УДК 004.056.3

СИСТЕМЫ РЕЗЕРВНОГО КОПИРОВАНИЯ

Широв П.Д.[1], студент гр.153503, Тушинская Е.В.[2], магистрант гр.256241

Белорусский государственный университет информатики и радиоэлектроники[1]
г. Минск, Республика Беларусь

Рогов М.Г. – ассистент кафедры информатики

Аннотация. Данная научная статья представляет собой обзор и анализ систем резервного копирования. В статье рассматриваются основные принципы функционирования систем резервного копирования, их роль в обеспечении сохранности данных и защите от потери информации. Проводится обзор традиционных методов резервного копирования, таких как полное, дифференциальное и инкрементное копирование. Анализируются их преимущества и недостатки, а также области их применения. Статья обсуждает важные аспекты, связанные с выбором и реализацией системы резервного копирования, такие как определение целей и требований, выбор подходящих технологий и инструментов, управление ресурсами и обеспечение безопасности данных. В целом, данная статья предоставляет полный обзор систем резервного копирования, их компонентов и применения.

Ключевые слова. Система резервного копирования, система избыточного резервирования, кластерная архитектура, реплицирование, физическое резервное копирование, логическое резервное копирование, автономное резервное копирование, полное резервное копирование, дифференциальное резервное копирование, инкрементальное резервное копирование, топология, показатель точки восстановления, время восстановления, отставание, несоразмерность.

Современные организации и предприятия все больше осознают важность сохранности своих данных. Они сталкиваются с угрозами, такими как аппаратные сбои, программные ошибки, вредоносные атаки и природные катастрофы, которые могут привести к потере ценных данных. В случае таких потерь, компании могут столкнуться с серьезными финансовыми и юридическими последствиями, а также с потерей доверия клиентов. Наличие полной резервной копии, независимо от того, каким методом она создана, означает, что локальный набор данных полностью зарезервирован. Для небольших наборов данных это довольно обычное дело. Для 10 Тбайт это может занять невероятное количество времени.

В процессе организации резервного копирования ставятся две основные задачи: восстановление инфраструктуры при сбоях (Disaster Recovery) и ведение архива данных в целях последующего обеспечения доступа к информации за прошлые периоды. Классическим примером резервной копии для Disaster Recovery является образ системной партиции сервера, созданный программой Acronis True Image. Примером архива может выступить ежемесячная выгрузка баз данных из «1С», записанная на кассеты с последующим хранением в специально отведенном месте.

Есть несколько факторов, по которым отличают резервную копию для быстрого восстановления от архива:

1 Период хранения данных. У архивных копий он достаточно длительный. В некоторых случаях регламентируется не только требованиями бизнеса, но и законодательно. У копий для аварийного восстановления он сравнительно небольшой. Обычно создают одну или две (при повышенных требованиях к надежности) резервные копии для Disaster Recovery с максимальным интервалом в сутки-двое, после чего они перезаписываются свежими. В особо критичных случаях возможно и более частое обновление резервной копии для аварийного восстановления, например, раз в несколько часов.

2 Быстрота доступа к данным. Скорость доступа к длительно хранящемуся архиву в большинстве случаев не критична. Обычно необходимость «поднять данные за период» возникает в момент сверки документов, возврата к предыдущей версии и т.д., то есть не в аварийном режиме. Другое дело – аварийное восстановление, когда необходимые данные и работоспособность сервисов должны быть возвращены в кратчайшие сроки. В этом случае скорость доступа к резервной копии является крайне важным показателем.

3 Состав копируемой информации. В архивной копии обычно содержатся только пользовательские и бизнес-данные за указанный период. В копии, предназначенной для аварийного восстановления, помимо этих данных, содержатся либо образы систем, либо копии настроек операционной системы и прикладного программного обеспечения, а также другой информации, необходимой для восстановления.

Иногда возможно совмещение этих задач. Например, годовой набор ежемесячных полных «снимков» файлового сервера, плюс изменения, сделанные в течении недели. В качестве инструмента для создания такой резервной копии подойдет True Image.

При разборе методов резервного копирования нужно определиться, какие именно методы резервного копирования мы собираемся анализировать.

Первой классификацией можно поделить методы резервного копирования на физические и логические.

При физическом резервном копировании базы данных создаются резервные копии реальных файлов, в которых хранятся данные. Это означает, что поддерживаются свойственные базе данных форматы файлов, и обычно в базе есть набор метаданных, которые определяют, какие есть файлы и какие структуры БД в них находятся. Если, создавая резервные копии файлов, вы ожидаете, что другой экземпляр базы данных сможет их использовать, то вам потребуется сделать резервную копию и сохранить связанные с ней метаданные, на которые опирается эта база данных, чтобы резервная копия была переносимой.

При создании логической резервной копии данные экспортируются из базы в формат, который теоретически можно перенести в любую систему. Обычно метаданные тоже сохраняются, но, скорее всего, они будут актуальны для того момента, когда выполнялось резервное копирование. Примером является экспорт всех операторов вставки, необходимых для заполнения пустой базы данных при ее обновлении. Другой пример – строка в формате JSON. В итоге логическое резервное копирование, как правило, занимает очень много времени, потому что это не физическая операция копирования и записи, а построчное извлечение данных. Аналогично восстановление сопровождается всеми обычными издержками базы данных, такими как блокировки и создание журнальных записей повторов и отмены.

Отличный пример такого разделения – различие между репликацией на основе строк и репликацией на базе операторов. Во многих реляционных базах данных репликация на основе операторов означает, что после записи в систему контроля версий к ним добавляется журнал операторов языка манипулирования данными (DML, или вставка, обновление, замена, удаление). Эти операторы передаются в реплики, в которых воспроизводятся. Другой подход к репликации основан на строках или захвате данных изменений (Change Data Capture, CDC). Также методы резервного копирования могут делиться на автономные и оперативные. Рассмотрим каждый из них.

Автономное (или холодное) резервное копирование – такое копирование, при котором экземпляр базы данных, использующий файлы, отключен. Благодаря этому можно быстро скопировать файлы, не беспокоясь о сохранении состояния на текущий момент, пока другие процессы читают и записывают данные. Это идеальное, но очень редко встречающееся состояние. При оперативном (или горячем) резервном копировании вы все равно копируете все файлы, но есть дополнительная сложность, связанная с необходимостью получить согласованный моментальный снимок данных, который должен существовать то время, в течение которого выполняется резервное копирование. Кроме того, если текущий трафик обращается к базе данных во время резервного копирования, необходимо быть осторожными и следить за тем, чтобы не превысить пропускную способность подсистемы ввода-вывода на уровне хранилища. Даже ограничивая нагрузку, вы можете обнаружить, что механизмы, используемые для поддержания согласованности, вносят необоснованные задержки в работу приложения.

