`). В SLAVE-состоянии запускается сетевой драйвер клиента `dbc_tcp`, утилита архивирования `lhb` в режиме первоначального копирования БД. По завершении первоначального копирования БД запускаются утилита `lhb` в режиме дозагрузки изменений в файлы системного журнала и ядро СУБД ЛИНТЕР в специальном режиме, которое переносит изменения из системного журнала в БД. После этого сервер переходит в режим `SLAVE_OK` – режим горячего резерва.

В состояние `SHUT_DOWN` сервер переходит при получении команды на завершение работы от пользователя или другого сервера по сети, либо сигнала на завершение работы сервера или системы резервирования (завершение работы, инициированное внешними событиями).

Состояние БД

Состояние БД – логическое состояние БД сервера резервирования в момент текущей или предшествующей работы сервера резервирования с данной БД.

Состояние БД	Характеристика состояния
UNDEFINED	При первом запуске системы резервирования для этой БД (характеризуется отсутствием файла `state`, который содержит информацию о предыдущем состоянии БД).

Состояние БД	Характеристика состояния
MONO	БД на главном сервере в одномашинном режиме.
MAIN	БД на главном сервере в многомашинном режиме.
SLAVE	БД на резервном сервере.
SLAVEFAILER	БД на резервном сервере, остановленном из-за ошибки запуска на нём одного из процессов: `dbc_tcp`, `linter`, `testdb`, `lhb`.
SLAVECRASH	БД на резервном сервере, остановленном из-за неудачного копирования БД при старте системы резервирования. БД непригодна для использования. В это состояние БД переходит также при кодах завершения `testdb`, `linter`, информирующих о разрушении целостности БД.
MAINFAILER	БД на главном сервере, остановленном из-за ошибки запуска на нём одного из процессов – `dbs_tcp`, `linter`, `lhb`.
MAINCRASH	БД на главном сервере, остановленном из-за повреждения на нём БД. БД непригодна для использования. Повреждение определяется по соответствующему коду завершения `linter`.

События в системе резервирования

Информирование пользователей о внутренней жизни системы резервирования и происходящих в ней изменениях выполняется с помощью механизма *событий*.

Термин «событие» означает явление, которое произошло в системе резервирования и относится к категории времени. При изменении состояния сервера резервирования программа управления генерирует соответствующее событие. Обработка событий выполняется программой управления системой резервирования в соответствии с заложенным в ней алгоритмом, однако пользователь может потребовать информировать его о событиях путем вызова специальной программы с тем, чтобы он имел возможность контролировать работу системы резервирования и/или выполнять дополнительную обработку интересующих его событий.

Список генерируемых программой управления сервером резервирования событий приведен в таблице 1.

Таблица 1. Список событий системы резервирования

Мнемоника события	Описание события	Дополнительное условие
UNDEFINED	Определение состояния запущенного сервера.	
SERVEREXIT	Программа управления сервером завершила свою работу.	
MONO	Сервер стал главным сервером. Резервных серверов нет.	
MAIN	К главному серверу подключился первый резервный.	
SLAVE	На резервном сервере идет первоначальное скачивание данных.	
SLAVE_WAIT	Резервный сервер ждет подтверждения готовности главного сервера.	

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Мнемоника события	Описание события	Дополнительное условие
SLAVE_OK	Резервный сервер перешел в состояние готовности (горячего резерва).	
SW_TO_MONO	Резервный сервер начал процесс перехода в состояние главного сервера.	
SHUT_DOWN	Управляющая программа сервера приступила к завершению своей работы. Останов инициирован внешними командами.	
SLAVEFAILER	Работа управляющей программы резервного сервера завершена из-за ошибки запуска на нем одного из процессов.	
SLAVECRASH	Работа управляющей программы резервного сервера завершена при старте системы резервирования из-за ошибки при скачивании БД с главного сервера.	
MAINFAILER	Работа управляющей программы главного сервера завершена из-за ошибки запуска на нем одного из процессов.	
MAINCRASH	Работа управляющей программы главного сервера завершена из-за повреждения БД.	
STARTMONO	Сервер начал переход из UNDEFINED-состояния в MONO-состояние.	
STOPPED	Управляющая программа сервера остановлена.	
END_STATUS	Зарезервировано.	
WAIT_OLDER	Сервер начал переход к ожиданию готовности главного сервера для дальнейшего перехода в состояние резервного сервера.	
NOT_FOUND	Истек тайм-аут прослушивания сети при старте. Статус серверу (главный/резервный) пока не назначен.	
NOT_SHUT_DOWN	Сервер проигнорировал команду на обмен состояния или останов по причине своей неготовности (запрещен останов главного или резервного сервера, если процесс первоначального скачивания БД не закончен).	
SERVERRESTART	Выполнен перезапуск управляющей программы `server`.	

Мнемоника события	Описание события	Дополнительное условие
	Зарезервировано (используется для совместимости с предыдущими версиями системы резервирования).	
E_PROCESS_START	На сервере стартовал процесс.	Только при наличии ключа `/altstproc`
E_PROCESS_EXIT	На сервере завершился процесс.	Только при наличии ключа `/altstproc`
E_SHUT_COMMAND	Получена команда на завершение работы управляющей программы всех серверов.	Только при наличии ключа `/altstproc`
E_STOP_COMMAND	Сервер получил команду на останов своей работы.	Только при наличии ключа `/altstproc`
E_NET_INFO	Произошло изменение состояния одного из серверов или узлов.	Только при наличии ключей `/watchnet`, `/altstproc`
E_TIME_CHANGE	Системное время на сервере переведено.	Только при одновременном наличии ключа `/tclog`
E_SERVER_TIME_DIFF	Выявлена разность времени между данным сервером и одним из других серверов.	Только при наличии ключа `/tclog`
W_DEADLOCK	Ядро СУБД ЛИНТЕР перестало реагировать на запросы системы резервирования (зависло).	Только при наличии ключа `/altstproc`
E_TESTDB	Начало или окончание процедуры тестирования БД.	
A_SLAVE_OK	Главный сервер получил уведомление о готовности резервного сервера.	
A_SHUT_DOWN	Зарезервировано.	
A_STOPPED	Резервный сервер получил уведомление о том, что главному серверу подана команда на останов.	

Тестовые пакеты

Тестовые пакеты – небольшие блоки данных, посылаемые управляющими программами системы резервирования по локальной сети для мониторинга состояния линий связи и серверов системы резервирования. Каждый тестовый пакет, посланный сервером-источником, предполагает получение ответа от сервера-получателя в течение определенного интервала времени (тайм-аута).

Интервал посылки тестовых пакетов задает «тактовую частоту», на основе которой в системе резервирования выполняются все временные операции. Например, если интервал посылки тестовых пакетов равен 3 секундам, а интервал посылки тестовых запросов к ядру СУБД ЛИНТЕР – 2 секунды, то реально посылка тестовых запросов будет выполняться с периодом 3 секунды.

Идентификатор сервера резервирования

Идентификатор сервера резервирования – уникальное (в пределах системы резервирования) внутреннее 128 битное значение. Идентификатор формируется на основе

IP-адреса компьютера, его сетевого имени, случайного числа и, возможно, значения ключа `/prior` командной строки запуска сервера резервирования.

Компьютер, на котором работает сервер резервирования, может иметь больше одного сетевого интерфейса. Каждый такой интерфейс задается своим сетевым адресом в описании узла в файле сетевой конфигурации `nodetab`.

Идентификатор используется для проверки принадлежности того или иного узла локальной сети, описанного в файле `nodetab`, конкретному серверу системы резервирования, а именно: если полученные от разных узлов идентификаторы совпадают, это означает их принадлежность одному серверу резервирования.

На основании идентификатора определяются и собственные узлы сервера резервирования (идентификатор узла совпадает со сгенерированным идентификатором).

Старший сервер

Старший сервер – сервер резервирования, на который возложена функция разрешения конфликтов при конкуренции за выполнение операций, как правило, при старте системы резервирования (когда все компьютеры равны).

Старшим назначается сервер, запущенный на компьютере с наибольшим идентификатором.

Узел по умолчанию

Узел удаленного сервера, к которому удалось установить соединение раньше других узлов этого сервера, назначается узлом по умолчанию для удаленного сервера. По линии связи, соединенной с этим узлом, будет осуществляться сетевой обмен, передаваться и приниматься данные. Остальные линии связи, ассоциированные с другими узлами данного сервера, назначаются запасными. Периодически все линии связи тестируются на целостность путем посылки тестовых пакетов.

Принципы построения системы резервирования

Архитектура системы резервирования СУБД ЛИНТЕР базируется на основе решений, принятых в следующих системах:

- классическая система с резервированием;
- отказоустойчивая система;
- система высокой готовности.

От классической системы резервирования заимствованы следующие элементы:

1) использование в программно-аппаратном комплексе двух и более серверов, каждый из которых может иметь одну и более линии связи;
2) реализация механизма «главный сервер –> резервный сервер». Главный сервер отрабатывает пользовательские запросы к СУБД и хранит данные в рабочей БД;
3) резервный сервер (их может быть несколько) ведет копию БД главного сервера. Каждый резервный сервер непрерывно получает данные с главного сервера по протоколу TCP/IP об изменениях в его рабочей БД посредством утилиты `lhb`. В минимальной конфигурации должен быть хотя бы один резервный сервер;
4) управление запуском и остановом компонентов системы резервирования берет на себя специализированная управляющая программа `server`;

5) управляющая программа сервера резервирования при аварийном завершении внутренних процессов анализирует их коды завершения и, если возможно, осуществляет автоматический перезапуск процессов;
6) управляющие программы серверов системы резервирования обмениваются информацией друг с другом и отслеживают взаимное состояние серверов;
7) при отказе (или принудительном останове) главного сервера на одном из резервных серверов запускается СУБД ЛИНТЕР для работы с копией БД, полученной ранее с главного сервера. С этого момента данный сервер станет главным сервером;
8) запросы клиентских приложений к СУБД ЛИНТЕР всегда адресуются только главному серверу системы резервирования. При смене функции главного сервера переадресация запросов новому главному серверу выполняется автоматически;
9) в процессе работы может происходить смена ролей серверов системы резервирования по команде оператора либо в случае неисправности главного сервера или линий связи. Классическая система резервирования базируется, в свою очередь, на двух конфигурациях:
   - конфигурация избыточных аппаратных средств с горячим резервом (Hot Standby);
   - усовершенствованная классическая модель дублирования (Improved Classic Backup).

Усовершенствованная классическая модель дублирования предполагает, что:

- БД на резервном сервере содержит полную копию рабочей БД на момент запуска системы резервирования и накопленные изменения в БД на текущий момент;
- отсутствует аппаратное переключение с главного сервера на резервный;
- регулярно опрашивается состояние главного и резервного серверов с целью определения их работоспособности;
- опрос состояния серверов и каналов связи является функцией непрерывной готовности, которая обеспечивает минимизацию среднего времени до восстановления (ремонта) MTTR (mean time to repair);
- поддерживается восстановление готовности резервного сервера без прерывания работы информационной системы.

От архитектуры отказоустойчивых систем заимствованы следующие элементы:

- идентификация неисправных компонентов (программ) системы резервирования;
- перезапуск программ без прерывания работы системы резервирования;
- избыточность серверов.

Характеристики системы резервирования

Основные характеристики система резервирования СУБД ЛИНТЕР:

- главный и резервные серверы не привязаны жестко к своим компьютерам;
- в процессе работы системы резервирования может происходить смена ролей серверов по команде оператора либо автоматически при обнаружении неисправности главного сервера или линий связи;
- для того чтобы СУБД ЛИНТЕР полностью выполняла свои функции, достаточно работы главного сервера. Однако в этом случае надежность информационной системы снижается;

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

- резервный сервер может быть запущен в любое время после запуска главного сервера. Он «на ходу» получает копию БД с главного сервера и непрерывно поддерживает ее в состоянии немедленной готовности к использованию путем копирования изменений, происходящих в БД главного сервера. Доступ к БД при этом не приостанавливается (подключение незаметно для пользователей СУБД ЛИНТЕР);
- компьютер может выполнять функцию резервного сервера только тогда, когда работает главный сервер;
- при отказе главного сервера его функции берёт на себя резервный сервер (становится главным). При этом все запросы к СУБД ЛИНТЕР, выполняющиеся в данный момент, возвращаются клиентским приложениям с соответствующим кодом завершения;
- время переключения на резервный сервер равно сумме тайм-аута обнаружения аварии и времени запуска СУБД. Величина тайм-аута обнаружения аварии может назначаться администратором системы резервирования;
- резервный сервер перед началом копирования БД с главного сервера может делать архивную копию существующей на нем БД. Кроме того, он может делать архивные копии БД периодически или по расписанию, при этом архивная копия БД может тестироваться и проверяться на целостность;
- для ускорения приведения резервного сервера в состояние готовности используется смена назначения каталога БД. Может быть использовано два каталога, один из которых является рабочим, куда накапливаются изменения, а второй – резервным каталогом, в нем находится предыдущая рабочая БД. Рабочий и резервный каталоги могут изменять свою функциональность – резервный становится рабочим, а рабочий – резервным. В результате этого на резервном сервере почти всегда (за исключением первоначального старта системы) есть БД, на которой может быть запущено ядро СУБД ЛИНТЕР;
- на момент окончания копирования рабочей БД резервный сервер содержит полную копию рабочей БД главного сервера и накопленные за время копирования изменения в БД, хранящиеся в файлах системного журнала;
- поскольку суммарный размер файлов системного журнала на резервном сервере постоянно возрастает в объеме, на нем также работает ядро СУБД ЛИНТЕР в специальном режиме, которое переносит изменения из системного журнала в таблицы БД. После переноса изменений файлы системного журнала освобождаются. Дополнительно это сокращает время перехода резервного сервера в режим главного.

Программные компоненты

В системе резервирования задействованы общие программные компоненты СУБД ЛИНТЕР и специализированные компоненты, которые, собственно, управляют процессом резервирования.

Компоненты системы резервирования реализованы на основе интерфейса нижнего уровня СУБД ЛИНТЕР (call-интерфейса) и (как любое приложение) обращаются к БД через ядро СУБД ЛИНТЕР. Схема взаимодействия специализированных компонентов системы резервирования и компонентов СУБД ЛИНТЕР показана на рисунке 1. Общая схема взаимодействия программных компонентов, участвующих в резервировании, показана на рисунке 2.

Примечание
На рисунках 1, 2 штриховые линии обозначают связь процессов по запуску типа (родитель->сын), сплошные линии обозначают направление информационного обмена.

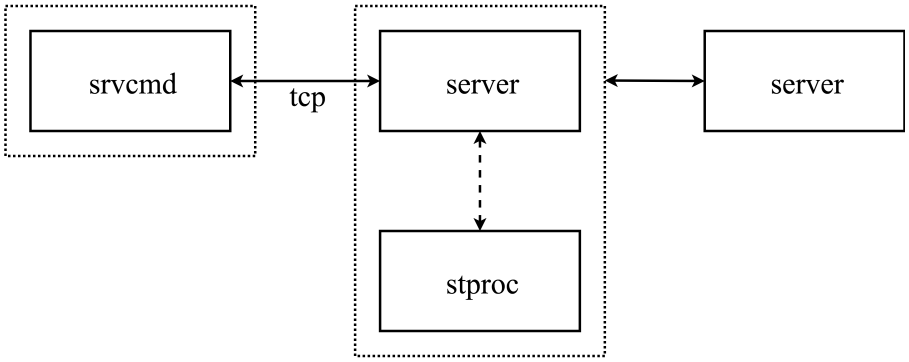


Рисунок 1. Структура специализированного программного обеспечения системы резервирования

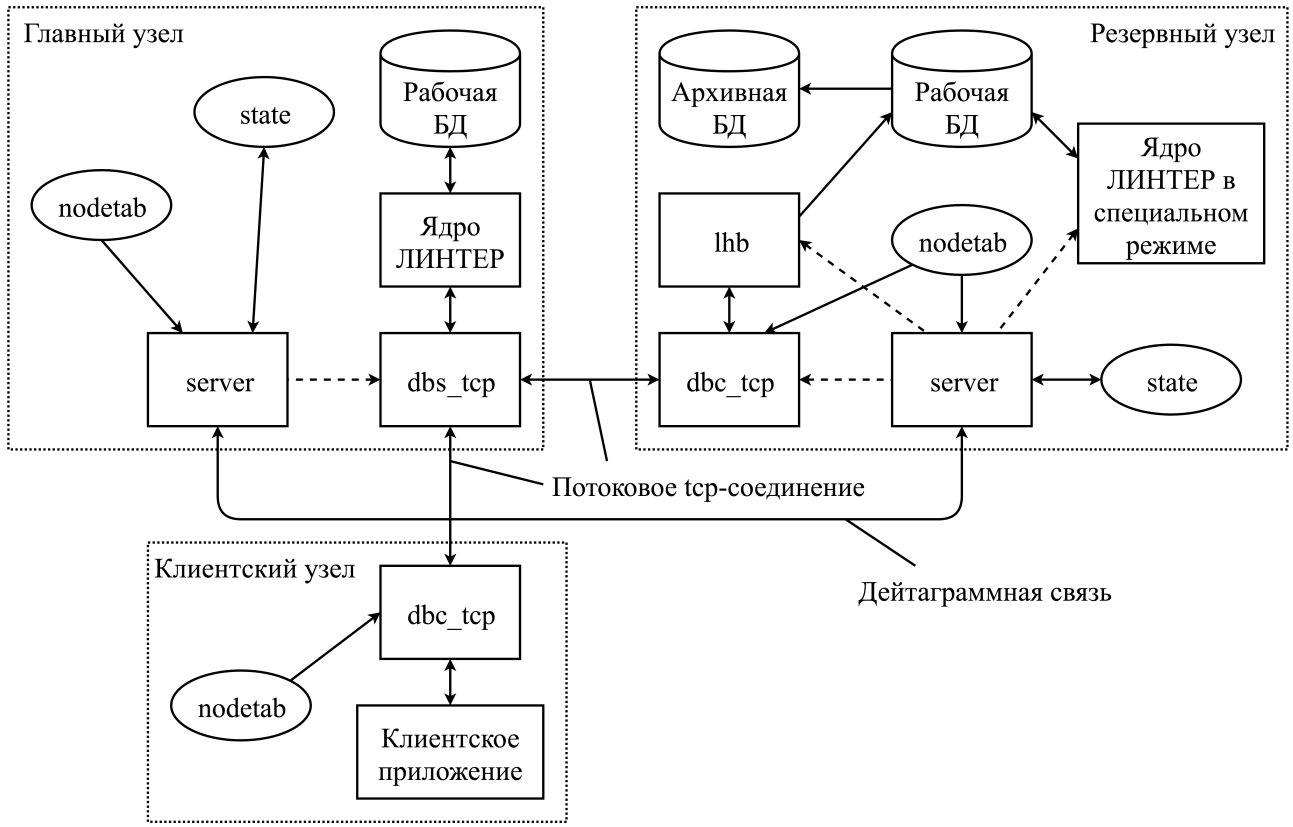


Рисунок 2. Схема взаимодействия СУБД ЛИНТЕР и системы резервирования

© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Администрирование системы резервирования

Установка системы резервирования

Система резервирования является частью СУБД ЛИНТЕР и устанавливается в процессе установки СУБД ЛИНТЕР (см. документ «Установка СУБД ЛИНТЕР в среде ОС Linux» или «Установка СУБД ЛИНТЕР в среде ОС Windows» в зависимости от ОС). Дополнительных действий по установке системы резервирования предпринимать не требуется.

Система резервирования состоит из управляющей программы (`server`), утилит удаленного управления (`srvcmd` или `hresctl`) и примера обработчика событий системы резервирования (`stproc`).

Примечание
Программа `srvcmd` работает напрямую с сетью, а программа `hresctl` использует hresapi.
В дистрибутиве СУБД ЛИНТЕР для ОС Windows поставляется только программа `hresctl`.

В процессе своей работы управляющая программа запускает следующие утилиты из состава СУБД ЛИНТЕР:

- `linter` – главная программа ядра СУБД ЛИНТЕР, осуществляющего собственно обработку SQL-запросов;

Примечание
Для обеспечения синхронного с системой резервирования сохранения данных к ключам запуска ядра СУБД ЛИНТЕР необходимо добавить ключ `/defcomm`.

- `sql`, `tsp`, `intsrt` – дополнительные программы ядра СУБД ЛИНТЕР. Запускаются программой `linter`;
- `dbc_tcp` – сетевой драйвер клиента. Используется для доступа к удаленной СУБД ЛИНТЕР;
- `dbs_tcp` – сетевой драйвер сервера. Используется для доступа удаленных клиентов к СУБД ЛИНТЕР;
- `lhb` – утилита архивирования данных. Используется для копирования данных с главного сервера на резервные, а также для управления контрольными точками;
- `testdb` – утилита проверки целостности БД.

Запуск системы резервирования

Система резервирования приводится в активное состояние путем запуска управляющей программы `server` на каждом из своих серверов. Поведение системы резервирования модифицируется путем задания ключей запуска управляющей программе. Управляющая программа может быть запущена либо как сервис, либо как приложение из командной строки оператором, а также из пользовательской программы.

Примечание
В случае запуска управляющей программы как сервиса ОС Windows при неправильной конфигурации возвращается код 186 (`invalid flag`). В случае получения такого кода

нужно проверять опции запуска и файл `nodetab`. Для выяснения подробностей смотреть файл `server.log` в каталоге базы данных.

Формат командной строки запуска:

```
server [<ключ> …]
<ключ>::= <формат 1> | <формат 2>
<формат 1>::= -<имя ключа> [{<пробел> | = } <значение ключа>]
<формат 2>::= /<имя ключа>[= <значение ключа]
```

Синтаксические правила:

- квадратные скобки указывают на необязательный элемент командной строки или ключа. Внутри ключа он может быть заменен пробелом;
- имена ключей регистронезависимы (можно задавать как большими, так и малыми буквами);
- название ключа предваряется символом "-" (минус) или "/";
- значение ключа может отделяться от имени ключа пробелом или символом "=";
- если признаком ключа является символ "/", то значение ключа должно быть отделено от имени символом "=" во избежание интерпретации аргумента, представляющего полный путь, как несуществующего ключа;

Допустимые написания ключей:

```
-<ключ> <значение>
-<ключ>=<значение>
/<ключ>=<значение>
```

Недопустимое написание ключа:

```
/<ключ> <значение>
```

- если название ключа указано неправильно, то управляющая программа сервера резервирования запущена не будет. Сообщение об ошибочном ключе будет выведено на консоль.

Ключи запуска управляющей программы системы резервирования

Ниже перечислены используемые утилитой ключи и их назначение.

Ключ	Назначение
Управление режимом запуска	
`/wait`	Продолжительность прослушивания сети
`/port`	Порт сервера резервирования
`/nq`	Автоматический запуск системы резервирования
`/wp`	Разрешение на запуск системы резервирования по сигналу
`/input`	Периодичность запросов интерактивного запуска
`/nservers`	Минимально допустимое количество серверов системы резервирования
`/nodaemon`	Запрет функционирования в фоновом режиме
`/local`	Запрет запуска сервисом

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Ключ	Назначение
/down	Допустимый интервал простоя системы резервирования
/prior	Приоритет сервера резервирования
/cmpsyslog	Выбор рабочей БД на основе системного журнала
/pf	Файл регистрационных данных администратора БД
/testslave	Предотвращение потери данных при останове системы резервирования
Управление выдачей справочной информацией	
/h или /?	Справочная информация о ключах
/version	Справочная информация о версии
/show	Предоставление информации о состоянии сервера и БД
Пути к объектам	
/ntab	Путь к файлу сетевой конфигурации
/pathtodb	Путь к каталогу рабочей БД
/pathtoarc	Путь к каталогу архивной БД
Управление состоянием	
/cu	Активация проверки полномочий пользователя для управления сервером
/setstate	Перевод рабочей БД в заданное состояние
/forceshut	Длительность ожидания завершения операций
Управление обработкой событий	
/stproc	Имя обработчика событий
/altstproc	Генерирование альтернативных кодов событий
/tclog	Генерирование событий, относящихся к контролю времени
/watchnet	Контроль за сетевыми событиями
/nostprocqueue	Отмена очередности обработки событий
Управление ресурсами	
/pool	Размер пула памяти ядра СУБД ЛИНТЕР
/spool	Размер пула сортировки СУБД ЛИНТЕР
Управление отладочной информацией	
/lintlog	Трассировка запросов к СУБД ЛИНТЕР
/debug	Протоколирование работы сервера резервирования
/log	Стандартное протоколирование
/pid	Протоколирование pid
Управление запуском программ	
/lintadd	Дополнительные ключи запуска СУБД ЛИНТЕР
/lretry	Ограничение повторных запусков ядра СУБД ЛИНТЕР
/pretry	Количество повторных запусков программ системы резервирования
/dbsctout	Задержка запуска сетевых драйверов
/cf	Файл конфигурации ключей запуска программы

Ключ	Назначение
/nocf	Запрет использования файла конфигурации запуска программы
Управление архивами	
/archivate	Создание архивного файла
/barc	Местоположение архивного файла
/mklhbarc	Дублирование архивируемых данных
/devenv	Работа с базами данных на многих устройствах
/crash	Разрешение на восстановление БД из архивного файла
/arcint	Периодичность копирования рабочей БД в архивный каталог
/exchdir	Автоматический обмен статуса каталогов БД резервного сервера
/diff	Копирование только отличий БД
/syslogcount	Управление размером системного журнала
/cmdarcadd	Шаблон командной строки для сохранения БД в архивном файле
/cmdarcext	Шаблон командной строки для восстановления БД из архивного файла
Сетевые настройки	
/testint	Посылка тестовых пакетов
/tstlimit	Критерий разрыва соединения
/dbtestint	Периодичность сохранения состояния баз данных серверов
Управление тестированием	
/testdb	Регулярность тестирования БД резервного сервера
/tdbadd	Дополнительные указания по тестированию БД
Управление слежением за процессами	
/treq	Тайм-аут и интервал проверки ядра СУБД ЛИНТЕР
/st	Облегченная проверка активности ядра СУБД ЛИНТЕР
/parent	Слежение за родительским процессом
/wd	Слежение за управляющей программой системы резервирования
/tcorrect	Скорость подстройки внутреннего времени системы резервирования
/sc	Управление запуском компонентов
/instsrv	Создание сервиса
/delsrv	Удаление сервиса

Управление режимом запуска

Продолжительность прослушивания сети (wait)

Синтаксис

```
/wait=<интервал>
<интервал>::=целочисленное положительное значение.
```

Значение <интервала> задается в секундах.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Описание

Задаёт интервал времени, в течение которого выполняется прослушивание локальной сети (процесс установления сетевого соединения между серверами резервирования) при начальном запуске системы резервирования.

Если в течение этого интервала создаются условия для автоматического запуска (все узлы активны, или количество запущенных серверов равно заданному в ключе `/nservers` количеству узлов) система стартует, не дожидаясь его истечения.

Если автоматический запуск не разрешен, то после истечения заданного интервала допускается запуск системы резервирования по разрешению администратора или по специальному сигналу управляющей программе `server`.

Если ключ не задан, значение по умолчанию равно 20 секунд.

Пример

```
-wait 25
```

Порт сервера резервирования (port)

Синтаксис

```
/port=<номер порта>
```

Описание

Задает номер UDP-порта сервера резервирования. Через этот порт сервер резервирования принимает пакеты обмена от других серверов. Заданный <номер порта> должен быть согласован с номером порта в файле сетевой конфигурации `nodetab`, используемом данным сервером.

Если ключ не задан, по умолчанию используется порт 1060.

Примечание
Если сервер находится в MONO или MAIN -режиме, то сетевой драйвер сервера `dbs_tcp` при работе по TCP/IP-протоколу использует порт, заданный в ключе `/port`.

Пример

```
/port=1062
```

Автоматический запуск системы резервирования (nq)

Синтаксис

```
/nq
```

Описание

Если после прослушивания сети (см. ключ /wait) не все узлы сети активны, то сервер резервирования должен иметь четкие инструкции по дальнейшей работе. Возможны следующие варианты:

- возложить решение о продолжении работы на администратора системы резервирования (он может разрешить или запретить запуск в неполном составе).

Для этого на консоль выдается запрос на запуск системы резервирования в неполном составе. Разрешение может быть дано в виде ответа на запрос администратором системы резервирования или в виде сигнала, посланного из какой-либо пользовательской программы;

- разрешить автоматический запуск системы резервирования в неполном составе. Для этого используется ключ `/nq`, который автоматически, без ожидания разрешения администратора, запускает систему резервирования в неполном составе активных узлов.

При использовании ключа `/nq` система резервирования запускается автоматически всегда (за исключением случая, когда БД кандидата на роль главного сервера находится в разрушенном (MAINCRASH или SLAVECRASH) состоянии, см. пункт «Вывод подтверждения запуска») либо в полном составе серверов (все узлы сети подтвердили активность до истечения указанного в ключе `/wait` интервала времени), либо в неполном составе (часть серверов или вообще ни один из них не подтвердили свою готовность к работе в течение указанного в ключе `/wait` интервала времени).

При необходимости безусловного старта без участия администратора необходимо использовать ключ `/wp` и подтверждать старт сигналом по тайм-ауту. Сигнал также может быть послан из обработчика событий при наступлении события `NOT_FOUND`.

Для исключения запроса на старт системы резервирования в случае необходимости копирования БД следует применять ключ исключения копирования `/exchdir`.

По умолчанию (ключ `/nq` не задан) управляющая программа всегда запрашивает разрешение на запуск системы резервирования при неактивности хотя бы одного узла системы резервирования или при необходимости восстановления на нем рабочей БД.

Примечание
Режим запуска системы резервирования должен быть согласован между серверами резервирования, то есть все управляющие программы должны запускаться либо с ключом `/nq` (автоматический запуск), либо все – без ключа `/nq` (ручной запуск).

Разрешение на запуск системы резервирования по сигналу (wp)

Синтаксис

```
/wp[=<номер сигнала>]
<номер сигнала>::=целочисленное положительное значение.
```

Описание

Разрешает запуск управляющей программы системы резервирования по сигналу.

Ключ предназначен для возможности выбора альтернативного (по отношению к интерактивному способу) варианта запуска системы резервирования. Посылка серверу резервирования `<номера сигнала>` эквивалентна интерактивному ответу, разрешающему запуск системы резервирования.

Если номер сигнала не задан, по умолчанию предполагается SIGINT. Если ключ не задан, то при получении сигнала производится действие по умолчанию для данного сигнала.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Пример

```
/wp=12
```

Периодичность запросов интерактивного запуска (input)

Синтаксис

```
/input=<интервал>
<интервал>::=положительное целочисленное значение или значение с
 десятичной точкой.
```

Значение <интервала> задается в секундах.

Описание

Задаёт периодичность повторения интерактивных запросов на разрешение запуска системы резервирования с неполным составом активных узлов сети (см. ключ /wait).

Если ключ не задан, по умолчанию запросы выводятся на консоль с периодичностью 20 секунд.

Если произошло изменение состояния предполагаемого MONO-сервера, то запрос выводится вне очереди.

Если одновременно задан ключ /nq, то данный ключ игнорируется.

Пример

```
/input=60
```

Минимально допустимое количество серверов системы резервирования (nservers)

Синтаксис

```
/nservers=<количество>
<количество>::=положительное целочисленное значение.
```

Описание

Задаёт минимальное количество активных (за время прослушивания сети) серверов системы резервирования, при котором разрешается её автоматический запуск.

Без этого ключа система резервирования стартует автоматически, если в течение интервала прослушивания сети, заданного в ключе /wait, все узлы подтвердили активность. Если же хотя бы одна линия связи неактивна, то по истечению интервала прослушивания выводится запрос на разрешение старта системы резервирования. Это будет происходить даже тогда, когда все серверы активны, но неисправна одна из запасных линий связи между серверами, поскольку в файле сетевой конфигурации nodetab нет указания на принадлежность линии связи серверу.

Чтобы в такой ситуации не запрашивать разрешение на запуск системы резервирования, можно обеспечить её автоматический запуск с помощью ключа /nservers. Как только в системе резервирования количество активных серверов достигнет заданного значения, произойдет автоматический старт системы резервирования. При этом остальные неактивные линии будут считаться запасными и могут быть подключены позднее.

Выбор значения ключа должен быть продуманным (соответствовать количеству компьютеров), иначе может произойти потеря данных.

Пример:

- в системе резервирования предусмотрено 3 сервера, но разрешен запуск при двух активных серверах;
- в локальной сети произошел сбой в питании;
- при перезапуске системы резервирования после сбоя питания бывший главный сервер стартует последним;
- в этом случае в качестве нового MONO-сервера выбирается один из серверов, бывших до сбоя питания в режиме резервного сервера;
- есть вероятность, что до сбоя питания не все данные были переданы с главного сервера на этот резервный сервер.

