Перспективы кластерных центров данных

Данная публикация является видением компании VERITAS на развитие своего продукта VERITAS Cluster Server (или Next Generation Clustering) в рамках концепции VERITAS Utility Computuing. Статья продолжает серию материалов (SN №№ 2/20, 1/19, 4/18), связанных с этой концепцией и посвященных наиболее острым проблемам управления IT-инфраструктурой в высокодоступных открытых системах.

Введение

Метод кластеризации является одним из основных при достижении необходимых требований доступности и надежности любых IT-систем. Корни вопроса уходят в середину 80-х, и в историческом разрезе все типы кластеров – с самых первых до перспективных – строятся на основе 5 архитектур. В "классическом" представлении выделяют 3 типа наиболее распространенной кластерной организации: асимметричная – с одним активным сервером); симметричная – с 2 активными серверами; N-to-1 – с одним запасным сервером на N (рис. 1), каждая из которых имеет свои преимущества и недостатки (табл. 1). Асимметричная имеет высокие стоимость и надежность; симметричная – высокую сложность, более низкую производительность и меньшую надежность из-за проблем с совместимостью приложений разных версий/платформ; N-to-1 – более низкую стоимость при меньшей доступности из-за необходимости проведения процедур по рестарту после отказа.

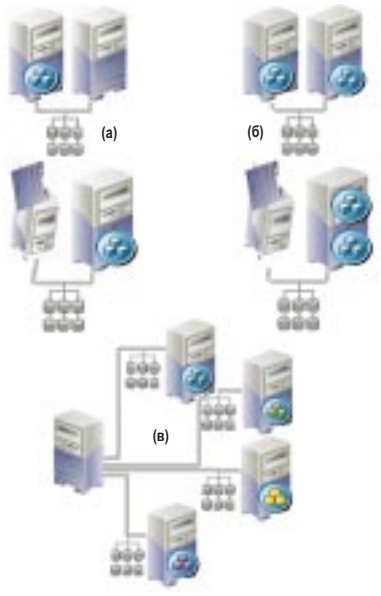

Рис. 1. Три типа архитектуры "классических" кластеров: (а) асимметричный, (б) симметеричный и (в) N-to-1.

Табл. 1. Сравнение различных типов "классических" кластеров

	асимметричная (1 акт.сервер)	симметричная (2 акт.сервера)	N-to-1 (1 запасной сервер на N)
стоимость	$$$$	$$$	$$
производительность	*****	**	*****
сложность	*	*****	****
доступность	*****	***	**

Более совершенный N+1 кластер имеет общее сетевое хранилище данных (рис. 2) и в отличие от N-to-1 кластера он не требует рестарта системы после отказа вследствие наличия равного доступа серверов к сетевым ресурсам хранения. После восстановления отказавшего сервера он становится резервом, т.н. архитектура с "плавающим" запасным сервером.

Дальнейшее развитие кластерной архитектуры идет в направлении повышения ее эффективности использования и доступности (N-to-N архитектура, рис. 3). Особые признаки такой архитектуры – резервирование всем пулом, а не отдельным выделенным резервным сервером; выполнение, как правило, на каждом сервере множества приложений на множестве гетерогенных платформ; наличие процедур по оптимизации балансировки нагрузки (в соответствии с приоритетностью приложений и их совместимостью) и бесшовному перемещению приложений между серверами в случае изменения нагрузки или отказа.

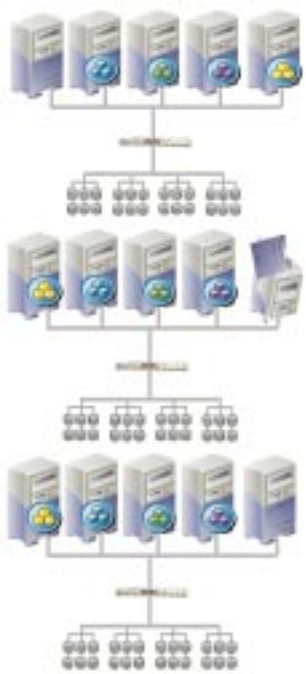

Рис. 2. Современный тип N+1 кластерной архитектуры – "плавающий" запасной сервер.

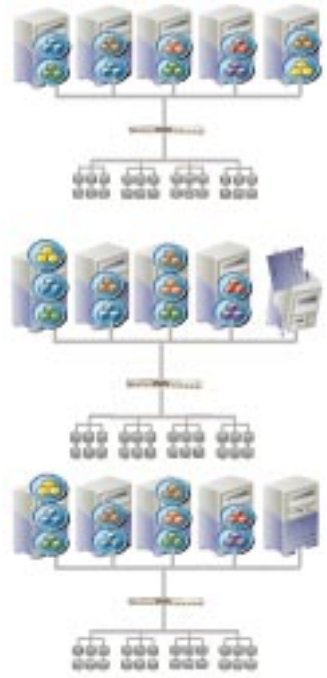

Рис. 3. Перспективный тип N-to-N кластерной архитектуры.