Полное резервное копирование (или Full backup) является главным и основополагающим методом создания резервных копий, при котором выбранный массив данных копируется целиком. Это наиболее полный и надежный вид резервного копирования, хотя и самый затратный. В случае необходимости сохранить несколько копий данных общий хранимый объем будет увеличиваться пропорционально их количеству. Для предотвращения подобного расточительства используют алгоритмы сжатия, а также сочетание этого метода с другими видами резервного копирования: инкрементным или дифференциальным. И, конечно, полное резервное копирование незаменимо в случае, когда нужно подготовить резервную копию для быстрого восстановления системы с нуля.

Инкрементное копирование в отличие от полного резервного копирования в этом случае копируются не все данные (файлы, сектора и т.д.), а только те, что были изменены с момента последнего копирования. Для выяснения времени копирования могут применяться различные методы, например, в системах под управлением операционных систем семейства Windows используется соответствующий атрибут файла (архивный бит), который устанавливается, когда файл был изменен, и сбрасывается программой резервного копирования. В других системах может использоваться дата изменения файла. Понятно, что схема с применением данного вида резервного копирования будет неполноценной, если время от времени не проводить полное резервное копирование. При полном восстановлении системы нужно провести восстановление из последней копии, созданной полным резервным копированием, а потом поочередно «накатить» данные из инкрементных копий в порядке их создания. В случае создания архивных копий этот метод необходим, чтобы сократить расходуемые объемы на устройствах хранения информации (например, сократить число используемых ленточных носителей). Также это позволит минимизировать время выполнения заданий резервного копирования,

что может быть крайне важно в условиях, когда приходится работать в плотном графике 24x7 или прокачивать большие объемы информации.

У инкрементного копирования есть один нюанс, который нужно знать. Поэтапное восстановление возвращает и нужные удаленные файлы за период восстановления. Допустим, по выходным дням выполняется полное копирование, а по будням инкрементное. Пользователь в понедельник создал файл, во вторник его изменил, в среду переименовал, в четверг удалил. Так вот при последовательном поэтапном восстановлении данных за недельный период мы получим два файла: со старым именем за вторник до переименования, и с новым именем, созданным в среду. Это произошло потому, что в разных инкрементных копиях хранились разные версии одного и того же файла, и в итоге будут восстановлены все варианты. Поэтому при последовательном восстановлении данных из архива «как есть» имеет смысл резервировать больше дискового пространства, чтобы смогли поместиться в том числе и удаленные файлы.

Дифференциальное резервное копирование отличается от инкрементного тем, что копируются данные с последнего момента выполнения полного резервного копирования. Данные при этом помещаются в архив «нарастающим итогом». В системах семейства Windows этот эффект достигается тем, что архивный бит при дифференциальном копировании не сбрасывается, поэтому измененные данные попадают в архивную копию, пока полное копирование не обнулит архивные биты. В силу того, что каждая новая копия, созданная таким образом, содержит данные из предыдущей, это более удобно для полного восстановления данных на момент аварии. Для этого нужны только две копии: полная и последняя из дифференциальных, поэтому вернуть к жизни данные можно гораздо быстрее, чем поэтапно накатывать все инкременты. К тому же этот вид копирования избавлен от вышеперечисленных особенностей инкрементного, когда при полном восстановлении старые файлы, подобно птице Феникс, возрождаются из пепла. Возникает меньше путаницы. Но дифференциальное копирование значительно проигрывает инкрементному в экономии требуемого пространства. Так как в каждой новой копии хранятся данные из предыдущих, суммарный объем зарезервированных данных может быть сопоставим с полным копированием. И, конечно, при планировании расписания (и расчетах, поместится ли процесс бэкапа во временное «окно») нужно учитывать время на создание последней, самой «толстой», дифференциальной копии.

Рассмотрим, какие бывают топологии резервного копирования:

Децентрализованная схема. Ядром этой схемы является некий общий сетевой ресурс (рисунок 1).

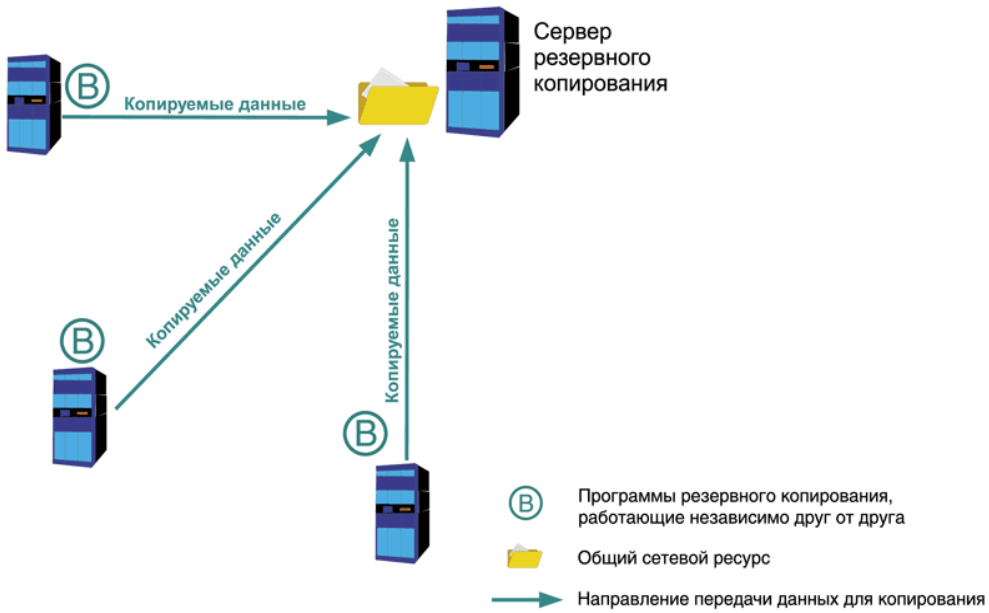

Рисунок 1 – Децентрализованная схема резервного копирования

Например, общая папка или FTP-сервер. Необходим и набор программ для резервного копирования, время от времени выгружающих информацию с серверов и рабочих станций, а также других объектов сети (например, конфигурационные файлы с маршрутизаторов) на этот ресурс. Данные программы установлены на каждом сервере и работают независимо друг от друга. Несомненным плюсом является простота реализации этой схемы и ее дешевизна. В качестве программ копирования подойдут штатные средства, встроенные в операционную систему, или программное обеспечение, такое как СУБД. Например, это может быть программа ntbackup для семейства Windows,

