Принципы практической реализации современных архивных хранилищ данных

Г.А. Егоров, В.И. Шяудкулис, М. Финотти, М.И. Беляков

Аннотация. Рассмотрены вопросы практической реализации архивных хранилищ на базе современных носителей данных. Значительное внимание уделено методам интеграции и поддержки различных устройств хранения информации (облачные хранилища, Blu Ray-диски, современные магнитные ленты, твердые диски, и т.д.) в стандартных операционных системах. Подробно рассмотрены назначение и функции программного обеспечения, реализующего основные требования к современным архивным системам.

Ключевые слова: архивация данных, сменные носители, облачные хранилища, магнитная лента, миграторы, операционные системы.

Введение

Неудержимое увеличение объемов информации в электронной форме предъявляет новые требования к методам и принципам ее хранения. Это связано не только с проблемой хранения огромных объемов данных, но и с их классификацией, защитой от несанкционированного доступа, постоянным контролем хранимых данных на их достоверность и др. Особые требования предъявляются к архивным хранилищам, предназначенным для долгосрочного хранения данных. В [1] рассмотрены общие требования к современным архивным хранилищам данных и основные направления их развития. В настоящей статье основное внимание уделяется принципам практической реализации современных хранилищ данных на примере архивной системы QStar Archive Storage Manager (QStar ASM), разработанной компанией QStar Technologies [2], имеющей 25-летний опыт работы на рынке архивных систем.

1. Основные проблемы практической реализации архивных хранилищ

К основным требованиям, которые определяют общие характеристики архивных систем и должны учитываться при их разработке [1], относятся: независимость от аппаратуры; доступность архивной информации по мере необходимости; аудит информации; потребление электроэнергии; независимый от производителей доступ к информации; интеллектуальность устройств хранения данных; предотвращение катастроф; расширяемость; цена.

Архивные хранилища должны быть в полной мере независимы от операционных систем и структуры файловых систем. В то же время стандартные файловые системы не удовлетворяют в полной мере требованиям архивных хранилищ и не могут быть основой для создания архивных систем. Для обеспечения этих требований разработчикам архивных систем

остается единственное решение – создание специализированных программных продуктов. Для доступа к архивным хранилищам необходимо использовать открытые, общепризнанные сетевые протоколы (NFS, CIFS и др.), чтобы обеспечить достаточную независимость от конкретной реализации архивной системы.

Практическая реализация архивных систем связана с решением ряда проблем, обусловленных следующими требованиями.

Интеграция (портация) в современные операционные системы. **А**рхивная система, как любая программная система, как правило, должна выполняться в конкретной операционной среде (Windows, UNIX, Linux, AIX, Solaris, MAC и др.), осюда вытекает важное требование – независимость архивных систем от операционной системы (ОС). Другая проблема интеграции связана с необходимостью инкорпорации архивной файловой системы в конкретную ОС. В большинстве ОС файловые системы включены в ядро, однако сложность и значительный объем программного обеспечения архивных файловых систем не позволяют их реализацию в составе ядра, кроме того, необходимо учитывать постоянное развитие ОС, что приводит к значительным затратам по поддержке программного обеспечения архивных систем. Отсюда вытекает требование реализации архивных файловых систем в пространстве пользователя с обеспечением какого-либо механизма взаимодействия с ядром ОС.

Поддержка различных типов устройств хранения данных. Каждый новый тип устройств хранения данных связан с новыми стандартами их программирования. Например, программирование магнитных лент значительно отличается от программирования дисков или оптических накопителей, хотя все они поддерживают протокол SCSI. В то же время программирование облачных хранилищ не зависит от аппаратных интерфейсов, а основано на использовании сетевых протоколов (TCP/IP, http). Отсюда вытекает необходимость в разработке специального слоя программного обеспечения, обеспечивающего поддержку разнотипных устройств хранения данных, включая: облачные хранилища различных производителей; Blu Ray-диски и другие оптические диски; твердотельные диски (SSD); магнитные ленты.

Поддержка библиотек носителей данных. Емкость современных архивов данных достигает многих тера- и петабайт. При этом, как правило, для ответственных применений требуется иметь, по крайней мере, три копии архивных данных: локальную, удаленную и автономную (резервную). Создание современных архивных хранилищ возможно только с применением библиотек ленточных носителей, и задача управления такими носителями является одним из важных требований.

Поддержка базового набора операций с файловой системой. Архивная файловая система должна предоставлять базовый набор операций для возможности работы через сетевые протоколы NFS или CIFS, включая: операции над каталогами (opendir, readdir, closedir, …); операции над файлами (open, read, close, write, …); операции над расширенными атрибутами файлов (Windows-streams, MAC-forks, расширенные атрибуты в UNIX); операции над атрибутами доступа к файлам (ACL в Windows-системах, POSIX ACL в UNIX-системах); операции над файловой системой (монтирование, состояние, проверки, восстановление и др.).

Управление файловыми системами со сменными носителями. В отличие от дисковых систем хранения данных, где носители всегда находятся в режиме on-line (постоянного подключения), использование сменных носителей (обычно магнитных лент) требует специальных средств для их управления. Например, при обращении к файлу, отсутствующему в дисковом кэше, должна быть выполнена команда загрузки носителя из библиотеки носителей. При этом необходимо зарезервировать ленточное устройство (drive, в дальнейшем – устройство), подождать в очереди, если все устройства заняты, загрузить ленту, позиционировать ее, прочитать данные в кэш и передать их в пользовательскую программу. В случае, если требуемая лента находится вне библиотеки (off-line), необходимо послать сообщение оператору, чтобы он достал ленту из хранилища и импортировал ее в библиотеку. Для выполнения всего набора таких операций требуется большой объем программного обеспечения с участием в этом процессе достаточно ненадежного звена – человека.