В такой ситуации непереданные данные будут потеряны.

Если за время прослушивания сети необходимого количества активных серверов не набралось, то на консоль будет выдан запрос на запуск системы резервирования.

Если совместно с ключом `/nservers` задан ключ `/nq`, то система резервирования запускается по правилам ключа `/nq`.

Пример

```
/nserves=4
```

Запрет функционирования в фоновом режиме (nodaemon)

Синтаксис

```
/nodaemon
```

Описание

В ОС Linux, ЗОСРВ Нейтрино:

- задаёт режим работы сервера резервирования без отключения от терминала;
- при задании ключа сервер после определения своего состояния не переходит в фоновый режим;
- по умолчанию сервер отсоединяется от управляющего терминала и переходит в фоновый режим работы после принятия решения о функциональном назначении каждого сервера.

В ОС типа Windows:

- программа не будет пытаться запускаться как сервис. Будет принудительно осуществлен запуск как приложения.

Пример

```
/nodaemon
```

Запрет запуска сервисом (local)

Синтаксис

```
/local
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Описание

Примечание
Ключ действует только в ОС семейства Windows.

Программа не будет пытаться запускаться как сервис. Будет принудительно осуществлен запуск как приложения.

Пример

```
/local
```

Допустимый интервал простоя системы резервирования (down)

Синтаксис

```
/down=<интервал>
<интервал>::=положительное целочисленное значение или значение с
 десятичной точкой.
```

Значение <интервала> задается в минутах.

Описание

Задаёт максимально допустимое время простоя (нахождение в выключенном состоянии) сервера в минутах, при котором управляющая программа системы резервирования при последующем старте не выдает на консоль интерактивный запрос на старт системы резервирования.

Если от момента останова управляющей программы до ее перезапуска прошло меньше времени, чем указано в ключе, и сервер является старшим в группе или единственным, то он инициирует автоматический запуск системы резервирования (аналогично ключу /nq), в противном случае выдается интерактивный запрос на разрешение запуска системы резервирования.

Если ключ не задан, по умолчанию допустимый интервал простоя равен 1 минутах.

Пример

```
/down=1.5
```

Приоритет сервера резервирования (prior)

Синтаксис

```
/prior=<приоритет>
<приоритет>::=символьная строка.
```

Описание

Задаёт приоритет сервера резервирования.

Значение приоритета задаётся в виде произвольной символьной строки длиной не более 16 символов. Иерархия приоритетов определяется на основе лексикографического сравнения символьных строк.

Приоритет сервера учитывается в следующих ситуациях:

- при автоматическом запуске системы резервирования. Сервер с большим приоритетом объявляет себя старшим и определяет статусы каждого сервера системы резервирования (ключ `/nq`);
- при выборе главного сервера системы резервирования. В случае равных остальных условий главным сервером становится сервер с большим приоритетом.

По умолчанию приоритет сервера выбирается случайно. Для исключения совпадений часть его формируется на основе IP-адреса сервера.

Примеры

```
/prior=RS
/prior=F
-prior=RTF
-prior=12345
-prior 12wq5
```

Выбор рабочей БД на основе системного журнала (cmpsyslog)

Синтаксис

```
/cmpsyslog
```

Описание

Задаёт специальный критерий выбора главного сервера при запуске системы резервирования (во время процедуры выбора главного сервера).

По умолчанию в качестве критерия выбора главного сервера используется дата последней работы с БД, а именно: главным сервером становится сервер с самой «свежей» БД.

При задании ключа `/cmpsyslog` критерием выбора главного сервера будет не дата работы с БД, а положение последней записанной в системный журнал этой БД отметки, то есть количество добавленных или измененных данных, поскольку модификация данных находит свое отражение в системном журнале. Главным будет выбран сервер с наибольшим количеством данных в системном журнале БД.

Ключ `/cmpsyslog` является симметричным, то есть должен задаваться одновременно для любой возможной пары сравниваемых серверов, ибо, если для разных серверов заданы различные критерии определения рабочей БД, то выбор главного сервера может стать невозможным. Если все же возникла такая ситуация, то на консоль будет выдано соответствующее предупреждение.

Если при первоначальном запуске в качестве резервного сервера подключается компьютер, содержащий файлы БД ЛИНТЕР, то перед запуском сервера резервирования с ключом `/cmpsyslog` необходимо вручную удалить эти файлы из рабочих и архивных каталогов БД.

Если для обоих сравниваемых серверов состояние выбранных на них баз данных - MONO (то есть оба сервера заканчивали работу без синхронизации друг с другом), то сравнение журнальных адресов баз данных на этих серверах не выполняется.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Файл регистрационных данных администратора БД (pf)

Синтаксис

```
/pf=<спецификация файла>
```

Описание

Задаёт спецификацию текстового файла, в котором хранятся имя администратора БД и его пароль. Используется в случае, когда имя и пароль администратора БД отличны от используемых по умолчанию значений SYSTEM и MANAGER8 соответственно.

Формат файла:

```
USER=<имя пользователя>
PASSWORD=<пароль пользователя>
```

Примечание
Правила определения переменных должны подчиняться правилам определения переменных командного файла соответствующей ОС.

Значение переменной может быть заключено в кавычки. В одной строке может быть инициализирована только одна переменная.

Примеры файла:

```
USER=adm
PASSWORD=7cw@1
USER="Админ"
PASSWORD="гуру"
```

Примечание
Следует максимально защитить этот файл средствами ОС от доступа к нему посторонних лиц.

Пример

```
/pf=/usr/linter/account.txt
```

Предотвращение потери данных при останове системы резервирования (testslave)

Синтаксис

```
/testslave
```

Описание

Запрещает завершать работу системы резервирования в случае возможной потери данных.

Если главный сервер запущен с этим ключом, то при получении из сети команды или управляющего сигнала на завершение своей работы, он проверяет полную готовность всех резервных серверов (а именно, завершение копирования данных из рабочей БД

главного сервера на резервный сервер). Если хотя бы один из резервных серверов не завершил копирование данных с главного, останов главного сервера не производится. На резервном сервере при получении сигнала на завершение системы резервирования остановка в этом случае откладывается (сигнал не игнорируется).

Отключение запуска ядра СУБД в безопасном режиме (unsafe_mode)

Примечание
Поддерживается со сборки 6.0.17.95.

Синтаксис

`/unsafe_mode`

Описание

При запуске с данным ключом сервер резервирования будет запускать подчиненный процесс ядра СУБД в обычном режиме, безопасный режим запуска ядра использован не будет.

Ключ работы с защищенной базой (cryptadd)

Примечание
Поддерживается со сборки 6.0.20.3.

Синтаксис

`/cryptadd`

Описание

Ключ для работы с защищенной базой, с помощью которого возможно задать дополнительный ключ для запуска ядра, `testdb` и `lhb`.

Первый символ в значении ключа /cryptadd может быть "/" или "-" (для ядра в командную строку подставляется "/", для остальных "-").

Пример

/CRYPTADD=/passfile=/usr/linter/pass.des.txt

Управление выдачей справочной информацией

Справочная информация о ключах (h)

Синтаксис

`/h или /?`

Описание

Вывод на консоль краткой информации о ключах управляющей программы `server`. По окончании вывода управляющая программа завершает свою работу.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Справочная информация о версии (version)

Синтаксис

```
/version
```

Описание

Вывод на консоль информации о версии программы `server` и продукта в целом. По окончании вывода управляющая программа завершает свою работу.

Предоставление информации о состоянии сервера и БД (show)

Синтаксис

```
/show
```

Описание

Задаёт вывод на консоль терминала информацию о состоянии сервера резервирования. После вывода информации управляющая программа `server` завершает свою работу. Состояние сервера резервирования в результате запуска не меняется (даже если на нем работает другая ранее запущенная управляющая программа `server`).

Выводится следующая информация:

- путь к файлу сетевой конфигурации `nodetab`;
- время, прошедшее после последней записи отметки о состоянии БД сервера;
- текущее системное время;
- путь к архивному файлу архивной БД и время его создания (при наличии архива);
- полный путь к рабочей БД;
- последнее состояние рабочей БД;
- наличие рабочей БД;
- время [:адрес_системного_журнала] рабочей БД;
- полный путь к архивной БД;
- последнее состояние архивной БД;
- наличие архивной БД;
- время [:адрес_системного_журнала] архивной БД;
- начальные (стартовые) размеры архивной и рабочей БД;
- текущее состояние сервера;
- информация об активности сервера.

Примечания

1. Если сервер резервирования работает в режиме обмена каталогов (задан ключ `/exchdir`), то для правильного определения статуса каталогов БД (рабочий – архивный) программа `server` должна запускаться с парой ключей: `/show`, `/exchdir`.
2. Отображаемая информация о состоянии сервера может меняться в зависимости от версии системы резервирования.

Пример

Пусть сервер резервирования находится в состоянии главного сервера (запущена управляющая программа `server` только на этом компьютере).

Запускаем на компьютере еще один экземпляр управляющей программы `server` с ключом `show`: `server /show /exchdir`.

Этот экземпляр программы выдает информацию о состоянии сервера и завершает свою работу.

Ранее запущенная управляющая программа системы резервирования продолжает работу в прежнем состоянии.

Пути к объектам

Путь к файлу сетевой конфигурации (ntab)

Синтаксис

```
/ntab=<спецификация файла>
```

Описание

`<Спецификация файла>` задает местоположение файла сетевой конфигурации `nodetab` на файловой системе. В файле `nodetab` должны быть описаны все узлы системы резервирования (см. документ «Сетевые средства»). Если ключ не задан, то поиск файла `nodetab` выполняется в следующей последовательности:

1) в каталоге, заданном переменной окружения `SY00`;
2) в каталоге, заданном переменной окружения `SERVER_HOME`;
3) в текущем каталоге;
4) в домашнем каталоге;
5) в каталоге размещения исполняемого файла программы `server`;
6) в каталогах, заданных переменной окружения `PATH`.

Примеры

```
/ntab=/usr/linter/bin/nodetab
-ntab=/usr/linter/bin/node_res.cfg
-ntab='/home/server/program fies/linter/bin/nodetab'
```

Путь к каталогу рабочей БД (pathtodb)

Синтаксис

```
/pathtodb=<спецификация каталога>
```

Описание

`<Спецификация каталога>` задает путь к каталогу рабочей БД (см. раздел «Основные понятия», понятие Рабочая БД). Если ключ не задан, то для рабочей БД назначается каталог, определяемый переменной окружения `SY00`, или текущий каталог.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Примечание
В случае использования ключа `/exchdir` определяется один из двух используемых каталогов. Назначение каталога определяется автоматически.

Примеры

```
/pathtodb=/usr/linter/bin/db
-pathtodb='/home/server/program files/linter/bin/db'
```

Путь к каталогу архивной БД (pathtoarc)

Синтаксис

```
/pathtoarc=<спецификация каталога>
```

Описание

`<Спецификация каталога>` задает путь к каталогу архивной БД (см. раздел «Основные понятия», понятие Архивная БД).

Если ключ не задан, то каталогом архивной БД назначается каталог, определяемый переменной окружения ARJPT, или подкаталог `ARC` каталога рабочей БД.

Примечание
В случае использования ключа `/exchdir` определяется второй из двух используемых каталогов (первый задается ключом `/pathtodb`). Назначение каталога определяется автоматически.

Примеры

```
/pathtoarc=/usr/linter/db/arcdb
-pathtoarc '/home/server/program files/linter/bin/arcdb'
```

Управление состоянием

Активация проверки полномочий пользователя для управления сервером (cu)

Синтаксис

```
/cu
```

Описание

Разрешает проверку полномочий пользователя при удаленном управлении сервером (см. также ключ `/u` утилиты hresctl). Если ключ не задан, то проверка полномочий не выполняется.

Для возможности управления сервером резервирования пользователь должен быть владельцем БД.

Без этого ключа проверка полномочий не производится, поэтому указание в командной строке утилиты `hresctl` имени пользователя и пароля излишне.

Перевод рабочей БД в заданное состояние (setstate)

Синтаксис

```
/setstate=<состояние>
<состояние>::=SLAVE|MONO|MAIN|MAINCRASH|SLAVECRASH
              |MAINFAILER|SLAVEFAILER
```

Описание

Устанавливает заданное состояние рабочей БД сервера резервирования. Новое состояние запоминается в файле `state` (находится в каталоге рабочей БД), после чего сервер завершает свою работу. Информация о дате БД не изменяется.

Ключ может быть использован для изменения выбора главного сервера до старта системы резервирования (БД в CRASH-состоянии не может быть выбрана в качестве рабочей БД).

Примечание

1. Примечание. Если задано состояние, отличное от перечисленных в ключе, то ошибка не диагностируется, однако поведение системы резервирования будет не определено.
2. Использование данного ключа не рекомендуется при работающем экземпляре управляющей программы.
3. Наименования состояний регистрозависимы. Они должны быть написаны прописными буквами.

Пример

```
/setstate=SLAVE
```

Длительность ожидания завершения операций (forceshut)

Синтаксис

```
/forceshut=<интервал>
<интервал>::=целочисленное положительное значение.
```

Значение `<интервала>` задается в секундах.

Описание

Задаёт интервал ожидания завершения:

- работы ядра СУБД ЛИНТЕР.

  После отсылки управляющей программой главного сервера системы резервирования команды на останов работы ядра СУБД ЛИНТЕР происходит ожидание завершения останова в течение заданного `<интервала>` времени. Если в течение этого времени ядро не завершило свою работу, управляющая программа главного сервера выполняет останов ядра принудительно (сигналом KILL).

- обновления БД на резервном сервере.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

После получения резервным сервером команды останова или обмена состояния происходит ожидание в течение заданного <интервала> завершения работы утилиты lhb, выполняющей обновление («докачку») данных. Если в течение этого времени обновление БД не закончилось, управляющая программа резервного сервера выполняет останов утилиты lhb принудительно.

По умолчанию длительность ожидания завершения указанных процессов равна 600 секундам.

Примечание
Значение ключа не может быть меньше 5 секунд.

Пример

```
-forceshut 3600
```

Управление обработкой событий

Имя обработчика событий (stproc)

Синтаксис

```
/stproc[=<имя программы>]
```

Описание

Задает путь к программе-обработчику событий сервера резервирования. Обработчик событий автоматически вызывается при наступлении каждого события (см. раздел «Основные понятия», понятие События в системе резервирования). Все события ставятся в очередь в порядке их возникновения. Вызов обработчика событий для обработки следующего события происходит только после завершения обработки предыдущего события (однако администратор системы резервирования может отменить очередность обработки событий с помощью ключа /nostprocquery).

Обработчику событий в качестве аргументов передаются код состояния сервера резервирования в момент наступления события, а также дополнительная информация о событии.

Анализируя события системы резервирования, обработчик событий может инициировать дополнительные действия, связанные со сменой состояния сервера резервирования или остановом/запуском на нем определенных программ. Например, если предполагается поддерживать несколько версий архивных файлов БД, то в процессе обработки события «Программа архивирования создала архив БД» можно переименовать созданный со стандартным именем архивный файл с добавлением к имени файла даты его создания и удалить устаревшие версии, кроме трех последних.

Поиск программы-обработчика событий производится в текущем каталоге запуска сервера резервирования либо в каталогах, перечисленных в переменной окружения PATH. Также может быть указан полный путь к программе-обработчику.

Если значение ключа не задано, предполагаемое имя программы-обработчика – stproc.

Если ключ не задан, обработчик событий не запускается.

Примечание
Для распознавания некоторых событий можно использовать альтернативные коды событий (это упрощает анализ таких событий). Альтернативные коды событий генерируются и, соответственно, передаются обработчику событий только при наличии дополнительных ключей (см. столбец «Дополнительные условия» в таблице 1).

Пример

```
/stproc handler
```

Генерирование альтернативных кодов событий (altstproc)

Синтаксис

```
/altstproc
```

Описание

Заставляет генерировать альтернативные коды событий (см. раздел «Основные понятия», понятие События в системе резервирования).

При задании ключа дополнительно генерируются события, относящиеся к процессам системы резервирования (запуск/останов/завершение работы процессов).

Ключ работает только в паре с ключом `/stproc`.

Примечание
События, относящиеся к состоянию процессов на сервере резервирования, генерируются и без ключа `/altstproc`. Однако в этом случае обработка события затрудняется тем, что для его детализации необходимо анализировать дополнительные параметры. Например, при запуске процесса – имя процесса, при останове процесса – имя процесса и код завершения. При использовании ключа `/altstproc` дифференциацию событий выполняет сервер резервирования, что позволяет более простыми способами выполнять их обработку (например, отделять события, связанные со сменой состояния сервера, от других событий).

Генерирование событий, относящихся к контролю времени (tclog)

Синтаксис

```
/tclog
```

Описание

Генерирование событий, относящихся к контролю времени.

При задании этого ключа сервер резервирования дополнительно генерирует события `E_SERVER_TIME_DIFF` и `E_TIME_CHANGE`.

Событие `E_SERVER_TIME_DIFF` (код 41) информирует о разности времени данного сервера и удаленных серверов (см. пункт Событие E_SERVER_TIME_DIFF).

Событие `E_TIME_CHANGE` (код 40) возникает в случае резкой подстройки времени на данном компьютере (см. пункт Событие E_TIME_CHANGE).

© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Ключ работает только одновременно с ключами `/stproc` и `/altstproc`.

Примечание
Оба указанных события могут применяться в целях отладки.

Пример

```
/stproc handler /altstproc /tclog
```

Контроль за сетевыми событиями (watchnet)

Синтаксис

```
/watchnet
```

Описание

Заставляет генерировать дополнительные коды сетевых событий (см. пункт Событие E_NET_INFO).

Отмена очередности обработки событий (nostprocqueue)

Синтаксис

```
/nostprocqueue
```

Описание

Отменяет режим очередности обработки событий.

По умолчанию все события передаются программе-обработчику событий (см. ключ /stproc) в порядке их возникновения, при этом обработка очередного события возможна только после завершения обработки предшествующего события. Это потенциально может привести к увеличению числа процессов программы `server`, ожидающих своей очереди на запуск обработчика (`exec`).

Если обработчик событий по каким-то причинам будет выполняться достаточно долго, есть вероятность переполнения очереди ожидающих запуска процессов и исчерпания ресурсов ОС.

Для исключения подобной ситуации необходимо использовать ключ `/nostprocqueue`. В этом случае каждый следующий процесс обработки события не станет ожидать окончания предыдущего, а будет запущен сразу же при наступлении события. Как следствие отсутствия очереди может наблюдаться изменение логической последовательности обработки близких по времени событий.

Управление ресурсами

Размер пула памяти ядра СУБД ЛИНТЕР (pool)

Синтаксис

```
/pool=<количество страниц>
```

```
<количество страниц>::=целочисленное положительное значение.
```

Одна страница равна 4 Кбайт.

Описание

Задаёт размер пула памяти ядра СУБД ЛИНТЕР (см. документ «Запуск и останов СУБД ЛИНТЕР в среде ОС Windows», «Запуск и останов СУБД ЛИНТЕР в среде ОС Linux»). Значение ключа подставляется в командную строку запуска ядра СУБД ЛИНТЕР.

Если ключ не задан, по умолчанию пул памяти равен 100000 страницам.

Примечание
Поддерживается со сборки 6.0.17.95.

Примечание
Рекомендуется устанавливать требуемое значение pool, так как для конкретной задачи может быть недостаточен размер, устанавливаемый по умолчанию.

Пример

```
/pool=100000
```

Размер пула сортировки СУБД ЛИНТЕР (spool)

Синтаксис

```
/spool=<количество страниц>
<количество страниц>::=целочисленное положительное значение.
```

Одна страница равна 4 Кбайт.

Описание

Задаёт размер пула сортировки ядра СУБД ЛИНТЕР (см. документ «Запуск и останов СУБД ЛИНТЕР в среде ОС Windows», «Запуск и останов СУБД ЛИНТЕР в среде ОС Linux»). Значение ключа подставляется в командную строку запуска ядра СУБД ЛИНТЕР.

Если ключ не задан, по умолчанию пул сортировки равен 25000 страницам.

Примечание
Поддерживается со сборки 6.0.17.95.

Примечание
Рекомендуется устанавливать требуемое значение spool, так как для конкретной задачи может быть недостаточен размер, устанавливаемый по умолчанию.

Пример

```
/spool=25000
```


© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Управление отладочной информацией

Трассировка запросов к СУБД ЛИНТЕР (lintlog)

Синтаксис

```
/lintlog
```

Описание

Заставляет выполнять трассировку запросов к ядру СУБД ЛИНТЕР. При указании данного ключа в командную строку запуска ядра СУБД ЛИНТЕР подставляется ключ `/LOGALL`, что вызывает запись текстов запросов в файл `LINTER.LOG` в каталоге рабочей БД (см. документ «Запуск и останов СУБД ЛИНТЕР в среде ОС Windows», «Запуск и останов СУБД ЛИНТЕР в среде ОС Linux»).

По умолчанию ядро СУБД ЛИНТЕР стартует без трассировки запросов.

Протоколирование работы сервера резервирования (debug)

Синтаксис

```
/debug[=[<уровень>][:[|<программа>]
 <размер>[x<количество>]]<спецификация файла>|stdout|stderr]
<уровень>::=<числовое значение>|<символьное значение>
<числовое значение>::=<маска уровней>|<номер уровня>[+<номер
 уровня>…]
<номер уровня>::= 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 |
 1024 | 2048 | 4096 | 8192 | 16384 | 32768 | 65536
<символьное значение>::=<имя уровня>[,< имя уровня >…]
<имя уровня>::= "TRACE" | "ERROR"| "MSG" | "STATE" | "STPROC" |
 "INFO" | "INIT" | "EVENT" | "SIG" | "TIMER" | "PROC" | "REPEATE"
 | "USR" | "FILE" | "NET" | "NET_SEND" | "DBG"
<маска уровней>::=результат сложения номеров уровней. Может быть
 представлена в шестнадцатеричном виде.
```

Описание

Задаёт параметры трассировочной информации и возможные способы её обработки.

Уровни трассировки можно задавать в числовом или символьном виде.

Доступны следующие уровни:

Имя уровня	Номер уровня (десятичное значение)	Трассировочная информация
TRACE	1	о вызываемых функциях
ERROR	2	об ошибках
MSG	4	о ходе выполнения
STATE	8	о смене состояний
STRPROC	16	о запуске программы-обработчика событий

Имя уровня	Номер уровня (десятичное значение)	Трассировочная информация
INFO	32	о нештатных ситуациях, которые не привели к прекращению работы системы резервирования
INIT	64	при инициализации
EVENT	128	о событиях
SIG	256	о манипуляциях с сигналами
TIMER	512	о тайм-аутах
PROC	1024	о запуске и останове программ
REPEATE	2048	о периодическом добавлении записи в STATE-файл
USR	4096	о дополнительных событиях
FILE	8192	о работе с файлами (запись, копирование, удаление и т. д.)
NET	16384	о сетевом обмене
NET_SEND	32768	об отсылке сообщений
DBG	65536	отладочная трассировка

Числовые значения можно задавать в десятичном или шестнадцатеричном виде. При указании числового значения каждый установленный бит соответствует включенному уровню трассировки. Например, чтобы задать уровни трассировки TRACE, MSG, STATE в числовом виде, надо указывать число 1+4+8 = 1310 (158, 11012), и ключ, соответственно, будет иметь вид `/debug=13`. В символьном виде эти же уровни задаются ключом `/debug=TRACE,MSG,STATE`.

Примеры задания уровней трассировки:

```
-debug=2
-debug=0xFFFFFFFF
-debug=2+512+64
-debug=0x10+0x400
-debug=32+0x654
-debug=TRACE
-debug=TRACE,info,dbg
-debug="TRACE,NET,proc"
```

<Спецификация файла> задаёт полный путь и имя файла, в который будет записываться трассировочная информация. Если спецификация файла не обозначена, то по умолчанию используется файл `server.log` в каталоге, указанном переменной окружения `SERVER_HOME` или в рабочем каталоге. Если указанный файл не существует, он будет создан. Запись в файл выполняется до исчерпания ресурсов файловой системы или до достижения максимально допустимого размера файла. По умолчанию размер файла равен 1048576 байт. Этот размер можно изменить с помощью аргумента <размер> (задается в байтах).

После достижения файлом максимального размера будут создаваться новые трассировочные файлы с именами вида <исходное имя>1, <исходное имя>2 и т.д., например: `server.log`, `server.log1`, `server.log2`.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Необходимое количество одновременно хранящихся трассировочных файлов можно задать сразу после размера файла и символа "x" в опции <количество>.

Если <количество> не указано, по умолчанию кроме файла с заданным именем на файловой системе будет храниться 10 файлов с трассировочной информацией.

В случае превышения максимального количества файлов трассировки при создании нового файла последний из файлов (server.log10) удаляется, файл server.log9 переименовывается в server.log10 и т.д. Файл server.log переименовывается в server.log1.

Пример.

```
-debug=TRACE:200/usr/linter/debug/server_db1.log
-debug=:200x15/usr/linter/debug/server_db1.log
```

Если в качестве имени файла указать стандартные файлы вывода stdout или stderr без размера, то поток будет перенаправлен в стандартный поток вывода или ошибок соответственно.

Если необходимо только управление перенаправлением вывода, перечисление уровней трассировки можно опустить и значение ключа начинать сразу с символа ":". При этом уровни будут соответствовать значениям по умолчанию.

Если после символа ":" сразу следует символ "|", то следующая за ним строка будет восприниматься как пользовательская программа с аргументами, которую надо запустить при старте сервера резервирования и передать ей на вход поток трассировочной информации. Эта программа может использоваться, в частности, для разбивки потока на файлы и/или фильтрации некоторых сообщений.

Пример.

```
Поток трассировки передан на вход программы awk, которая
обрабатывает данные в соответствии с внутренним алгоритмом:
server -debug=":|awk -f program.awk"
```

Пример программы фильтрации трассировки приведен в приложении 1.

Умолчания:

- на стандартный поток ошибок выводятся только сообщения об ошибках (если ключ не задан);
- все уровни трассировки, осуществляющие периодическую запись, выключены, в log-файл помещается информация только о событиях системы резервирования;
- включены уровни MSG, EVENT, INIT, ERROR, PROC, TIMER, USR, NET_SEND, STATE. Этой информации обычно достаточно для отслеживания поведения сервера;
- трассировочный поток сервера резервирования направляется в файл server.log, который ищется в каталоге, указанном в переменной окружения SERVER_HOME;
- количество одновременно хранящихся трассировочных файлов равно 10;
- размер файла трассировки 1048576 байт.

Таким образом, значение ключа /debug по умолчанию:

```
MSG,INFO,INIT,ERROR,STATE,EVENT,SIG,TIMER,PROC,USR,DBG,STPROC
```

Формат трассировочных записей

Трассировочные записи имеют следующий формат:

```
<источник трассировки> <детализация трассировки> <дата/время
 трассировки> <трассировочная информация>
```

где:

```
<источник трассировки>::=HOTRES:<уровень трассировки>
<детализация трассировки>::=<имя исходного модуля>:<номер строки>.
<дата/время трассировки>::=ddmm:hh:mi:ss.ff
<трассировочная информация>::=символьная строка.
```

Примеры трассировочных записей:

```
HOTRES:INFO init.c:503 0306:10:10:54.945223 Can't open STATE
 file ./STATE, errno = 2, No such file or directory

HOTRES:TRACE init.c: 504 0306:10:10:54.945223 'ReadStateFromFile'
 Leave fopen

HOTRES: INFO init.c:1909 0306:10:10:54.945223 Error while reading
 STATE file of work DB. State set to UNDEFINED.

HOTRES: TRACE states.c:34 0306:10:10:54.945223 'GetStateByName'
 Call from init.c:1913

HOTRES: DBG states.c:  36 0306:10:10:54.945223 Get state with name
 UNDEFINED

HOTRES: TRACE states.c:  45 0306:10:10:54.945223 'GetStateByName'
 Leave UNDEFINED

HOTRES: TRACE backup.c:  87 0306:10:10:54.945223 'DbExist' Call
 from init.c:1916

HOTRES: TRACE backup.c: 100 0306:10:10:54.945223 'DbExist' Leave
 Db not found.

HOTRES: TRACE init.c: 499 0306:10:10:54.945223 'ReadStateFromFile'
 Call from init.c:1947

HOTRES: INFO init.c: 503 0306:10:10:54.945223 Can't open STATE
 file ./ARC/STATE, errno = 2, No such file or directory

HOTRES: TRACE init.c: 504 0306:10:10:54.945223 'ReadStateFromFile'
 Leave fopen

HOTRES: INFO init.c:1949 0306:10:10:54.945223 Error while reading
 STATE file of backup DB. State set to UNDEFINED.
```


© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

```
HOTRES: TRACE states.c:34 0306:10:10:54.945223 'GetStateByName'
 Call from init.c:1953

HOTRES: DBG states.c:36 0306:10:10:54.945223 Get state with name
 UNDEFINED

HOTRES: TRACE states.c:45 0306:10:10:54.945223 'GetStateByName'
 Leave UNDEFINED

HOTRES: DBG init.c:2011 0306:10:10:54.945223
 FullPreviousState.State = UNDEFINED

HOTRES: ERROR init.c:2031 0306:10:10:54.945223 nodetab file is not
 found.

HOTRES: TRACE init.c:2032 0306:10:10:54.945223 'Init' Leave FAILED

HOTRES: ERROR server.c:91 0306:10:10:54.945223 Init failed. Exit.

HOTRES: TRACE server.c:93 0306:10:10:54.945223 'server' Leave
 FAILED
```

Стандартное протоколирование (log)

Синтаксис

`/log`

Описание

Задаёт протоколирование работы сервера резервирования со средним уровнем трассировки в файл `server.log`, создаваемый в каталоге, определяемом переменной окружения `SERVER_HOME`.

Фактически данный ключ является специальной версией ключа `/debug`.

Установки трассировки для ключа `/log`:

```
MSG,INFO,INIT,ERROR,STATE,STPROC
```

.

Протоколирование pid (pid)

Синтаксис

`/pid=<спецификация файла>`

Описание

Заставляет управляющую программу системы резервирования записывать свой pid в указанный файл.

Примечание
После перехода управляющей программы системы резервирования в фоновый режим работы ее pid будет изменен и заново записан в указанный файл. Сервер переключается в фоновый режим работы перед началом перехода из UNDEFINED-состояния в SLAVE или MONO-состояние.

Пример

```
/pid=/tmp/serverpid.txt
```

Управление запуском программ

Дополнительные ключи запуска СУБД ЛИНТЕР (lintadd)

Синтаксис

```
/lintadd=<ключ>|'<список ключей>'
<список ключей>::=<ключ>[<пробел><ключ>…]
<ключ>::=символьная строка.
```

Символьную строку необходимо заключать в кавычки, если она содержит пробелы.

Описание

Задаёт дополнительные ключи запуска ядра СУБД ЛИНТЕР. Значение ключа `/lintadd` подставляется в конец командной строки запуска ядра СУБД ЛИНТЕР. Возможные дополнительные ключи см. в документе «Запуск и останов СУБД ЛИНТЕР в среде ОС Windows», «Запуск и останов СУБД ЛИНТЕР в среде ОС Linux».

Логическая противоречивость ключей не проверяется.

Если добавляемый ключ дублирует заданный в командной строке запуска сервера резервирования ключ (например, в командной строке задан ключ `/pool` и дополнительно задан также `/pool`), то последний ключ в сформированной командной строке запуска подавляет все предыдущие аналогичные ключи.

Примеры

```
-lintadd=/nooutput
/lintadd="/nolog /notsp"
```

Ограничение повторных запусков ядра СУБД ЛИНТЕР (lretry)

Синтаксис

```
/lretry=<количество перезапусков>
<количество перезапусков>::=целое положительное число.
```

Описание

Задаёт допустимое количество повторных неудачных запусков ядра СУБД ЛИНТЕР. Неудачным считается запуск, если программа завершилась не по команде сервера резервирования, а самостоятельно. По истечении контрольного интервала времени

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

после последнего неудачного запуска счетчик неудачных запусков сбрасывается. Для ядра СУБД ЛИНТЕР контрольный интервал времени равен величине интервала тестирования (см. описание ключа /treq), умноженной на 10; значение по умолчанию 50 секунд. При превышении количества повторных запусков рабочая БД сервера переходит в состояние SLAVEFAILER, MAINFAILER, MAINCRASH или SLAVECRASH, и управляющая программа системы резервирования завершает свою работу. Переход в состояние MAINCRASH, MAINFAILER происходит в случае, если предыдущее состояние этой БД было MAIN или MONO. Конкретное значение, в которое переходит БД системы резервирования, определяется кодом возврата ядра СУБД ЛИНТЕР.

Если ключ не задан, по умолчанию допускается 5 повторных запусков ядра СУБД ЛИНТЕР.

Пример

```
-lretry=7
```

Количество повторных запусков программ системы резервирования (pretry)

Синтаксис

```
/pretry=<количество перезапусков>

<количество перезапусков>::=целое положительное число.
```

Описание

Задаёт допустимое количество повторных неудачных запусков программ `dbs_tcp`, `dbc_tcp`, `lhb` (кроме ядра СУБД ЛИНТЕР), поддерживающих функционирование системы резервирования. Запуск считается неудачным, если программа завершилась не по команде управляющей программы системы резервирования, то есть самостоятельно. При превышении количества повторных неудачных запусков любой из перечисленных выше программ в течение контрольного интервала времени сервер переходит в состояние SLAVEFAILER или MAINFAILER и завершает свою работу. Переход в состояние MAINFAILER происходит в случае, если предыдущее состояние БД было MAIN или MONO. Контрольный интервал для процессов, отличных от `linter`, равен 60 секундам.

Если ключ не задан, по умолчанию допускается 5 повторных запусков перечисленных программ.

Пример

```
-pretry=3
```

Задержка запуска сетевых драйверов (dbsctout)

Синтаксис

```
/dbsctout=<интервал>
<интервал>::=положительное целочисленное значение или значение с
 десятичной точкой.
```

Значение <интервала> задается в секундах.

Описание

Задаёт временной интервал рестарта сетевых драйверов сервера и клиента (dbs_tcp, dbc_tcp).

После завершения работы сетевых утилит нельзя производить их немедленный рестарт, поскольку их дочерние процессы могут завершаться с некоторым запозданием и на это время удерживать сетевые порты. В связи с этим рестарт сетевых драйверов должен производиться с некоторым запозданием.

По умолчанию интервал рестарта равен 5 секундам. Не рекомендуется задавать это значение менее 3 секунд во избежание попадания сервера резервирования в FAILER-состояние из-за превышения попыток повторного запуска сетевых компонентов. Кроме того, необходимо учитывать, что пока не обнаружен разрыв соединения с главным сервером, сетевой драйвер клиента dbc_tcp будет периодически перезапускаться. Поэтому значение «интервал повторного запуска сетевых компонентов», помноженное на значение «максимальное количество повторных запусков», должно быть больше значения «интервал посылки тестовых пакетов», помноженного на значение «максимальное количество потерянных пакетов» (см. ключи /testint, /tstlimit).

Пример

```
-dbsctout=15
```

Файл конфигурации ключей запуска программы (cf)

Синтаксис

```
/cf=<спецификация файла>
<спецификация файла>::=текстовая строка.
```

Описание

В файле задаются значения ключей запуска управляющей программы server и ее переменных окружения. Файл параметров запуска может включать две секции: секция ключей и секция переменных окружения.

Секция ключей должна начинаться строкой.