программа tar для UNIX-like операционных систем или набор скриптов, содержащих встроенные команды SQL-сервера для выгрузки баз данных в файлы резервных копий. Еще одним плюсом является возможность использования различных программ и систем, лишь бы все они могли получить доступ к целевому ресурсу для хранения резервных копий. Минусом является неповоротливость этой схемы. Так как программы установлены независимо друг от друга, то и настраивать приходится каждую по отдельности. Довольно тяжело учитывать особенности расписания и распределять временные интервалы, чтобы избежать конкуренции за целевой ресурс. Мониторинг также затруднен, процесс копирования с каждого сервера приходится отслеживать отдельно от других, что в свою очередь может привести к высоким трудозатратам. Поэтому данная схема применяется в небольших сетях, а также в ситуации, когда невозможно организовать централизованную схему резервного копирования имеющимися средствами. Более подробное описание этой схемы и практическую организацию можно найти в.

Централизованное резервное копирование. В отличие от предыдущей схемы в этом случае используется четкая иерархическая модель, работающая по принципу «клиент-сервер» (рисунок 2).

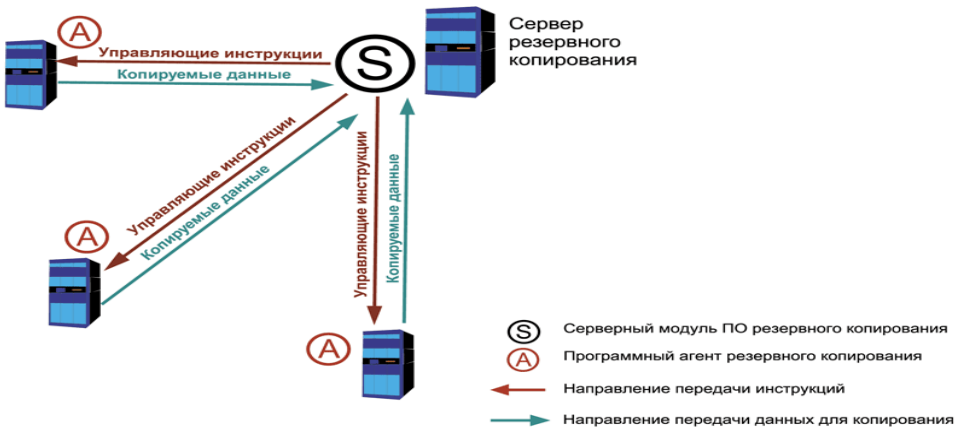

Рисунок 2 – Централизованная схема резервного копирования

В классическом варианте на каждый компьютер устанавливаются специальные программы-агенты, а на центральный сервер – серверный модуль программного пакета. Эти системы также имеют специализированную консоль управления серверной частью. Схема управления выглядит следующим образом: с консоли создаем задания для копирования, восстановления, сбора информации о системе, диагностики и так далее, а сервер дает агентам необходимые инструкции для выполнения указанных операций. Помимо различных агентов для большинства операционных систем существуют разработки для резервного копирования популярных баз данных и корпоративных систем, например, для MS SQL Server, MS Exchange, Oracle Database и так далее.

Для совсем небольших компаний в некоторых случаях можно попробовать упрощенный вариант централизованной схемы резервного копирования без применения программ-агентов (рисунок 3).

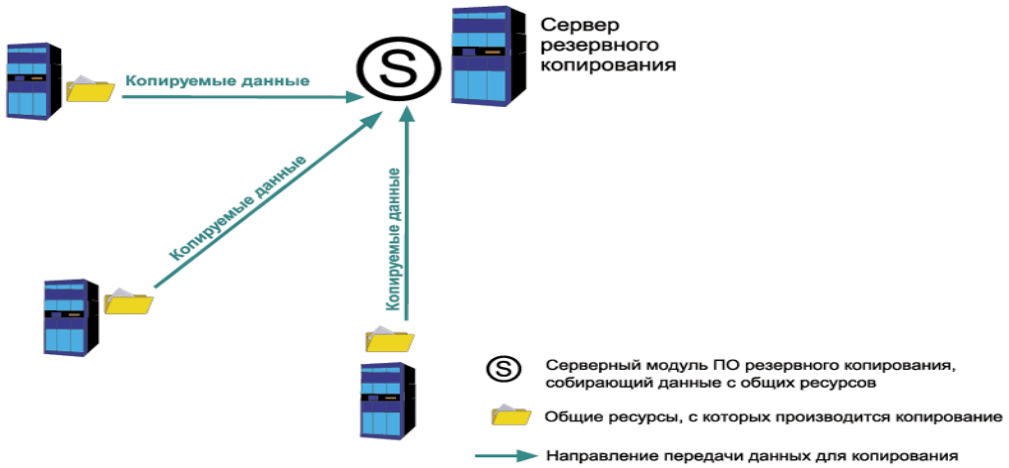

Рисунок 3 – Упрощенная централизованная схема резервного копирования

Также эта схема может быть задействована, если не реализован специальный агент для используемого ПО резервного копирования. Вместо этого серверный модуль будет использовать уже существующие службы и сервисы. Например, «выгребать» данные из скрытых общих папок на Windows-серверах или копировать файлы по протоколу SSH с серверов под управлением UNIX-систем. Данная схема имеет весьма существенные ограничения, связанные с проблемами сохранения файлов, открытых для записи. В результате подобных действий открытые файлы будут либо пропущены и не попадут в резервную копию, либо скопированы с ошибками. Существуют различные методы обхода данной проблемы, например, повторный запуск задания с целью скопировать только ранее открытые файлы, но нет ни одного надежного. Поэтому такая схема подходит для применения только в определенных ситуациях. Например, в небольших организациях, работающих в режиме 5x8, с дисциплинированными сотрудниками, которые сохраняют изменения и закрывают файлы перед уходом домой. Для организации такой усеченной централизованной схемы, работающей исключительно в среде Windows, неплохо подходит ntbackup.