Верификация данных. В стандартных файловых системах контроль данных обеспечивается на аппаратном уровне. Это не удовлетворяет требованиям архивных хранилищ данных, где сменные носители данных в связи со значительным периодом их сохранности могут подвергаться различным внешним воздействиям, поэтому для архивных хранилищ должны создаваться различные методы верификации хранимых данных.

Уничтожение данных. Архивные системы должны обеспечивать возможность уничтожения данных файла с целью недопустимости последующего их восстановления. При использовании сменных носителей, с учетом удаленных и автономных копий архивных данных, процесс уничтожения данных является достаточно сложной процедурой. Программное обеспечение современных архивных систем должно обеспечивать уничтожение данных с участием их владельца.

В последующих разделах рассматривается структура программного обеспечения архивной системы QStar ASM с учетом приведенных требований.

2. Структура программного обеспечения архивной системы QStar ASM

В структуре программного обеспечения архивной системы (Рис. 1) можно выделить следующие основные модули.

1. **File system Interface (FSI)** (интерфейс файловой системы) обеспечивает доступ к системе хранения данных с помощью протоколов (NFS/CIFS, DFS, HTTP и др.). Модуль DFS, обеспечивающий локальный интерфейс к файловой системе, реализуется в ядре операционной системы или использует транспортный модуль файловых запросов ядра FUSE [3], CBFS [4], чтобы обеспечить возможность реализации файловых функций в пространстве пользователя. Для ОС, где механизм FUSE не реализован (Solaris, AIX), QStar ASM предоставляет собственные аналогичные средства, реализованные в ядре ОС.

2. **Cache File System (CFS)** (кэш-файловая система) является центральным ядром программного обеспечения, представляющим полноценную файловую систему, обслуживающую запросы от FSI. Кроме того, CFS предоставляет

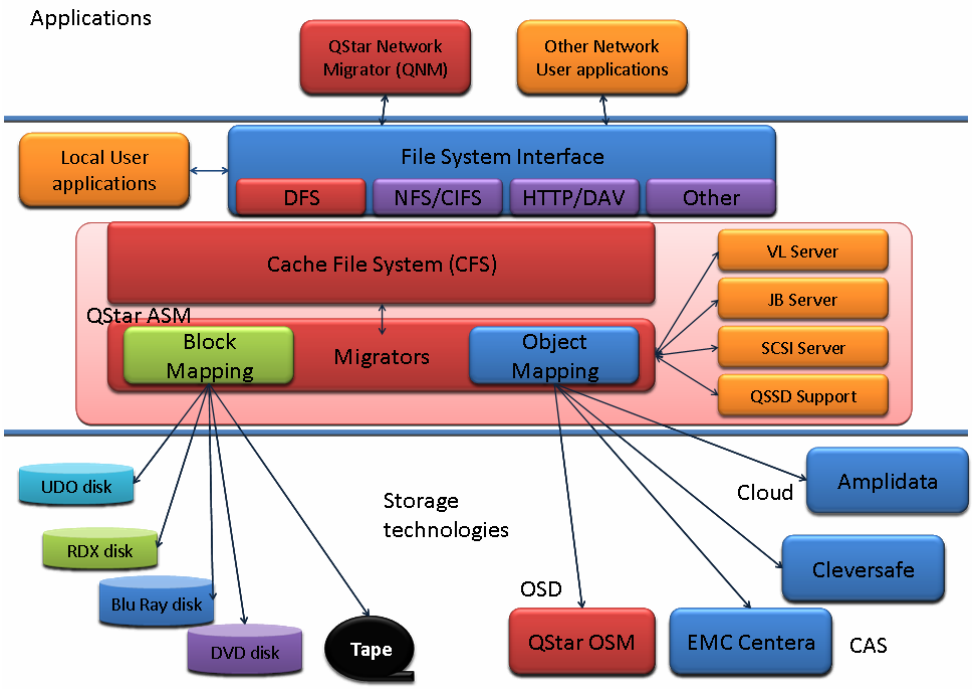

Рис. 1. Структура архивной системы QStar ASM

средства миграции данных на различные устройства хранения с помощью специальных модулей-миграторов.

3. **Migrators (миграторы)** предоставляют стандартизованный интерфейс для CFS и обеспечивают поддержку различных устройств и форматов хранения данных (форматов файлов). Миграторы взаимодействуют с вспомогательными серверами, управляющими общими ресурсами, прежде всего библиотеками сменных носителей.

4. **Серверы** обеспечивают управление общими ресурсами:

- SCSI-сервер выполняет конфигурационные услуги и координирует использование устройств (drives);
- JB-сервер управляет библиотекой носителей, включая выбор устройств, загрузку и выгрузку носителей и др.;
- VL-сервер выполняет информационные услуги и управляет библиотекой носителей на логическом уровне, например, если мигратору требуется новая лента для продолжения записи, он обращается к VL-серверу, который выбирает подходящий носитель и предоставляет его мигратору; логическое открытие этого носителя связано с обращением к JB- и SCSI-серверам для выбора подходящего устройства;
- QSSD-модуль не является непосредственно сервером, т.к. поддерживает ресурсы, управляемые независимо и не требующие поэтому дополнительной координации; к таким ресурсам, например, относится библиотека функций и методов доступа к сетевым устройствам хранения (облачные хранилища, другие файловые системы, и т.д.).