```
.arg:
```

Секция переменных окружения должна начинаться строкой.

```
.env:
```

Каждая из секций может состоять из нескольких строк.

Синтаксис ключей такой же, как и в командной строке запуска управляющей программы server.

В файле конфигурации запуска можно задавать переменные окружения

```
<имя переменной1> = <значение1> <имя переменной2> = <значение2>.
```

Допустимо использование кавычек для задания значений, содержащих пробелы. Кавычки должны закрываться на той же строке.

Конец строки считается разделителем, то есть ключ или выражение вида

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

```
<имя переменной1> = <значение1>
```

нельзя продолжать на следующую строку. Строки, начинающиеся символом #, игнорируются.

Размер строки файла конфигурации должен быть ограничен 4000 байт. Поэтому, несмотря на возможность указания нескольких ключей в строке, рекомендуется обозначать по одному ключу в каждой строке.

Имя файла конфигурации по умолчанию `hotreserve.conf`.

Порядок поиска файла конфигурации запуска программы такой же, как и порядок поиска файла `nodetab` (см. описание ключа /ntab), за исключением того, что поиск в каталогах, заданных переменными окружения `SY00` и `SERVER_HOME`, не осуществляется.

С помощью ключа `/cf=<путь к файлу конфигурации>` можно задать полный путь файла. При этом имя файла может отличаться от значения по умолчанию. Если отсутствует файл, полный путь к которому задан, программа завершается.

Если файл найден, выдается сообщение при старте `server`:

```
Configuration file found: <путь к файлу>
```

Такое же сообщение записывается в log-файл с атрибутом MSG.

Обработка особых случаев:

1) файл конфигурации не найден, ключ `/cf=<имя файла>` не задан.

   Выдается сообщение

```
The configuration file <имя файла> not found
```

   и программа продолжает работу, игнорируя настройку параметров из файла.

2) файл конфигурации не найден, ключ `/cf=<имя файла>` задан.

   Выдается сообщение

```
The configuration file <имя файла> not found
```

   и программа завершается.

3) синтаксическая ошибка в файле конфигурации.

   Выдается сообщение

```
Error in configuration file string: <строка с ошибкой>
```

   и программа завершается.

4) Если заданы одноименные ключи в файле конфигурации и в командной строке запуска программы `server`, берется значение из командной строки.

Пример файла конфигурации:

```
#argument section
```

```
.arg:
/setstate=MONO   /tcorrect=20

#environment section
.env:
SERVER_HOME= /home/linter/HOME
SY00 =  /home/linter/DB
```

Запрет использования файла конфигурации запуска программы (nocf)

Синтаксис

```
/nocf
```

Описание

Запрещает использование файла конфигурации программы. Управляющая программа настраивается только ключами запуска. Попытка поиска файла конфигурации не осуществляется.

Управление архивами

Создание архивного файла (archivate)

Синтаксис

```
/archivate
```

Описание

Заставляет создавать архивный файл архивной БД после копирования рабочей БД в архивный каталог.

После переноса или копирования рабочей БД в архивный каталог можно создать архивный файл архивной БД. По умолчанию используется архиватор `zip`. Чтобы использовать другой архиватор, нужно задать командную строку этого архиватора в ключе `/cmdarcadd`, при этом имя архивной БД может быть указано в ключе `/barc`.

При успешном создании архивного файла генерируется соответствующее событие. Если задан обработчик событий (ключ `/stproc`), то ему передается имя созданного архивного файла (для возможной дальнейшей обработки).

Местоположение архивного файла (barc)

Синтаксис

```
/barc=<спецификация файла>
```

Описание

Задаёт полный абсолютный путь архивного файла БД.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Заданная <спецификация файла> используется:

- для поиска архивного файла при обработке ключа /crash;
- для размещения архивного файла при обработке ключа /archivate;
- для архивирования по команде;
- для периодического архивирования.

По умолчанию архивный файл располагается в архивном каталоге и имеет имя DB.zip.

Если расширение файла не задано, по умолчанию используется zip.

Примеры

```
-barc=/usr/linter/arc/db0712.arc
/barc='/home/server/program files/linter/arc/db_sale.zip'
```

Дублирование архивируемых данных (mklhbarc)

Синтаксис

```
/mklhbarc
```

Описание

Задаёт режим сохранения получаемых с главного сервера данных одновременно в БД и в её архивном (lhb) файле.

Когда резервный сервер получает данные с главного сервера, то он сохраняет их непосредственно в своей рабочей БД. В этом случае в архивный файл dbhotres.lhb записывается только служебная информация. При использовании данного ключа в архивный файл БД записываются и сами данные.

Примечание
Размер архивного файла может только увеличиваться. Использование ключа может привести к большим размерам архивного файла и переполнению файловой системы. Ключ рекомендуется использовать только в целях отладки. Файл dbhotres.lhb переименовывается в dbhotres.lhb_ при переходе сервера в MONO-состояние.

Работа с базами данных на многих устройствах (devenv)

Синтаксис

```
/devenv=<устройство>[<разделитель> <устройство> …]
<устройство>::=<переменная окружения>=<рабочий каталог>:<архивный
 каталог>
<переменная окружения>::=имя переменной окружения, используемой
 для указания устройства с БД
<рабочий каталог> и <архивный каталог> – спецификация пути к
 рабочему и архивному каталогу БД соответственно
<разделитель> – для ОС типа Windows ";",для ОС типа Linux, ЗОСРВ
 Нейтрино ":"
```

Описание

По умолчанию система резервирования предполагает, что все файлы БД расположены на одном устройстве (в одном каталоге). Это устройство задается либо переменной окружения SY00, либо ключами командной строки `pathtodb`, `pathtoarc`.

Ключ `/devenv` позволяет работать с базами данных, размещенными на нескольких устройствах. Значение этого ключа определяет, какие переменные окружения должны быть установлены при запуске ядра СУБД ЛИНТЕР и других утилит для данной БД.

Ключ имеет приоритет над переменными среды окружения и остальными опциями выбора каталогов.

Работа с базами данных на нескольких устройствах имеет следующие особенности:

- при очистке БД удаляются все файлы, имеющие такой же вид, как файлы БД, независимо от того, принадлежат ли они БД или нет;
- поскольку БД расположена в нескольких каталогах, невозможно создать список файлов БД, находящихся в одном каталоге. Это ограничивает применение шаблона %LIST% (список файлов БД без указания путей) при создании архивов. Вместо этого шаблона рекомендуется применять шаблон %FULLIST%;
- при архивировании шаблон %FILE% определяет только имя файла без каталога. Соответственно, файлы в архиве оказываются нераспределенными по каталогам;
- БД также не может быть восстановлена из архива автоматически (ключ `/crash`), поскольку архив не может быть развернут в один каталог. Восстановление из архива должно производиться администратором системы резервирования в ручном режиме;
- при использовании ключа `/show` должен быть установлен и ключ `/devenv` для отображения корректной информации. Для получения информации предпочтительно использовать утилиту `hresctl` (`srvcmd`);
- при отображении информации утилитами `srvcmd`, `hresctl` или `server` ключом `/show` выводится только каталог, соответствующий устройству SY00. Остальные устройства не отображаются;
- для работы системы резервирования с базой, расположенной на нескольких устройствах, недопустимо указывать пути этих устройств явно, то есть запросом create or replace device, или установкой переменных окружения. Вместо этого необходимо использовать опцию `/devenv` для указания пар каталогов устройств рабочей и резервной базы данных. То есть система резервирования работает только с незарегистрированными устройствами;
- если базы будут использоваться вне системы резервирования, то устанавливать пути к устройствам необходимо установкой переменных окружения, соответствующих устройствам;
- для работы с незарегистрированными устройствами нужно назначить соответствующие привилегии пользователям. В простейшем случае запросом:

```
grant access on unlisted device to all;
```

См. документ «Администрирование комплекса средств защиты данных».

Примечание
На ОС Windows работа с несколькими устройствами не реализована.

© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Пример

Задается установка 2-х переменных окружения SY00 и SY01. Устройство SY00 располагается в каталогах /path/to/work/db или /path/to/arc/db, а устройство SY01 в каталогах /path/to/work/db_dir или /path/to/arc/db_dir, соответственно:

```
SY00=/path/to/work/db:/path/to/arc/db:SY01=/path/to/work/db_dir:/
path/to/arc/db_dir
```

Разрешение на восстановление БД из архивного файла (crash)

Синтаксис

```
/crash
```

Описание

Разрешает архивному файлу сервера резервирования участвовать в конкурсе на выбор активной БД при отсутствии на нем рабочей или архивной БД при старте системы.

После старта системы резервирования и выбора данного сервера в качестве главного сервера резервирования архивный файл на этом сервере будет развернут в каталог рабочей БД. После чего система резервирования завершит работу (и главная и резервная управляющие программы). При последующем ручном рестарте системы резервирования будет использована развернутая из архивного файла рабочая БД, а сервер получит состояние из восстановленного файла state.

По умолчанию сервер не восстанавливает БД из архивного файла.

Периодичность копирования рабочей БД в архивный каталог (arcint)

Синтаксис

```
/arcint=<периодическое копирование>
        |[']<копирование по расписанию>[']
<периодическое копирование>::={0|-1|<интервал>}
<интервал>::=положительное целочисленное значение.
```

Значение <интервала> задается в секундах.

```
<копирование по расписанию>::=
AT[<шаблон времени>[{<запятая>|<пробел>}<шаблон времени>…]]
<шаблон времени>::=<год><месяц><день><час><мин>.
<мин>::=значение в диапазоне от 0 до 59.
<час>::=значение в диапазоне от 0 до 23.
<день>::=<номер дня>|<день недели>.
<номер дня>::=значение в диапазоне от 1 до 31.
<день недели>::=MON|TUE|WED|THU|FRI|SAT|SUN
<месяц>::=значение в диапазоне от 1 до 12.
<год>::=[<г><г>[<г><г>]].
<г>::=значение в диапазоне от 0 до 9.
```

В шаблоне времени любые впереди стоящие элементы даты/времени могут быть опущены (год, месяц, день, день недели, час, минута).

Если какие-либо элементы даты в шаблоне времени не заданы, то вместо них берутся такие значения, чтобы следующее копирование БД выполнялось в ближайшее будущее время, у которого совпадут все непропущенные элементы.

Если указан день недели, то все предшествующие поля шаблона (месяц, год) будут игнорированы.

Год может быть задан как двумя символами, так и четырьмя. В случае указания только двух символов считается, что год относится к 21 веку.

Для запуска используется локальное время сервера резервирования с учетом часового пояса.

В случае если числовое значение элемента <день> больше количества дней в данном месяце, то копирование БД будет проводиться в последний день месяца независимо от номера месяца.

Если указана полная дата, то копирование БД будет выполнено только один раз, в указанную конкретную дату. Если эта дата уже истекла, то копирование отменяется.

Примеры

Шаблон	Дата/время тестирования
SUN0300	Каждое воскресенье в три часа ночи
121530	Ежемесячно двенадцатого числа в 15:30
311815	Ежемесячно в последний день месяца в 18:15
00	В начале каждого часа
01010000	В начале каждого года
200807181500	В три часа дня восемнадцатого июля 2006 года

Если в <копировании по расписанию> задан только один <шаблон времени> и нет символов пробела, ограничивающие кавычки можно опустить.

Примеры

```
/arcint=at311815
/arcint='at 311815'
```

Если указано несколько шаблонов времени, то копирование БД будет выполняться по всем шаблонам в ближайшее для каждого шаблона время. Например, значение ключа 'at00,10,20,30,40,50' предписывает копирование БД каждые 10 минут.

Если <шаблоны времени> перечислены только через запятую, ограничивающие кавычки можно не ставить.

Примеры

```
/arcint=at311815,SUN0300
/arcint='at311815 SUN0300'
/arcint='at311815,SUN0300 121530'
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Пустой шаблон времени (ключ `/arcint=at`) означает запуск копирования каждую минуту.

В случае если наступило время копирования БД, но предыдущее копирование еще не закончено, новое копирование будет запущено позже. Поэтому для данного ключа реализована возможность копирования по расписанию.

Описание

Задаёт распорядок копирования рабочей БД резервного сервера в архивный каталог на резервном сервере.

Для повышения надежности системы резервирования желательно на резервном сервере выполнять периодическое копирование рабочей БД в архивный каталог. На время выполнения этой операции ядро СУБД ЛИНТЕР в специальном режиме останавливается с целью предотвращения потери информации. С этой же целью сначала копируются все файлы таблиц, а потом уже файлы системного журнала в порядке их возрастания.

Если значение ключа равно 0 (значение по умолчанию), то выполняется только одноразовое копирование рабочей БД резервного сервера в его архивный каталог перед получением рабочей БД с главного сервера. Последующие копирования рабочей БД в архивный каталог не производятся. Данное значение имеет смысл только для случая отключенного обмена каталогов (`/exchdir=0`). При обмене каталогов автоматически самая свежая БД оказывается в резервном каталоге и копирования не требуется.

Если опция `<периодическое копирование>` равна -1, то запрещено любое, даже первоначальное, копирование рабочей БД. Более того, в данном случае архивная БД не будет принимать участия в конкурсе на рабочую БД, как при старте резервного сервера, так и при переходе его в MONO-состояние. Работать в этом режиме не рекомендуется, поскольку при выходе из строя главного сервера во время получения от него резервным сервером рабочей БД, последний не сможет взять на себя роль главного по причине отсутствия рабочей БД. Данное значение имеет смысл только для случая отключенного обмена каталогов (/exchdir=0). При обмене каталогов автоматически самая свежая БД оказывается в резервном каталоге. Первоначальное копирование не производится.

Если задан интервал времени (или расписание), то на резервном сервере периодически (или по расписанию) выполняются следующие операции:

1) производится останов ядра СУБД ЛИНТЕР, работающего в специальном режиме;
2) удаляются старые файлы БД из архивного каталога;
3) рабочая БД копируется в архивный каталог;
4) по завершении копирования БД ядро СУБД ЛИНТЕР снова стартует в специальном режиме и функционирует в нем в течение заданного в ключе интервала времени;
5) после истечения интервала времени сервер повторяет перечисленные операции (начиная с 1).

Интервал времени определяет продолжительность работы резервного сервера между операциями копирования рабочей БД в архивный каталог.

Задаваемая периодичность копирования должна быть выбрана с таким расчетом, чтобы ядро СУБД ЛИНТЕР успевало перенести накопленные в системном журнале данные в таблицы БД. В противном случае время перехода сервера в MONO-режим будет длительным из-за большого количества необработанных файлов системного журнала. Также по этой причине возможна нехватка дискового пространства из-за роста количества файлов системного журнала.

В случае указания расписания копирование стартует в заданные моменты времени. Если к заданному моменту времени не завершилась предыдущая операция копирования, то новая операция копирования пропускается.

При использовании данного ключа совместно с ключом `/archivate` после каждого копирования запускается операция создания архивного файла архивной БД.

По умолчанию используется ключ `/arcint=0`.

Пример

```
-arcint=300
```

Автоматический обмен статуса каталогов БД резервного сервера (exchdir)

Синтаксис

```
/exchdir[=0]
```

Описание

Разрешает/запрещает взаимно менять статус рабочего и архивного каталогов БД.

При задании этого ключа копирование файлов БД из архивного каталога в рабочий каталог при старте системы или при смене состояний не производится, каталоги просто меняют свой текущий статус. Это обеспечивает ускоренный переход резервного сервера в MONO-состояние за счет исключения операции копирования БД. Данный режим является режимом по умолчанию. При значении параметра 0 сервер работает в режиме совместимости со старыми версиями с копированием БД.

Определение статуса каталогов происходит при конкурсе на определение рабочего каталога (например, при старте системы резервирования) по результатам предшествующей работы. Для этого сравнивается время последней записи в файле `STATE` рабочей и архивной БД или положение последней записи в системном журнале. Каталог с более «свежей» БД получает статус рабочего каталога на главном и резервном серверах.

На резервном сервере предпринимается попытка докачки системного журнала с контрольной точки. Если это невозможно, то в качестве рабочего каталога назначается каталог с самой старой БД, поскольку на резервном сервере рабочий каталог подвергается очистке перед получением БД.

Ключ влияет на выдаваемую по ключу `/show` информацию:

1) `/show /exchdir=0` – отображает статус каталогов на основании только аргументов командной строки;
2) `/show` – отображает реальный (текущий) статус каталогов (который мог измениться в результате обмена статусами).

Копирование только отличий БД (diff)

Синтаксис

```
/diff
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Описание

Активизирует копирование только отличий БД (вместо их полного копирования).

На этапе перехода компьютера в SLAVE-режим после неудачной попытки докачивания БД в инкрементном режиме необходимо получение всей БД с главного сервера. Обычно для этого очищается каталог с наиболее старой БД, и в этот каталог начинается полное копирование БД путем запуска утилиты `lhb` в startinc wait режиме. При указании данного ключа очистка каталога с наименее актуальной БД производиться не будет. Вместо этого для этой БД будет запущена утилита `lhb` с ключом `/diff`, которая вместо поблочного копирования всех файлов таблиц БД будет копировать только контрольные суммы файловых блоков и сравнивать их с контрольными суммами соответствующих блоков локального каталога. Если контрольные суммы блоков совпадают, то их копирование не производится.

За счет этого объем переданных по сети данных может быть значительно сокращен. При быстрых файловых системах и медленных линиях связи это может дать прирост производительности в несколько раз, при условии наличия на SLAVE-сервере не очень старой БД в обоих каталогах. То есть данный режим применяется тогда, когда в каталоге уже есть БД. На практике это означает, что при запуске сервера резервирования с пустыми каталогами БД данный режим активируется только при третьем запуске: первый запуск заполняет пустой рабочий каталог, второй запуск – пустой резервный каталог. Исходя из этого, при больших объемах БД рекомендуется при введении нового сервера в систему разворачивать копию БД из архива в оба каталога.

Применение данного метода на медленных файловых системах и быстрых сетях не дает выигрыша в скорости, поскольку в любом случае файловый блок должен быть прочитан. Если время чтения блока является узким местом системы, то это и будет в основном определять скорость работы системы в целом. В этом случае возможно лишь небольшое улучшение производительности или даже небольшое ее ухудшение по сравнению с обычным копированием БД.

При этом если БД главного и резервного сервера совершенно различны, то использование данного режима несколько ухудшит производительность системы резервирования, поскольку, кроме копирования всех различных файловых блоков, потребуется получать их контрольные суммы. В результате суммарный трафик увеличивается.

Примечание
Работа утилиты `lhb` с ключом `/diff` и одновременное сохранение данных в lhb-файл невозможны, поэтому при указании ключа `/mklhbarc` режим копирования только отличий автоматически отключается.

Управление размером системного журнала (syslogcount)

Синтаксис

```
/syslogcount=<количество файлов>
```

Описание

Устанавливает количество файлов системного журнала, при достижении которого производится удаление контрольных точек на главном сервере. Поскольку размер файла является постоянным, то указание количества файлов определяет и размер системного журнала. Пока контрольная точка не удалена, резервный сервер при старте может

продолжить докачку системного журнала вместо скачивания всей БД. Это ускоряет приведение системы резервирования в состояние готовности.

Проверка размера системного журнала производится периодически с интервалом посылки тестовых запросов ядру СУБД ЛИНТЕР. Если количество файлов превысило заданное значение, то удаляется самая старая контрольная точка, не принадлежащая активному в настоящий момент резервному серверу.

По умолчанию количество файлов равно 5. При указании нулевого количества файлов производится удаление всех контрольных точек, не принадлежащих ни одному работающему резервному серверу. Таким образом, как только обнаруживается завершение работы резервного сервера, его контрольная точка удаляется при следующей проверке размеров файла системного журнала.

Шаблон командной строки для сохранения БД в архивном файле (cmdarcadd)

Синтаксис

```
/cmdarcadd=<шаблон командной строки>
<шаблон командной строки>::=символьная строка.
```

Описание

Задает командную строку для добавления файла архивной БД в файл архива.

В командной строке можно использовать следующие предопределенные переменные:

1) `%FILE%` – имя архивируемого файла без указания каталога;
2) `%ARCHIVE%` – спецификация архивного файла;
3) `%LIST%` – спецификация файла, содержащего список архивируемых файлов без указания каталога.
4) `%FULLIST%` – спецификация файла, содержащего список архивируемых файлов с полным путем к каждому из файлов.

По ключу `/archivate` при окончании копирования БД происходит запуск заданной командной строки из каталога архивируемой БД с подстановкой значений вместо шаблонов. Замену шаблонов реальными значениями выполняет сервер резервирования.

На место шаблона `%ARCHIVE%` подставляется спецификация архивного файла (полный путь и собственно имя файла). Имя файла извлекается из ключа `/barc` (если он задан) или используется по умолчанию (`DB.zip`).

На место шаблона `%FILE%` последовательно подставляются имена всех файлов архивируемой БД без указания каталога БД, и каждый раз производится запуск архиватора. Так как текущий каталог является каталогом архивируемой БД, то эти имена можно использовать напрямую, без добавления пути к файлу. Команда архивирования с шаблоном `%FILE%` будет запущена по одному разу для каждого архивируемого файла, то есть всего столько раз, сколько файлов входит в состав БД.

На место шаблона `%LIST%` подставляется спецификация сформированного сервером резервирования файла, в котором содержится список всех файлов архивируемой БД. Имена файлов перечислены по одному в каждой строке. Команда архивирования в

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

случае указания шаблона %LIST% будет выполнена только один раз. Файл списка размещается в каталоге, определяемой переменной окружения SERVER_HOME, или в рабочем каталоге и имеет имя dbarclist.

Шаблон %FULLIST% аналогичен шаблону %LIST% за исключением того, что в файле сохраняются не список имен файлов, а список полных путей к файлам БД.

В шаблоне возможно указание только %LIST%, %FULLIST% или %FILE%. Одновременное использование хотя-бы двух из шаблонов недопустимо.

Данный ключ позволяет использовать любые архиваторы, отличные от архиватора по умолчанию zip.

Для архивирования БД с помощью архиватора tar задать следующее значение ключа:

```
/cmdarcadd='tar cfz %ARCHIVE% -T %LIST%'
```

В этом случае утилита tar за один вызов создаст архивный файл с именем, заданным в ключе /barc (или DB.zip по умолчанию).

Для архивирования БД с помощью архиватора 7z.exe задать следующее значение ключа:

```
/cmdarcadd="7zG.exe a -tzip -ssw -mx1 -r0 %ARCHIVE% @%LIST%"
```

Если операционная система не поддерживают автоматический запуск команды сжатия данных утилитой tar, необходимо написать небольшой shell-скрипт для запуска архивирования, например:

```
#!/bin/sh
tar cf - -T %1 | bzip2 > $2
exit $?
```

В этом случае командная строка запуска будет выглядеть так:

```
/cmdarcadd='/path/to/script.sh %LIST% %ARCHIVE%'
```

В приведенном скрипте не обязательно использовать шаблон для спецификации архивного файла – можно явно указать его другое местоположение. Дополнительно в скрипте можно прописать выполнение некоторых действий с архивным файлом, например:

- следить за хранением 10 последних архивных файлов БД;
- дублировать на компакт-диск;
- передавать по сети на другой компьютер.

Скрипт открывает дополнительные возможности манипуляции с архивными файлами БД.

Можно также ускорить создание архивного файла архиватором zip, если архивировать не по одному файлу, а списком. Для этого можно использовать следующий скрипт:

```
#!/bin/sh
cat $1 | zip  $2.zip -@
exit $?
```

Если ключ не задан, по умолчанию используется значение
"zip -j %ARCHIVE% %FILE%".

Шаблон командной строки для восстановления БД из архивного файла (cmdarcext)

Синтаксис

```
/cmdarcext=<шаблон командной строки>
<шаблон командной строки>::=символьная строка.
```

Описание

Задает командную строку для извлечения БД из архивного файла.

В командной строке можно использовать предопределенную переменную (шаблон) %ARCHIVE%, в которую сервер резервирования подставит полный путь и имя архивного файла (см. описание ключа /cmdarcadd).

Команда будет запущена из каталога, куда необходимо развернуть БД. По умолчанию используется командная строка unzip -o %ARCHIVE%.

Сетевые настройки

Посылка тестовых пакетов (testint)

Синтаксис

```
/testint=<интервал>
<интервал>::=положительное число (в виде целого числа или числа с
 десятичной точкой).
```

Значение <интервала> задается в секундах.

Описание

Задает интервал посылки тестовых пакетов удаленным серверам системы резервирования и дискретность временных операций сервера. Все временные операции сервера резервирования будут производиться с интервалами, кратными указанному. Например, если заданы ключи /testint=5.5 и /treq=30, то реальное тестирование активности ядра СУБД ЛИНТЕР будет производиться с интервалом от 33 (5.5*6) секунды.

Рекомендуется согласовывать интервалы посылки тестовых пакетов на всех серверах системы резервирования.

Если ключ не задан, по умолчанию посылка тестовых пакетов выполняется с интервалом в 2 секунды.

Примеры

```
-testint=3
/testint=0.5
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Критерий разрыва соединения (tstlimit)

Синтаксис

```
/tstlimit=<количество пропусков>
<количество пропусков>::=положительное целое число.
```

Описание

Задает количество пропущенных тестовых пакетов, при котором связь с удаленным сервером резервирования считается потерянной. Если ключ не задан, то используется значение по умолчанию – 7 пакетов.

Величина тайм-аута потери соединения вычисляется как произведение значений ключей `/testint` и `/tstlimit`.

Критерий потери соединения можно сформулировать следующим образом: если в течение этого тайм-аута не принято ни одного пакета от узла, то связь с удаленным узлом считается потерянной.

Ключ определяет также максимальное количество не отправленных и повторных посылок тестовых пакетов удаленному серверу.

С периодичностью тайм-аута удаленного сервера резервный сервер в SLAVE_WAIT-состоянии производит опрос готовности главного сервера. Если соединение не потеряно, он остается в SLAVE_WAIT-состоянии.

Примеры

```
-tstlimit=10
/tstlimit=5
```

Периодичность сохранения состояния баз данных серверов (dbtestint)

Синтаксис

```
/dbtestint=<интервал>
<интервал>::=положительное целочисленное значение или значение с
 десятичной точкой.
```

Значение `<интервала>` задается в секундах.

Описание

Задаёт периодичность получения информации о количестве данных в журнале БД, отсылки этой информации другим серверам и записи информации в файл `state`. Все эти действия выполняются при включенном сравнении БД по журналу (ключ `/cmpsyslog`). При сравнении БД по времени только на главном сервере производится запись даты БД в файл `state` и рассылка ее резервным серверам, на резервном сервере никаких действий не выполняется.

Эта информация используется, при необходимости, для ранжирования серверов.

Если ключ не задан, по умолчанию проверка состояния серверов резервирования выполняется с периодичностью в 5 секунд.

Пример

```
-dbtestint=7
```

Управление тестированием

Регулярность тестирования БД резервного сервера (testdb)

Синтаксис

```
/testdb[=
{ <периодическое тестирование>
| <тестирование по расписанию>
| <тестирование по сигналу> }
]
<периодическое тестирование>::=<интервал>
<интервал>::=положительное целое число.
```

Значение <интервала> задается в секундах.

```
<тестирование по расписанию>::=см. описание опции <копирование по
 расписанию> в ключе /arcint.
<тестирование по сигналу>::=SIG<номер сигнала>
<номер сигнала>::=положительное целочисленное значение.
```

Описание

Задаёт регулярность тестирования рабочей БД резервного сервера. Тестирование БД выполняется утилитой `testdb` периодически, по расписанию или по сигналу (см. документ «Тестирование базы данных»).

Тестирование БД выполняется по следующим правилам:

1) производится останов ядра СУБД ЛИНТЕР, работающего в специальном режиме на резервном сервере;
2) из архивного каталога удаляется хранящаяся там БД и на её место копируется рабочая БД резервного сервера. Копирование выполняется в порядке добавления данных – самые последние данные копируются также в последнюю очередь;
3) производится запуск ядра СУБД ЛИНТЕР в специальном режиме на рабочей БД этого сервера;
4) в архивном каталоге для только что сделанной копии рабочей БД запускается новый экземпляр ядра СУБД ЛИНТЕР в обычном режиме.

   Повторный запуск ядра СУБД на скопированной БД необходим по следующим причинам:

   - после останова ядра СУБД ЛИНТЕР, работавшего в специальном режиме на резервном сервере, могло продолжаться добавление данных в файлы системного журнала (утилитой `lhb`);
   - ядро СУБД ЛИНТЕР в специальном режиме на резервном сервере не обрабатывает последний файл системного журнала рабочей БД и не закрывает журнал, поэтому при копировании рабочей БД в архивный каталог журнал остается не до конца обработанным. Запущенная в архивном каталоге СУБД выполняет окончательный докат всех транзакций по системному журналу;

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

5) как только второй экземпляр СУБД ЛИНТЕР (на БД в архивном каталоге) выйдет в состояние готовности (докатит записи системного журнала в БД), он принудительно останавливается программой `server`;
6) в архивном каталоге запускается утилита тестирования `testdb` для проверки архивной БД. Вывод `testdb` направляется в файл `testdb.out`, который создается в каталоге, задаваемом переменной окружения `SERVER_HOME`;
7) в случае обнаружения ошибок в тестируемой БД на консоль и в файл трассировки выводятся соответствующие сообщения.

Если сервер завершает свою работу в режиме резервного сервера, а параллельный процесс тестирования архивной БД или копирования рабочей БД в архивный каталог работают, то он будет немедленно завершен сигналом SIGKILL. В этом случае обработчику событий будет передан соответствующий код события (см. пункт «Событие E_TESTDB»).

При эксплуатации системы резервирования необходимо учитывать, что ошибки БД диагностируются на резервном сервере, а устранять их необходимо на главном сервере, поскольку БД на резервном сервере является точной копией БД главного сервера. Кроме того, устранение ошибок в БД невозможно без остановки работы всей системы резервирования, поскольку при остановке только главного сервера один из резервных серверов возьмет на себя функции главного сервера, и новые данные уже не попадут в ту БД, с которой будут проводиться профилактические работы. Для устранения ошибок необходимо использовать утилиту `testdb`.