Иногда организуют смешанную схему резервного копирования (рисунок 4). Например, с серверов, для которых есть в наличии программы-агенты резервного копирования, данные собираются посредством этих агентов. Для всех остальных ресурсов используется децентрализованная схема.

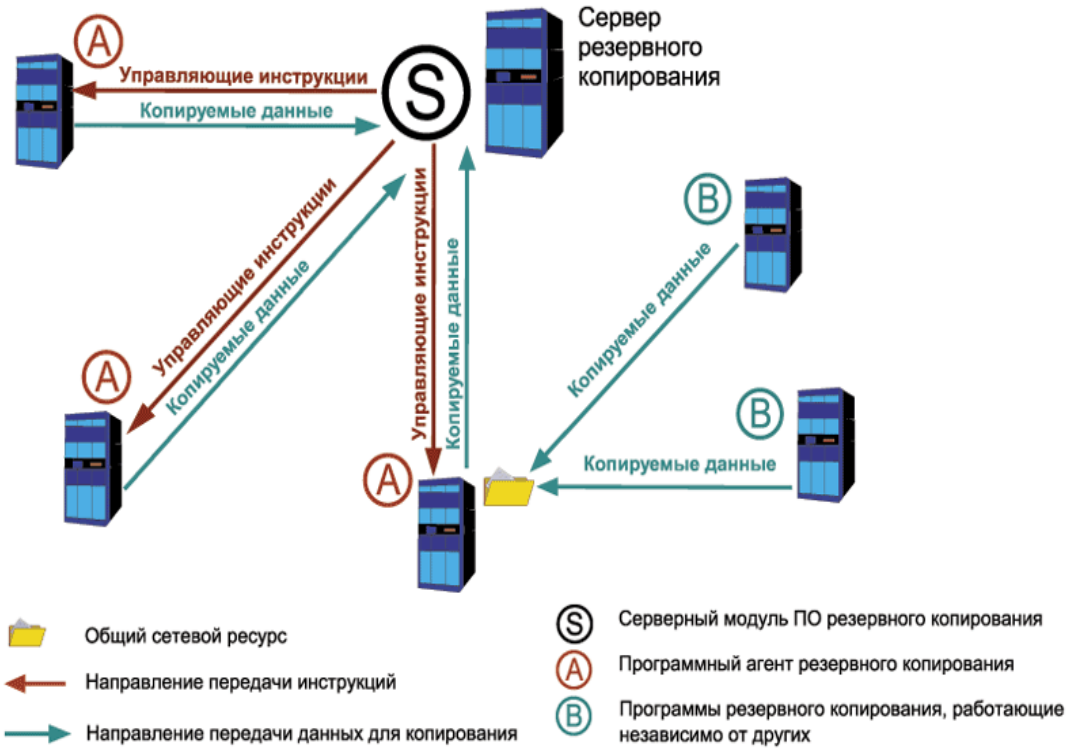

Рисунок 4 – Смешанная схема резервного копирования

Инкрементное резервное копирование – это такое, при котором последовательные копии данных содержат только ту часть, которая изменилась с момента создания предыдущей резервной копии (рисунок 5). Когда требуется полное восстановление, для процесса восстановления потребуется последняя полная резервная копия плюс все инкрементные резервные копии до момента восстановления. Инкрементное резервное копирование часто желательно, поскольку оно сокращает использование дискового пространства и выполняется быстрее, чем дифференциальное резервное копирование. Основная форма инкрементного резервного копирования состоит в идентификации, записи и, следовательно, сохранении только тех файлов, которые изменились с момента последнего резервного копирования. Поскольку изменений обычно немного, инкрементное резервное копирование намного меньше и быстрее, чем полное резервное копирование. Например, после полной резервной копии в пятницу резервная копия в понедельник будет содержать только те файлы, которые изменились с пятницы. Резервная копия во вторник содержит только те файлы, которые изменились с понедельника, и так далее. Полное восстановление данных, естественно, будет медленнее, поскольку должны быть восстановлены все приращения. В случае сбоя какой-либо из созданных копий, включая первую (полную), восстановление будет неполным.

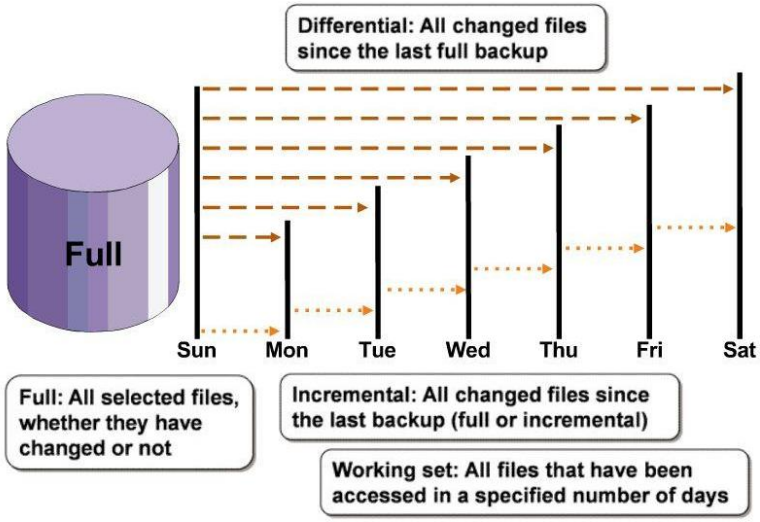

Рисунок 5 – Инкрементное резервное копирование

Более сложная формой инкрементного резервного копирования является многоуровневое инкрементное резервное копирование (рисунок 6). Оно включает несколько пронумерованных уровней резервного копирования. Полное резервное копирование имеет уровень 0. Резервная копия уровня n будет создавать резервные копии всего, что изменилось с момента последней резервной копии уровня n-1. Предположим, например, что резервная копия уровня 0 была сделана в воскресенье. Резервная копия уровня 1, сделанная в понедельник, будет включать только изменения, внесенные с воскресенья. Резервная копия уровня 2, сделанная во вторник, будет включать только изменения, внесенные с понедельника. Резервная копия уровня 3, созданная в среду, будет включать только изменения, внесенные со вторника. Если в четверг была сделана резервная копия уровня 2, она будет включать все изменения, внесенные с понедельника, поскольку понедельник был самой последней резервной копией уровня n-1.