Такие устройства получили название QStar Structured Storage Devices (OSSD).

При дальнейшем рассмотрении наиболее важных компонентов архивной системы QStar ASM внимание уделяется, прежде всего, вопросам выбора принципов их практической реализации.

3. Реализация стандартного файлового интерфейса

Встраивание (интеграция) в ОС новых (дополнительных) системных модулей, в частности, файловой системы, является достаточно сложной задачей по следующим причинам:

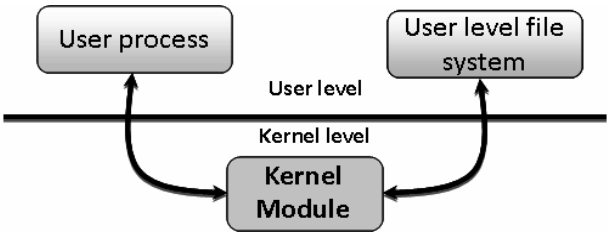

Рис. 2. Транспортировка файловых запросов в пользовательское пространство

1) программирование в ядре ОС является мало документированной и сложной процедурой, связанной с возможностью краха системы;

2) производители при развитии ОС могут менять внутренние интерфейсы без гарантии их совместимости при выпуске новой версии;

3) при необходимости поддержки нескольких ОС задача поддержки собственных разработок значительно усложняется;

4) производители ОС могут разорвать контракты на обслуживание или гарантийные обязательства, если в ядро инсталлированы программные модули третьих сторон.

В силу приведенных причин при разработке системы QStar ASM реализация программного обеспечения, в основном, выполнялась на пользовательском уровне (в пространстве пользователя). В то же время в ряде ОС (Solaris и AIX) некоторые компоненты архивной системы из-за отсутствия других возможностей и для повышения производительности были реализованы как модули ядра (Kernel Module). Такие модули являются агентами, транспортирующими пользовательские запросы через ядро в пространство пользователя (Рис. 2).

Очевидно, что модули ядра должны удовлетворять всем стандартам ОС, и их программная реализация должна учитывать особенности, перечисленные выше.

Другим способом интеграции является предоставление специализированных сетевых серверов NFS или CIFS. Структура FSI, реализованная на таком механизме, показана на Рис. 3.

В этом случае исключается необходимость программирования в ядре ОС, но имеется ряд ограничений:

- - необходимо поддерживать собственную реализацию протоколов NFS или CIFS;

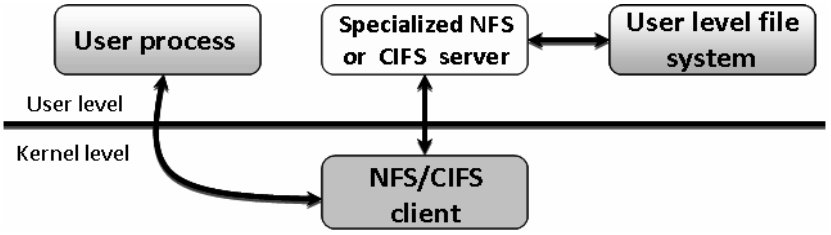

Рис. 3. Структура FSI с использованием специализированных серверов NFS или CIFS

– производительность системы значительно зависит от сетевых протоколов;

– не все ОС предоставляют возможность поддержки специализированных серверов NFS или CIFS из-за занятости портов TCP/IP.

В системе QStar ASM такой подход используется в ОС HP-UX и ранних реализациях Linux и Windows.

С появлением открытого исходного кода FUSE (File system in Userspace – файловая система в пространстве пользователя) [3] и CBFS (Callback File System) [4] поддержка FSI значительно упростилась (Рис. 4).

Этот механизм отличается от предыдущего (Рис. 3) реализацией протокола между ядром и модулем «User level file system» (пользовательский уровень файловой системы). Необходимо отметить, что с появлением механизма FUSE/CBFS разработчики получили возможность написания своих файловых систем без необходимости программирования в ядре ОС, чем достигается также независимость от ОС. В настоящее время только в Linux уже написан целый ряд таких файловых систем [5].

В системе QStar ASM такой подход реализован для операционных систем Linux, Windows и MAC OSX в модуле CFS. Этот модуль совместно с модулем FSI выполняет также функции монтирования файловой системы, т.е. подключения файловой системы к ОС. В UNIX-системах точка монтирования становится вершиной дерева файлов, в Windows после монтирования файловая система доступна через буквы (E:, F:, …) или как INC-путь.

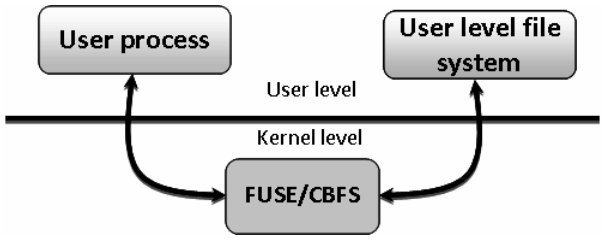

Рис. 4. Структура FSI с использованием FUSE/CBFS транспорта файловых запросов

4. Кэш-файловая система (CFS)

Основной задачей CFS является предоставление полноценного файлового интерфейса для экспорта файлов через сетевые протоколы. CFS является также интерфейсом к миграторам, и в этом отличие CFS от остальных файловых систем. Если в традиционных дисковых файловых системах требуется, чтобы все каталоги и файлы находились на диске и были доступны в любой момент времени, то CFS позволяет удалять как файлы (оставляя только описательную информацию о файле), так и целые деревья каталогов (оставляя информацию только о каталоге). Кроме того, CFS рассматривает файл не как единое целое, а как набор страниц, что позволяет хранить в дисковом кэше только часть данных файла, необходимую в данный момент.