Результат последнего тестирования БД сохраняется в файле `state`. Для его просмотра можно использовать ключ `/show` или утилиту `hresctl` (`srvcmd`) (на работающем сервере). Результат тестирования содержит информацию о факте обнаружения ошибок в БД или их отсутствии, и время окончания тестирования БД.

Примечание

Так как тестируется копия рабочей БД резервного сервера в архивном каталоге, а последняя является копией рабочей БД главного сервера, то результат тестирования распространяется на все три перечисленные БД.

Если задано периодическое тестирование БД, то утилита `testdb` будет периодически запускаться на резервном сервере через заданный `<интервал>` времени после завершения предыдущего тестирования. Тестирование сильно нагружает систему резервирования, поэтому, если время очередного тестирования совпало с повышенными нагрузками на систему резервирования, её производительность может понизиться.

При использовании расписания, если предыдущая операция тестирования не была завершена, новая операция тестирования пропускается.

Для запуска тестирования БД по сигналу рекомендуется использовать сигнал SIGALRM.

В целях безопасности тестирование выполняется только при условии, что БД целиком получена с главного сервера.

Часть операций, задаваемых ключами `/arcint` и `/testdb`, совпадают. Если эти ключи заданы одновременно, началось копирование БД, инициированное ключом `/arcint`, и при этом наступает время тестирования БД, то дополнительная операция копирования отменяется. И наоборот, если началось копирование БД, инициированное ключом `/testdb`, и при этом наступает время копирования БД по ключу `/arcint`, то

повторное копирование также отменяется. Если при этом совпадении указан еще и ключ `/archivate`, то тестирование БД начнется только после создания архивного файла архивной БД.

По умолчанию (при отсутствии ключа `/testdb`) тестирование БД не выполняется, однако, если ключ `/testdb` задан, но без параметров, то тестирование будет выполняться ежедневно в полночь.

Примеры

```
-testdb=300
-testdb
-testdb=sig12
/testdb=ATwed0500,sun0300
```

Дополнительные указания по тестированию БД (tdbadd)

Синтаксис

```
/tdbadd=[']<ключ>[<пробел><ключ>…][']
<ключ>::=символьная строка.
```

Описание

Задаёт дополнительные ключи запуска утилиты тестирования `testdb`. Значение ключа `/tdbadd` подставляется в конец командной строки запуска утилиты. Возможные дополнительные ключи см. в документе «Тестирование базы данных».

При добавлении одного ключа ограничивающие кавычки можно не ставить.

Логическая противоречивость ключей не проверяется.

Примеры

```
-tdbadd='-td'
/tdbadd=-td
/tdbadd='-p 1000 -i 3'
```

Управление слежением за процессами

Тайм-аут и интервал проверки ядра СУБД ЛИНТЕР (treq)

Синтаксис

```
/treq=<тайм-аут>[:<интервал>]
<тайм-аут>::=целочисленное положительное значение.
<интервал>::=целочисленное положительное значение.
```

Описание

Задаёт величину тайм-аута (в секундах) и интервал (в секундах) посылки тестовых запросов ядру СУБД ЛИНТЕР. Тайм-аут ядра – это временной интервал, в течение которого ядро СУБД ЛИНТЕР считается «живым» (даже в случае, если от ядра нет ответов на тестовые запросы). Если в течение тайм-аута ядро не вернуло ответ на тестовый запрос, то выполняется принудительный рестарт ядра СУБД ЛИНТЕР.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Сервер резервирования посылает циклически 4 вида тестовых запросов последовательно, между запросами, с интервалом, заданным величиной <интервал>. Запросы предназначены не только для проверки активности ядра СУБД ЛИНТЕР, но и его компонентов (программ sql, tsp, intsrt).

Если задан ключ /st (облегченный тест), то посылается всегда один вид запроса, который проверяет только ядро.

Проверка активности ядра продолжается и после посылки команды останова ядру СУБД ЛИНТЕР.

Если ключ не задан, по умолчанию тайм-аут ядра равен 20 секундам, а интервал – 5 секундам.

Если задано только значение <тайм-аут>, то отсутствующее значение <интервал> вычисляется по формуле: <тайм-аут> = <интервал> * 4.

Если заданы и <тайм-аут>, и <интервал>, то может быть рассчитано максимальное количество пропущенных запросов.

В случае если по истечении интервала посылки (в момент посылки следующего тестового запроса) ядро еще не вернуло ответ на отосланный запрос, то будет сгенерировано событие W_DEADLOCK (возможно, ядро СУБД ЛИНТЕР зациклилось).

Необходимо различать тайм-аут проверки активности ядра СУБД ЛИНТЕР и интервал посылки ядру тестовых пакетов. Интервал посылки определяет периодичность посылки тестовых пакетов ядру, а тайм-аут ядра – допустимое время задержки ответа на тестовую посылку.

Примечание
Все временные интервалы должны быть кратны в меньшую или большую сторону интервалу посылки тестовых пакетов (см. ключ /testint).

Примеры

```
-treq=15
/treq=15:3
```

Облегченная проверка активности ядра СУБД ЛИНТЕР (st)

Синтаксис

```
/st
```

Описание

Задает упрощенную проверку активности ядра СУБД ЛИНТЕР (см. ключ /treq) с помощью команды desc (получить описание БД) интерфейса нижнего уровня, а не специальными запросами. Этот метод ускоряет проверку и уменьшает нагрузку на ядро СУБД ЛИНТЕР, но не тестирует все компоненты СУБД.

Слежение за родительским процессом (parent)

Синтаксис

```
/parent[=<слежение за процессом>|<слежение за лидером сессии>]
```

```
<слежение за процессом>::=идентификатор отслеживаемого процесса.
<слежение за лидером сессии>::=SID
```

Описание

Заставляет отслеживать существование заданного родительского процесса.

Для некоторых платформ можно указать специальное значение SID (слежение за лидером сессии, в которой запущен сервер резервирования).

При завершении работы процесса, за которым производится слежение, сервер резервирования также завершает свою работу. Это может быть полезно при запуске управляющей программы из пользовательской программы.

Если параметр ключа не задан, отслеживается существование родительского процесса сервера резервирования.

Если ключ не задан, по умолчанию существование родительских процессов не отслеживается.

Примеры

```
-parent
-parent=24
/parent=SID
```

Слежение за управляющей программой системы резервирования (wd)

Синтаксис

```
/wd[=<файловый дескриптор>|E]
<файловый дескриптор>::=целочисленное значение.
```

Описание

При отсутствии значения ключа порождается отдельный процесс, который следит за работоспособностью управляющей программы системы резервирования. В случае зависания управляющей программы производится ее автоматический рестарт.

Если значение ключа равно E, то при отсутствии активности управляющей программы ее рестарт не выполняется, выполняется только останов. Внешняя программа может обнаружить завершение процесса управляющей программы и выполнить рестарт.

Если задан <файловый дескриптор>, то процесс слежения не запускается, а сервер резервирования будет записывать в файловый дескриптор символ «A» с периодичностью, указанной в ключе /testint. Слежение за работоспособностью сервера резервирования в этом случае может взять на себя внешнее приложение. Отсутствие очередного символа в потоке указывает на то, что отслеживаемый процесс «мертв» (например, произошло зацикливание, «зависание» и т.п.), и требуется перезапуск сервера резервирования.

При использовании ключа /wd необходимо учитывать, что в этом случае на компьютере могут работать два процесса сервера резервирования: один – контролирующий, второй – основной. Для пользовательского приложения необходимо знать, какой из процессов

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

является основным, поскольку управлять работой сервера резервирования оно может посылкой сигналов только основному процессу. Посылка сигнала контролирующему процессу вызовет завершение работы сервера. Для определения идентификатора основного процесса сервера резервирования необходимо использовать ключ `/pid`. При этом в файл, указанный в ключе `/pid`, будет записан идентификатор (pid) основного процесса сервера резервирования. Пользовательское приложение должно прочитать из файла и сохранить у себя этот идентификатор для последующего управления работой сервера резервирования.

Примеры

```
-wd=E
/wd=5
```

Скорость подстройки внутреннего времени системы резервирования (tcorrect)

Синтаксис

```
/tcorrect=<процент>
<процент>::=положительное целочисленное значение или значение с
 десятичной точкой.
```

Описание

Задаёт, на сколько процентов ускоряется/замедляется ход внутренних часов системы резервирования по отношению к часам операционной системы при несоответствии внутреннего и системного времени.

Наличие внутренних часов системы резервирования обусловлено следующими причинами. Система резервирования может не использовать часы операционной системы, например, потому, что в случае их скачкообразного перевода (вручную или операционной системой) может произойти ложное срабатывание событий по тайм-аутам. Это приведет к непредсказуемому поведению системы резервирования.

Поэтому система резервирования использует свои внутренние часы с плавной подстройкой к системному времени.

Если ключ не задан, по умолчанию берется 10%.

Значение этого ключа должно быть в пределах 0 – 50. При выходе значения за указанные границы аргумент принимается равным 10.

При скорости подстройки, для примера, 20%, после перевода системных часов на 10 минут вперед, внутренние часы будут идти на 20% быстрее системных, пока не догонят их. Это произойдет через t = 10 * (100 / 20) = 50 минут.

Для отмены автоматической подстройки времени значение <процента> должно быть равным 0. При этом в случае перевода системного времени внутренние часы будут продолжать идти без перевода и подстройки.

Алгоритм подстройки времени:

- внутреннее время высчитывается как сумма базового времени и истекшего интервала некорректируемого времени;

- базовое время сначала равно значению системного времени на момент старта системы резервирования;
- при скачке системного времени базовое время плавно изменяется, пока внутреннее время не подстроится к системному;
- базовое время изменяется так, что измеряемые интервалы растягиваются или сжимаются на количество процентов, заданное в ключе `/tcorrect`;
- подстройка производится только при разнице системного и внутреннего времени больше 0.01 сек.;
- каждый раз, когда суммарный скачок системного времени превышает 0.5 сек., генерируется событие `E_TIME_CHANGE` (код 40) с указанием величины скачка в секундах в формате с плавающей запятой.

Управление запуском компонентов

Синтаксис

```
/sc[=OFF]
```

Описание

При запуске с ключом без указания значения OFF будет производиться принудительный запуск компонентов СУБД ЛИНТЕР (`linternt`, `linter64`, `dbs_tcp`, `dbc_tcp`) как сервисов. Если опция имеет значение OFF – запуск компонентов СУБД ЛИНТЕР как приложений. Если опция /sc не указана, то при запуске сервера резервирования как приложения будут запускаться как приложения и его компоненты. Если же сервер резервирования запускается как сервис, то и компоненты будут запускаться как сервисы.

При запуске системы резервирования как сервиса им создается серия сервисов (LinNetClnt_Backup, LinNetLstr_Backup, LinSQLSrvr_Backup, LinSQLSrvr_Backup_Rapid, LinSQLSrvr_Backup_Testdb) для управления компонентами СУБД ЛИНТЕР.

В дальнейшем при работе системы резервирования эти сервисы стартуют и останавливаются сервером системы резервирования.

Примечание
При запуске с ключом `/sc` пользователь должен иметь права на управление сервисами. Ключ действует только в ОС семейства Windows.

Пример

```
-sc=OFF
```

Создание сервиса

Синтаксис

```
/instsrv=<имя_сервиса>
```

Описание

Устанавливает сервис с именем `<имя_сервиса>` для запуска системы резервирования. При запуске сервиса будут использованы те же аргументы, которые были переданы системе резервирования при установке сервиса (кроме собственно `instsrv`).

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Переменные окружения при запуске программы как сервиса не должны использоваться для настройки. Аналог `sc create`.

Примечание
Ключ действует только в ОС семейства Windows.

Пример

```
-instsrv=LinNetClnt_Backup
```

Удаление сервиса

Синтаксис

```
/delsrv=<имя_сервиса>
```

Описание

Удаляет ранее установленный сервис с именем `<имя_сервиса>`. Аналог `sc delete`.

Примечание
Удаляется сервис по имени. Дополнительных проверок на принадлежность сервиса системе резервирования не производится. Ключ действует только в ОС семейства Windows.

Пример

```
-delsrv=LinNetClnt_Backup
```

Переменные окружения, влияющие на работу системы резервирования

PATH

Задает список путей для поиска исполняемых файлов. Необходимо добавить каталог исполняемых файлов системы резервирования в список, хранимый в данной переменной. После этого становится возможным запуск сервера резервирования без указания полного пути к нему (просто по имени).

Пример (для ОС Linux, ЗОСРВ Нейтрино):

```
PATH=/usr/linter/bin:$PATH; export PATH
```

Пример (для ОС Windows)

```
set PATH=C:\Program Files\Linter\bin;%PATH%
```

SY00

Задает полный путь к каталогу рабочей БД. Если переменная не определена, по умолчанию предполагается, что рабочая БД находится в текущем каталоге сервера резервирования (программы `server`).

Пример (для ОС Linux, ЗОСРВ Нейтрино)

```
SY00=/usr/linter/RESDB; export SY00
```

Пример (для ОС Windows)

```
set SY00=C:\Linter\RESDB
```

ARJPT

Задает полный путь к каталогу архивной БД. В этом же каталоге хранится файл состояния сервера резервирования `state`, относящийся к архивной БД. Если переменная не определена, по умолчанию предполагается значение `$SY00/ARC`.

Пример (для ОС Linux, ЗОСРВ Нейтрино)

```
ARJPT=/usr/linter/RESDB/ARJ; export ARJPT
```

Пример (для ОС Windows)

```
set ARJPT=C:\Linter\RESDB\ARJ
```

SERVER_HOME

Задает полный путь к каталогу файлов трассировки системы резервирования. Если переменная не определена, по умолчанию используется значение переменной окружения `SY00` или каталог рабочей БД.

Пример (для ОС Linux, ЗОСРВ Нейтрино)

```
SERVER_HOME=/usr/linter/RESDB/HOME; export SERVER_HOME
```

Пример (для ОС Windows)

```
set SERVER_HOME=C:\Linter\RESDB\HOME
```

HOTRES_LOG

Определяет уровень трассировки работы сервера резервирования. Значения переменной аналогичны значениям ключа `/debug` командной строки запуска сервера резервирования, но имеют приоритет перед ним. Без необходимости значение этой переменной устанавливать не рекомендуется.

Пример (для ОС Linux, ЗОСРВ Нейтрино)

```
HOTRES_LOG=0xFFFFFFFF; export SERVER_HOME
```

Пример (для ОС Windows)

```
set HOTRES_LOG=0xFFFFFFFF
```

LINTER_MBX

Задает идентификатор механизма межпроцессного обмена ядра СУБД ЛИНТЕР с клиентскими приложениями. Значение этой переменной должно быть числовым в диапазоне от 1 до 65535. Оно не должно совпадать ни с одним из значений для существующих почтовых ящиков или других механизмов обмена. Значение по умолчанию – 20561.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

NET_MBX

Задает идентификатор механизма межпроцессного обмена клиентского приложения с сетевым драйвером клиента (`dbc_tcp`). Значение этой переменной должно быть числовым в диапазоне от 1 до 65535. Оно не должно совпадать ни с одним из значений для существующих почтовых ящиков или других механизмов обмена. Значение по умолчанию – 20562.

SRVCMD_LOG

Определяет уровень трассировки работы утилиты `srvcmd`. Значения переменной аналогичны значениям ключа `/debug` командной строки запуска сервера резервирования. Однако трассировка выполняется в файл `srv_cmd.log`. Без необходимости значение этой переменной устанавливать не рекомендуется.

Завершение работы системы резервирования

Работа системы резервирования завершается с остановом управляющих программ системы резервирования. Управляющие программы могут быть остановлены по отдельности, либо все одновременно.

В первом случае желателен останов сначала всех управляющих программ SLAVE-серверов, а потом уже и главного сервера.

Примечание
При завершении работы и смене состояний возможно завершение `lhb` с ненулевым статусом.

Завершение работы отдельной управляющей программы системы резервирования

Работу управляющей программы системы резервирования можно завершить сигналами SIGINT, SIGTERM, SIGQUIT, если они не были определены в ключе `/wp` или в других ключах. Также останов управляющей программы может быть выполнен удаленно командой `stop` утилиты `hresctl` (`srvcmd`).

Существует несколько ограничений на останов управляющей программы системы резервирования:

1) сигнал на останов игнорируется, если одновременно выполнены условия:
   - сервер – главный и запущен с ключом `/testslave`;
   - в системе имеются резервные серверы, находящиеся в состоянии первоначального копирования БД.
2) сигнал принимается и останов произойдет после копирования БД, если:
   - сервер является резервным сервером;
   - запущен с ключом `/testslave`;
   - находится в состоянии первоначального копирования БД.
3) команда `stop` на останов отвергается, если одновременно выполнены условия:
   - сервер является резервным сервером;

- находится в состоянии первоначального копирования БД.

Возвращается соответствующий код завершения.

4) команда `stop` на останов отвергается, если одновременно выполнены условия:
   - сервер является главным сервером;
   - в системе имеются резервные серверы, находящиеся в состоянии первоначального копирования БД.

   Возвращается соответствующий код завершения.

В случае возникновения указанных выше ситуаций попытка останова управляющей программы должна быть повторена позднее.

Завершение работы всех управляющих программ

Для завершения работы всех управляющих программ системы резервирования необходимо процессу управляющей программы послать сигнал TRAP (например, с помощью системной утилиты `kill` для ОС Linux, ЗОСРВ Нейтрино) или выполнить команду `shut` в утилите `hresctl` (`srvcmd`).

В некоторых случаях сигнал или команда на завершение могут игнорироваться или быть отложены.

Сигнал должен быть послан именно процессу управляющей программы, а не процессу слежения (см. ключ /wd) и не процессам обработки событий (см. ключ /nostprocqueue). Идентификатор управляющего процесса может быть получен из файла, куда он сохраняется при использовании ключа `/pid`.

Конфигурирование и настройка системы резервирования

Перед первым запуском системы резервирования необходимо выполнить её конфигурирование и настройку. Эта процедура предполагает выполнение на каждом компьютере системы резервирования следующих операций:

1) синхронизацию времени данного компьютера с остальными компьютерами системы резервирования;
2) настройку переменных окружения системы резервирования;
3) редактирование или создание файла сетевой конфигурации `nodetab`;
4) при необходимости (если БД еще не существует ни на одном из компьютеров системы резервирования), создание новой БД на одном из компьютеров системы резервирования или перенос на него уже существующей БД.

Некоторые из этих действий, такие, как подстройка времени и настройка переменных окружения, могут производиться перед каждым запуском управляющей программы системы резервирования.

Синхронизация времени

Для работы системы резервирования нет необходимости в синхронизации времени на её серверных и клиентских компьютерах. Однако для удобства эксплуатации

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

рекомендуется установить одинаковое время на всех компьютерах системы резервирования с точностью до нескольких секунд.

Синхронизация времени (по Гринвичу) выполняется средствами операционной системы до запуска управляющей программы системы резервирования.

Настройка переменных окружения

Система резервирования может функционировать без настройки переменных среды окружения. В этом случае необходимо указывать полный путь для запуска сервера резервирования и запуск производить из каталога рабочей БД.

Однако установка переменных среды окружения повышает удобство эксплуатации системы резервирования, с их помощью можно менять поведение сервера резервирования.

Рекомендуется устанавливать значения переменных окружения `PATH`, `SY00`, `SERVER_HOME`.

Создание и настройка файла сетевой конфигурации

На всех серверах системы резервирования и каждом клиентском компьютере должен присутствовать файл сетевой конфигурации `nodetab` (его стандартное местоположение на серверном компьютере – подкаталог `bin` установочного каталога СУБД ЛИНТЕР), среди записей которого обязательно должны присутствовать строки вида:

```
<узел1> TCPIP <адрес1> <порт1> <тайм-аут1> <тайм-аут2> <тайм-аут3>
<узел2> TCPIP <адрес2> <порт2> <тайм-аут1> <тайм-аут2> <тайм-аут3>
…,
<узелN> TCPIP <адресN> <портN> <тайм-аут1> <тайм-аут2> <тайм-аут3>
<имя>   REZ   <узел1> <узел2>    …        <узелN>
```

где:

`<узел1>, <узел2>, …<узелN>` – логические имена узлов, входящих в систему резервирования. Для каждого сервера может быть задано несколько узлов – по количеству линий связи к нему;

`<адрес1>, <адрес2>, … <адресN>` – сетевые адреса узлов системы резервирования;

`<порт1>, <порт2>, … <портN>` – IP-порты узлов системы резервирования;

Номера портов для всех узлов одного сервера должны быть одинаковые.

`<имя>` – условное имя собственно системы резервирования. Оно (для узла связи с протоколом `rez`) необходимо только для работы системы горячего резерва и не несет на себе никакой функциональной нагрузки;

`<тайм-аут1>` – тайм-аут передачи в минутах;

`<тайм-аут2>` – тайм-аут приема в секундах;

`<тайм-аут3>` – тайм-аут установки соединения в секундах.

В файле nodetab должны быть описаны все сетевые интерфейсы всех серверов системы резервирования, которые могут принимать участие в передаче данных между серверами. Для одного сервера резервирования может быть задано несколько интерфейсов.

Для корректной работы системы резервирования должны быть заполнены все поля каждого узла системы, так как принятые в сетевых средствах СУБД ЛИНТЕР значения по умолчанию для протокола TCPIP (TCPIPS) не подходят для работы системы резервирования.

Файлы nodetab должны быть абсолютно одинаковы на всех серверах системы резервирования. В файле должен присутствовать один и только один узел с протоколом rez. В качестве адреса в этом протоколе должны быть перечислены все узлы системы резервирования. Не рекомендуется описывать в файле nodetab серверного компьютера узлы, которые не предполагается включать в список узлов с протоколом rez.

Настройка клиентских компьютеров

На каждом клиентском компьютере должен быть запущен сетевой драйвер клиента (dbc_tcp) и пользовательское приложение. Настройка dbc_tcp заключается в создании файла конфигурации nodetab.

В клиентский файл сетевой конфигурации (nodetab) кроме вышеуказанных данных в файлах сетевой конфигурации серверов необходимо добавить запись для доступа к главному серверу системы резервирования:

```
<имя2> LASSP <узел1> <узел2> … <узелN>
```

где:

<имя2> – логическое имя главного узла системы горячего резерва.

Клиентское приложение может работать:

- с единственной системой резервирования;
- с несколькими системами резервирования;
- с системой резервирования и другими серверами СУБД ЛИНТЕР.

Работа с единственной системой резервирования

В данном варианте клиентские приложения, запущенные на одном или нескольких компьютерах, работают с несколькими серверами резервирования, объединенными в целостную систему резервирования. Из всех серверов резервирования только один является главным.

В этом случае файл nodetab для клиентских компьютеров аналогичен серверному. Отличие состоит в том, что в файле nodetab должны быть перечислены только узлы серверов резервирования, входящих в систему резервирования (то есть узлы, не принадлежащие ни одному серверу резервирования, должны отсутствовать). Строка с протоколом rez на клиентском компьютере не обязательна, но может присутствовать. Файл настройки nodetab должен располагаться в одном из каталогов поиска конфигурационного файла сетевым драйвером клиента, или же путь к файлу nodetab должен быть указан явно в значении ключа запуска dbc_tcp (см. документ «Сетевые средства»).

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

На клиентском компьютере должен быть запущен драйвер сетевого клиента dbc_tcp с ключом -s=<имя2> (см. документ «Сетевые средства»). При обращении прикладной программы к узлу по умолчанию (пробелы в качестве имени сервера) сетевой драйвер клиента dbc_tcp автоматически устанавливает по сети соединение с активным в данный момент ядром СУБД ЛИНТЕР и осуществляет передачу ему запросов и прием от него ответов. При потере соединения dbc_tcp будет пытаться установить связь с новым главным сервером из числа узлов, указанных в файле nodetab, но клиентскому приложению необходимо повторно выполнить подключение.

При необходимости можно задать драйверу ключ -ERR1001 для получения приложением кода завершения «Нет активного ядра СУБД ЛИНТЕР» в случае невозможности установки соединения ни с одним из серверов.

Например, командная строка запуска драйвера может выглядеть так:

```
>dbc_tcp -s=<имя2> -ERR1001
```

Одновременная работа с несколькими системами резервирования

При необходимости одновременной работы с системой резервирования и другим сервером или несколькими системами резервирования на клиентском компьютере нужна специальная настройка файла nodetab с использованием протокола LASSP (более подробно о протоколе LASSP см. в документе «Сетевые средства»).

В этом файле должны быть перечислены узлы всех серверов всех систем резервирования и отдельных серверов СУБД ЛИНТЕР, как это делается для обычного сетевого доступа. Нужно только учитывать, что обнаружение отказа любого сервера происходит по тайм-ауту, и, следовательно, указание тайм-аутов является обязательным. После перечисления всех узлов обычным образом необходимо определить, какие из них объединены в системы резервирования. При этом необходимо учитывать ограничение: один обычный узел может входить только в одну систему резервирования или быть обычным сетевым ЛИНТЕР-сервером.

Узлы, входящие в одну и ту же систему резервирования, объединяются с использованием протокола LASSP. Для этого после перечисления всех обычных узлов заводится одна или несколько строк с описанием узлов LASSP по одной для каждой системы резервирования. Первое поле этих строк, как обычно, содержит имя узла. Второе поле содержит имя протокола LASSP. В последующих полях должны быть перечислены через пробел все имена узлов данной системы резервирования.

Например, файл nodetab может выглядеть следующим образом:

```
SRVA1 TCPIP SRVA1 1060  1 10 10
SRVA2 TCPIP SRVA2 1060  1 10 10
SRVB1 TCPIP SRVB1 1060  1 10 10
SRVB2 TCPIP SRVB2 1060  1 10 10
SRV   TCPIP SRV   1060  1 10 10
SRVA  LASSP SRVA1 SRVA2
SRVB  LASSP SRVB1 SRVB2
```

В приведенном примере узлы SRVA1 и SRVA2 принадлежат одной системе резервирования и объединены в узел SRVA, узлы SRVB1 и SRVB2 принадлежат второй системе резервирования и объединены узлом SRVB. SRV – это отдельный ЛИНТЕР-сервер, не объединенный ни в одну из систем резервирования.

Пользовательское приложение должно обращаться к системам резервирования и обычным ЛИНТЕР-серверам по их именам, перечисленным в файле `nodetab`. Для обращения к системе резервирования используется имя узла с протоколом LASSP. К сетевому ЛИНТЕР-серверу обращение происходит как обычно, то есть по имени его узла.

Обращение к узлу по умолчанию будет осуществляться обычным способом: к локальному ЛИНТЕР-серверу или к первому в списке `nodetab` сетевому узлу. Поэтому такое обращение в случае работы с несколькими системами резервирования недопустимо. Указание имени узла при открытии канала является обязательным. Обращение к серверу по умолчанию может быть использовано для коммуникации с локальным ЛИНТЕР-сервером.

Выбор временных интервалов

В большинстве случаев требуется быстрая реакция системы резервирования на отказ одного из серверов. Для этого при запуске сервера резервирования необходимо установить в минимальное значение интервал посылки тестовых пакетов и максимальное количество пропущенных тестовых пакетов.

Время реакции системы резервирования на нештатную ситуацию равно произведению интервала тестовой посылки (ключ `/testint`) и максимального количества пропущенных пакетов (ключ `/tstlimit`). Необходимо учитывать, что при посылке UDP-пакетов один или даже несколько подряд посланных пакетов могут потеряться (сеть не гарантирует доставку), поэтому устанавливать количество пропущенных пакетов меньше 3 – 4 не рекомендуется. В такой ситуации время реакции лучше регулировать уменьшением интервала посылки.

Интервал посылки может быть задан в командной строке не только целым числом, но и вещественным (то есть реально можно задать интервал посылки в одну десятую секунды). Минимальный интервал необходимо подбирать опытным путем на реальной сети при реальной нагрузке на неё.

Для этого:

1) запустить на выполнение какие-либо задачи, эмулирующие максимальную нагрузку сети. Количество задач должно быть достаточным для тестирования устойчивости системы резервирования при повышенных нагрузках на сеть;
2) создать БД такого размера, чтобы её первоначальное копирование с главного компьютера на резервный компьютер без нагрузки на сеть было порядка нескольких минут;
3) запустить систему резервирования с выбранным для эксперимента (эмпирическим путем) интервалом посылки и количеством пакетов. Если соединение между серверами устойчиво и первоначальное копирование БД прошло успешно с первой попытки, то такое время реакции системы резервирования приемлемо в данной конфигурации сети;
4) уменьшить время реакции системы резервирования, например, в 2 раза, и повторить проверку её работы. Если хотя бы один из резервных серверов перешел за время проверки из SLAVE-состояния в SLAVE_WAIT-состояние (повторил первоначальное копирование БД) или вообще завершил работу, значит, выбранное время реакции является ошибочным. Его необходимо увеличить в несколько раз и повторить попытку. Таким образом, экспериментальным путем подбирается минимальное время реакции системы резервирования для конкретной сети. Для надежности найденное значение можно увеличить в 2 раза.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Время реакции системы резервирования необходимо учитывать при подготовке файла `nodetab`. В нем значения всех тайм-аутов должны быть больше времени реакции системы резервирования. Необходимо также учитывать, что минимальные тайм-ауты в файле `nodetab` должны быть не менее 4 секунд, иначе работа сетевых драйверов СУБД ЛИНТЕР может стать нестабильной.

Тайм-аут проверки ядра СУБД ЛИНТЕР (примечание к ключу `/treq`) должен выбираться, исходя из особенностей пользовательского приложения и остальных настроек сервера резервирования:

1) не рекомендуется выбирать тайм-аут меньше, чем учетверенный интервал посылки тестовых сообщений, поскольку все временные операции сервера резервирования тактируются с периодичностью интервала посылки;
2) интервал посылки тестовых запросов ядру СУБД ЛИНТЕР рекомендуется выбрать равным четвертой части тайм-аута СУБД ЛИНТЕР;
3) тайм-аут ядра СУБД ЛИНТЕР рекомендуется выбирать немного больше, чем время выполнения самого длительного запроса приложения при максимальной нагрузке на ядро;
4) при выборе тайм-аута ядра необходимо учитывать возможное накопление данных в процессе работы системы резервирования и, если возможно, проверять длительность выполнения пользовательских запросов на реальных объемах данных и в окружении всех прикладных задач (при увеличении объемов время исполнения запроса также увеличивается). В связи с тем, что минимальный интервал посылки тестового запроса ядру СУБД ЛИНТЕР не может быть установлен меньше 1 секунды, тайм-аут ядра СУБД ЛИНТЕР не рекомендуется выбирать менее 3-4 секунд;
5) тайм-аут ожидания останова ядра СУБД ЛИНТЕР (ключ `/forceshut`) должен выбираться с таким расчетом, чтобы обеспечивать передачу всех данных на резервный сервер после получения ядром СУБД ЛИНТЕР команды останова.

   Для определения этого тайм-аута необходимо:

   - запустить систему резервирования с пиковой (максимальной) нагрузкой;
   - предоставить главному серверу возможность отработать в период пиковой нагрузки в реальной системе;
   - после этого подать главному серверу команду останова. Время, за которое главный сервер завершит свою работу, и будет рекомендуемым тайм-аутом ожидания останова. Для надежности это время необходимо увеличить в полтора-два раза;
6) интервал ожидания готовности всех серверов резервирования (ключ `/wait`) должен быть больше, чем разница во времени готовности компьютеров резервирования после включения. Если автоматический запуск системы резервирования запрещен, то этот ключ игнорируется, так как момент запуска системы резервирования будет определяться разрешением на запуск (администратором системы резервирования или сигналом). Если разрешен автоматический запуск системы резервирования, то данный тайм-аут должен гарантировать запуск всех компьютеров системы резервирования до его истечения.

Создание файла регистрационных данных администратора

Если регистрационные данные администратора системы резервирования отличаются от стандартных (`SYSTEM/MANAGER8`), то необходимо создать файл, где хранятся его имя и пароль (см. ключ /pf).

Создание файла ключей запуска

Если командная строка запуска управляющей программы слишком длинная, или необходимо сохранить в секрете ключи запуска программы от пользователей ОС, может использоваться файл ключей запуска (см. ключ /cf).

В данном файле, кроме перечисления ключей, могут быть определены и переменные окружения, поэтому файл ключей запуска может использоваться как альтернатива командного интерпретатора ОС.

© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Запуск системы резервирования

Первоначальный запуск

Запуск системы резервирования сводится к запуску управляющей программы `server` на каждом из серверов системы резервирования. Перед первым запуском системы резервирования необходимо выполнить ее настройку.

Настройка системы резервирования предполагает настройку серверов резервирования и клиентов, работающих с системой резервирования.

Ниже приведены действия, выполнение которых может понадобиться при первоначальном запуске системы резервирования.