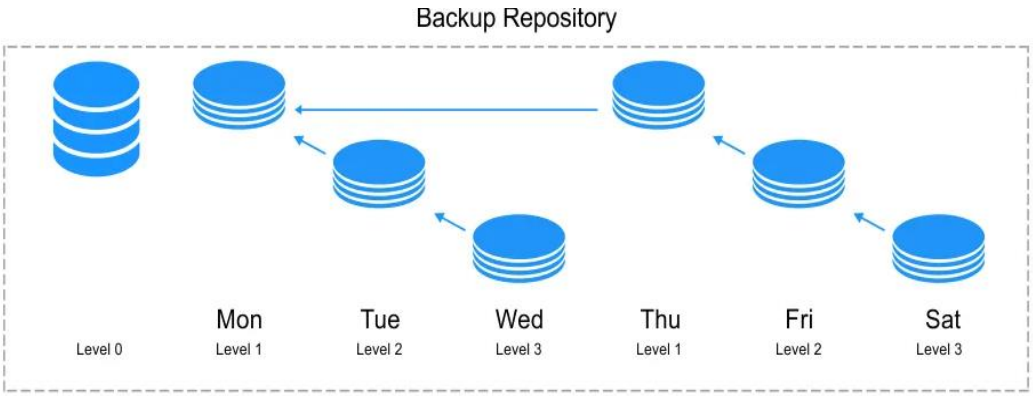

Рисунок 6 – Многоуровневое инкрементное резервное копирование

Инкрементное резервное копирование изменений, внесенных между двумя экземплярами данных, может быть прямым или обратным.

Если самая старая версия данных рассматривается как базовая, а самая новая версия – как исправленная, создаваемая инкрементная копия является прямой инкрементной.

Если новейшая версия данных рассматривается как базовая, а самая старая версия – как исправленная / измененная версия, полученная инкрементная копия является обратной инкрементной.

При создании резервных копий с использованием обратных инкрементных резервных копий каждый раз, когда выполняется обратная инкрементная резервная копия, она применяется (в обратном порядке) к предыдущей полной (синтетической) резервной копии (рисунок 7). Таким образом, текущая полная (синтетическая) резервная копия всегда является резервной копией самого последнего состояния системы. Это является отличием от прямых инкрементных резервных копий, где текущая полная резервная копия является резервной копией самой старой версии системы, и чтобы получить резервную копию самого последнего состояния системы, все прямые инкрементные резервные копии должны быть последовательно применены к этой самой старой версии.

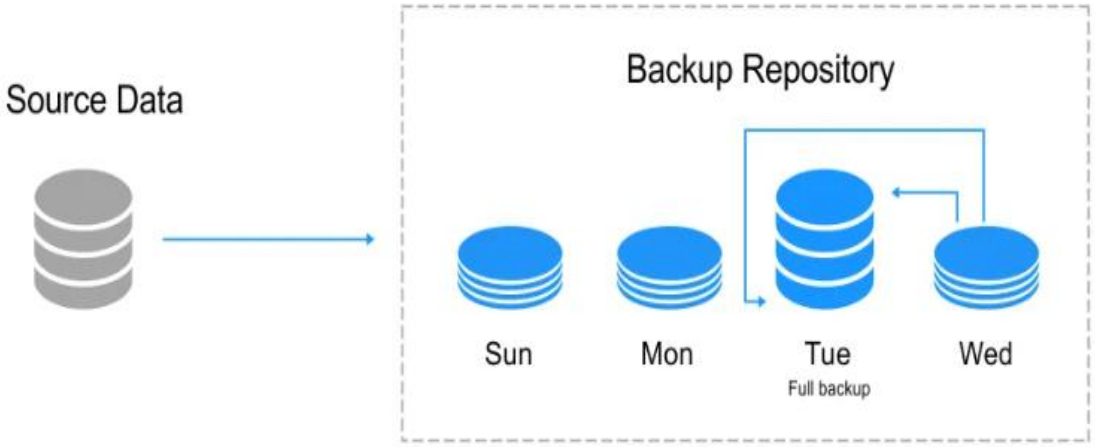

Рисунок 7 –Обратное инкрементное резервное копирование

При применении обратного инкрементного копирования к данным результатом будет предыдущая версия данных. Это дает возможность вернуться к любой предыдущей версии данных. Другими словами, после первоначального полного резервного копирования каждая последующая инкрементная резервная копия применяет изменения к предыдущей полной, каждый раз создавая новую синтетическую полную резервную копию, сохраняя при этом возможность возврата к предыдущим версиям.

Основным преимуществом этого типа резервного копирования является более эффективный процесс восстановления, поскольку самая последняя версия данных (которая является наиболее часто восстанавливаемой версией) представляет собой (синтетическую) полную резервную копию, и при восстановлении к ней не нужно применять дополнительные файлы. Обратное инкрементное резервное копирование работает как для лент, так и для дисков, но на практике, как правило, лучше работает с дисками.

Инкрементное «Forever». Этот стиль аналогичен концепции синтетического резервного копирования. После первоначального полного резервного копирования только инкрементные резервные копии отправляются в централизованную систему резервного копирования (рисунок 8). Этот сервер отслеживает все приращения и отправляет соответствующие данные обратно клиенту во время восстановления. Это может быть реализовано путем отправки каждого приращения непосредственно на ленту по мере его создания, а затем рефакторинга лент по мере необходимости. При наличии достаточного объема дискового пространства можно поддерживать онлайн-зеркало вместе с предыдущими инкрементными изменениями, чтобы можно было восстановить текущие или более старые версии создаваемых резервных копий систем. Это подходящий метод в случае банковских систем.