К отличительным особенностям CFS прежде всего относятся:

– поддержка семантик файловых систем как в UNIX, так и в Windows (включая streams и ACL в Windows, MAC-forks, расширенные атрибуты в UNIX);

– возможность удаления из дискового кэша как данных файлов, так и каталогов;

– страничная структура с размером страниц от 4 Кбайт до 1 Мбайт, что имеет большое значение при значительных размерах файлов;

– поддержка различных способов обработки данных файлов, таких как шифрование и подсчет контрольных сумм для проверки целостности данных;

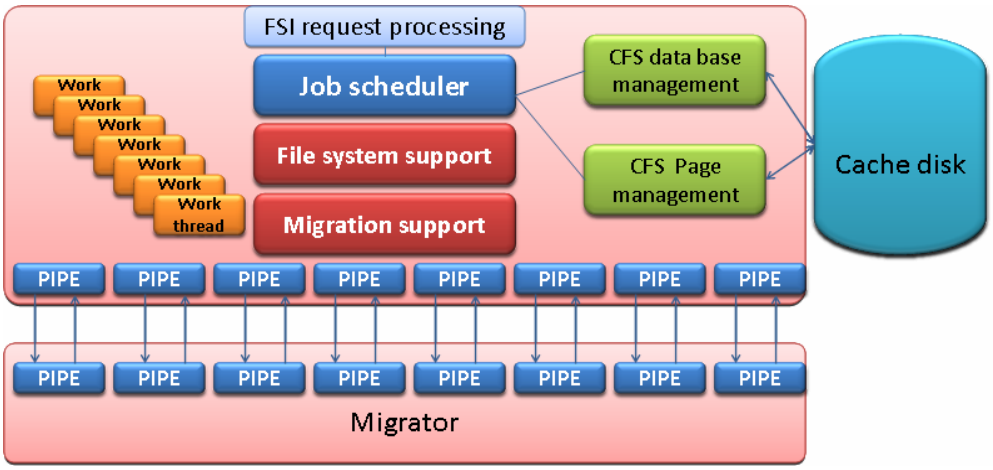

Рис. 5. Структура CFS

– поддержка интерфейса с миграторами, что является наиболее важной особенностью.

Следует отметить, что страничная структура позволяет решать вопросы модификации файлов. В отличие от других аналогичных систем [6, 7], CFS не требует передачи с носителя всего файла, а считывает только те страницы, которые подлежат модификации. Другая отличительная особенность CFS заключается в обработке каталогов. CFS позволяет удалять редко используемые каталоги и целые деревья файлов с целью освобождения ресурсов. Если к этим каталогам в дальнейшем происходит обращение, они будут восстанавливаться из базы данных мигратора. На Рис. 5 представлена структура CFS.

Файловые запросы, полученные модулем обработки запросов (FSI request processing) от FSI, распределяются диспетчером (Job scheduler). Каждый запрос получает свой контекст и рабочую нить (thread). Если запрос может быть выполнен сразу (когда все необходимые данные уже находятся в дисковом кэше), то ответ запрашивающему процессу возвращается немедленно. Если же данные были удалены с диска (мигрированы), то модуль поддержки миграции (Migration support) посылает запрос через один из каналов PIPE в мигратор и ожидает ответа.

Для поддержки структуры файловой системы CFS ведет базу данных в дисковом кэше. Данные файлов содержатся в виде набора страниц в дисковом кэше, и размер такого кэша задается пользователем. Именно база данных позволяет определить состояние каталогов, файлов и наличие их страниц данных. Таким образом, кэш может содержать файлы, часть из которых находятся только в кэше, а другая часть в кэше и на носителях или только на носителях. Динамика миграции данных определяется политикой управления CFS, где пользователю предоставляются различные возможности.

Если по какой-либо причине необходимо произвести миграцию данных из кэша (например, при полном кэше пользователь затребовал начать миграцию или сработала миграционная политика кэша), модуль «Migration support» начинает посылать запросы мигратору по одному или нескольким каналам (PIPE). Каждый запрос определяется как «событие» файловой системы (например, «создать файл», «записать страницу», «переименовать файл» и др.). Мигратор выполняет задание и возвращает ответ с кодом завершения. Следует отметить, что «событие» файловой системы не обязательно совпадает с операциями пользователя. Например, пользователь может писать файл порциями по 10 байт, в то время как CFS образует «событие», когда будет набрана полная страница данных.