Синхронизация времени

Перед запуском системы резервирования необходимо синхронизировать время на всех серверах резервирования средствами ОС.

Установка переменных окружения

Перед запуском системы резервирования желательна установка переменных окружения `PATH`, `SY00`, `SERVER_HOME`:

1) `PATH` – добавить путь каталога исполняемых файлов СУБД ЛИНТЕР;
2) `SY00` – установить путь каталога, содержащего БД;
3) `SERVER_HOME` – установить путь каталога для хранения файлов состояния системы резервирования и файлов трассировки.

Пример (для ОС Linux, ЗОСРВ Нейтрино)

```
PATH=/usr/linter/bin:$PATH; export PATH
SY00=/usr/linter/db; export SY00
SERVER_HOME=/usr/linter/hres
```

Пример (для ОС Windows)

```
set PATH=C:\Program Files\Linter\bin;%PATH%
set SY00=C:\Linter\db
set SERVER_HOME=C:\Linter\hres
```

Создание БД

Если на компьютерах системы резервирования отсутствуют БД, то они могут быть созданы с помощью утилиты `gendb` СУБД ЛИНТЕР (см. документ «Создание и конфигурирование базы данных»). БД создается в рабочем каталоге на компьютере, который при первом запуске станет главным (на остальных компьютерах системы резервирования БД не создается).

БД, используемая в системе резервирования, имеет следующее ограничение: все её файлы должны располагаться в одном каталоге, заданном переменной окружения `SY00`. Недопустимо размещение каких-либо файлов БД в разных каталогах.

Пример создания новой БД:

```
>gendb
gendb>create database "DATABASE";
gendb>exit
```

Может использоваться и готовая БД. В этом случае достаточно установить переменную SY00, указывающую на каталог БД, или скопировать файлы существующей БД в каталог SY00.

Создание файла сетевой конфигурации

Создание файла сетевой конфигурации nodetab описано в пункте «Создание и настройка файла сетевой конфигурации».

Например, он может выглядеть следующим образом:

```
SRV1 TCPIP srv1.org.com 1061 1 10 10
SRV2 TCPIP srv2.org.com 1061 1 10 10
DEMO REZ SRV1 SRV2
```

Файл nodetab обычно размещают в каталоге исполняемых файлов СУБД ЛИНТЕР. Но если он размещен в другом месте, то необходимо указывать путь к нему при запуске управляющей программы системы резервирования.

Запуск управляющей программы системы резервирования на главном сервере

После выполнения предварительной настройки (синхронизация времени, установка переменных окружения, создание файла сетевой конфигурации) на всех серверах резервирования может быть осуществлен запуск управляющей программы на главном сервере (сервере с существующей БД).

Запуск управляющей программы заключается в запуске программы server. Первоначальный запуск рекомендуется осуществлять с ключом /debug (при условии установки переменной окружения SY00):

```
>server /debug /pf=name.txt /port=1061
```

где:

/pf=name.txt – спецификация файла с именем пользователя и паролем (по умолчанию используются - SYSTEM/MANAGER8);

/port=1061 – порт сервера резервирования (по умолчанию используются порт 1060).

Непосредственно после запуска будут выведены сообщения о состоянии системы резервирования.

Примерно через 30 секунд на терминал выведется запрос:

```
Do you want to start  another server?
```

В ответ на этот запрос ввести **N** и нажать клавишу **Enter**.

Сервер резервирования должен перейти к фоновому режиму работы и вывести строку "Server new state is MONO", что свидетельствует об успешном запуске управляющей программы на главном сервере.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Запуск управляющей программы системы резервирования на резервных серверах

После успешного запуска управляющей программы на главном сервере производится запуск управляющих программ системы резервирования на резервных серверах. Порядок запуска резервных серверов может быть произвольным.

Для запуска управляющей программы на резервном сервере необходимо использовать ту же командную строку, что и для главного сервера:

```
>server /debug
```

После запуска управляющая программа резервного сервера переходит в фоновый режим работы. После копирования БД с главного сервера (может занимать от нескольких секунд до нескольких минут) на консоль выводится сообщение:

```
"Server is in SLAVE_OK state. It is ready."
```

Это означает переход резервного сервера в режим готовности.

Повторный запуск

Повторный запуск системы резервирования (после принудительного останова системы резервирования или после сбоя) аналогичен первоначальному и предполагает выполнение следующих операций:

1) синхронизацию времени серверов резервирования;
2) настройку переменных окружения;
3) запуск программы `server` на компьютерах системы резервирования.

Некоторые подготовительные операции, необходимые при первоначальном старте, исключаются. В частности, нет необходимости в создании БД и файла сетевой конфигурации `nodetab`. Кроме того, запуск управляющих программ может осуществляться одновременно или с небольшой задержкой на всех серверах резервирования.

При повторных запусках возможно добавление дополнительных ключей, модифицирующих работу системы резервирования. Рекомендуется явно задавать следующие ключи:

- `/ntab` – для указания пути к файлу сетевой конфигурации `nodetab`;
- `/pathtodb` или переменную окружения `SY00` – для указания пути к рабочей БД;
- `/pathtoarc` – для указания пути к файлу архивной БД;
- `/testint`, `/tstlimit`, `/treq`, `/forceshut` – для задания периодичности проверки активности линий связи (определяется как `/testint*/tstlimit`), максимального времени выполнения пользовательского запроса и максимального времени пересылки не переданных с главного сервера на резервный сервер данных при останове главного сервера;
- `/pool` – для установки размера оперативной памяти, выделяемой для работы ядра СУБД ЛИНТЕР;
- `/stproc`, `/pid`, `/wp` – в случае управления работой системы резервирования не администратором системы резервирования, а специальной программой;

- /wd – рекомендуется задавать всегда. При управлении системой резервирования из программы может потребоваться указывать также и значение этого ключа;
- /nservers – при наличии более одной линии связи у каждого компьютера системы резервирования;
- /exchdir – включен по умолчанию. /exchdir=0 не рекомендуется использовать как устаревшее;
- /debug – для сохранения протокола работы системы резервирования. В отдельных случаях, например, при возникновении непонятного поведения системы резервирования, необходимо задавать более детальный уровень трассировки.

В случае повторного запуска системы резервирования (то есть после того, как система резервирования функционировала, а затем её работа по каким-то причинам была остановлена) следует быть осторожным при использовании ключей запуска /nq и /nservers.

Если первым запустить сервер с ключом /nq и устаревшей БД, то он может стартовать как главный без запроса на старт, и в рабочие каталоги остальных серверов будет скопирована устаревшая БД.

При старте серверов в неполном составе с уменьшенным против реального значением ключа /nservers и при отсутствии в их составе сервера с наиболее свежей БД, система начнет работать с устаревшей БД; при последующем подключении сервера со свежей БД свежая БД будет перемещена в архивный каталог, а на ее место запишется более старая БД, что приведет к потере информации.

Безопасные способы повторного запуска:

1) запустить серверы одновременно в полном составе без ключа /nq и задать /nservers=<количеству запускаемых серверов>. При этом система сама произведет ранжирование серверов и запустит сервер со свежей БД как главный;
2) запустить первым сервер с наиболее свежей БД и, когда он перейдет в режим MONO, запустить остальные серверы системы резервирования.

Для повторного запуска системы резервирования рекомендуется создать скрипт, который выполняет все подготовительные действия и производит запуск управляющей программы.

Просмотр профиля сервера резервирования

Для просмотра текущего состояния сервера резервирования после принудительного останова или во время работы можно выполнить на компьютере сервера команду:

```
server -show
```

По данной команде на компьютере запускается управляющая программа server, которая выдает на консоль информацию о текущем состоянии сервера резервирования и завершает свою работу.

Выдается следующая информация:

```
Inactive time of the server = <время, прошедшее после последнего
 обновления БД>
Current  time = <текущая системная дата и время>
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

```
NODETAB found: <путь к файлу nodetab>
Archive file <имя файла архива и его наличие>
Work database <каталог рабочей БД>
Last work database state=<состояние рабочей БД>
Work database is <наличие рабочей БД>
Last work database time=<дата последней работы с рабочей БД>
<Результат тестирования рабочей БД и дата запуска testdb>
Backup database <каталог архивной БД>
Last backup database state=<состояние архивной БД>
Backup database is <наличие архивной БД>
Last backup database time=<дата последней работы с архивной БД>
<Результат тестирования архивной БД и дата запуска testdb>
Database size=<размер рабочей БД>
Backup database size=<размер архивной БД>
server state=<текущее состояние сервера резервирования>
<Информация о работе сервера резервирования>
```

В связи с тем, что в файле состояния сервера `state` фиксируются исключительно главные состояния БД, а не сервера (переходные состояния игнорируются), параметр `<текущее состояние сервера резервирования>` может принимать только следующие значения:

```
MONO, MAIN, SLAVE, SLAVEFAILER, SLAVECRASH, MAINFAILER, MAINCRASH
```

Примечание

Если система резервирования работает в режиме обмена каталогов (ключ `/exchdir`), то для правильного отображения функционального назначения каталогов ключи `/show` и `/exchdir` необходимо применять совместно.

Получить информацию о текущем состоянии работающей системы резервирования можно также с использованием ключа `-getstate` утилиты `hresctl` (`srvcmd`). Результат выполнения этой команды аналогичен результату программы `server` с ключом `/show`, но с помощью утилиты можно просматривать профиль как локального, так и удаленного сервера резервирования.

Пример выдаваемого по ключу /show профиля сервера резервирования

```
The "hotreserve.conf" has not been found
Inactive time of the server = 5711 min

Current  time = Mon May 12 14:46:43 2008
NODETAB found:/usr/linter/bin/nodetab
Archive file  /usr/linter/db/ARC/DB.zip is present.
Its time is Wed Apr 02 17:46:48 2008

Work database /usr/linter/db
Last work database state    = MONO
Work database is              PRESENT
Last work database time     = Thu May 08 15:35:24 2008
Work database has not been checked
```

```
Backup database /usr/linter/db/ARC
Last backup database state    = UNDEFINED
Backup database is              PRESENT
Last backup database time     = Thu Jan 01 00:00:01 1970
Backup database has not been checked

Work database size = 3084288
Backup database size = 3084288
server state = UNDEFINED

The server is not started
```

Примеры запуска системы резервирования

Примеры приведены для ОС Linux, ЗОСРВ Нейтрино.

Исходные данные:

- в системе резервирования два сервера, каждый из которых имеет один сетевой интерфейс;
- рабочая БД главного сервера будет размещена в каталоге `/usr/linter/RESDB`;
- исполняемые файлы СУБД ЛИНТЕР находятся в каталоге `/usr/linter/bin`;
- архивная БД должна размещаться в каталоге `/usr/linter/RESDB/ARJ`;
- файл состояния рабочей БД главного сервера и файлы трассировки будут размещаться в каталоге `/usr/linter/RESDB/HOME`.

Первоначальный запуск на главном сервере

Шаг 1. Создание несуществующих каталогов

```
>mkdir /usr/linter/RESDB
>mkdir /usr/linter/RESDB/ARJ
>mkdir /usr/linter/RESDB/HOME
```

Шаг 2. Установка переменных окружения

```
SY00=/usr/linter/RESDB; export SY00
PATH=/usr/linter/bin:$PATH; export PATH
ARJPT=/usr/linter/RESDB/ARJ; export ARJPT
SERVER_HOME=/usr/linter/RESDB/HOME; export SERVER_HOME
```

Шаг 3. Если БД еще не существует, создание БД с именем DEMO

```
>gendb
gendb>create database DEMO;
gendb>exit
```

Шаг 4. Пусть серверы резервирования имеют сетевые интерфейсы с адресами SRV1, SRV2. Создание (добавление) в файл `/usr/linter/bin/nodetab` строк:

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

```
S1   TCPIP  SRV1  1060  1  20  20
S2   TCPIP  SRV2  1060  1  20  20
DEMO REZ    S1    S2
```

Шаг 5. Запуск главного сервера с регистрацией событий

```
>server  -exchdir -debug
```

Шаг 6. Подождать истечения интервала обнаружения других серверов системы резервирования. При этом сервер резервирования выведет запрос на ожидание старта всех серверов системы: "`Do You want to start another server?(Y/N/E):`". Необходимо ответить "`N`" на данный запрос. Затем сервер резервирования должен продолжить работу в фоновом режиме и перейти в режим `MONO`. Об успешном старте в режиме `MONO` сервер выведет сообщение на консоль: "`Server is in MONO state`".

Первоначальный запуск на резервном сервере

Описанная ниже процедура запуска применима как к первоначальному, так и повторному запуску сервера резервирования на резервном компьютере.

Шаг 1. Аналогичен шагу 1 из пункта «Первоначальный запуск на главном сервере».

Шаг 2. Аналогичен шагу 2 из пункта «Первоначальный запуск на главном сервере».

Шаг 3. Проверить наличие на компьютере как самого файла `nodetab`, так и строк, задающих в нем сетевую конфигурацию системы резервирования.

Примечание
Файлы `nodetab` должны быть одинаковы на всех компьютерах серверов резервирования.

Шаг 4. Убедиться, что главный сервер работает (если нет – запустить его).

Шаг 5. Запустить сервер резервирования на резервном компьютере

```
>server -debug -exchdir
```

Шаг 6. Дождаться синхронизации рабочих БД главного и резервного серверов. При её успешном завершении на консоль будет выдано сообщение:

```
Server is in SLAVE_OK state. It is ready.
```

В результате главный сервер должен перейти в состояние `MAIN`.

Пример командного файла запуска системы резервирования

Пример приведен для ОС Linux, ЗОСРВ Нейтрино. Повторные запуски системы резервирования рекомендуется производить специальным командным файлом. Ниже приведен пример такого командного файла.

```
#!/bin/sh
SERVER_HOME=/usr/linter/hres
SY00=/usr/linter/db
PATH=$PATH:/usr/linter/bin
```

```
export SERVER_HOME SY00 PATH

server  /testint=5 /tstlimit=6 /pool=16000 /nservers=2 /wait=10 /
input=120 /lintadd="/tracelog" /stproc /treq=120 /forceshut=3600 /
down=1 /crash /wp /testdb=sig14 /watchnet /exchdir
# /wd /debug /nq /pid="$SERVER_HOME"/server.pid  /
BARC="$SERVER_HOME"/db.tar.bz2 /input=120 /WAIT=10 /prior=baas /
cmdarcadd='/usr/linter/bin/arc.sh %LIST% %ARCHIVE%'
```

В примере используется командный файл создания архива `arc.sh`. Он может выглядеть следующим образом:

```
#!/bin/sh -x

tar cf - -T $1 | bzip2 > $2
#cat $1 | zip  $2.zip -@
exit $?
```

В данном примере приведено использование как архиватора `tar` совместно с `bzip2`, так и архиватора `zip` (в комментарии).

© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Внешнее управление системой резервирования

Система резервирования СУБД ЛИНТЕР предусматривает два варианта режима управления её работой:

1) полностью автоматический: после настройки и запуска все функции по управлению системой резервирования берут на себя управляющие программы;
2) полуавтоматический: часть функций по управлению системой резервирования выполняется администратором этой системы или внешней программой (например, останов работы сервера резервирования, обмен состояниями серверов, останов всей системы резервирования).

Управление может осуществляться удаленно по IP-протоколу с помощью программ `hresctl` (`srvcmd`) или сигналами.

Описание удаленного управления системой резервирования и управления сигналами приведено ниже.

Управление с помощью программ hresctl и srvcmd

Для удаленного управления системой резервирования предназначены утилиты `hresctl` и `srvcmd`.

Удаленное управление осуществляется с помощью команд, передаваемых по локальной сети по протоколу UDP управляющей программе (в частном случае, если указан адрес локального интерфейса, выполняется управление локальным сервером). Каждый запуск утилиты соответствует передаче одной команды управления и ожидания ответа от сервера резервирования в течение тайм-аута. Для программы `srvcmd` тайм-аут ожидания постоянен и равен 5 секундам. Для программы `hresctl` тайм-аут задается в значении ключа `/t`.

Так как протокол UDP не гарантирует доставку сетевых пакетов на удаленный сервер, то в случае потери сетевого пакета команда должна быть повторена (путем повторного запуска программы).

Примечание
В дистрибутиве СУБД ЛИНТЕР для ОС Windows поставляется только программа `hresctl`.

Для программы `srvcmd` имеется возможность получения файла трассировки путем установки переменной окружения SRVCMD_LOG=0xFFFFFFFF (см. описание переменной окружения SRVCMD_LOG).

Программа `hresctl` при старте информирует о параметрах запуска:

```
"command = show" – текущая команда;
"timeout (sec) = 5 " – таймаут ожидания ответа;
"HostAddr=localhost" – IP адрес сервера (может быть опущен);
"Port=1060" – порт сервера.
```

Передаваемая команда задается с помощью аргументов командной строки.

Далее по тексту при указании программы `hresctl` также подразумевается программа `srvcmd`.

Командная строка запуска утилиты имеет следующий формат:

```
hresctl -<команда> [-u=<пользователь/пароль>] [-t=<значение>]
 [<адрес сервера> [<номер порта>]]
```

Параметры командной строки:

1) `<команда>`:
   - `getstate|show` – опросить текущее состояние сервера;
   - `stop` – остановить сервер резервирования;
   - `shut` – остановить все серверы системы резервирования;
   - `exch` – изменить статус сервера резервирования;
   - `main` – перевести резервный сервер в режим главного;
   - `slave` – перевести главный сервер в режим резервного;
   - `testdb` – протестировать БД. Команда может быть выполнена только SLAVE-сервером;
   - `copy` – скопировать БД в архивный каталог. Команда может быть выполнена только SLAVE-сервером;
   - `arc` – запустить на выполнение процесс архивирования. Путь архива и архиватор задаются из командной строки утилиты `server`, или используются значения по умолчанию. Архивирование запускается, даже если не задан ключ `/archivate` при запуске `server`;
   - `dead_node=<имя узла>` – отсоединить узел системы резервирования;
   - `dead_server=<имя узла>` – отсоединить сервер системы резервирования.
2) `-u=<пользователь>/<пароль>`

   Задает регистрационные данные (имя и пароль) пользователя СУБД ЛИНТЕР. Используется в случае применения защиты сервера от несанкционированного управления (ключ сервера `/cu`).
3) `-t=<значение>`

   Задает длительность тайм-аута ожидания ответа от сервера резервирования.

Примечание
Ключ поддерживается только программой `hresctl`.

4) `<адрес сервера>`

   Задает сетевое имя или адрес узла, на котором работает сервер резервирования. Если адрес не задан, предполагается работа с `localhost (127.0.0.1)`.
5) `<номер порта>`

   Задает номер порта сервера резервирования, который идентичен ключу `/port` при запуске программы `server` (по умолчанию 1060). При обозначении порта обязательно явное указание адреса сервера.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Синтаксические правила:

- квадратные скобки указывают на необязательный элемент командной строки или ключа. Внутри ключа он может быть заменен пробелом или пропущен;
- имена ключей регистронезависимы (можно задавать как большими, так и малыми буквами);
- название ключа предваряется символом "-" (минус);
- значение ключа может отделяться от имени ключа пробелом или символом "=".
- допускается предварять имя ключа символом "/". Однако в этом случае значение должно быть отделено от имени только символом "=".

Если ключ указан неправильно, то утилита не выполнит команду и выведет на консоль краткую информацию о своих ключах.

Примечание
В описании ключей значение в скобках является значением по умолчанию.

Утилита `hresctl` всего лишь передает серверу резервирования соответствующую команду – отработку команды выполняет непосредственно сервер резервирования. Команда может быть отвергнута сервером резервирования, если ее выполнение способно привести к потере данных или противоречит логике работы системы резервирования.

Для команд, изменяющих состояние сервера (всех, кроме `getstate` и `show`) в случае, если сервер работает в защищенном режиме (запущен с ключом `/cu`), необходимо задавать имя пользователя и пароль пользователя СУБД ЛИНТЕР. Этот пользователь к тому же должен быть владельцем БД.

Обмен статусами серверов резервирования

Синтаксис

`exch`

Описание

Сервер резервирования, получивший команду, изменяет свой статус:

- главный сервер становится резервным (если выполняются необходимые логические условия);
- резервный сервер становится главным (если выполняются необходимые логические условия).

Команда на изменение состояния может быть выполнена резервным сервером только в случае готовности всех резервных серверов. Команда не выполняется, если сервер находится в режиме переключения из одного состояния в другое. После получения резервным сервером команды на обмен состоянием (то есть на переход в состояние главного сервера) он посылает команду текущему главному серверу на переход того в состояние резервного сервера, а остальным резервным серверам – команду на рестарт сервера резервирования.

Команда на изменение состояния главным сервером выполняется только в случае присутствия хотя бы одного и готовности всех резервных серверов. Какой из резервных

серверов возьмет на себя роль главного, определяется в результате конкурса, который проводится, как если бы главный сервер вышел из строя.

Остановить работу сервера резервирования

Синтаксис

```
stop
```

Описание

Команда инициирует останов управляющей программы на заданном сервере.

На резервном сервере команда выполняется только в состоянии готовности данного сервера.

Остановить работу системы резервирования

Синтаксис

```
shut
```

Описание

Останавливает работу управляющих программ всех серверов резервирования. Остановка всех управляющих программ означает остановку работы всей системы резервирования.

Команда выполняется только в случае готовности всех резервных серверов.

Получить состояние сервера резервирования

Синтаксис

```
getstate | show
```

Описание

Выводится информация о текущем состоянии сервера, аналогичная выводу по ключу сервера резервирования `/show`.

Команда выполняется для любого пользователя, даже если он не имеет прав администратора или вообще не является пользователем СУБД ЛИНТЕР.

Перейти в режим главного сервера

Синтаксис

```
main
```

Описание

Команда на перевод резервного сервера в режим главного.

Команда применяется только к резервному серверу. Все резервные серверы должны быть в состоянии готовности.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Перейти в режим резервного сервера

Синтаксис

`slave`

Описание

Главный сервер переводится в режим резервного сервера.

Команда применяется к главному серверу и выполняется только в том случае, если существует хотя бы один резервный сервер, и все резервные серверы находятся в состоянии готовности.

Протестировать БД резервного сервера

Синтаксис

`testdb`

Описание

Команда на проверку рабочей БД резервного сервера. Проверка выполняется путем копирования рабочей БД в резервный каталог с последующей проверкой скопированной БД посредством утилиты `testdb`.

Команда применима только к резервному серверу, который должен находиться в состоянии готовности. В случае выполнения резервным сервером в момент получения команды операции проверки или копирования БД команда игнорируется.

Скопировать рабочую БД в архивный каталог

Синтаксис

`copy`

Описание

Команда копирования рабочей БД резервного сервера в архивный каталог.

Команда применима только к резервному серверу. Он должен находиться в состоянии готовности. В случае выполнения резервным сервером в момент получения команды операции проверки или копирования БД команда игнорируется.

Создать архивный файл рабочей БД

Синтаксис

`arc`

Описание

Команда копирования рабочей БД резервного сервера в архивный каталог с последующим архивированием.

Местоположение архивного файла и используемый архиватор определяются на основании соответствующих ключей командной строки запуска сервера резервирования (если ключи не были заданы, используются значения по умолчанию).

Примечание
Команда выполняется даже при отсутствии ключа `/archivate` в командной строке запуска сервера резервирования.

Команда применима только к резервному серверу. Он должен находиться в состоянии готовности. В случае выполнения резервным сервером в момент получения команды операции проверки или копирования БД команда игнорируется.

Отсоединить узел системы резервирования

Синтаксис

```
dead_node=<имя узла>
```

Описание

Команда объявляет заданный узел сервера резервирования отсоединенным, то есть объявляет разорванной (неактивной) линию связи между данным сервером и заданным узлом. При этом реакция системы резервирования на данную команду аналогична ее реакции при самостоятельном обнаружении разрыва линии связи. Эта команда дает возможность управлять обнаружением неисправности серверов и линий связи внешними программами по алгоритмам, отличным от использующихся в системе резервирования. Например, сервер может быть объявлен неисправным в случае выхода из строя какого-либо оборудования, не отслеживаемого сервером резервирования, но использующимся прикладным приложением.

Примечание
Присоединение узла (активизация линии связи) обнаруживается и обрабатывается сервером резервирования автоматически.

Отсоединить сервер системы резервирования

Синтаксис