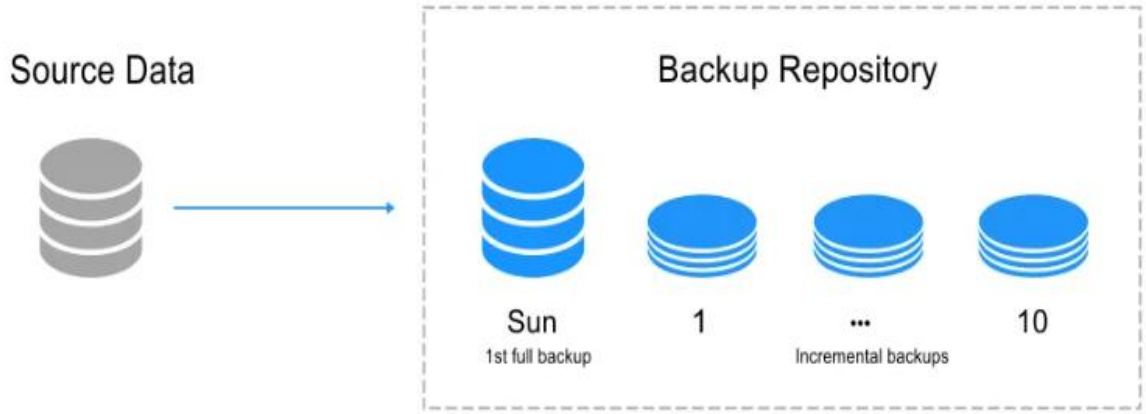

Рисунок 8 – «Forever» инкрементное резервное копирование

Инкрементное резервное копирование на уровне блоков. Этот метод создает резервные копии только блоков в измененном файле. Это требует более высокого уровня интеграции между отправителем и получателем.

Инкрементное резервное копирование на уровне байтов. Эта технология резервного копирования аналогичны методу инкрементного резервного копирования "на уровне блоков"; однако метод инкрементного резервного копирования в байтах (или двоичных файлах) основан на двоичном изменении файлов по сравнению с предыдущей резервной копией: в то время как блочные технологии работают с большими изменяющимися единицами (блоками по 8 КБ, 4К или 1 КБ), байтовые технологии работают с минимальной единицей, экономя место при отражении изменений в файле. Еще одно важное отличие заключается в том, что они работают независимо в файловой системе. На данный момент это технологии, которые обеспечивают максимальное относительное сжатие данных, что становится большим преимуществом для безопасного копирования, осуществляемого через сеть.

Рассмотрим методы оценки эффективности систем резервного копирования.

Рассмотрим еще два важных показателя резервного копирования Показатели точки восстановления (RPO – Recovery Point Objective) и времени восстановления (RTO – Recovery Time Objective). Показатель времени восстановления (RTO) определяет количество времени с момента наступления разрушительного события до момента, когда затронутые ресурсы должны быть полностью работоспособны и готовы поддерживать цели организации.

Когда ресурс выходит из строя, может потребоваться несколько действий, например, замена поврежденных компонентов, перепрограммирование и тестирование, прежде чем ресурс будет снова введен в эксплуатацию и начнется обычный режим работы. Существует обратная зависимость между временем восстановления и затратами, необходимыми для поддержки восстановления. В частности, чем короче RTO по времени, тем больше затраты на восстановление, и наоборот. Поэтому очень важно, чтобы при определении значений RTO участвовали руководители бизнес-подразделений. Они могут захотеть, например, чтобы целевое время восстановления составляло 30 минут, но затраты на достижение этой цели могут оказаться непомерно высокими. Иллюстрацию RTO можно увидеть ниже (рисунок 9):

Рисунок 9 – Демонстрация RTO

Показатель точки восстановления (RPO) особенно важен, когда речь идет о резервном копировании и восстановлении данных. Наглядная иллюстрация RPO ниже (рисунок 10):

Рисунок 10 – Демонстрация RPO

Организациям – например, банкам, компаниям, выпускающим кредитные карты – которые проводят много операций в течение дня, вероятно, потребуется более частое резервное копирование, почти в режиме реального времени, чтобы иметь в наличии самые актуальные критические данные для

своих конкретных нужд, доступные для будущих операций. Это означает, что данные не должны сильно устареть с момента последнего резервного копирования, то есть данные должны быть как можно более актуальными. В этом и есть суть RPO, чтобы резервные копии данных были как можно более актуальными. На рисунке мы видим обратную зависимость между значением RPO и затратами на его достижение. Очень короткое RPO, например, от 10 до 30 секунд, означает, что резервное копирование данных должно выполняться очень часто, что требует использования высокоскоростных технологий резервного копирования, таких как зеркалирование или репликация данных, особенно если резервные копии хранятся вне площадки в облаке или другом месте. Добавьте к этому пропускную способность сети, необходимую для передачи больших объемов данных, и затраты могут быть значительными для достижения требуемой доступности данных.

Оба показателя являются важными элементами, используемыми в планах резервного копирования и восстановления данных. В идеале оба показателя должны быть ключевыми элементами резервного копирования и восстановления, чтобы обеспечить доступность критически важных данных и систем в случае необходимости, особенно после разрушительного события.

Кроме случаев их использования в планах восстановления, на практике они совершенно разные. RTO назначаются после наступления события. RPO используются до наступления события. Впрочем, когда эти два показателя связаны между собой, короткий RTO обычно требует столь же короткого RPO, особенно если речь идет о защите данных. Если мы рассматриваем только резервное копирование и восстановление систем, значения RTO может быть достаточно для определения того, как будет происходить восстановление. Однако если восстанавливаемая система также работает с критическими данными, то обе метрики должны быть согласованы.

Снижение затрат на хранение информации [1] сопровождается ростом объёмов текущей информации. Высвобождение места для новой информации приводит к тому, что устаревшая информация часто оказывается недоступной. Восстановление компьютерной системы, пришедшей в неработоспособное состояние, состоит из устранения физических поломок оборудования и восстановления сохранённого состояния информации. Для этой цели производится резервное копирование информации. Поскольку ошибка в данных, приводящая к краху системы, может быть замечена не сразу, одновременно должно храниться несколько резервных копий данных системы. При поломке, вызванной хакерским вторжением, сохранение прошлых состояний системы позволяет уточнить источник, характер и возможные масштабы повреждений и организовать не только восстановление системы, но и предупреждение рецидивов. При длительной работе над кодом и другими документами потребность в доступе к старым версиям настолько высока, что привела к отдельному классу популярного программного обеспечения – системам управления версиями. Разрабатываются файловые системы, сохраняющие историю изменений и позволяющие восстанавливать прежние состояния, некоторые из них (CDP-continuous data protecting) сохраняют все без исключения изменения. Все перечисленные ситуации роднят общие особенности: · Сохранение всех версий требует места и усложняет доступ. · Актуальность версий со временем снижается. · Старые версии со временем приходится удалять. · Невозможно предсказать, когда и какая именно из старых версий потребуется в будущем. Проблема резервного копирования сложна и рассматривалась в разных аспектах в многочисленных публикациях и патентах последних лет. Удивительно, что среди них не удалось обнаружить ни одной публикации об оптимизации различной продолжительности существования сохранённых изменений. Практически всегда проблема бездумно возлагается на администратора – либо как, например в [2], предполагается, что все изменения хранятся одинаковое время, либо администратору предлагается вручную изменять индивидуальный срок для версии. Отсюда следует, что такой способ повышения эффективности использования памяти для хранения резервных копий не является общеизвестным. По всей видимости, задача такой оптимизации ставится впервые даже в простейшем описанном ниже варианте. Будем считать, что в определённые моменты времени делаются снимки системы (резервные копии) и каждая из них сохраняется в течение некоторого времени. Рассмотрим множество $V = [t_1, t_2] \subset R$ отрезков временной оси. Определим расписание $S \subset V$ как множество отрезков $s = [s_1, s_2] \in S$, представляющих отдельные снимки (версии). Каждый отрезок $s \in S$ однозначно идентифицируется моментом s_1, в который сделан снимок. Правый конец отрезка s_2 означает момент уничтожения снимка. В каждый момент времени t имеется подмножество