Политика миграции – это совокупность правил, определяющих методы управления страницами файлов в кэше. Самое простое правило определяется верхним и нижним пределами заполнения кэша страницами, которые не имеют копии на внешнем носителе (так называемые «primary pages» – первичные страницы). При

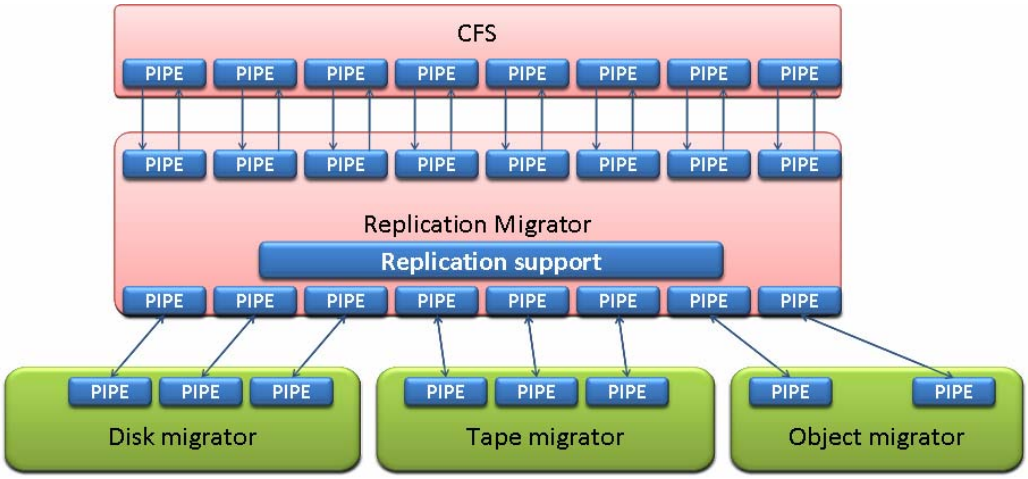

Рис. 6. Многослойные миграторы (мигратор репликаций)

достижении верхнего предела (как правило, 80% емкости кэша) CFS начинает миграцию первичных страниц, при достижении нижнего предела процесс останавливается. Более сложные правила позволяют периодически инициировать принудительную миграцию всех первичных страниц или зафиксировать страницы выбранных файлов в кэше. Для различных устройств хранения данных интерфейс с мигратором является единым. В Табл. 1 приведены возможные запросы к миграторам, включая события файловой системы и их описание.

Такая стандартизация позволяет создавать многослойные, или «виртуальные», миграторы, как это реализовано в специальном миграторе репликаций.

На Рис. 6 показаны три мигратора (число каналов уменьшено для наглядности), подчиненных мигратору репликаций (Replication migrator), который распараллеливает запросы от CFS с целью создания трех синхронных копий архивных файлов. Если какой-либо из миграторов рассинхронизировался, мигратор репликаций предоставляет средства его синхронизации. Необходимо подчеркнуть, что в данном случае репликация выполняется не на уровне блоков, а на уровне файлов, что позволяет хранить данные на совершенно несовместимых устройствах хранения.

Виртуальные миграторы позволяют создавать неограниченные многослойные комбинации миграторов. Например, кластерный мигратор позволяет представить две файловые системы (представленные двумя миграторами) как продолжение одна другой. Это используется в таких случаях, когда одна файловая система уже переполнена или по каким-либо причинам закрыта для записи. Добавление второй файловой системы позволяет продолжить запись и предоставляет пользователю интегрированный вид обеих файловых систем.

Табл. 1. Возможные запросы к миграторам

Событие файловой системы	Описание
create	Создать файл
link	Создать жесткую ссылку
mkdir	Создать каталог
remove	Удалить файл
rmdir	Удалить каталог
Setattr	Установить атрибуты (защита, владелец, длина, времена и т.д.)
symlink	Создать символическую ссылку
Page_fault	Прочитать страницу данных
Lookup/lookup_path	Найти файл / найти объект по пути
Readdir	Прочитать каталог
Readlink	Прочитать символическую ссылку
Get_file	Получить информацию об атрибутах файла
Setacl	Установить Access Control List (ACL)
Getacl	Получить Access Control List (ACL)
Access	Проверить права доступа
Dir_fault	Прочитать каталог
Write_Page	Записать страницу данных
Mksdir	Создать каталог для Streams
Rmsdir	Удалить каталог для Streams
Retention	Получить информацию о сроке хранения файла

5. Интегрированный набор томов

Файловые системы создаются и поддерживаются на конкретном пространстве памяти (диски, CD/DVD, лента). Если носитель сам по себе не предполагает никакой структуры данных, то носитель с инициализированной файловой системой определяется как «том», и несколько томов группируются в наборы.

В архивных системах, где большие объемы хранимых данных, а сами носители могут быть съемными, файловая система должна поддерживать многотомную структуру. Такая структура в QStar ASM получила название «Integral Volume Set» (IVS – интегрированный набор томов). IVS состоит из следующего набора физических компонент (Рис. 7):

- идентификатор IVS (Set ID) – это некоторое число, позволяющее уникально идентифицировать все компоненты IVS;
- пространство на диске, где хранятся база данных, дисковый кэш и миграторы, для чего в QStar ASM не требуются отдельные диски или разделы; практически кэш может состоять из одного-четырех каталогов, назначенных пользователем;
- мигратор, тип которого должен быть определен пользователем на основе типа носителя или по другим соображениям (например, для обеспечения совместимости со стандартами);
- набор носителей, назначенный для данного IVS; каждый носитель проходит стадию инициализации, во время которой идентификатор IVS записывается на носитель; форматированный носитель становится томом и это позволяет отслеживать такой набор томов на разных стадиях жизненного цикла системы.