```
dead_server=<имя узла>
```

Описание

Команда объявляет отсоединенным сервер, которому послана команда, от сервера, принадлежащего заданному узлу, то есть объявляет разорванными (неактивными) все линии связи к отсоединяемому серверу. В качестве <имени узла> необходимо задавать узел, с которым уже установлено соединение. Реакция сервера резервирования на данную команду аналогична его реакции при самостоятельном обнаружении разрыва всех линий связи с каким-либо сервером.

Необходимо учитывать, что принадлежность линии связи серверу определяется на этапе установки соединения. Поэтому, если по линии связи не установлено соединение, то принадлежность данного узла серверу определить невозможно, и команда будет проигнорирована. В качестве аргумента необходимо указывать имя узла, к которому уже установлено соединение.

Команда может быть использована для управления сервером резервирования внешним программным обеспечением.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Сообщения утилиты srvcmd

Утилита `srvcmd` может выдавать диагностические сообщения, если доступ к управлению системой резервирования запрещен.

Linter error <NNNN>

Причина

Ошибка доступа к СУБД ЛИНТЕР.

Способ устранения

Способ устранения зависит от кода завершения <NNNN> (см. документ «Справочник кодов завершения»).

Unknown user name

Причина

Неизвестно или не задано имя пользователя.

Способ устранения

Задать имя пользователя-администратора БД ЛИНТЕР главного сервера.

Invalid password

Причина

Неправильный пароль пользователя.

Способ устранения

Указать правильный пароль.

Inproper system state

Причина

Сервер не может выполнить команду, поскольку находится в процессе запуска.

Способ устранения

Повторить команду через некоторое время.

Not enough user privilegies

Причина

Пользователь не имеет привилегий на данную команду.

Способ устранения

Задать имя пользователя и пароль, соответствующие администратору СУБД ЛИНТЕР.

Unknown error

Причина

Ошибка в процессе проверки привилегий.

Способ устранения

Задать правильные параметры команды (длину и/или содержимое полей).

Управление с помощью сигналов

Помимо командного управления внешнее управление системой резервирования возможно с помощью сигналов. Этот вариант управления поддерживается только в рамках одного компьютера. Обычно он применяется для управления сервером резервирования какой-либо пользовательской программой.

Распознаваемые системой резервирования сигналы приведены в таблице 2. Часть из них может быть использована для управления, остальные не должны использоваться системой резервирования (см. сноску к таблице 2).

Примечание
В ОС Windows управление с помощью сигналов не поддерживается.

Таблица 2. Сигналы, распознаваемые системой резервирования

Сигнал	Назначение сигнала
SIGINT	Сигнал имеет двойное назначение: 1) завершение действий управляющей программы системы резервирования в процессе работы; 2) разрешение на запуск системы резервирования по запросу управляющей программы системы резервирования. В этом случае сервер резервирования должен быть запущен с ключом `/wp` без параметра или с параметром `SGINT` (см. описание ключа /wp). Сигнал должен быть передан только во время запроса управляющей программы. В остальные моменты времени он будет воспринят как сигнал на завершение работы.
SIGTERM	Завершение работы сервера резервирования.
SIGQUIT	Завершение работы сервера резервирования.
SIGTRAP	Завершение работы всех управляющих программ системы резервирования при получении данного сигнала любой из управляющих программ.
SIGCHLD	Для обнаружения завершения дочерних процессов.[1)]
SIGKILL	Запрещен к использованию (поскольку он немаскируемый).[1)]
SIGUSR2	Для извещения сервера резервирования об успешной инициализации сетевого драйвера сервера `dbs_tcp` или `dbc_tcp`.[1)]
SIGHUP	Для извещения сервера резервирования об успешной инициализации ядра СУБД ЛИНТЕР.[1)]

Сигнал	Назначение сигнала
SIGIO (или SIGUSR1)	Используется библиотекой интерфейса нижнего уровня.[1)]
Поддерживаемые ОС	Для запуска системы резервирования по сигналу (должен быть задан в ключе `/wp`) или для запуска тестирования БД (см. ключ `/testdb`).

[1)] Запрещается применять для управления работой сервера резервирования, так как этот сигнал используется сервером резервирования или ОС для внутренних нужд.

По сигналу SIGTERM, SIGQUIT, SIGINT сервер резервирования завершает свою работу немедленно и, если не используется ключ `/testslave`, не отслеживает возможную потерю данных резервными серверами. Если же ключ `/testslave` задан, то на резервном сервере происходит ожидание завершения первоначального копирования БД перед завершением его работы.

При получении главным сервером сигнала SIGTERM, SIGQUIT, SIGINT или SIGTRAP этот сигнал игнорируется, если главный сервер запущен с ключом `/testslave` и есть резервные серверы, которые в данный момент выполняют первоначальное копирование БД.

При получении сигнала SIGTRAP резервным сервером, если задан ключ `/testslave` на главном и есть серверы в процессе первоначального копирования БД, система резервирования останавливается после того, как БД будет скопирована на все серверы. При отсутствии ключа `/testslave` на главном сервере ожидания завершения копирования БД не происходит, и система останавливается немедленно.

Если значение ключа `/testdb` начинается с SIG, то для управления запуском процедуры тестирования БД будет использоваться сигнал, номер которого идет непосредственно после SIG. Процедура тестирования может быть выполнена только на сервере в состоянии SLAVE, во всех остальных случаях заданный сигнал будет игнорироваться.

Структура системы резервирования

Горячее архивирование – основа системы резервирования

Одной из основных задач СУБД является обеспечение целостности данных. СУБД ЛИНТЕР выполняет эту задачу за счет использования системного журнала. Прежде чем изменяемые данные будут записаны в таблицу БД, они помещаются в системный журнал в специальном виде. В случае отката транзакции или при включении компьютера после сбоя питания восстановление целостности и непротиворечивости данных СУБД выполняется по записям системного журнала.

Журнал устроен таким образом, что данные в него помещаются только в конец. Относительное постоянство размера журнала обеспечивается хранением нескольких файлов журнала, размер каждого из которых ограничен. Старые файлы журнала (с уже обработанными транзакциями, данные которых уже внесены в таблицы БД), удаляются. В случае длинной транзакции количество файлов журнала временно увеличивается.

Механизм записи данных только в конец журнала используется утилитой архивирования `lhb` для создания архива БД без останова СУБД ЛИНТЕР. Основной опасностью при копировании файлов таблиц СУБД в режиме «горячего архивирования» является нарушение целостности данных. Чтобы исключить это, утилита `lhb` после копирования файлов таблиц БД осуществляет также и копирование файлов журналов, в которых продолжалось накопление изменений в БД во время процесса копирования. Этот процесс продолжается до тех пор, пока не будет получено последнее изменение, записанное в системном журнале СУБД ЛИНТЕР.

Для отключения механизма автоматического удаления старых файлов журнала ядром СУБД ЛИНТЕР перед началом архивирования ставится так называемая «контрольная точка». Контрольная точка указывает ядру СУБД ЛИНТЕР, что данный файл журнала и его более поздние версии не подлежат удалению до снятия контрольной точки. Контрольная точка снимается утилитой `lhb` по окончании получения всей БД: файлов таблиц и файлов журнала.

Таким образом, по окончании копирования в архиве lhb находятся файлы таблиц БД и файлы системного журнала. Они могут быть развернуты из архива в отдельный каталог, который будет содержать файлы таблиц БД и файлы системного журнала. Причем часть данных, измененных во время процесса архивирования, уже находится в таблицах БД, но все измененные данные находятся в файлах системного журнала.

Ядро СУБД ЛИНТЕР при старте распознает, что некоторые данные не перенесены в таблицы БД, и осуществляет «докат» данных по системному журналу (перенос данных, записанных в журнал, в таблицы БД). При этом незаконченные транзакции в таблицы не попадают, то есть после старта ядра СУБД ЛИНТЕР БД находится в непротиворечивом состоянии, соответствующем оригинальной БД на какой-либо момент времени.

Расширением горячего архивирования является непрерывный процесс архивирования. В этом случае после архивирования последних добавленных в системный журнал данных утилита `lhb` не завершает своей работы, а ожидает следующих изменений, которые также добавляются в архив, что обеспечивает механизм непрерывного горячего архивирования.

Именно этот механизм положен в основу системы резервирования. На главном сервере работает ядро СУБД ЛИНТЕР и обеспечивает обычную работу клиентов с СУБД. На резервном сервере работает утилита `lhb` в режиме непрерывного

горячего архивирования. В случае отказа главного сервера на резервном сервере необходимо развернуть архив и запустить на новой БД ядро СУБД ЛИНТЕР. При использовании данного механизма непрерывного горячего архивирования возникают несколько проблематичных моментов:

1) данные при архивировании накапливаются в одном файле – архиве БД, который позже разворачивается в файлы таблиц БД и системного журнала СУБД. Разворачивание архива требует времени, что, несомненно, снижает скорость перехода системы резервирования в состояние готовности после сбоя главного сервера;
2) данные могут накапливаться в системном журнале в течение продолжительного времени, поэтому и «докат» также отнимает продолжительное время (часто гораздо большее, чем разворачивание архива), что также снижает время перехода системы в состояние готовности;
3) так как данные могут только добавляться к концу журнала, то и файлы системного журнала могут только расти. Их обработка будет произведена лишь при запуске ядра СУБД ЛИНТЕР. Непрерывный рост системного журнала очень нежелателен, поскольку требует «бесконечных» ресурсов для размещения журнала.

Для устранения перечисленных недостатков используются особые режимы работы утилиты архивирования `lhb` и ядра СУБД ЛИНТЕР, а именно:

1) утилита `lhb` не накапливает данные в архивном файле, а сразу же разворачивает их в файлы таблиц и системного журнала БД на резервном сервере. Таким образом, устраняется первый недостаток, и создаются предпосылки для решения двух остальных;
2) ядро СУБД ЛИНТЕР функционирует в специальном режиме периодического «доката» изменений по системному журналу. В этом режиме оно не работает с клиентскими приложениями и не обслуживает их запросов, а всего лишь анализирует данные, добавленные в системный журнал утилитой `lhb`, и переносит невнесенные изменения в таблицы БД. Таким образом, устраняется второй недостаток – медленный «докат» данных;
3) кроме того, поскольку ядро СУБД обрабатывает вновь поступившие данные системного журнала и переносит накопленные изменения в таблицы БД, то обработанные файлы журнала, не содержащие не сохраненные в БД транзакции, могут быть удалены. Это также выполняет ядро СУБД в специальном режиме, устраняя последний недостаток.

Поскольку при работе с журналом, с одной стороны, выполняются операции добавления в него данных утилитой `lhb`, с другой стороны, идет обработка этих данных ядром СУБД ЛИНТЕР, то необходимо исключить одновременную работу с одними и теми же данными для предотвращения обработки незавершенных данных. Это достигается работой `lhb` и ядра СУБД ЛИНТЕР с разными файлами системного журнала. Утилита `lhb` накапливает данные в последний файл журнала. Как только он будет заполнен, запись информации в него прекращается, и он поступает в распоряжение СУБД ЛИНТЕР (то есть ядро СУБД ЛИНТЕР в специальном режиме работает только с полностью завершенными файлами системного журнала).

При переходе резервного сервера в режим главного сервера для БД запускается ядро СУБД ЛИНТЕР в нормальном режиме, которое обрабатывает при своем старте и незавершенный файл системного журнала. Поскольку объем этого файла невелик, то его обработка выполняется быстро.

В стационарном режиме движение изменяемых данных выглядит следующим образом:

- приложение путем отправки SQL-запроса модифицирует БД главного сервера системы резервирования;
- при модификации данных изменяются файлы таблиц БД и файлы системного журнала, причем в системный журнал они добавляются всегда в конец;
- утилита `lhb`, запущенная на резервном сервере, получает добавленные в системный журнал данные и сохраняет их в копию файлов системного журнала в БД на резервном сервере;
- утилита `lhb` получает данные с помощью сетевого драйвера сервера `dbs_tcp`, работающего на главном сервере, и сетевого драйвера клиента `dbc_tcp`, работающего на резервном сервере;
- ядро СУБД ЛИНТЕР в специальном режиме, запущенное на резервном сервере, переносит измененные данные из системного журнала в таблицы БД с отставанием на один файл.

Доступ клиентских приложений к главному серверу резервирования автоматически обеспечивается сетевым драйвером клиента `dbc_tcp`. Он обращается к одному из нескольких ЛИНТЕР-серверов и проверяет их доступность, производя, при необходимости, перенаправление запросов на доступный сервер.

Управляющая программа – средство автоматизации резервирования

Как было сказано выше, для обеспечения работы системы резервирования достаточно запустить на главном сервере ядро СУБД ЛИНТЕР и сетевой драйвер сервера `dbs_tcp`, а на резервном – сетевой драйвер клиента `dbc_tcp`, утилиту горячего архивирования `lhb` и ядро СУБД ЛИНТЕР в специальном режиме. Этого достаточно для работы системы резервирования в минимальном составе. Администратор системы резервирования может самостоятельно определить назначение каждого из серверов (главный/резервный) и запустить на них необходимые утилиты. В случае отказа главного сервера администратору необходимо на резервном сервере остановить все запущенные утилиты и запустить те, что должны работать на главном. После такого запуска резервный сервер станет главным.

Выполнение этих (и ряда других) задач берет на себя специально разработанная управляющая программа системы резервирования. Основным ее назначением является автоматизация управления системой резервирования, состоящей из нескольких серверов, а именно:

- определение назначения данного компьютера в системе резервирования при старте системы (главный или резервный);
- запуск утилит в соответствии с назначением данного компьютера при старте или при смене назначения;
- слежение за работоспособностью компьютеров, входящих в систему резервирования (серверов резервирования);
- выявление неработоспособности серверов резервирования;
- смена функционального назначения сервера в случае выхода из строя главного сервера;
- проведение конкурса на роль главного сервера среди резервных серверов в случае выхода из строя главного сервера;

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

- взаимодействие с администратором системы резервирования, информирование его о своих действиях, и обработка команд администратора;
- завершение работы отдельного сервера или системы резервирования в целом.

Кроме основных задач управляющая программа берет на себя ряд дополнительных функций:

- поддержку нескольких линий связи между серверами резервирования и автоматическую смену линии связи на запасную при выходе из строя основной;
- определение функциональности сервера на основании времени БД или по системному журналу;
- отслеживание работоспособности утилит и ядра СУБД ЛИНТЕР;
- автоматический рестарт утилит и ядра СУБД ЛИНТЕР при их завершении, инициированном причинами, отличными от команды управляющей программы;
- запуск специальной программы (обработчика событий) при обнаружении каких-либо действий или изменении состояния системы резервирования. Данная программа может быть использована для дополнительного управления системой резервирования или реагирования пользовательского приложения на события системы резервирования;
- плавная подстройка времени системы резервирования к системному времени;
- протоколирование (трассировка) работы управляющей программы и системы резервирования в целом;
- проверку санкционированности команд администратора системы резервирования;
- создание копии БД при старте системы резервирования, либо по расписанию или по команде администратора системы резервирования;
- создание архивного файла копии БД;
- восстановление БД из архивного файла;
- проверку целостности БД утилитой `testdb` по расписанию или по команде администратора системы резервирования;
- чтение файла конфигурации системы резервирования;
- хранение и чтение предыдущего состояния данного сервера резервирования и БД;
- ведение двух каталогов БД на каждом сервере для увеличения надежности и скорости переключения на нужную БД.

Ниже описание работы системы резервирования приведено в хронологическом порядке, однако при необходимости (в случае невозможности или нецелесообразности такого изложения) возможны отступления от хронологического порядка.

В общем случае работа управляющей программы системы резервирования включает в себя следующие стадии (режимы):

- запуск (старт) системы резервирования;
- работа в стационарном режиме;
- переключение в другое состояние при обнаружении выхода из строя главного сервера;
- завершение работы.

Старт управляющей программы

Во время старта управляющая программа выполняет следующие действия:

- определение предыдущего состояния данного сервера (на основе информации файла `state`);

- анализ файла настройки системы резервирования (nodetab);
- инициализацию сетевой связи с другими серверами системы резервирования;
- информирование других серверов системы резервирования о своей конфигурации и сбор аналогичных сведений об их конфигурации;
- выбор по заданным критериям сервера на роль главного сервера;
- ожидание условий запуска системы резервирования;
- вывод подтверждения о запуске системы резервирования;
- принятие решения о статусе серверов системы резервирования;
- запуск утилит в режиме главного или резервного сервера.

Определение предыдущего состояния сервера

Предыдущее состояние сервера резервирования – это состояние на момент останова (или состояние до выхода из строя) данного сервера резервирования и его баз данных.

Каждый сервер имеет по две БД – рабочую и архивную, которые располагаются в рабочем и архивном каталоге соответственно. Как было сказано ранее, это повышает надежность системы резервирования.

Файл state резервной БД располагается в резервном каталоге. Файл STATE рабочей БД располагается в рабочем каталоге.

В файле state, кроме состояния БД (главная, резервная, разрушенная), хранится также и время последней работы сервера с этими БД. На основании состояния и времени баз данных впоследствии принимается решение о статусе сервера после запуска на нем управляющей программы.

Анализ файла настройки системы резервирования

На данном этапе осуществляется поиск файла настройки системы резервирования nodetab (для настройки системы резервирования и сетевой конфигурации сетевого клиента используется один и тот же файл, см. документ «Сетевые средства»). Алгоритм поиска файла nodetab приведен в описании ключа /ntab. Из файла конфигурации извлекается информация обо всех узлах, описанных в строке с протоколом REZ. Управляющая программа считает, что именно эти узлы входят в систему резервирования. Какие из узлов являются удаленными, а какие принадлежат данному серверу – определяется позднее.

Инициализация сетевой связи с другими серверами

На данном этапе программа открывает сетевой UDP-сокет. Управляющая программа использует тот же номер порта, что и сетевой драйвер сервера dbs_tcp, но по UDP-протоколу. Также на данном этапе генерируется уникальный случайный идентификатор данного сервера резервирования, который в дальнейшем используется для определения старшего сервера.

На основании информации файла настройки nodetab управляющая программа посылает специальные UDP-пакеты каждому из узлов, входящих в систему резервирования. Рассылка UDP-пакетов производится на протяжении всей последующей работы управляющей программы. Назначение этих пакетов различно

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

на определенных этапах работы управляющей программы. На этапе инициализации UDP-пакет содержит уникальный идентификатор сервера. На основании этого идентификатора определяется принадлежность узлов данному серверу. Те узлы, с которых пришел пакет с совпадающим идентификатором, принадлежат данному серверу. В последующем сервер сам себе пакеты уже не посылает.

Те же самые действия выполняют и другие серверы резервирования при старте. Поэтому, принимая пакеты от других серверов, программа управления по одинаковым идентификаторам определяет принадлежность узла различным серверам. В случае повторения идентификаторов, принятых с другого узла, этот узел считается тем же самым сервером с запасной линией связи. В случае наличия нескольких линий связи одна из них (по которой поступил первый пакет) назначается основной, а остальные – запасными. По основной линии связи происходит передача данных. Все линии связи периодически проверяются путем отсылки тестовых пакетов.

В то же самое время управляющая программа принимает пакеты от других серверов. В случае пропадания пакетов на интервал, превышающий тайм-аут, линия считается разорванной. В случае разрыва последней линии связи данного сервера он считается вышедшим из строя.

Информирование других серверов о своей конфигурации и сбор аналогичных сведений от других серверов

Сразу же, как только управляющая программа обнаружила существование еще одного сервера резервирования (этот факт устанавливается путем приема пакета с другим идентификатором сервера), она посылает в ответ информацию о состоянии своих БД (их предыдущее состояние и время). Рассылка данных пакетов осуществляется периодически с интервалом посылки тестовых пакетов, что обеспечивает обмен информацией между серверами системы резервирования о состоянии своих БД и состоянии самих серверов резервирования.

Выбор по заданным критериям сервера на роль главного сервера

Как только управляющая программа получила состояние какого-либо удаленного сервера, или это состояние было изменено, проводится конкурс серверов на определение наиболее подходящего из них на роль главного сервера. Конкурс на роль главного сервера проводится также по истечении тайм-аута прослушивания сети периодически с интервалом запроса интерактивного запуска или при получении сигнала старта системы резервирования (см. ключи /input, /wait, /wp). После проведения конкурса его результаты выводятся на консоль.

По окончании конкурса, в случае наступления условий старта, определяется назначение сервера (главный, резервный) и осуществляется запуск на выполнение необходимых утилит.

Ожидание условий запуска

В процессе работы управляющие программы серверов резервирования постоянно обмениваются своими состояниями: отсылают свое состояние и принимают состояния других. Поэтому, если при старте управляющей программы будет получена информация, что один из серверов резервирования уже является главным (находится в

состоянии MAIN или MONO), то данный сервер немедленно начинает переход в режим резервного сервера.

Если же ни один из серверов не находится в состоянии главного (обычно при старте все серверы находятся в UNDEFINED-состоянии), то управляющая программа ожидает наступления событий запуска, которыми могут быть:

- истечение тайм-аута прослушивания сети и длительность паузы после останова менее интервала простоя системы (см. ключ /down);
- получение явного сигнала на запуск системы резервирования (см. ключ /wp);
- достижение заданного количества серверов (см. ключ /nservers);
- активизация всех узлов системы резервирования (это также означает запуск всех серверов резервирования);
- истечение тайм-аута прослушивания сети (см. ключ /wait) и разрешение на автоматический запуск в неполном составе (см. ключ /nq);
- истечение тайм-аута прослушивания сети и явного разрешения старта системы резервирования администратором.

Вывод подтверждения запуска

В случае наступления события запуска «истечение тайм-аута прослушивания сети и явного разрешения старта системы резервирования администратором» и при наличии хотя бы одного выключенного сервера резервирования (хотя бы один узел определен как выключенный) периодически на консоль выводится информация о наилучшем сервере на роль главного и запрос на подтверждение выбора данного сервера в качестве главного. Это необходимо, потому что выключенный сервер (с незапущенной управляющей программой) не может сообщить о своем состоянии, и, следовательно, управляющая программа может ошибиться в выборе главного сервера. Поэтому решение о запуске системы резервирования в неполном составе должен принимать её администратор. Информировать о своем решении администратор может интерактивным вводом или посылкой сигнала. На данном этапе администратор может дать команду серверу на завершение работы (интерактивно, путем ответа на запрос или посылкой сигнала).

Запрос администратору на старт системы резервирования будет выведен всегда (несмотря на ключ `-nq`) в случае нахождения рабочей БД будущего главного сервера в SLAVECRASH или MAINCRASH состоянии, или если будет необходимо делать копию резервной БД в рабочий каталог.

Принятие решения о статусе

Как только наступило событие запуска, управляющая программа системы резервирования должна начать запуск утилит и ядра СУБД ЛИНТЕР. Для исключения конфликтов решение о назначении (статусе) всех серверов системы резервирования принимает единолично старший сервер. Старшим сервером назначается тот сервер, на котором администратор выполнил подтверждение старта системы резервирования (сигналом или интерактивно). В случае автоматического старта системы резервирования старшим является сервер с максимальным идентификатором. Идентификатор передается с каждым пакетом (определяется при инициализации сетевого обмена управляющей программы и не изменяется на всем протяжении работы). О своем решении управляющая программа старшего сервера уведомляет все остальные серверы путем отсылки сетевой команды.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Алгоритм выбора главного сервера

При конкурсе на роль главного сервера проводятся следующие действия:

1) среди рабочей БД, архивной БД и архивного файла БД выбирается объект с максимально свежими данными (по времени или по системному журналу);
2) выполняется попарно сравнение серверов по объектам с наиболее свежими данными. Если один из серверов имеет более свежую рабочую или архивную БД, а другой имеет только свежий архивный файл БД, то выбирается сервер с БД, а не с архивным файлом;
3) если у сравниваемых серверов нет БД, то подходящий сервер выбирается на основе сравнения времени архивных файлов;
4) если у сравниваемых серверов есть БД и если включен режим сравнения по системному журналу, то сравнение происходит по системному журналу, иначе этот пункт опускается;
5) если сравнение по системному журналу не выявило предпочтительного сервера или не производилось, то выполняется сравнение по времени БД;
6) если и сравнение по времени БД не выявило наиболее подходящий сервер, то производится сравнение по состояниям БД. MONO или MAIN состояния имеют преимущество перед всеми остальными;
7) если и на данной стадии не обнаружено наиболее подходящего сервера, то главным выбирается сервер с наибольшим идентификатором (старший).

Таким образом, после попарного сравнения всех серверов системы на роль главного подбирается наиболее подходящий сервер.

Принципы сравнения баз данных по времени

При своей работе главный сервер периодически обновляет информацию о состоянии своей рабочей БД в файле `state` (см. ключ /dbtestint). Каждый раз при этом в файл `state` записывается текущее время главного сервера.

Резервный сервер записывает свое состояние и текущую дату в файл `state` только после окончания первоначального копирования данных с главного сервера.

При последующем сравнении БД по датам выбирается наиболее свежая.

Принципы сравнения баз данных по системному журналу

Сравнение БД может производиться не только по времени, но и по системному журналу БД.

Системный журнал имеет отметки о выполненных транзакциях. На основании этих отметок можно принять решение о более актуальной БД.

Так как транзакции записываются в конец журнала последовательно, то наиболее заполненный журнал будет иметь более свежие транзакции. На этом и основано сравнение данных по журналу – наиболее заполненный журнал имеет более свежие данные (см. описание ключа /cmpsyslog).

Запуск ядра и утилит СУБД в режиме главного сервера

Как только данный сервер получает команду от старшего сервера на начало своей работы в режиме главного (или он сам является старшим и определил свое назначение

как главный сервер), управляющая программа переходит в фоновый режим работы и меняет свое состояние с UNDEFINED на STARTMONO.

В режиме STARTMONO осуществляется запуск необходимых для главного сервера утилит и ядра СУБД ЛИНТЕР, при этом управляющая программа выполняет следующие действия:

1) выбор лучшей БД для запуска ядра СУБД ЛИНТЕР.

   Наиболее подходящей БД является самая свежая БД по критериям времени или системного журнала.

   В случае если подходящей БД является БД из архивного каталога, выполняется копирование этой БД в рабочий каталог (при работе с ключом `/exchdir` меняется функциональность каталогов без копирования).

   В случае отсутствия БД в обоих каталогах и наличия ключа `/crash` выполняется распаковка архивного файла в рабочий каталог и завершение работы системы резервирования (для дальнейшей работы необходим повторный запуск системы резервирования).

2) запуск ядра СУБД ЛИНТЕР.

   Запуск ядра СУБД ЛИНТЕР производится для наиболее подходящей БД. Управляющая программа ожидает сигнала от ядра СУБД ЛИНТЕР об успешном старте. Производится удаление lhb-файла, если он существует в каталоге рабочей БД. Это выполняется для исключения докачки системного журнала в БД, на которой работало ядро СУБД ЛИНТЕР, в случае, если данный сервер станет `SLAVE`.

3) получение списка контрольных точек.

   После успешного старта ядра СУБД ЛИНТЕР осуществляется запуск утилиты `lhb` для получения списка контрольных точек. Контрольные точки могут остаться от предыдущих сеансов работы `lhb`. Ведение их списка позволяет контролировать накопление файлов системного журнала на главном сервере. Как только системный журнал превысит заданный размер, производится удаление старых контрольных точек. Количество хранимых файлов системного журнала задается ключом `/syslogcount`. С другой стороны, откладывание удаления контрольных точек на некоторое время позволяет продолжить выполнение утилиты `lhb` на SLAVE-сервере после запуска в инкрементном режиме вместо скачивания всей БД.

4) запуск сетевого драйвера сервера.

   После получения списка контрольных точек производится запуск сетевого драйвера сервера `dbs_tcp`.

Алгоритм выбора базы данных при переходе сервера в режим главного

При переходе сервера в состояние главного наиболее подходящая БД определяется по следующему алгоритму:

1) проверяется возможность рабочей и архивной БД сервера быть кандидатами на роль новой рабочей БД сервера. Для этого БД должна удовлетворять следующим требованиям:
   - реально существовать на сервере;

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

- иметь состояние, отличное от SLAVECRASH или MAINCRASH;
- время БД должно быть отлично от 0 (существующая БД всегда имеет хотя бы на секунду позже начала эпохи);
- БД должна быть актуальной (если сравнение происходит не в стартовом (UNDEFINED) состоянии).

2) если какая-либо БД не может быть кандидатом на роль рабочей БД главного сервера, то автоматически ею становится другая;
3) если обе БД удовлетворяют перечисленным требованиям, то кандидат выбирается последовательно на основании сравнения системных журналов, времени, состояний;
4) если на этапе сравнения состояний наиболее подходящая БД не выбрана, то используется рабочая БД;
5) если обе БД не подходят на роль рабочей БД, то используется архивный файл БД (только для UNDEFINED-состояния с ключом `/crash`, после разворачивания система останавливается);
6) возможна также ситуация, когда сервер вообще не имеет данных и, следовательно, не может работать в режиме главного.

Запуск утилит в режиме резервного сервера

После получения извещения, что данный сервер должен быть резервным, он переходит в фоновый режим и начинает работу в режиме `SLAVE_WAIT`. В данном режиме управляющая программа периодически посылает запрос главному серверу на разрешение запуска этого сервера в SLAVE-режиме. От главного сервера он ожидает приема разрешения на запуск работы в SLAVE-режиме.

В режиме резервного сервера управляющая программа выполняет следующие действия:

1) выбор рабочей БД.

   Как только разрешение на запуск получено, резервный сервер переходит в состояние SLAVE. В данном состоянии сначала определяется наиболее подходящий каталог для рабочей БД. Он подбирается по критерию самой актуальной информации.

2) запуск сетевого драйвера клиента.

   После выбора каталога с рабочей БД производится запуск сетевого драйвера клиента `dbc_tcp`. При успешном запуске `dbc_tcp` сигналом извещает об этом управляющую программу.

3) очистка контрольной точки.

   Управляющая программа запускает утилиту `lhb` для очистки контрольной точки данной БД (утилита `lhb` с ключом `-stopinc`). Очистка контрольной точки требуется для освобождения файлов системного журнала главного сервера и возможности их удаления.

4) попытка продолжения докачки журнала.

   Если рабочая или резервная БД присутствует и в её каталоге есть lhb-файл, то производится попытка продолжить существующий lhb-архив. Это возможно, только если на главном сервере не было интенсивного изменения данных, и размер системного журнала не превысил максимально допустимое значение. Если этого не

случилось, то контрольная точка не будет удалена главным сервером и сохранится возможность продолжения скачивания системного журнала (успешно продолжится докачка существующего системного журнала). Об окончании этого процесса утилита `lhb` информирует управляющую программу, и она сменит свое состояние на SLAVE_OK. Описанные далее пункты будут пропущены (в том числе и копирование (архивирование) БД при старте).

Если же попытка продолжения скачивания системного журнала будет неудачна по причине отсутствия БД, lhb-файла или по неуспешному продолжению докачки, то будут произведены описанные ниже действия.

5) сохранение копии БД.

   В случае использования устаревшего режима совместимости (`/exchdir=0`) и если рабочая БД имеет более актуальную БД, чем резервная БД, производится перемещение рабочей БД в архивный каталог. Наличие архивной БД необходимо на случай отказа главного сервера в ситуации, когда БД с главного сервера получена резервным сервером не полностью. Предыдущая БД в архивном каталоге при этом удаляется.

   В случае работы в режиме по умолчанию используется обмен каталогов и копирование никогда не выполняется. Для рабочего каталога выбирается каталог с наиболее старой БД, поскольку БД в нем будет замещена новой БД.

6) очистка рабочего каталога.

   Выполняется предварительная очистка (удаление файлов) рабочего каталога от файлов БД. Таким образом, подготавливается место для получения новой БД с главного сервера.

7) создание архивного файла.

   На следующем этапе, если это задано ключом командной строки `/archivate`, выполняется создание архивного файла резервной БД. По умолчанию в ОС Linux, ЗОСРВ Нейтрино для архивации используется архиватор `pkzip`. Местоположение каталога для архивного файла берется из ключа `/barc`. Если этот ключ не задан, по умолчанию архивный файл БД создается в каталоге архивной БД под именем `DB.zip`.

8) первоначальное копирование данных.

   Следующим шагом является запуск утилиты `lhb` в режиме первоначального копирования данных (с ключом `-startinc`). Утилита горячего архивирования `lhb` размещает копию рабочей БД главного сервера в рабочем каталоге резервного сервера. Она сохраняет копируемые данные не в архивный файл (как обычно – архив `lhb`), а сразу же размещает их в файлы БД. При таком способе архивирования БД архивный файл утилиты `lhb` используется только для хранения служебной информации и не содержит собственно данных (а, следовательно, и не растет в объеме).

9) передача информации о контрольной точке главному серверу.

   Сразу же после старта утилиты `lhb` анализируется выдаваемая ею информация об установленной контрольной точке. Процесс копирования рабочей БД может выполняться параллельно на несколько резервных серверов. Для каждого резервного сервера в системном журнале копируемой рабочей БД создается своя контрольная точка, которая идентифицируется только своим порядковым номером и датой создания.

Утилита lhb сохраняет информацию о контрольной точке в выходном файле. Эта информация анализируется и передается главному серверу, чтобы исключить её удаление при росте системного журнала. Контрольные точки активных SLAVE-серверов главным сервером не удаляются.

10) извещение о готовности.

    По окончании копирования данных (в режиме startinc или при попытке докачки системного журнала) утилита lhb сигналом информирует об этом управляющую программу. Управляющая программа генерирует событие SLAVE_OK, извещающее о готовности резервного сервера, и рассылает его всем остальным серверам системы резервирования. Если задан ключ /stproc, то вызывается обработчик событий, которому передается информация о размерах обеих БД и времени завершения копирования БД (докачки системного журнала) с главного сервера.

    Утилита lhb остается в состоянии WAIT для немедленного получения информации об изменившихся данных и сохранении её в файлах системного журнала. В дальнейшем по файлам системного журнала на резервном сервере может быть восстановлена вся измененная информация.

11) смена своего состояния.

    С этого момента (окончание скачивания данных утилитой lhb) на резервном сервере в рабочем каталоге будет находиться готовая к использованию БД – копия рабочей БД главного сервера. Таким образом, данный резервный сервер готов в любой момент взять на себя функции главного сервера (находится в состоянии SLAVE_OK).

    Рабочая БД переводится в состояние SLAVE, что фиксируется в файле state рабочей БД вместе с текущим временем.

12) запуск ядра СУБД ЛИНТЕР в специальном режиме.

    Для переноса данных из файлов системного журнала в файлы таблиц БД управляющая программа запускает ядро СУБД ЛИНТЕР в специальном режиме – режиме совместной работы с утилитой lhb. Ядро СУБД после переноса данных удаляет уже обработанные файлы системного журнала.

    Резервный сервер переходит в стационарный режим работы.

Стационарный режим работы

В стационарном режиме:

1) на главном сервере работают ядро СУБД ЛИНТЕР и сетевой драйвер сервера dbs_tcp;
2) на резервном сервере работают:
   - сетевой драйвер клиента dbc_tcp;
   - утилита горячего архивирования lhb в режиме ожидания данных;
   - ядро СУБД ЛИНТЕР в специальном режиме. Отличительной особенностью работы ядра в этом режиме является отсутствие процессов sql, intsrt, tsp.
3) на каждом из серверов работает управляющая программа системы резервирования.

Ниже рассматриваются процессы, или события, обрабатываемые управляющей программой в стационарном режиме работы.

В стационарном режиме управляющая программа выполняет:

- мониторинг состояния удаленных серверов и линий связи;
- мониторинг работоспособности самой управляющей программы (самоконтроль);
- обработку команд (сетевых и по сигналам);
- синхронизацию внутреннего и системного времени;
- определение невозможности дальнейшего продолжения работы и прекращение работы управляющей программы;
- запуск обработчика событий в системе резервирования;
- мониторинг работы утилит и ядра СУБД ЛИНТЕР;
- перезапуск процессов в случае необходимости.

Управляющая программа всегда при выходе из стационарного состояния выполняет завершение работы запущенных ею процессов.

В стационарном режиме главный сервер осуществляет подключение резервных серверов к системе резервирования и хранение соответствия номера контрольной точки серверу резервирования.

Резервный сервер в стационарном состоянии может производить копирование рабочей БД, создание архивного файла после копирования, тестирование БД. В случае отказа главного сервера управляющие программы резервных серверов проводят конкурс на замещение главного сервера, и один из них осуществляет переход в режим главного.

Мониторинг состояния удаленных серверов и линий связи

Основным назначением системы резервирования является автоматическая смена главного сервера в случае выхода из строя главного на данный момент сервера. Поэтому на протяжении всего времени работы управляющая программа осуществляет мониторинг работы удаленных серверов.

Мониторинг линий связи

Мониторинг линий связи выполняется с использованием локальной сети. Каждая из управляющих программ на всем протяжении своей работы осуществляет периодическую (см. ключ /testint) посылку тестовых пакетов всем удаленным узлам системы резервирования. Посылка тестовых пакетов выполняется независимо от факта работоспособности (включен или выключен) удаленного сервера резервирования. Посылка пакетов производится по UDP-протоколу на тот же номер порта, который используется для работы сетевого драйвера сервера `dbs_tcp` по TCP-протоколу.

Каждый из серверов принимает тестовые пакеты и следит за количеством пропущенных пакетов, посылаемых по данной линии связи (от данного узла). Если от узла не поступает ни одного из заданного количества пакетов (см. ключ /tstlimit), линия связи к удаленному узлу считается разорванной. Другими словами, линия связи считается разорванной в случае истечения тайм-аута, определяемого произведением значений ключей `/testint` и `/tstlimit`.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Действия серверов при обрыве линии связи:

- главный сервер при обнаружении обрыва линии связи не предпринимает никаких действий, за исключением, возможно, вызова обработчика этого события. Он считает, что переход на одну из запасных линий связи произойдет автоматически;
- резервный сервер при обнаружении разрыва основной линии связи прекращает работу утилиты `lhb` (если она не прекратила работу самостоятельно), ядра СУБД ЛИНТЕР в специальном режиме и затем выполняет их перезапуск. При этом `lhb` осуществляет соединение с другим узлом главного сервера. В случае разрыва запасной линии связи резервный сервер не предпринимает никаких действий. Вызов обработчика осуществляется при обрыве любой линии связи.

Мониторинг серверов

Мониторинг удаленных серверов резервирования осуществляется путем мониторинга его узлов (линий связи к удаленному серверу). В случае разрыва всех линий связи удаленный сервер считается вышедшим из строя. При выходе из строя удаленного сервера также может вызываться обработчик события.

Действия серверов при выходе из строя удаленного сервера:

- главный сервер при выходе из строя резервного сервера осуществляет перечитывание информации о контрольных точках. При этом все контрольные точки, не принадлежащие активным SLAVE-серверам, попадают в список подлежащих удалению контрольных точек. При превышении размера системного журнала происходит удаление самой старой контрольной точки из списка, после чего список перечитывается заново. Проверка размера системного журнала происходит периодически с интервалом посылки тестовых пакетов ядру СУБД ЛИНТЕР.

  При выходе из строя последнего резервного сервера системы резервирования главный сервер переходит из состояния MAIN в состояние MONO.
- если вышедший из строя сервер был резервным, то никаких действий другими резервными серверами не предпринимается. Если же из строя вышел главный сервер, то резервные серверы завершают свою работу в состоянии SLAVE, и среди них проводится конкурс на роль нового главного сервера.

Мониторинг состояния серверов

Каждый из серверов при изменении своего состояния, а главный сервер периодически (см. ключ /dbtestint), извещает о своем состоянии все остальные серверы системы резервирования. Сетевой обмен пакетами реализован таким образом, что, несмотря на UDP-протокол, извещение все равно достигнет удаленного сервера (выполняется повтор отсылки в случае необходимости).

Управляющая программа хранит состояние каждого из удаленных серверов. В момент смены состояния удаленного сервера управляющая программа может предпринять некоторые действия. Например, при смене состояния главного сервера в результате его останова может начаться конкурс на роль нового главного сервера.

Запуск обработчика событий системы резервирования

При смене своего состояния или наступлении какого-либо контролируемого события управляющая программа может вызывать обработчик события (см. ключ /stproc). Для этого управляющая программа порождает отдельный поток, который следит за

очередностью запуска обработчика событий (см. ключ /nostprocqueue) и выполняет запуск собственно обработчика с требуемыми ключами. Запуск обработчика осуществляется единообразно при любом состоянии системы резервирования.

Конкурс на замещение главного сервера

В случае отказа главного сервера его функции должен взять на себя один из резервных серверов. Для этого среди оставшихся резервных серверов проводится конкурс на выбор лучшего кандидата на роль главного.

Если выбрано сравнение баз данных по системному журналу, то конкурс производится не сразу, а с некоторой задержкой. Эта задержка необходима для получения самой последней информации о состоянии баз данных резервных серверов. Задержка обусловлена разницей моментов времени обнаружения выхода из строя главного сервера разными резервными серверами, а также задержками в сети.

Алгоритм выбора преемника главного сервера практически такой же, как и при старте системы резервирования. Отличие в том, что конкурс проводится только среди серверов с состоянием `SLAVE`, `SLAVE_OK`, `SLAVE_WAIT`, а в конкурсе БД участвуют только актуальные БД.

БД является актуальной только в том случае, если она изменялась или могла изменяться с момента старта управляющей программы сервера резервирования. Таким образом, актуальными могут быть:

- БД, которые были рабочими на главном сервере (если когда-либо этот сервер был главным);
- рабочие БД резервных серверов, полученные с главного сервера после старта управляющей программы;
- резервные БД резервных серверов, полученные копированием актуальных рабочих БД.

Понятие актуальности введено для исключения потери данных при завершении работы главного сервера. Если главный сервер завершил работу, и при этом ни на одном резервном сервере нет актуальной БД, то система резервирования завершит функционирование. Поэтому данные, добавленные на главный сервер непосредственно после его старта до получения актуальной БД хотя бы одним из резервных серверов, не будут потеряны.

Отличаются также состояния в процессе переключения. Если при старте управляющая программа проходит через состояние `STARTMONO`, то при переключении она проходит через состояние `SW_TO_MONO`.

Серверы, не прошедшие конкурс на главного, переводятся в состояние старта резервного сервера `SLAVE_WAIT` и начинают заново свою работу в режиме резервного сервера.

Подключение резервных серверов к системе резервирования

Управляющая программа в начале работы сервера в режиме резервного при старте или в результате переключения переводится в режим `SLAVE_WAIT`, в котором она периодически опрашивает готовность главного сервера и просит его разрешения на старт. Без разрешения главного сервера ни один из резервных серверов не начинает

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

запуск своих процессов, то есть главный сервер фактически осуществляет подключение резервных серверов к системе резервирования.

Главный сервер дает разрешение на старт резервного только после своего перехода в стационарный режим работы. Если в процессе мониторинга состояния удаленных серверов обнаружено, что хотя бы один из резервных серверов перешел в состояние `SLAVE_OK` (закончил первоначальное копирование БД), то главный сервер резервирования переводится из MONO в состояние MAIN, что обеспечивает режим повышенной надежности.

Синхронизация внутреннего и системного времени

Система резервирования не нуждается в синхронизации времени между серверами. Каждый принятый сетевой пакет состояния сервера содержит время пославшего его сервера, поэтому каждому серверу известна разница времени между серверами. Эта разница учитывается при проведении сравнения времени БД, поэтому синхронизация времени серверов является желательной (но не обязательной).

Если же время на сервере изменяется не монотонно (скачкообразно), например, после подстройки системных часов сервера, то это может вызвать неверное поведение сервера. Подстройка вызовет резкий скачок времени, что может привести к ложному обнаружению выхода из строя удаленного сервера (скачок эквивалентен мгновенному истечению тайм-аута) или к выбору в качестве нового главного сервера резервного сервера с менее актуальными данными.

Во избежание подобных проблем управляющая программа ведет свой внутренний отсчет времени. В момент старта внутренние часы устанавливаются по системным. В случае обнаружения скачка системных часов внутренние часы постепенно начинают подстройку. Обычно подстройка выполняется за удесятеренное время скачка. Этот параметр может регулироваться ключом `/tcorrect`.

Рестарт процессов

Управляющая программа в стационарном режиме работы постоянно следит за работоспособностью запущенных ею утилит и ядра СУБД ЛИНТЕР: отслеживает завершение процессов и осуществляет мониторинг их работы.

В случае обнаружения неожиданного (без команды управляющей программы) завершения отслеживаемого процесса управляющая программа автоматически производит его рестарт с необходимыми параметрами.

Управление очередностью запуска

При рестарте процессов отслеживается порядок и особенности их запуска. На главном сервере, например, при самостоятельном завершении работы ядра СУБД ЛИНТЕР одновременно завершает работу и сетевой драйвер сервера `dbs_tcp`. Запуск их производится в необходимом порядке (сначала ядро, затем драйвер сервера).

На резервном сервере также существует иерархия запуска: сетевой драйвер клиента `dbc_tcp`, утилита архивирования `lhb`. При завершении `dbc_tcp` завершится также и `lhb`.

Задержка рестарта

Зависимые процессы завершаются через некоторое время, необходимое им для корректного завершения своей работы после завершения основного процесса. Кроме

того, сам процесс завершения занимает время, поэтому производить немедленный рестарт сетевых компонентов нельзя: рестарт сетевых компонентов выполняется с задержкой, определяемой ключом /dbsctout.

Анализ кода завершения процессов

При завершении процесса управляющая программа анализирует код завершения. В случае получения кода завершения, свидетельствующего о невозможности продолжения работы (разрушение БД или несоответствие версий БД и ядра СУБД), управляющая программа заканчивает свою работу с предварительным переходом в MAINCRASH, SLAVECRASH, MAINFAILER, SLAVEFAILER состояние и запоминанием этого состояния в state файле соответствующей БД. В разрушенное (CRASH) состояние БД переводится при невозможности дальнейшей работы (при соответствующем коде завершения ядра СУБД ЛИНТЕР или при завершении утилиты lhb на этапе первоначального копирования данных).

Ограничение попыток рестарта

Управляющая программа ограничивает количество неудачных запусков процессов. Для каждого из них вводится счетчик неудачных запусков. При превышении количества запусков, определяемого ключами /pretry и /lretry, происходит завершение работы управляющей программы с переходом в FAILER состояние. Если в течение тайм-аута, задаваемого ключом /treq для ядра СУБД ЛИНТЕР, или 60 секунд для остальных процессов рестарта данного процесса не происходило, счетчик неудачных запусков обнуляется.

Мониторинг работы утилит и ядра СУБД ЛИНТЕР

Способы мониторинга

Управляющая программа, кроме слежения за завершением процессов, постоянно выполняет мониторинг их состояния с помощью периодической проверки работоспособности процесса путем посылки запросов и ожидания ответов, либо путем приема от процессов сообщений «я жив».

По первому способу контролируются сетевой драйвер клиента и ядро СУБД ЛИНТЕР, работающее в обычном режиме.

По второму способу – сетевой драйвер сервера dbs_tcp, утилита lhb и программа управления резервированием server, причем dbs_tcp и server осуществляют самоконтроль, порождая для этого отдельный следящий процесс.

Действия при обнаружении сбоя

При обнаружении сбоя, проявляющегося в невыполнении процессом своих функций, следящий процесс принудительно завершает основной и перезапускает его, или завершается сам в случае невозможности перезапуска. Перезапуск процесса осуществляется на общих основаниях.

Мониторинг состояния ядра СУБД ЛИНТЕР на главном сервере

Мониторинг состояния ядра СУБД ЛИНТЕР и его компонентов (sql, tsp, intsrt) выполняется путем периодической посылки им специальных тестовых запросов

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

(частота запросов задается в ключе /treq). Если в течение тайм-аута управляющая программа не получит ответа от ядра СУБД ЛИНТЕР, то она считает, что обнаружен сбой в работе ядра, и посылает ему сигнал безусловного завершения работы (SIGKILL). Впоследствии ядро будет повторно запущено, как и при аварийном завершении.

Для мониторинга ядра может использоваться упрощенная методика (см. ключ /st).

Мониторинг сетевого драйвера сервера dbs_tcp

Сетевой драйвер сервера dbs_tcp выполняет самоконтроль за своей работой. Для этого он запускается со специальным ключом, который заставляет драйвер следить за всеми своими процессами. В случае сбоя какого-либо процесса следящий процесс сетевого драйвера сервера производит перезапуск сетевого драйвера сервера.

Мониторинг работы утилиты lhb

Для слежения за работой утилиты lhb сервер резервирования запускает её со специальным ключом, который заставляет lhb выводить хотя бы один символ в заданный файловый дескриптор каждые 30 секунд или чаще. При запуске lhb для возможности чтения этих символов сервер резервирования создает pipe-файловый дескриптор, который становится стандартным потоком вывода lhb. Если управляющая программа не смогла прочитать из этого файлового дескриптора ни одного символа в течение 1 минуты, то она считает, что имеются сбои в работе утилиты lhb, и завершает её работу сигналом SIGKILL. После этого производится перезапуск lhb аналогично запуску после ее обычного завершения.

Мониторинг сетевого драйвера клиента

Для отслеживания активности сетевого драйвера клиента ему периодически посылается специальный тестовый запрос. В случае отсутствия ответа в течение времени, отведенного на обнаружение разрыва соединения, работа драйвера dbc_tcp прерывается сигналом SIGKILL, и он перезапускается, как и после обычного завершения.

Самоконтроль управляющей программы

Автоматический рестарт

При задании ключа /wd управляющая программа может осуществлять самоконтроль. Для этого порождается еще один дополнительный процесс управляющей программы, который выполняет функции следящего процесса (контроль работы основного процесса). Механизм контроля аналогичен механизму контроля над работой утилиты lhb. В случае обнаружения сбоев в работе основного процесса управляющей программы производится его автоматический рестарт. Сервер перезапускается из UNDEFINED-состояния с разрешением автоматического запуска системы резервирования, то есть автоматически воспроизводится ключ /nq.

Рестарт пользовательской программой

Автоматический механизм самоконтроля может оказаться неудобным при запуске сервера резервирования пользовательским приложением, а именно – приложение может не узнать о перезапуске сервера. Если для пользовательского приложения необходимо знать об останове сервера резервирования, запуск сервера резервирования может

выполняться с параметром E в ключе /wd. Это запрещает автоматический перезапуск сервера. О перезапуске сервера резервирования в данном случае должно позаботиться пользовательское приложение.

Пользовательский мониторинг состояния сервера резервирования

При необходимости мониторинг состояния сервера резервирования можно возложить на пользовательское приложение. Для этого перед запуском сервера резервирования пользовательское приложение должно создать канал pipe и передать его файловый дескриптор вывода (записи) серверу резервирования. Этот файловый дескриптор предварительно должен быть переведен в неблокирующий режим работы. При запуске сервера резервирования необходимо в качестве значения ключа /wd указать номер полученного файлового дескриптора. В данном случае сервер резервирования будет писать в этот файловый дескриптор один или более символов за интервал времени, заданный в ключе /testint, а контролирующий процесс не будет запущен. Пользовательское приложение также должно перевести файловый дескриптор ввода (чтение) созданного канала pipe в неблокирующий режим и периодически пытаться из него читать символы. Если в течение удвоенного времени, заданного в ключе /testint, приложение не прочитает ни одного символа, то предполагается, что сервер резервирования имеет проблемы в работе и должен быть принудительно завершен (возможно, с последующим запуском).

Отслеживание контрольных точек

Контрольные точки создаются на главном сервере утилитой lhb, запущенной на каждом резервном сервере, который подключается к системе резервирования. Наличие контрольной точки не позволяет удалять файл системного журнала, пока он не будет скопирован lhb. При своей работе lhb производит сдвиг контрольной точки на более поздние файлы системного журнала по мере их получения, поэтому в нормальном режиме работы файлы системного журнала удаляются сразу же после переноса их на резервные серверы.

При выходе из строя резервного сервера закрепленная за ним контрольная точка может привести к неограниченному росту объема БД главного сервера, поэтому при старте lhb управляющая программа считывает номер установленной контрольной точки и передает эту информацию на главный сервер.

Главный сервер ведет список подлежащих удалению контрольных точек. Список создается при переходе сервера в режим главного и обновляется при выходе из строя любого резервного сервера. При получении контрольной точки от резервного сервера она удаляется из списка подлежащих удалению контрольных точек. Если резервный сервер завершает свою работу или выходит из строя, то список контрольных точек перечитывается, и контрольная точка вышедшего из строя сервера снова заносится в список подлежащих удалению контрольных точек.

Главный сервер производит периодическую проверку размера системного журнала. Интервал проверки равен интервалу посылки тестовых запросов ядру СУБД ЛИНТЕР. Если размер системного журнала (количество его файлов) превысил заданный, то производится удаление самой старой контрольной точки. При этом все файлы системного журнала от этой контрольной точки до следующей удаляются ядром СУБД ЛИНТЕР. Это предотвращает рост системного журнала на главном сервере. Кроме того, удаление самой старой контрольной точки производится при превышении общего количества контрольных точек. Максимальное значение контрольных точек – 3 точки на сервер, не считая самого сервера. Количество серверов в системе определяется ключом,

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

или, если он не задан, количеством узлов в системе резервирования, описанном в файле `nodetab`.

Управляющая программа сохраняет соответствие имени сервера и соответствующей ему контрольной точки. В случае выхода из строя резервного сервера управляющая программа производит запуск `lhb` для удаления контрольной точки по ее номеру.

Управление рабочей БД

Управление рабочей БД подразумевает выполнение следующих операций:

1) копирование рабочей БД;
2) создание архивного файла после копирования;
3) тестирование БД.

Эти три операции в системе резервирования тесно связаны.

Создание архивного файла возможно только после копирования рабочей БД в архивную БД. Тестирование выполняется только на копии архивной БД после выполнения операции копирования. В общем случае эти операции выполняются в следующей последовательности:

- производится очистка резервного каталога от файлов БД;
- останавливается работа ядра СУБД ЛИНТЕР в специальном режиме;
- производится копирование файлов рабочей БД в резервный каталог;
- запускается ядро СУБД ЛИНТЕР в специальном режиме для рабочей БД;
- создается файл архива резервной БД;
- для архивной БД запускается ядро СУБД ЛИНТЕР с целью «доката» всех незаконченных транзакций по системному журналу;
- ядро СУБД ЛИНТЕР останавливается немедленно после старта;
- запускается утилита тестирования БД `testdb`.

В случае отсутствия необходимости выполнения каких-либо действий они пропускаются. Например, если не задан ключ создания архивного файла (`/archivate`), то пропускается запуск архиватора. Для тестирования предназначены 3 последних действия в списке.

Очистка резервного каталога

Очистка резервного каталога необходима для освобождения в нем места для хранения новой БД. Очистка осуществляется путем удаления всех файлов БД по очереди. Операции удаления периодически прерываются для выполнения программой управления проверки линий связи.

Останов ядра СУБД ЛИНТЕР в специальном режиме

Ядро СУБД ЛИНТЕР в специальном режиме осуществляет перенос изменений из системного журнала БД в файлы БД. Только в этом режиме ядро изменяет файлы таблиц БД. Его останов обеспечивает неизменность файлов таблиц БД во время копирования, а значит, и возможность последующего восстановления данных по системному журналу резервной БД.

Копирование файлов рабочей БД в резервный каталог

Копирование файлов рабочей БД в резервный каталог происходит в пофайловом режиме (последовательно по одному файлу). Сначала выполняется копирование файлов таблиц БД. Поскольку ядро СУБД ЛИНТЕР в специальном режиме остановлено, эти файлы не изменяются.

Так как файлы системного журнала БД сортируются в порядке возрастания, то копируются сначала наиболее старые файлы журнала. Копирование каждого файла журнала осуществляется с его начала. Поскольку утилита `lhb` добавление данных производит только в конец последнего файла системного журнала, такое копирование не приводит к нарушению структуры журнала БД.

Копирование файлов может выполняться как самой управляющей программой (путем поблочного чтения и записи), так и запуском системной утилиты `cp`. Если файлы БД короткие, то управляющая программа копирует их самостоятельно, в противном случае операция копирования одного файла выполняется медленно, поэтому для исключения пропуска обработки сетевых событий запускается внешняя утилита копирования. Пропуски обработки сетевых событий могут привести к ложному обнаружению выхода из строя линии связи или сервера резервирования.

Запуск ядра СУБД ЛИНТЕР в специальном режиме для рабочей БД

Поскольку вся информация рабочей БД скопирована, то для нее снова может быть запущено ядро СУБД ЛИНТЕР в специальном режиме, которое осуществляет перенос измененных данных из системного журнала в таблицы БД.

Создание файла архива резервной БД

Если задан ключ `/archivate`, то после каждого копирования производится создание архивного файла (система резервирования не имеет отдельной команды запуска процесса архивирования). Архивирование выполняется при окончании копирования, которое может быть инициировано отдельной командой или по расписанию. Если выполняется проверка БД, то архивирование не осуществляется. Архивирование всегда выполняется внешней программой (см. ключ /cmdarcadd).

Управляющая программа предполагает два способа работы с внешним архиватором: либо последовательность вызовов для добавления каждого файла резервной БД, либо однократный вызов архиватора с передачей ему всего списка архивируемых файлов. Подробности выбора режима приведены в описании ключа `/cmdarcadd`.

Запуск ядра СУБД ЛИНТЕР для резервной БД

Данная операция является первой в цепочке команд, запускаемых для проверки БД. Всегда выполняется проверка специально созданной копии рабочей БД в архивном каталоге, поэтому перед проверкой обязательно копирование рабочей БД в архивный каталог. Ядро в режиме обычной работы запускается для переноса оставшихся в системном журнале изменений в таблицы БД. После запуска изменения, внесенные незаконченными транзакциями, будут отменены, и БД примет непротиворечивое логическое состояние.

Перед запуском устанавливается отличное от умолчания значение переменной окружения `LINTER_MBX`, поэтому клиентские приложения не смогут работать с данным экземпляром СУБД ЛИНТЕР.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Поскольку от ядра СУБД необходима только процедура начального восстановления БД, то сразу же после запуска оно останавливается.

Если запуск завершился неудачей, это говорит о неуспешной проверке БД и необходимости остановки всей системы резервирования для восстановления.

Тестирование БД

Для непосредственной проверки БД производится запуск утилиты `testdb`. При необходимости передачи ей дополнительных параметров можно использовать ключ `/tdbadd`.

Утилита проверяет внутреннюю структуру архивной БД резервного сервера на непротиворечивость и пригодность для дальнейшего использования. Поскольку эта БД является копией рабочей БД резервного сервера, а та, в свою очередь, копией рабочей БД главного сервера, то фактически осуществляется проверка сразу трех БД: обеих БД резервного сервера и рабочей главного сервера.

Обработка команд оператора

Удаленное управление

Для управления работой системы резервирования используется утилита `hresctl` (`srvcmd`). Она позволяет передавать команды управляющей программе для выполнения заданных действий.

Команда передается по локальной сети с использованием UDP-протокола. Для коммуникации между утилитой `hresctl` (`srvcmd`) и управляющей программой используется тот же порт, что и для обмена между серверами резервирования. После отсылки команды утилита `srvcmd` в течение 5 секунд ждет от управляющей программы ответ на эту команду (для утилиты `hresctl` время ожидания ответа устанавливается ключом `-t`). Если ответа не приходит, утилита `hresctl` (`srvcmd`) завершает свою работу. В данном случае нет гарантии, что команда была доставлена, и ее рекомендуется повторить.

Проверка санкционированности команды

Если управляющая программа запущена с ключом `/cu`, то перед выполнением полученной команды проверяется её санкционированность. Команды получения статистики (`show`, `getstate`) на санкционированность не проверяются. Для остальных команд перед выполнением осуществляется посылка запросов ядру СУБД ЛИНТЕР на открытие канала с данным именем пользователя и паролем, проверку наличия у пользователя прав администратора СУБД ЛИНТЕР и закрытие канала. Для резервного сервера эти запросы пересылаются по сети. В случае неудачи при открытии канала, или если пользователь не имеет прав администратора СУБД, выполнение команды отклоняется.

Так как при проверке санкционированности обязательно наличие активного ядра СУБД ЛИНТЕР на главном сервере, то при его отсутствии, например, в переходных режимах, все команды будут игнорироваться.

Выполнение сетевых команд управляющей программой

После проверки санкционированности команды или непосредственно после её приема управляющая программа анализирует команду на возможность выполнения. Если

команда не может быть выполнена, об этом информируется утилита `hresctl` (`srvcmd`) в ответном сообщении. Причина отказа описывается в текстовой форме.

В случае успешной проверки всех условий выполнения команды управляющая программа приступает к её выполнению. Выполнение команды часто заключается в смене состояния управляющей программы и/или отсылке команд на изменение состояние другим управляющим программам данной системы резервирования.

Причины отказа команд `arc`, `copy`, `testdb`:

"`This server is not SLAVE_OK or is changing state`" – для выполнения команд резервный сервер должен находиться в состоянии готовности;

"`Testdb is already run or scheduled for running`" – получена команда на проверку БД, но уже выполняется копирование, архивирование или проверка БД;

"`Copy or testdb are already run or scheduled for running`" – уже выполняется копирование, архивирование или проверка БД.

Причины отказа команд `main`, `slave`, `exch`:

"`Server is already in MONO or MAIN or SW_TO_MONO or STARTMONO state`" – сервер уже является главным или переключается на роль главного;

"`Server is not in MAIN state`" – сервер в состоянии, отличном от главного;

"`Command exchange disabled in starting mode`" – обмен состояниями невозможен в стартовом (UNDEFINED) режиме.

При переключении состояний и необходимости проведения конкурса на роль главного сервера сервер, получивший команду, становится старшим. То есть он выполняет функции по выбору главного сервера и рассылает команды остальным серверам на изменение своего состояния.

Для выполнения команды на переключение или завершение работы всей системы резервирования используется передача команд управляющей программы одного сервера управляющей программе другого.

Причины отказа выполнения команды

При проверке санкционированности:

"`linter error`" – неуспешный код возврата ядра СУБД ЛИНТЕР;

"`unknown user name`" – неизвестное имя пользователя СУБД ЛИНТЕР;

"`invalid password`" – неверный пароль;

"`inproper system state`" – в данном состоянии ядро СУБД ЛИНТЕР не запущено, и команда выполнена быть не может;

"`not enough user privilegies`" – пользователь не имеет привилегий администратора;

"`server busy, try later`" – выполняется проверка санкционированности другой команды.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

При командах dead_node, exch, dead_server:

"This server is not found or it is already disconnected" – узел не найден или не входит в систему резервирования;

"I'm in going to SLAVE state. I'm not ready" – резервный сервер не готов взять на себя роль главного. Первоначальное скачивание данных не завершено;

"I'm switching to other state" – сервер в переходном состоянии;

"I cannot switch to next server. It doesn't exist" – переключение невозможно из-за отсутствия кандидатов;

"One or more SLAVE isn't ready: …" – некоторые из резервных серверов не находятся в состоянии готовности. После двоеточия перечисляются имена узлов этих серверов;

"There are no servers with good database" – ни один из серверов не может быть главным;

"I'm ready to switch from SLAVE to MONO. But some SLAVE isn't ready: ..." – некоторые из других резервных серверов не находятся в состоянии готовности. После двоеточия перечисляются имена узлов этих серверов.

Для команд stop, shut:

"I'm in going to SLAVE state. I'm not ready" – резервный сервер не находится в состоянии готовности;

"Some SLAVE servers is not ready: …" – некоторые из резервных серверов не находятся в состоянии готовности. После двоеточия перечисляются имена узлов этих серверов.

Управление сигналами

Если управление системой резервирования осуществляется с помощью сигнала, то управляющая программа не выполняет проверку их санкционированности. При выполнении команды по сигналу так же, как и при удаленном управлении, управляющая программа производит переключение состояний и/или передачу команд по сети остальным управляющим программам.

Завершение работы запущенных процессов при переходе в новое состояние

Завершение процессов главным сервером

Перед переходом сервера резервирования в новое состояние управляющая программа предварительно останавливает все запущенные на нем процессы. В режиме главного сервера команду на останов получает ядро СУБД ЛИНТЕР. Оно прекращает обслуживание всех запросов, за исключением запросов утилиты lhb, и ожидает приема всех измененных данных всеми утилитами lhb. Управляющая программа отводит на эту операцию время, определяемое ключом /forceshut. Если в течение заданного интервала времени работа ядра СУБД не завершилась, производится принудительный останов ядра сигналом KILL.

Сетевой драйвер сервера завершает свою работу самостоятельно при обнаружении завершения работы ядра СУБД, но может быть остановлен и управляющей программой после останова ядра СУБД ЛИНТЕР.

Завершение процессов резервным сервером

Управляющая программа резервного сервера, так же, как и главного, при смене состояний завершает все свои процессы. Прежде всего, немедленно завершаются процессы копирования, архивирования и проверки БД, а также функционирование ядра СУБД ЛИНТЕР в специальном режиме. Так как эти операции выполняются только над рабочей БД в состоянии готовности, то нет необходимости в ожидании завершения этих операций. Завершение работы утилиты `lhb` производится либо принудительно, если останавливается только данный сервер, либо ожидается ее самостоятельное завершение вследствие завершения работы ядра СУБД ЛИНТЕР на главном сервере. Тайм-аут ожидания определяется ключом `/forceshut`. После завершения работы утилиты `lhb` завершает работу и сетевой драйвер клиента.

Самостоятельный останов управляющей программы

При невозможности рестарта процессов управляющая программа завершает свою работу, предварительно переходя в специальное состояние останова. При самостоятельном завершении работы выполняются те же действия, что и при обычной смене состояний. Предварительно, при необходимости, изменяются состояния БД, хранящиеся в `state` файлах.

Самостоятельный останов возможен также при отсутствии кандидата на роль главного сервера системы резервирования. Например, если ни один резервный сервер еще не имеет актуальной БД, то работа всей системы резервирования будет завершена. В данном случае администратор или пользовательская программа должны стартовать систему резервирования заново для выбора главного сервера среди оставшихся. При этом администратору необходимо принять решение о новом главном сервере (например, путем поочередного запуска серверов).

Разрешение конфликтов главных серверов

В случае, когда все линии связи с одним из серверов были нарушены, а потом восстановлены, в системе резервирования некоторое время могут присутствовать сразу два «главных» сервера резервирования. В подобном случае старшим из этих серверов принимается решение о том, кто из них должен перейти в состояние резервного. Решение принимается на основании максимального времени нахождения в состоянии главного сервера или по системному журналу. После принятия решения старший сервер либо сообщает об этом второму серверу, либо сам переходит в режим резервного.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Регистрация событий системы резервирования

Информация о функционировании системы резервирования (изменение состояния серверов резервирования, запуск и останов программ, входящих в состав системы резервирования, и другие важные события) может быть доступна обработчику событий – специальной пользовательской программе обработки событий системы резервирования. Имя обработчика событий задается в ключе `/stproc` командной строки запуска сервера резервирования. При вызове обработчика событий ему передаются следующие значения:

- имя программы обработчика события (устанавливается ОС);
- код произошедшего события;
- дополнительная информация о событии (для некоторых событий может отсутствовать).

Обработчик событий

Если пользовательский обработчик событий предусмотрен, то при наступлении любого из событий, перечисленных в таблице 3, управляющая программа системы резервирования вызывает его с передачей соответствующего событию списка аргументов. При этом гарантируется, что до завершения обработки данного события следующее событие не будет передано обработчику. Это обеспечивается механизмом очереди событий на обработку (если очередность не отключена с помощью ключа `/nostprocqueue`).

Формат вызова обработчика событий

Головной модуль пользовательской программы-обработчика событий при разработке ее с использованием языка программирования С должен иметь вид:

```
int main(int argc, char **argv)
  {
   ...
  }
```

где:

1) argc – количество передаваемых аргументов;
2) argv[0] – всегда имя программы обработчика событий (берется из ключа `/stproc`);
3) argv[1] – числовой код события (см. приложение 2);
4) argv[2] – первый дополнительный аргумент;
5) argv[3] – второй дополнительный аргумент.

Значения дополнительных аргументов специфичны для конкретных событий.

При использовании других языков программирования для разработки обработчика событий необходимо использовать их средства для получения аргументов командной строки запуска обработчика.

Нулевой аргумент передается всегда и устанавливается операционной системой. В контексте системы резервирования этот аргумент не важен.

Первым аргументом является код события (представлен в виде числового значения). В аргументе числовое значение кода события передается в виде строки. Первый аргумент всегда присутствует при вызове обработчика событий.

Второй и третий аргументы не являются обязательными. Их формат зависит от кода события.

Наборы событий

Система резервирования имеет два набора событий – стандартный и альтернативный. Альтернативный набор событий порождается при задании ключа `/altstproc`.

В стандартном наборе событий в первом аргументе передается только код (номер) текущего состояния сервера и некоторые коды псевдосостояний. Например, отдельно не выделяются события старта и завершения процессов и т.д. При запуске и останове какого-либо процесса в первом аргументе устанавливается состояние управляющей программы. О том, что конкретно произошло в этом состоянии, можно узнать из дополнительных аргументов.

В альтернативном наборе событий вместо кода состояния в первом аргументе передается код события. Код состояния передается только при смене состояния сервера. Дополнительные аргументы событий при этом остаются без изменений.

Альтернативный набор событий включает дополнительные коды событий:

`E_PROCESS_START`, `E_PROCESS_EXIT`, `E_SHUT_COMMAND`, `E_STOP_COMMAND`, `E_NET_INFO`, `E_TIME_CHANGE`, `E_SERVER_TIME_DIFF`.

Кроме альтернативного набора кодов событий имеются еще и дополнительные события, которые генерируются всегда, независимо от набора, при наличии разрешающих ключей. Например, при указании ключа `/tclog` дополнительно генерируются события, связанные с подстройкой времени: `E_TIME_CHANGE`, `E_SERVER_TIME_DIFF`.

При указании ключа `/watchnet` генерируется дополнительное событие `E_NET_INFO`, информирующее о подключении или отключении удаленных серверов резервирования. Дополнительные аргументы данного события позволяют судить об активных и неактивных в настоящий момент серверах, а также о принадлежности узлов серверам резервирования.

События системы резервирования

Список генерируемых системой резервирования событий приведен в таблице 3.

Таблица 3. События системы резервирования

Имя события	Описание события
UNDEFINED	Генерируется при старте системы резервирования после инициализации сети и до получения первого сетевого сообщения. Информирует об удачном разборе конфигурационных файлов и начале обмена сетевыми сообщениями с другими серверами.
SERVEREXIT	Генерируется непосредственно перед завершением работы управляющей программы.
MONO	Сервер перешел или находится в MONO-состоянии. Главный сервер находится в стационарном одномашинном режиме. В данном состоянии резервные серверы отключены, или

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Имя события	Описание события
	ни один из них не находится в состоянии готовности. В это состояние сервер переходит из состояний SW_TO_MONO или STARTMONO. Также он может возвращаться в MONO-состояние из MAIN-состояния. Кроме того, событие MONO генерируется вместо событий из группы расширенных событий (имена которых начинаются с префикса E_) в случае запрета на их распознавание (сервер запущен без ключа /altstproc).
MAIN	Сервер перешел или находится в состоянии MAIN. Состояние главного сервера при готовности хотя бы одного резервного сервера. В это состояние сервер переходит из MONO-состояния.
SLAVE	Сервер находится или перешел в состояние SLAVE (одно из возможных состояний резервного сервера). В это состояние сервер переходит после SLAVE_WAIT-состояния. Сервер находится в данном состоянии до готовности резервного (SLAVE_OK).
SLAVE_WAIT	Сервер перешел или находится в состоянии ожидания готовности главного сервера к работе. С этого состояния сервер начинает работу в режиме резервного. При получении сообщения о готовности главного сервера из данного состояния резервный переходит с состояние SLAVE.
SLAVE_OK	Резервный сервер перешел или находится в состоянии готовности. В это состояние резервный сервер переходит после SLAVE-состояния.
SW_TO_MONO	Главный сервер начал запуск процессов после работы в любом из режимов, исключая стартовый (UNDEFINED). Обычно предыдущим режимом работы является SLAVE. Из данного режима сервер переходит в MONO-состояние.
SHUT_DOWN	Управляющая программа сервера начала завершение своей работы. Оно начинается с останова процессов.
SLAVEFAILER	Завершение управляющей программы резервного сервера по превышению попыток рестарта процессов или при невозможности запуска процессов, не подлежащих рестарту (копирование, архивирование).
SLAVECRASH	Завершение работы резервного сервера на этапе первоначального копирования данных из-за завершения работы или выхода из строя главного сервера. База данных после такого завершения непригодна к использованию.
MAINFAILER	Завершение управляющей программы главного сервера по превышению попыток рестарта процессов.
MAINCRASH	Завершение работы главного сервера из-за неудачного запуска ядра СУБД ЛИНТЕР для данной БД. Причина определяется по коду завершения ядра СУБД ЛИНТЕР. Обычно после этого рабочая БД главного сервера не пригодна для дальнейшего использования, но можно попытаться ее восстановить. Часто причиной перехода в данное состояние является нехватка дискового пространства на главном сервере.

Имя события	Описание события
STARTMONO	Главный сервер начал запуск процессов после стартового состояния (UNDEFINED). Из данного режима сервер переходит в MONO-состояние.
STOPPED	Управляющая программа главного или резервного сервера завершается. Генерируется непосредственно перед событием SERVEREXIT.
END_STATUS	Зарезервировано.
WAIT_OLDER	Событие генерируется при переходе в SLAVE_WAIT-состояние из UNDEFINED-состояния.
NOT_FOUND	Истек тайм-аут прослушивания сети (см. ключ /wait).
NOT_SHUT_DOWN	Команда на останов системы или сервера игнорирована.
SERVERRESTART	Зарезервировано.
E_PROCESS_START	Старт процесса.
E_PROCESS_EXIT	Завершение процесса.
E_SHUT_COMMAND	Получена удаленная команда на завершение работы системы.
E_STOP_COMMAND	Получена удаленная команда на останов.
E_NET_INFO	Изменение состояния соединения с сервером. Установка соединения или разрыв. Это событие генерируется, только если присутствует ключ `/watchnet` совместно с ключом `/altstproc`. Если ключ `/watchnet` присутствует, а ключ `/altstproc` не задан, то при изменении ситуации в сети обработчику передается вместо `E_NET_INFO` номер текущего состояния сервера.
E_TIME_CHANGE	Обнаружена резкая подстройка системного времени на резервном компьютере.
E_SERVER_TIME_DIFF	Обнаружена разность времени данного сервера и удаленных серверов.
W_DEADLOCK	Ядро СУБД ЛИНТЕР не ответило за интервал посылки. Есть вероятность, что оно зациклилось.
E_TESTDB	Запуск тестирования БД, доклад о результате тестирования.
A_SLAVE_OK	Главный сервер переходит в MAIN-состояние.
A_SHUT_DOWN	Зарезервировано.
A_STOPPED	Событие резервного сервера. Главный сервер получил команду на останов.

Значения дополнительных аргументов

Как было сказано, первый аргумент обработчика событий всегда содержит код события. Формат дополнительных аргументов для каждого кода события приведен ниже.

Событие SLAVE_OK

Первый дополнительный аргумент

Символьная строка с информацией о состоянии БД.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Формат строки:

```
<размер рабочей БД>M<пробел><размер резервной БД>M<пробел><время>
<размер рабочей БД>::= число с плавающей точкой.
<размер резервной БД>::= число с плавающей точкой.
<время>:: = положительное целочисленное значение.
```

Параметры <размер рабочей БД> и <размер резервной БД> характеризуют размеры указанных БД в мегабайтах (числовое значение заканчивается латинской буквой M).

Параметр <время> информирует о длительности (в секундах) загрузки (копирования) на резервный сервер БД с главного сервера.

Пример

```
45.7M 44.9M 236
```

Второй дополнительный аргумент

Отсутствует.

Событие SHUT_DOWN

Первый дополнительный аргумент

Символьная строка, содержащая сообщение о причине останова сервера резервирования или о причине отказа сервера выполнить команду.

Пример

```
By signal
```

Второй дополнительный аргумент

Отсутствует.

Событие NOT_SHUT_DOWN

Первый дополнительный аргумент

Символьная строка (заканчивается двоичным нулем), содержащая сообщение о причине игнорирования сервером команды останова своей работы или изменения состояния.

Пример

```
There are servers with lhb s -startinc worked. Signal ignored
```

Второй дополнительный аргумент

Отсутствует.

Событие SLAVEFAILER

Первый дополнительный аргумент

Командная строка запуска программы, завершение которой привело к состоянию SLAVEFAILER, или причина перехода сервера резервирования в данное состояние.

Если событие произошло при тестировании архивной копии БД, то оно выдается два раза подряд: первый раз в качестве дополнительного аргумента выдается путь к архивной копии БД, второй раз – строка запуска ядра СУБД ЛИНТЕР linter в режиме тестирования.

Пример

```
dbc_tcp /G /ERR1001 /AUTOCONNECTTIMEOUT=10 /PCHECK /K=12
```

Второй дополнительный аргумент

Отсутствует.

Событие SLAVECRASH

Первый дополнительный аргумент

Командная строка запуска программы, завершение которой привело к состоянию SLAVECRASH, или причина перехода сервера резервирования в данное состояние.

Пример

```
There are no servers with actual database
```

Второй дополнительный аргумент

Отсутствует.

Событие MAINFAILER

Первый дополнительный аргумент

Командная строка запуска программы, завершение которой привело к состоянию MAINFAILER, или причина перехода сервера резервирования в данное состояние.

Пример

```
Memory allocation
```

Второй дополнительный аргумент

Отсутствует.

Событие MAINCRASH

Первый дополнительный аргумент

Командная строка запуска программы, завершение которой привело к состоянию MAINCRASH, или причина перехода сервера резервирования в данное состояние.

Пример

```
cp ARC/*.?1 DB
```

Второй дополнительный аргумент

Отсутствует.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Событие A_SLAVE_OK

Первый дополнительный аргумент

Имя удаленного резервного сервера, который перешел в состояние готовности.

Пример

```
SRV1
```

Второй дополнительный аргумент

Отсутствует.

Событие A_STOPPED

Первый дополнительный аргумент

Имя удаленного главного сервера, который получил команду на завершение своей работы.

Пример

```
SRV1
```

Второй дополнительный аргумент

Отсутствует.

Событие E_TESTDB

При запуске тестирования

Первый дополнительный аргумент

Полный путь к тестируемой БД.

Пример

```
/mnt/hdb1/DATABASE/ARC
```

Второй дополнительный аргумент

Отсутствует.

При завершении тестирования

Первый дополнительный аргумент

Полный путь к тестируемой БД.

Пример

```
/mnt/hdb1/DATABASE/ARC 0
```

Второй дополнительный аргумент

Результат тестирования. Положительное значение свидетельствует об успешном завершении собственно процесса тестирования (но не о корректности протестированной БД), отрицательное – о неудачном (таблицы 4, 5).

Пример

```
"/mnt/hdb1/DATABASE/ARC" "0"
```

Таблица 4. Коды завершения в случае «Процесс тестирования завершился успешно»

Код завершения	Результат тестирования
0	БД не содержит ошибок.
1	Некорректная БД. Утилита тестирования обнаружила ошибки в БД.
2	Ядро СУБД ЛИНТЕР не смогло стартовать на тестируемой БД.

Таблица 5. Коды завершения в случае «Процесс тестирования был прерван»

Код завершения	Причина прерывания тестирования
-1	Тестирование прервано (по сигналу SIGKILL) из-за необходимости смены состояния сервера резервирования.
-2	Работа ядра СУБД ЛИНТЕР была завершена сервером резервирования (по сигналу SIGKILL) из-за перехода сервера резервирования в другое состояние.
-3	Невозможно удалить резервную БД перед её копированием с главного сервера.
-4	Неустранимая ошибка при копировании БД.
-5	Ошибка запуска утилиты тестирования `testdb`.
-6	Ошибка запуска ядра СУБД ЛИНТЕР.
-7	Тестирование прервано по сигналу.
-8	Работа ядра СУБД ЛИНТЕР завершена по сигналу.

Событие MONO

Так как событие MONO генерируется вместо событий из группы расширенных событий (имена которых начинаются с префикса `E_`) в случае запрета на их распознавание (сервер запущен без ключа `/altstproc`) дополнительные аргументы соответствуют дополнительным аргументам расширенных событий (за исключением событий, соответствующих `E_TIME_CHANGE`, `E_SERVER_TIME_DIFF`, которые генерируются только со своим кодом события).

Возможные комбинации значений дополнительных аргументов, передаваемых обработчику событий, приведены в таблице 6.

Таблица 6. Комбинации дополнительных аргументов события MONO

№ варианта	Первый дополнительный аргумент	Второй дополнительный аргумент
1	Отсутствует.	Отсутствует.

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

№ варианта	Первый дополнительный аргумент	Второй дополнительный аргумент
2	Командная строка запуска программы.	Отсутствует.
3	Командная строка запуска программы.	Код завершения вызванной программы.
4	Информация об изменении состояния других серверов системы резервирования.	Отсутствует.

Событие MONO с дополнительными аргументами:

1) по варианту № 1 – информирует о переходе сервера в данное состояние;
2) по варианту № 2 – информирует о том, что на сервере в состоянии MONO запущена указанная программа;
3) по варианту № 3 – информирует о том, что на сервере в состоянии MONO завершено выполнение указанной программы;
4) по варианту № 4 – информирует о том, что серверу в состоянии MONO была передана информация об изменении состояния серверов системы резервирования (например, добавлен новый сервер).

Формат передаваемой информации приведен в описании события `E_NET_INFO`.

Событие MAIN

Аналогично событию MONO.

Событие SLAVE

Аналогично событию MONO.

Событие SLAVE_WAIT

Аналогично событию MONO.

Событие SW_TO_MONO

Аналогично событию MONO.

Событие STARTMONO

Аналогично событию MONO.

Событие UNDEFINED

Аналогично событию MONO.

Событие E_PROCESS_START

Первый дополнительный аргумент

Командная строка запуска процесса.

Пример

```
dbs_tcp -p 1060 -K 12 -C -s -w -wd
```

Второй дополнительный аргумент

Отсутствует.

Событие E_PROCESS_EXIT

Первый дополнительный аргумент

Командная строка запуска процесса.

Второй дополнительный аргумент

Код завершения процесса.

Пример

```
"dbs_tcp -p 1060 -K 12 -C -s -w -wd"  "0"
```

Событие E_SHUT_COMMAND

Первый дополнительный аргумент

Shut by srvcmd.

Второй дополнительный аргумент

Отсутствует.

Событие E_STOP_COMMAND

Первый дополнительный аргумент

Stop by srvcmd.

Второй дополнительный аргумент

Отсутствует.

Событие E_NET_INFO

Первый дополнительный аргумент

Формат передаваемой информации:

```
NODES: <элемент описания><элемент описания>…
<элемент описания>::=<пробел><имя узла>[@{<имя сервера>| THIS}]
 <состояние узла>
<имя узла>::= символьная строка длиной до 8 символов.
<имя сервера>::= символьная строка длиной до 8 символов.
<состояние узла>::= символ, указывающий состояние узла.
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Возможные значения <состояние узла>:

- # – узел активен и является основной линией связи (по ней передаются данные);
- + – узел активен и является запасной линией связи;
- - – узел не активен (соединение с узлом отсутствует);
- * – узел принадлежит данному серверу.

Если <имя сервера> не задано, то в качестве имени сервера принимается имя основного узла, к которому по линии связи передаются данные.

Зарезервированное имя THIS указывает на принадлежность узла данному серверу.

Пример

```
NODES: MCBC0@THIS*  WIN-
NODES: MCBC0@THIS*  WIN@WIN#
```

Второй дополнительный аргумент

Отсутствует.

Событие E_TIME_CHANGE

Первый дополнительный аргумент

Величина коррекции времени в секундах.

Второй дополнительный аргумент

Отсутствует.

Событие E_SERVER_TIME_DIFF

Первый дополнительный аргумент

Имя удаленного сервера и разность времени данного сервера и удаленного.

Второй дополнительный аргумент

Отсутствует.

Пример

```
MCBC0  200
```

Событие W_DEADLOCK

Первый дополнительный аргумент

Отсутствует.

Второй дополнительный аргумент

Отсутствует.

Примеры вызова обработчика событий

В нижеследующих примерах пользовательский обработчик событий имеет имя `stat`. Для его вызова необходимо задать ключ `/stproc=stat` в командной строке запуска программы `server`.

Пример 1.

Пусть сервер резервирования сменил статус и перешёл в новое состояние «сервер главный, многомашинный режим» (значение статуса 4). Тогда обработчику событий `stat` будут переданы следующие значения аргументов (на примере языка программирования С. В другом языке программирования выборка аргументов будет осуществляться иным способом):

```
argc=2;
argv[0]="stat";
argv[1]=4.
```

Пример 2.

Пусть текущий статус сервера резервирования равен 4, и он запустил ядро СУБД ЛИНТЕР с командной строкой:

```
linter /pool=3000.
```

Тогда обработчику событий будут переданы следующие значения аргументов:

```
argc=3;
argv[0]="stat";
argv[1]=4;
argv[2]="linter /pool=3000".
```

Пример 3.

Пусть сервер резервирования с текущим статусом 4 зафиксировал завершение (с кодом 1) сетевого драйвера сервера `dbs_tcp`, запущенного командной строкой `dbs_tcp -port=1063`. Тогда обработчику событий будут переданы следующие значения аргументов:

```
argc=4;
argv[0]="stat";
argv[1]=4;
argv[2]="dbs_tcp -port=1063"
argv[3]="1".
```

Пример программы обработчика событий

```
#include <stdio.h>
int main(int argc,char **argv)
  {
if(argc<2  || argc>4)
  printf ("STPROC:  error. argc=%d\n",argc);
  else
    {
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

```
    printf("STPROC:  current  SERVER status =%s\n",argv[1]);
    switch (argc)
      {
      case 2:
      break;
      case 3:
      printf("STPROC: process start %s\n", argv[2]);
      break;
      case 4:
      printf("STPROC: process stop %s, exit status = %s\n",
argv[2], argv[3]);
     }
    }
   exit(0);
  }
```

Сообщения системы резервирования

При первоначальном запуске системы резервирования резервный компьютер, на котором первым был запущен сервер резервирования (программа `server`) считывает из файла `nodetab` адреса всех узлов системы резервирования и ждет от каждого из них информацию о текущем состоянии сервера резервирования (для этого на остальных резервных серверах тоже должен быть запущен сервер резервирования). Если в течение тайм-аута запуска системы резервирования хотя бы от одного узла не пришел ответ о готовности, то сервер резервирования, запущенный первым, на консоль оператора выдает вопрос:

```
Do You want to start another server (Y/N/E)?
(Надо ли запускать недостающий сервер резервирования?)
```

После этого сервер резервирования переходит в ожидание ответа администратора системы резервирования или активизации всех узлов, описанных в протоколе rez в файле `nodetab`. Оператор может либо стартовать систему резервирования в неполном составе (с отсутствием одного или нескольких узлов), либо устранить причину отсутствия сетевого соединения и повторить запуск системы резервирования позднее.

Варианты ответа:

1) "Y" или не отвечать (Y по умолчанию).

   Ответ подразумевает ожидание запуска недостающих серверов резервирования.

   После запуска всех недостающих серверов резервирования выполняется запуск системы резервирования. Статус каждого из серверов резервирования определяется путем сравнения их БД. Старший сервер резервирования будет назначен главным сервером, а остальные серверы – резервными. Каждый сервер резервирования автоматически продолжит работу в соответствии с назначенным ему статусом.
2) "N".

   Ответ разрешает запуск системы резервирования в неполном составе.

   Информация о предполагаемом главном сервере выводится перед запросом администратору на старт системы резервирования. При положительном ответе данный сервер станет главным, остальные серверы – резервными.
3) "E".

   Ответ запрещает работу этого сервера резервирования. Сервер резервирования завершает свою работу.

   С помощью данного ответа можно оставить нужный администратору главный сервер резервирования (если он не согласен с автоматическим выбором главного сервера, осуществленным самой системой резервирования).

Примечание
Если тайм-аут запуска системы резервирования достаточно большой, а запуск серверов резервирования выполняется с небольшими паузами, то при неготовности одного или более серверов резервирования приведенный выше вопрос могут задать несколько серверов резервирования. В этом случае отвечать «Y», «N» надо только на одном (любом) сервере резервирования. О принятом решении остальные серверы извещаются автоматически.

Проверка конфигурирования системы резервирования

Правильность установки и конфигурирования системы резервирования проверяется следующим образом:

1) выбрать один из компьютеров системы резервирования и запустить на нем сервер резервирования (программу `server`):

```
server -debug
```

2) при запросе запущенного сервера резервирования на ожидание подключения других серверов ответить "N".

   Сервер резервирования на выбранном компьютере должен запустить ядро СУБД ЛИНТЕР (`linter`) с его компонентами (`intsrt`, `sql`, `tsp`) и сетевой драйвер сервера (`dbs_tcp`). В конце запуска на консоль должно быть выдано сообщение:

```
Server is in MONO state
```

3) проверить состояние сервера командой

```
server -show
```

   На консоль выдается информация о состоянии системы резервирования, среди которой должна быть строка:

```
server state = MONO
```

4) проверить доступ клиентских приложений к главному серверу. Для этого:

   - на один из клиентских компьютеров скопировать конфигурационный файл `nodetab` с главного сервера резервирования;
   - запустить на клиентском компьютере сетевой драйвер клиента `dbc_tcp`:

```
dbc_tcp -G
```

   - запустить на клиентском компьютере программу `inl` и проверить работу с БД:

```
inl  -u SYSTEM/MANAGER8
SQL>create table test (i int);
SQL>insert into test i=1;
SQL>exit
```

5) проверить запуск резервного сервера. Для этого скопировать файл `nodetab` на резервный сервер и выполнить там команду:

```
server -debug
```

   Программа `server` должна обнаружить уже работающий главный сервер и запустить процесс архивирования БД. При окончании получения БД на консоль должно быть выдано сообщение:

```
Server is in SLAVE_OK state
```

6) проверить переход резервного сервера в режим главного. Для этого остановить главный сервер командой:

```
srvcmd -stop
```

```
-u <пользователь>/<пароль>
<имя/адрес главного узла> <порт главного узла>
```

На консоль главного сервера должно быть выдано сообщение:

```
Server stopped
```

На консоль бывшего резервного сервера через несколько секунд должно быть выдано сообщение:

```
Server is in MONO state
```

7) проверить, что объекты БД, созданные на главном сервере, переданы на резервный сервер. Для этого на клиентском компьютере выполнить команды:

```
inl -u SYSTEM/MANAGER8
SQL>select * from test;
```

Должно выдаться значение 1.

8) закончить проверку:

```
SQL>exit
```


© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Обращение к системе резервирования из клиентского приложения

Клиентское приложение обращается к системе резервирования как к обычному сетевому серверу СУБД ЛИНТЕР, то есть для доступа к системе резервирования используется драйвер сетевого клиента `dbc_tcp`. Это означает, что для работы с системой резервирования клиентское приложение может использовать любой программный интерфейс СУБД ЛИНТЕР. Дополнительные, присущие только системе резервирования, ресурсы не требуются.

При отказе главного сервера системы резервирования его функции берет на себя один из резервных серверов (при их наличии). В этом случае драйвер `dbc_tcp` автоматически переадресует запросы клиентского приложения к новому главному серверу, при этом возможна задержка ответа клиентскому приложению на время тайм-аута переключения с одного сервера на другой. Новый главный сервер не наследует номеров каналов клиентского приложения, курсоров и не гарантирует, что последние транзакции будут выполнены.

Клиентское приложение должно отслеживать момент переключения главного сервера на другой компьютер (по коду завершения ERROPENQUE или ERRNMRKAN) и самостоятельно снова устанавливать соединение. Сетевой драйвер клиента обеспечивает соединение с новым главным сервером резервирования, как только он будет готов. После этого приложение должно повторить все операции, которые не сохраняются в БД при переключении, а именно:

- блокировки данных;
- управление событиями (если это не сохраняемые в БД события);
- последние транзакции.

Пример клиентского приложения приведен в приложении 3.

Коды завершения при работе с системой резервирования

В таблице 7 приведены коды завершения СУБД ЛИНТЕР, возвращаемые клиентскому приложению при работе с системой резервирования (свидетельствуют о переключении серверов системы резервирования).

Реакция клиентского приложения на все перечисленные в таблице 7 коды завершения (кроме кода ERRNMRKAN) стандартная:

- закрыть канал;
- затем снова открыть канал;
- повторить последние транзакции.

Кроме кодов завершения, перечисленных в таблице 7, возможно также получение других кодов завершения, не относящихся непосредственно к системе резервирования. Такие коды завершения клиентские приложения должны обрабатывать так же, как и при работе без системы резервирования. При написании программ удобно обрабатывать все коды завершения, связанные с сетевым взаимодействием, в одном блоке. Для этого достаточно проверить, что значение кода завершения находится в диапазоне от 4000 до 5000.

Если в программе используется несколько каналов, то при получении по одному из них кода завершения, обработка которого предполагает закрытие этого канала, а затем его повторное открытие, необходимо также закрыть, а затем повторно открыть и остальные используемые каналы.

В случае использования интерфейса нижнего уровня (call-интерфейс) для закрытия канала достаточно обнулить поле NumChan во всех контрольных блоках программы. Такой прием используется, поскольку в большинстве случаев дальнейшая работа с сервером СУБД ЛИНТЕР невозможна, и посылка команды закрытия канала бессмысленна. Кроме того, это исключает совпадение номера вновь открытого канала и канала, работавшего с узлом, связь с которым была потеряна, то есть исключается возможность функционирования двух контрольных блоков с одним и тем же номером канала.

Таблица 7. Коды завершения системы резервирования

Обозначение	Числовое значение	Причина
ERR_CONNECT	4006	Ошибка при установке соединения с БД.
ERR_SEND	4007	Ошибка передачи.
ERR_RECV	4008	Ошибка приема.
ERRNMRKAN	1069	Канал не открыт данным процессом.
ERROPENQUE	1001	СУБД ЛИНТЕР на удаленном узле активна.
KernelShutdown	1046	Ядро СУБД ЛИНТЕР находится в процессе останова.
FORCED_ROLLB	6712	Транзакция отменена.
ERR_LINKDOWN	4061	Соединение закрыто сетевым драйвером сервера.
ERR_CONNCLOSED	4009	Соединение закрыто.

Обозначение	Числовое значение	Причина
ChannelBusy	1044	Канал занят.

Приложение 1
Пример фильтра трассировки

Программа `program.awk` выполняет разбивку потока трассировочной информации на отдельные файлы с последующим их архивированием.

```
BEGIN {
  Sum = 0;
  File="server.log";
  SizeLimit = 1000000;
  MaxFilesCount = 10;


  Index = 0;
  FirstFileIndex = 0;
  system("ls " File ".*.bz2 > /tmp/hotres_log_list");
  for(;;)
  {
    Tmp = "";
    getline Tmp < "/tmp/hotres_log_list";
    Files[Index] = Tmp;
    print "Files[" Index"]=" Tmp;
    if ( Tmp == "" )
    {
      break;
    }
    Index++;
  };
  system("rm -f /tmp/hotres_log_list");
  while ( Index - FirstFileIndex > MaxFilesCount)
  {
    print "Delete " Files[FirstFileIndex];
    system("rm -f " Files[FirstFileIndex]);
    delete Files[FirstFileIndex];
    FirstFileIndex++;
  }
}

#/Set timer 9 for/ { next; }
#/TIMEOUT EVENT Id=9/ {next;}
#/State MAIN has been written/ { next;}
#/Timeout 9 is ready/ {next;}

{
  Sum = length($0) + Sum;
  print $0 >> ""File;
```


© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

```
  if ( Sum > SizeLimit )
  {
    Sum = 0;
    close (File);
    FileName = File"."strftime("%y%m%d%H%M%S") Index;
    system("mv -f "File" "FileName);
    system("bzip2 "FileName);
    Files[Index] = FileName ".bz2";
    print "Index=" Index;
    while ( Index - FirstFileIndex >= MaxFilesCount)
    {
      print "Delete " Files[FirstFileIndex];
      system("rm -f " Files[FirstFileIndex]);
      delete Files[FirstFileIndex];
      FirstFileIndex++;
    }
    Index++;
  }
}
```

Приложение 2

Коды событий системы резервирования

```
# define UNDEFINED          -1
# define SERVEREXIT         2
# define MONO               3
# define MAIN               4
# define SLAVE              5
# define SLAVE_WAIT         6
# define SLAVE_OK           7
# define SW_TO_MONO         10
# define SHUT_DOWN          14
# define SLAVEFAILER        15
# define SLAVECRASH         17
# define MAINFAILER         20
# define MAINCRASH          21
# define STARTMONO          26
# define STOPPED            28
# define END_STATUS         29
# define WAIT_OLDER         30
# define NOT_FOUND          31
# define NOT_SHUT_DOWN      32
# define SERVERRESTART      34
# define E_PROCESS_START    35
# define E_PROCESS_EXIT     36
# define E_SHUT_COMMAND     37
# define E_STOP_COMMAND     38
# define E_NET_INFO         39
# define E_TIME_CHANGE      40
# define E_SERVER_TIME_DIFF 41
# define W_DEADLOCK         42
# define E_TESTDB           43
# define A_SLAVE_OK         57
# define A_SHUT_DOWN        64
# define A_STOPPED          78
```


© Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Приложение 3
Пример работы с системой резервирования

В качестве примера клиентского приложения приводится листинг программы `testrez.c`, которая демонстрирует обработку ситуации отказа главного сервера и переход на работу с резервным сервером. Взаимодействие с СУБД ЛИНТЕР выполняется с помощью интерфейса нижнего уровня (call-интерфейса).

Программа создает таблицу TEST, состоящую из 20 записей типа INT, затем в режиме `AUTOCOMMIT` циклически выполняет следующие операции:

1) открывает канал;
2) выполняет выборку каждой записи с выводом на экран;
3) увеличивает значение каждой записи на 1;
4) закрывает канал.

При получении ошибки выводится соответствующее сообщение, и повторяется последняя итерация цикла.

В режиме резервирования (одновременно работают главный и резервный серверы) изменения таблицы на главном сервере передаются в БД на резервной машине.

Если принудительно остановить главный сервер, то программа получит соответствующий код завершения и повторит последнюю итерацию уже с резервной БД. Выведенные значения записей таблицы TEST должны совпадать с теми, что были перед остановом главного сервера, или быть больше.

```
#include <stdio.h>
#include <signal.h>
#include "intlib.h"
#include "errors.h"

#define RECORDCOUNT 20
struct ErrString
{
  int ErrCode;
  char *ErrString;
};

static int Cbrk = 0;

static struct ErrString ES[]=
{
  {ERR_RECV,        "receive error"},
  {ERR_SEND,        "send error"},
  {ERR_CONNECT,     "connect error"},
  {ERR_RESLINK,     "reserver link lost "},
  {ERR_SEARCH,      "reserved server search timeout"},
  {ERR_LINKDOWN,    "connection has been closed"},
  {ERRNMRKAN,       "wrong channel number"},
```

```
  {ERROPENQUE,     "linter not found"},
  {KernelShutdown, "linter kernel is shutdowning"},
  {FORCED_ROLLB,   "forced rollback"},
  {ChannelBusy,    "channel is busy"},
  {0," "}
};

static int TestError(int Error)
{
  if ( Error >= 4000 && Error < 5000 )
    return 0;
  switch ( Error )
  {
    case ERRNMRKAN:
    case ERROPENQUE:
    case KernelShutdown:
    case FORCED_ROLLB:
    case ChannelBusy:
      return 0;
  }
  return 1;
}

static char  *ErrorString(int Err)
{
  struct ErrString *ESP=ES;

  while(ESP->ErrCode)
  {
    if(ESP->ErrCode==Err)
      return(ESP->ErrString);
    ESP++;
  }

  return(ESP->ErrString);
}

static int CloseChannel(TCBL *Cbl)
{
  memcpy(Cbl->Command, "CLOS", 4);
  inter(Cbl, NULL, NULL, NULL, NULL);
  if ( Cbl->CodErr )
  {
    printf("CLOSE error=%ld,%s\n", Cbl->CodErr, ErrorString(Cbl-
>CodErr));
    return -1;
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

```
  }
  return 0;
}

void SIGINTHandler(int Sig)
{
  printf("cbrk received\n");
  signal(SIGINT, SIGINTHandler);
  Cbrk = 1;
}

int main(int argc, char **argv)
{
  TCBL Cbl;
  int  i;
  int  AfterError = 0;

  signal(SIGINT, SIGINTHandler);

  /* Open channel */
  memset(&Cbl, 0, sizeof(Cbl));
  memcpy (Cbl.Command, "OPEN", 4);
  if(argc == 2)
    strncpy(Cbl.Node, argv[1], sizeof(Cbl.Node));
  inter(&Cbl, "SYSTEM/MANAGER8", NULL, NULL, NULL);
  if (Cbl.CodErr != 0)
  {
    printf("Channel Open error=%ld: %s\n", Cbl.CodErr,
 ErrorString(Cbl.CodErr));
    exit(1);
  }

  /* Create test table */
  memset(Cbl.Command, ' ', 4);
  inter(&Cbl, NULL, "CREATE TABLE TEST(I INTEGER);", NULL, NULL);
  if (Cbl.CodErr)
  {
    if(Cbl.CodErr == RELEXIST)
    {
      printf("table TEST already exist\n");
    }
    else
    {
      printf("Create Table error=%d:%s\n", Cbl.CodErr,
 ErrorString(Cbl.CodErr));
      CloseChannel(&Cbl);
```

```
      return -1;
    }
  }
  else
  {
    /* Full test table */
    for(i = 0; i < RECORDCOUNT; i++)
    {
      char ComString[128];

      sprintf(ComString, "INSERT INTO Test VALUES (%d);", i);
      inter(&Cbl, NULL, ComString, NULL, NULL);
      if (Cbl.CodErr != 0)
      {
        printf("INSERT error=%d:%s\n", Cbl.CodErr,
ErrorString(Cbl.CodErr));
        CloseChannel(&Cbl);
        return -1;
      }
    }
  }

  if ( CloseChannel(&Cbl) )
  {
    return -1;
  }

  while(!Cbrk)
  {
    int StrCnt = 0;

    /* Open channel */
    memcpy (Cbl.Command, "OPEN", 4);
    inter(&Cbl, "SYSTEM/MANAGER8", NULL, NULL, NULL);
    if (Cbl.CodErr != 0)
    {
      printf("Channel Open error=%ld:%s\n",
Cbl.CodErr,ErrorString(Cbl.CodErr));
      if( TestError(Cbl.CodErr) == 0)
      {
        AfterError = 1;
        sleep(5);
        printf("try again\n");
        continue;
      }
      return -1;
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

```
    }

    /* Select all data */
    memcpy(Cbl.Command, "SLCT", 4);
    Cbl.PrzExe = M_BINARY;
    Cbl.LnBufRow = sizeof(int);
    inter(&Cbl,NULL,"SELECT I FROM TEST;", NULL,(HPCHAR)&i);
    if (Cbl.CodErr != 0)
    {
      printf("SELECT error=%ld:%s
\n",Cbl.CodErr,ErrorString(Cbl.CodErr));
      if(TestError(Cbl.CodErr) == 0)
      {
        CloseChannel(&Cbl);
        continue;
      }
      CloseChannel(&Cbl);
      return -1;
    }

    /* Print data */
    printf("\n");
    while (Cbl.CodErr == 0)
    {
      StrCnt++;
      if ( StrCnt == 7 )
      {
        printf("\n");
        StrCnt=1;
      }
      printf(" i=%d", i);

      memcpy(Cbl.Command, "GETN", 4); /* Get Next Answer */
      inter(&Cbl, NULL, NULL, NULL, (HPCHAR)&i);
    } /* while */

    if (Cbl.CodErr == 2) /* End of Answer */
      printf("\n no more records\n");
    else
    {
      if ( TestError(Cbl.CodErr) == 0 )
      {
        CloseChannel(&Cbl);
        continue;
      }
      else
```

```
      {
        CloseChannel(&Cbl);
        return -1;
      }
    }

    /* Wait for compare */
    if(AfterError)
    {
      printf("press key to continue\n");
      getchar();
    }
    AfterError = 0;

    /* Update data */
    memset(Cbl.Command, ' ', 4); /* Command is not Select ! */
    inter(&Cbl, NULL,"update TEST set I = I+1;", NULL, NULL);
    if (Cbl.CodErr != 0)
    {
      printf("update error=%ld:%s\n",
 Cbl.CodErr,ErrorString(Cbl.CodErr));
      if ( TestError(Cbl.CodErr) == 0 )
      {
        CloseChannel(&Cbl);
        continue;
      }
      else
      {
        CloseChannel(&Cbl);
        return -1;
      }
    }

    if (CloseChannel(&Cbl))
    {
      continue;
    }
  }

  return 0;
}
```

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026

Указатель ключей

?, 30

A
altstproc, 36
arc, 89
archivate, 48
arcint, 51

B
barc, 48

C
cf, 46
cmdarcadd, 56
cmdarcext, 58
cmpsyslog, 28
copy, 89
crash, 51
cryptadd, 30
cu, 33

D
dbsctout, 45
dbtestint, 59
dead_node, 90
dead_server, 90
debug, 39
delsrv, 67
devenv, 49
diff, 54
down, 27

E
exch, 87
exchdir, 54

F
forceshut, 34

G
getstate, 88

H
h, 30

I
input, 25
instsrv, 66

L
lintadd, 44
lintlog, 39
local, 26
log, 43
lretry, 44

M
main, 88
mklhbarc, 49

N
nocf, 48
nodaemon, 26
nostprocqueue, 37
nq, 23
nservers, 25
ntab, 32

P
parent, 63
pathtoarc, 33
pathtodb, 32
pf, 29
pid, 43
pool, 37
port, 23
pretry, 45
prior, 27

S
sc, 66
setstate, 34
show, 31, 88
shut, 88
slave, 89
spool, 38
st, 63
stop, 88
stproc, 35
syslogcount, 55

T
tclog, 36
tcorrect, 65
tdbadd, 62
testdb, 60, 89
testint, 58
testslave, 29
treq, 62
tstlimit, 59

U
unsafe_mode, 30

V

version, 31

W

wait, 22
watchnet, 37
wd, 64
wp, 24

 © Система резервирования. АО НПП «РЕЛЭКС», 1990-2026