$$S_{1(t)} = s_1 | \, t \in s \in S\#(1)$$

снимков. Обозначим $\Delta(S, t) \subset V$ множество всех отрезков, образованных соседними точками множества $S_1(t)$. Эти отрезки образованы моментами соседних сохранившихся в момент t снимков.

Иными словами, концы такого такого отрезка $\left[s_1, s_1^{'}\right]$ лежат в $S_1(t)$, но его внутренность $(s_1, s_1^{'})$ не пересекается с $S_1(t)$. Зафиксируем границы $E_{min} > 0$ и:

$$E_{max} = (s_2 - t) > E_{min} \#(2)$$

продолжительности времени от снятия копии до её возможного использования.

В зависимости от системы, нижняя граница E_{min} может равняться либо длительности карантина (промежутка времени, в течение которого все изменения в информации сохраняются), либо времени технической задержки от момента снимка до его возможного использования, например, равняться шагу дискретизации времени. Верхняя граница E_{max} – это максимальная продолжительность хранения снимков, в теории, например, бесконечность. *Перспективой* прошлого D назовём пару функций D_i : $R \mapsto R$ с условием $D_1(t) < D_2(t) < t$, сопоставляющих каждому моменту времени интервал прошлого $D(t) = (D_1(t), D_2(t))$. Удобно для каждого $t \in$ R выделять одну или несколько перспектив прошлого. Например:

$$D^0(t) = [t - E_{max}, t - E_{min}] \#(3)$$

Перспективу прошлого D будем называть более узкой, чем перспектива D', если в каждый момент времени $D(t) \subset D'(t)$.

Основные качества расписания – доступность старых данных и экономность хранения – могут характеризоваться разнообразно. Попытаемся выделить достаточно простые, но адекватные показатели.

Простейшим показателем доступности данных является максимальный интервал резервных копий $\mathcal{A}(S)$, то есть наибольшая разница между временем искомого снимка и временем ближайшего сохранённого не позже. Она равна наибольшему интервалу времени между соседними сохранившимися снимками:

$$A(S, D) = (s_1 - s_1') \#(4)$$

Здесь и далее $\left(s_1, s_1^{'}\right)$ – это числовой интервал с концами s_1 и $s_1^{'}$. Максимальный интервал разумно использовать лишь в кратковременных перспективах, когда можно пренебречь ослаблением интереса к устаревающим версиям. Привычное удаление старых данных этому противоречит. Удобно учитывать снижение актуальности – версий со временем простой формулой, основанной на обратной соразмерности возрасту.

Отставанием расписания S в перспективе D назовём величину:

$$R(S, D) = \frac{t_2 - t_1}{t - t_2} \#(5)$$

Эта характеристика относительная: масштабирование времени на неё не влияет. Если, например, нам интересен момент x дней назад, то отставание, равное y, обеспечивает наличие снимка, который, если и старше раньше желаемого, то не более, чем на xy минут. Поэтому и используется термин отставание: если часы отстают на секунду в сутки, то за неделю они отстанут на семь секунд.

Экономность расписания можно оценивать средним количеством хранимых снимков за длительный промежуток времени. Однако выяснение близости такой оценки к оптимальной является трудной нерешённой задачей. Поэтому мы ограничимся более простыми критериями.

Назовём неравномерностью расписания S в перспективе D величину:

$$A_{\Delta}(S, D) = \frac{A(S, D)}{t2 - t1} - 1 \#(6)$$

В частности, неравномерность равна нулю, если сохранившиеся снимки сделаны через равные промежутки времени. Если же, например, из четырёх снимков подряд второй или третий удалить прежде, то максимальный интервал сохранения возрастёт вдвое, а неравномерность станет равна

единице. Неравномерность имеет относительный аналог, учитывающий снижение актуальности версий со временем. Назовём несоразмерностью расписания S в перспективе D величину:

$$B(S,D) = \frac{R(S,D)(t-t_2)}{t_2-t_1} - 1 \#(7)$$

Несоразмерность равна нулю в том и только в том случае, если отставание одинаково в любой области. Иными словами, это означает, что интервалы между сохранившимися снимками пропорциональны сроку хранения. Если один – из такого набора снимков (не крайний) удалить, то отставание возрастёт примерно в два раза, а несоразмерность более чем на единицу.

Нулевая несоразмерность может возникнуть лишь на мгновение. Однако, чем выше несоразмерность, тем меньшую перспективу охватывает то же количество снимков с тем же отставанием. Для уменьшения несоразмерности приходится частично удалять снимки.

Выясним, намного ли удаление снимков способно понизить несоразмерность.

ТЕОРЕМА. Пусть для расписания S в перспективе D удаляется снимок, а какой-нибудь более ранний и какой-нибудь более поздний на некоторое время остаются. Тогда $B(S,D) > \sqrt{1 + R(S,D)}$, причём оценка точна.