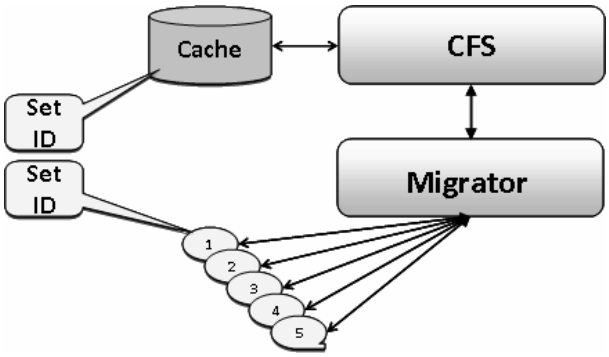

Рис. 7. Компоненты интегрированного набора томов

Необходимо отметить, что IVS оперирует с носителями, а не с устройствами или библиотеками носителей. Назначение устройств для носителей является функцией дополнительных серверов, т.к. в одной системе могут быть созданы несколько IVS и они будут конкурировать за общие ресурсы (устройства, робот, и т.д.). Эти вопросы рассматриваются в следующем разделе.

6. Управление устройствами хранения данных

Устройства хранения данных являются общими ресурсами системы. Использование библиотеки носителей предполагает, что только небольшая часть носителей может быть активна в определенный момент времени, а выгрузка/загрузка носителей занимает некоторое время (от десятков секунд до нескольких минут). Если IVS состоит из тысяч носителей и десятков устройств и система поддерживает несколько IVS, то становится очевидным, что появляется задача оптимизации (или, по крайней мере, координации) разделения общих ресурсов.

QStar ASM поддерживает три основных типа устройств хранения данных:

1) библиотеки носителей (БН);
2) виртуальные библиотеки носителей (ВБН);
3) структурированные устройства хранения (QSSD).

Библиотеки носителей (Storage libraries) в упрощенном виде представляют собой стойку, где размещены полки для хранения носителей, таких как кассеты с магнитной лентой [8], CD/DVD/Blu Ray-диски или UDO – кассетные носители. В стойке имеются, по крайней мере, одно устройство и специальные полки для импорта/экспорта носителей из библиотеки. Сердцем системы является роботизированная «рука» (или просто робот, возможно, несколько роботов), которая может перемещать носители между полками и устройствами, между полками и специальными полками для экспорта/импорта. На рынке имеется множество вариантов библиотек – от простых с одним устройством и 24 полками до гигантов с 100000 полками [9]. Очевидно, что управление библиотеками целиком зависит от программного обеспечения. Робот сам по себе ничего не делает, если ему не дана команда по какому-либо интерфейсу

(SCSI, iSCSI, SAS или Fibre Channel (FC)). Особое место занимают устройства вне БН, так называемые автономные устройства (stand-alone drives). В QStar ASM они поддерживаются с использованием виртуального робота, которым является человек-оператор. Если нужно заменить носитель, оператор получает сообщение с информацией о необходимом действии.

Виртуальные библиотеки носителей, по сути, эмулируют функциональные возможности физических библиотек. Так как взаимодействие с библиотеками осуществляется через некоторый интерфейс, это предоставляет возможность эмуляции команд с помощью программного обеспечения (ПО). Если такая эмуляция выполнена достаточно точно, то пользовательские программы могут и не знать, кто отвечает на их запросы, поэтому с точки зрения управления ВБН не отличается от БН.

Структурированные устройства хранения. БН и носители до процедуры инициализации не содержат никакой структуры данных. QSSD выделены в отдельный класс, потому что такие устройства содержат некоторую структуру данных. Например, облачные хранилища позволяют работать на уровне объектов произвольной длины [10, 11]. Другие устройства [12] предоставляют какие-то файловые структуры. Как правило, взаимодействие с такими устройствами выполняется с помощью протоколов TCP/IP и HTTP. QStar ASM предоставляет библиотеку функций для работы с QSSD-устройствами, т.к. здесь нет необходимости разделения ресурсов между миграторами и, следовательно, нет необходимости в специальном сервере.

В QStar ASM имеется три сервера, выполненные как отдельные процессы или нити, которые выполняют функции координации и управления общими ресурсами:

SCSI-сервер – осуществляет управление конфигурацией SCSI-устройств (Рис. 1); не участвует в передаче данных, но отвечает за координацию доступа к SCSI-устройствам. Например, операция открытия устройства направляется в SCSI-сервер, где выполняется проверка занятости устройства.

JB-сервер – выполняет управление библиотеками носителей с помощью следующих операций:

- начальное распознавание состояния БН; операция выполняется, как правило, во время старта JB-сервера и позволяет создать таблицу исходного состояния БН, которая содержит число полок хранения, наличие в них носителей, число устройств и наличие в них носителей, число роботов, число полок для импорта/экспорта, наличие магазинов носителей и печатающих устройств;
- отработка команд миграторов на различные перемещения носителей;
- поддержка очередей запросов;
- выгрузка неактивных носителей;
- ведение статистики (число загрузок, наличие ошибок, и т.д.).

Следует обратить внимание, что JB-сервер не связан с содержимым носителей.

VL-сервер – выполняет управление библиотекой носителей на логическом уровне с помощью нескольких основных операций:

- поддержка таблиц состояния носителей с некоторой логической структурой. Первоначально все носители в БН помечаются как «неизвестные». После выполнения операции обновления (что включает загрузку носителя в устройство, определение формата данных и возврат носителя на полку) VL-сервер помечает носитель как содержащий некоторый формат данных и принадлежащий какому-либо набору носителей;
- ведение набора пустых носителей;
- выбор носителя для определенного клиента (по сути, мигратора);
- обновление таблицы состояния БН при операциях импорта и экспорта носителей.