ДОКАЗАТЕЛЬСТВО. Пусть $t_1 < t_2 < t_3$ – моменты последовательных снимков, средний из которых удаляется. В момент $t' > t$ после удаления снимка t_2 пусть $I = [s_1, s_1']$ – тот отрезок из $\Delta(S, t')$, который содержит $[t_1, t_3]$. Тогда:

$$R(S,D) \geq \frac{s_1' - s_1}{t - s_1'} \geq \frac{t_3 - t_1}{t' - t_3} \#(8)$$

Предельный переход в неравенстве даёт $R(S,D) > \frac{t_3-t_1}{t-t_3}$. Нам понадобится вспомогательное неравенство:

ЛЕММА. Пусть $t_1 < t_2 < t_3 < t$ – вещественные числа и:

$$\mu = min\left(\frac{t_2-t_1}{t-t_2}, \frac{t_3-t_2}{t-t_3}\right) \#(9)$$

Тогда:

$$\frac{t_3-t_1}{t-t_3} \geq 2\mu + \mu^2 \#(10)$$

причём равенство достигается при:

$$\frac{t_2-t_1}{t-t_2} = \frac{t_3-t_2}{t-t_2} \#(11)$$

ДОКАЗАТЕЛЬСТВО. Линейной заменой $y = \frac{x-t}{t_3}$ общий случай сводится к частному $t = 0, t_3 = -1$. Обозначим $a = \frac{t_3-t_2}{t-t_3}$ и $b = \frac{t_2-t_1}{t-t_2}$. Получаем $\mu = min(a,b), t_2 = -1-a$,

$$t_1 = t_2 - b(1+a) = -(1+a)(1+b) \#(12)$$

и, соответственно,

$$\frac{t_3-t_1}{t-t_3} = a + ab + b \geq \mu + \mu^2 + \mu \#(13)$$

Условие превращения в равенство теперь очевидно. Решая квадратичное неравенство (10), получаем:

$$\mu \leq \sqrt{1 + \frac{t_3 - t_1}{t - t_3}} - 1 \leq \sqrt{1 + R(S,D)} - 1 \#(14)$$

Пусть $i \in 1{,}2$ такое, что $\mu = \frac{t_{i+1} - t_i}{t - t_{i+1}}$. Тогда:

$$B(S,D) \geq \frac{R(S,D)(t - t_{i+1})}{(t_{i+1} - t_i)} - 1 \geq$$

$$\geq \frac{R(S,D)}{\sqrt{1 + R(S,D)} - 1} - 1 =$$

$$= \sqrt{1 + R(S,D)} \#(15)$$

Равенство в лемме показывает, что равенство в теореме достигается если, например, снимков всего три.

Выше показано, что относительная несоразмерность не просто неизбежно присутствует, но всегда больше 1, и лишь при частых снимках (малом отставании) может быть приближена к 1.

Обычно для реляционных СУБД используется резервное копирование через заданный в конфигурации интервал времени, и все эти резервные копии хранятся в течение заданного длительного периода. По умолчанию в [3] указаны интервал между снимками 60 мин и срок хранения 30 дней. Это означает недоступность более давних чем месяц данных при ничтожном отставании данных трёхнедельной и более давности $(R(S, D_1) = 0.002)$. Это при перспективе вчерашнего дня $R(S, D_2) > 0.04$. Высокие отставание и несоразмерность в перспективе от вчерашних суток до месяца $R(S, D_3) > 0.04$ и $B(S, D_3) > 20$ показывают, что основное место в хранилище отводится на хранение наименее ценной информации. Соразмерное удаление старой информации позволило бы с тем же относительным отставанием и объёмом хранения хранить данные в десятки раз дольше. Более соразмерный подход используется грамотными системными администраторами.

С незапамятных времён с малосущественными вариациями приводится в руководствах системных администраторов (см, например, [4]) следующая рекомендация. Данные сохраняются ежесуточно и хранятся в течение недели. Еженедельно из них выделяется недельная копия, хранящаяся в течение последующего месяца, ежемесячная, хранящаяся в течение последующего года и ежегодная, хранящаяся в течение последующего десятилетия. Таким образом, предлагается набор перспектив, в каждой из которых снимки распределены равномерно. В общей перспективе от двух недель до десяти лет (диапазон 200 против 30 предыдущего примера) несоразмерность близка к 12. При этом отставание близко к единице (например, в начале года доступными за позапрошлый год могут оказаться только данные, близкие к новогодним праздникам).

В ходе исследования данной научной статьи выделены абсолютные и относительные показатели, позволяющие оценивать эффективность управления временем жизни резервных копий. Относительные показатели – отставание и несоразмерность – учитывают снижение актуальности со временем. Для них найдена точная оценка, обусловленная удалением снимков (версий). Выявлена несоразмерность общепринятых рекомендаций, более чем на порядок превышающая теоретически возможную. Поставлена задача поиска расписания с несоразмерностью, близкой к предельно возможной. Ожидается, что соразмерное расписание удаления старой информации способно обеспечить многократное снижение отставания при одновременной экономии требуемых для хранения резервных копий ресурсов памяти.

Список использованных источников:

1. Beagrie N., Chruszcz J., Lavoie B. Keeping research data safe : A cost model and guidance for uk universities. Sailsbury, UK :Charles Beagrie Limited, 2008. — 67 p.,

2. Frisch A. System Backup: Methodologies, Algorithms and Efficiency Models // Handbook of Network and System Administration / eds. Bergstra J., Burgess M. — Amsterdam : Elsevier, 2007, p. 1028.

3. Stuns D., Buterbaugh T., Bryla B. Ocp: Oracle 10g Administration II Study Guide : Exam 1z0-043 : John Wiley & Sons, 2006. — 752 p.

4. Ciampa M. Security Guide to Network Security Fundamentals.2nd ed. Boston, MA: Course Technology press, 2008. — 509 p.

UDC 004.056.3

BACKUP SYSTEMS

Shchirov P.D.[1], Tushinskaya E.V.[2]

Belarusian State University of Informatics and Radioelectronics[1], Minsk, Republic of Belarus

Rogov M.G. – Assistant of the Department of Computer Science

Annotation. This scientific article is a review and analysis of backup systems. The article discusses the basic principles of operation of backup systems, their role in ensuring data safety and protection against information loss. An overview of traditional backup methods such as full, differential and incremental backups is provided. Their advantages and disadvantages, as well as their areas of application are analyzed. The article discusses important aspects related to the selection and implementation of a backup system, such as defining goals and requirements, selecting appropriate technologies and tools, managing resources, and ensuring data security. Overall, this article provides a comprehensive overview of backup systems, their components and applications.

Keywords. Backup system, redundant backup system, cluster architecture, replication, physical backup, logical backup, offline backup, full backup, differential backup, incremental backup, topology, recovery point indicator, recovery time, lag, disparity.