Таким образом, SCSI-, JB- и VL-серверы предоставляют программный (а через него – графический и командный) интерфейс для управления общими ресурсами хранения данных. Клиентами этого интерфейса являются миграторы, рассматриваемые ниже.

7. Назначение и функции миграторов данных

Миграторы являются посредниками между CFS и устройствами хранения данных и должны удовлетворять следующим требованиям:

- поддержка структуры данных на носителе, включающая поддержку накопителей разно-

го типа и с разными форматами данных; в QStar ASM поддерживаются как стандартные форматы, такие как UDF и LTFS [13, 14], так и собственные форматы TDO и SDF, в случае QSSD-устройств миграторы обеспечивают необходимую поддержку объектных форматов и REST-протоколов;

- поддержка интерфейса с CFS с помощью запросов, представленных в Табл. 1;
- обеспечение интерфейса с VL-сервером, который предназначен для выполнения следующих функций:
  - восстановление базы данных (используемые форматы хранения данных предполагают, что база данных мигратора сохранена на носителе, и все события файловой системы могут быть восстановлены, поэтому файловая система может быть восстановлена, даже если компьютер полностью уничтожен, включая дисковый кэш),
  - получение новых носителей,
  - запросы на возврат носителей, если они были экспортированы из БН,
  - начальное форматирование носителя,
  - распознавание формата хранения данных на носителе;
- поддержка копирования и сравнения носителей, позволяющая изготавливать копии носителей для отправки их в удаленные хранилища, чтобы предотвратить потери данных при техногенных катастрофах;
- поддержка интерфейса с системой управления миграцией;
- обработка и регистрация различных ошибок; данная функция особо важна, потому что именно мигратор первым может обнаружить ошибки оборудования и оператора.

На Рис. 8 приведена структура функциональных модулей типичного мигратора.

Рассмотрим выполнение операции записи файла, начиная с операции его открытия пользователем. Пользовательский запрос «open» через FSI доставляется в CFS, и, если параметры защиты позволяют, создается файл в дисковом кэше. Пользователь записывает какое-то количество данных, и они сохраняются в кэше. Согласно политике кэша в какой-то момент времени инициализируется миграция данных, и CFS начинает посылать файловые события в

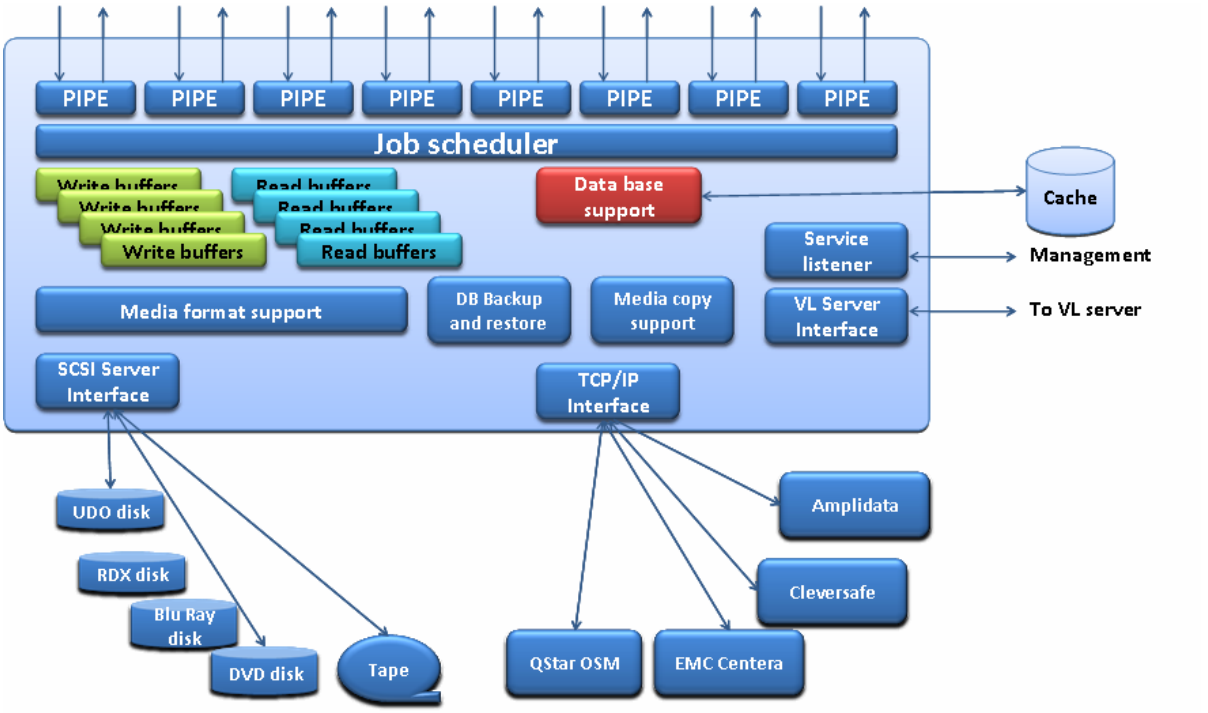

Рис. 8. Типичная структура мигратора данных

мигратор. Такие события попадают в буферы записи (write buffers) и далее через модуль поддержки формата данных на носителе (UDF, LTFS, TDO, HTTP и др.) и протоколы SCSI или TCP/IP на сам носитель. При этом осуществляется обновление базы данных и выполняется периодическая синхронизация с CFS. Такая синхронизация позволяет CFS освободить страницы кэш только в том случае, когда буферы записи синхронизированы с носителем. Операция чтения в миграторе инициируется, когда запрошенная страница данных отсутствует в кэше. В этом случае CFS посылает запрос в мигратор, который с помощью базы данных определяет, на каком носителе находится нужная страница. Если носитель не находится на устройстве, то SCSI-подсистема с помощью JB-сервера обеспечивает его загрузку. Если носитель не находится в БН, то по запросу к VL-серверу будет послано сообщение оператору, чтобы запрошенный носитель был импортирован. При чтении данных могут произойти значительные задержки из-за работы БН или из-за необходимости импортировать носитель. Тайм-ауты протоколов NFS и CIFS в какой-то мере могут быть увеличены, однако такие настройки находятся вне влияния QStar ASM.

8. Управление архивной системой QStar ASM

Как правило, стандартные файловые системы не нуждаются в специальных средствах управления. После начальной инициализации современные файловые системы работают без вмешательства оператора за исключением необходимости периодической проверки целостности или дефрагментации. При обслуживании архивных хранилищ возникает ряд задач, для решения которых требуется вмешательство оператора:

- обслуживание библиотек носителей. Оператор должен периодически импортировать и экспортировать носители из БН;
- БН как механические устройства могут выходить из строя и требуют обслуживания со стороны оператора;
- копирование архивных данных требует вмешательство оператора;
- система должна подвергаться периодическим аудитам как с точки зрения старения носителей, так и с точки зрения целостности данных;
- необходимо постоянное наблюдение за состоянием системы.

Для выполнения этих и других задач QStar ASM предоставляет ряд средств, состоящих из команд пользователя (CLI) и графического интерфейса (GUI). Команды пользователя позволяют выполнять операции управления серверами, инициировать носители и файловые системы, монтировать файловые системы и практически управлять всеми ресурсами QStar ASM. Как правило, опытные администраторы предпочитают CLI-интерфейс, потому что его команды могут быть использованы в составе командных файлов. Для других пользователей более удобным является GUI-интерфейс, где состояние системы представляется более наглядно и где управляющие воздействия могут быть инициированы посредством различных графических примитивов, таких как кнопки, меню, и т.д. В QStar ASM интефейс GUI реализован в двух видах: в Windows-среде с использованием стандартов Windows и как WEB-интерфейс с использованием стандартных браузеров (Internet Explorer, Safari, Chrome, FireFox). GUI поддерживает семь наиболее затребованных языков, таких как английский, немецкий, японский, китайский, итальянский, русский и голландский. Другие языки могут быть достаточно легко добавлены, т.к. все тексты вынесены в специальные файлы.

Заключение

Неудержимый рост объемов архивной информации предъявляет новые требования к способам и методам ее хранения [1]. В статье рассматриваются структура, назначение и функции программного обеспечения системы QStar ASM, разработанной с учетом приведенных требований к современным архивным системам. Особое внимание уделяется вопросам выбора принципов практической реализации наиболее важных компонентов архивной системы QStar ASM.

Литература

1. Егоров Г.А., Шяудкулис В.И., Финотти М.. Проблемы построения современных архивных хранилищ данных. – «Информационные технологии», 2012, № 12.
2. [QStar] http:// www. qstar.com/
3. [FUSE] http://fuse.sourceforge.net/
4. [CBFS] http://www.eldos.com/cbfs/
5. [FUSE-FS] http://sourceforge.net/apps/mediawiki/fuse/index.php?title=FileSystems
6. [QFS] http://wiki.lustre.org/images/6/6d/Wednesday_SAM-QFS.HPC.PUB.pdf
7. [SN] http://www.dscon.ru/quantumdocs/quantum_stornext_4.pdf
8. [ORACLE] http://www.oracle.com/us/products/servers-storage/storage/tape-storage/sl8500-modular-library-system/overview/index.html
9. [SPECTRA]http://www.spectralogic.com/index.cfm?fuseaction=products.displayContent&catID=1990&src=fly
10. [AMAZON] http://aws.amazon.com/s3/
11. [CLEVERSAFE] http://www.cleversafe.com/cleversafe-products/software-hardware
12. [HCP] http://www.hds.com/assets/pdf/hitachi-datasheet-content-platform.pdf
13. [UDF] http://www.osta.org/specs/
14. [LTFS] http://www.trustlto.com/LTFS_Format_To%20Print.pdf

Егоров Геннадий Алексеевич. Заместитель генерального директора ОАО «Институт электронных управляющих машин им. И.С. Брука». Доктор технических наук. Автор более 80 работ. Область научных интересов; управляющие вычислительные комплексы и системное программное обеспечение. E-mail: egorov_g@ineum.ru

Шяудкулис Владас Ионо. Ведущий архитектор разработчик QStar Technologies, Inc. Кандидат технических наук. Автор 30 работ. Область научных интересов: информационные технологии. E-mail: v.chiaoudkoulis@gstar.com

Финотти Массимилиани. Технический директор QStar Technologies, Inc. PhD. Автор 20 работ. Область научных интересов: архитектура систем хранения данных. E-mail: m.finotti@gstar.com

Беляков Михаил Иосифович. Ведущий разработчик-программист QStar Technologies, Inc. Кандидат технических наук. Автор более 50 работ. Область научных интересов: системное программирование. E-mail: m.beliakov@gstar.com