Компания VERITAS – одна из немногих последовательно развивающих кластерную архитектуру в своих решениях, а слоган "Business Without Interruption" длительное время оставался основным корпоративным лозунгом. VERITAS Cluster Server (VCS) – основной продукт, продаваемый компанией для поддержки кластерных решений сегодня (в представленной классификации – N+1 кластер). Начиная с 2005 г. будут поставляться отдельные компоненты, а во второй половине 2005 г. – продукт нового поколения VCS (см. рис. 3) – VERITAS Next Generation Clustering (NGC), развиваемый в рамках концепции VERITAS Utility Computing.

Данная публикация дает общее представление о базовой функциональности NGC и основные этапы его развития в свете требований, предъявляемых к современным информационным центрам данных высокой доступности.

Основные тенденции "текущего момента"

Снижение затрат на управление за счет консолидации серверного пула

Как показали исследования, большинство западных компаний уже существенно изменили структуру серверов IT-центров или планируют ее значительно изменить в ближайшей перспективе. Основная задача многих компаний – снизить затраты на управление. И чаще всего это решается за счет консолидации серверного пула: 1) за счет перехода на более мощные серверы; 2) за счет более плотной его "упаковки" – переход на блэйд-серверы. В обоих случаях цель – сделать больше с меньшими количеством оборудования, численностью персонала и стоимостью. В то же самое время, доступность критических приложений должна быть поддержана на прежнем уровне или увеличена.

Но это, в свою очередь, приводит к ряду других проблем. С переходом на более мощные серверы может снизиться уровень надежности/доступности системы в случае множественных отказов или сбоев, поэтому необходим и более высокий уровень управления каждым сервером для функционирования бóльшего количества приложений на одном сервере при условии поддержания требуемого уровня доступности. С переходом на блэйд-серверы возникает проблема управления большим количеством серверов, требующих гораздо бóльшей автоматизации их распределения/выделения (provisioning), а также и оптимизации их использования.

Приоритизация приложений

В любой среде все приложения имеют разную степень воздействия на бизнес, а отдельные, в случае даже не очень длительной остановки, могут вообще привести к прекращению бизнеса. В дополнение к этому, в высокодоступной кластерной среде, так же, как и в случае поддержания в среде автоматизированных процедур восстановления после катастроф, некоторые приложения должны иметь приоритет над ограниченными системными ресурсами, остающимися работоспособными после любого выхода из строя системы.

Автоматизация развертывания серверов

Добавление дополнительного количества серверов к сложной среде – трудоемкая и дорогостоящая процедура. Многие ищут способ ее упрощения, и во многих случаях таким способом является автоматизация развертывания серверов. Уже существует достаточное множество автономных инструментальных средств (например, Opforce) для автоматизации инициализации сервера и включения его в инфраструктуру. Не хватает в этом инструментарии: 1) конкретных механизмов управления в случаях, когда что-то уже распределено; 2) более жесткой их интеграции в общую систему управления. Если бы существовало бесконечное количество серверов в "свободном пуле", тогда не существовало бы проблемы. Однако никто не заинтересован в приобретении большого количества оборудования без его использования, и каждое новое распределение сервера приложению связано с выделением запасного сервера или перераспределением серверов между уже работающими приложениями. В обоих вариантах не должен снижаться уровень доступности других приложений, а перераспределение серверов должно осуществляться на основании вычисления каждый раз значения целевой функции, поддерживающей оптимальный уровень функционирования бизнеса.

Мониторинг производительности

Производительность все в большей степени используют как меру доступности и включают в соглашения сервисных уровней (Service Level Agreements – SLA). Традиционный упрощенный подход управления системой на основе интегрированных показателей доступности уже не может удовлетворять, т.к. простое добавление ресурсов в ряде случаев не устраняет проблему, приводит к большим издержкам, не обеспечивает более детальную детерминацию проблемы (которая могла бы быть ее решением без какого-либо реконфигурирования системы). Такой подход совершенно не работает и тогда, когда потребитель хочет видеть решение проблемы в перераспределении недоиспользуемых имеющихся средств в соответствии с задаваемыми приоритетами приложений.

Next Generation Cluster – кластеризация в концепции Utility Computing

Существенным сдвигом парадигмы в управлении приложениями является кластеризация следующего поколения от VERITAS – Next Generation Clustering (NGC). NGC позволяет управлять большими количеством приложений, основываясь на их приоритетах и требованиях к ресурсам на большом множестве типов серверов и гетерогенных операционных систем.

NGC снимает ограничение на 32 гомогенных узла, имеющееся в текущей реализации VCS (VERITAS Cluster Server), и может масштабироваться до 256 гетерогенных узлов с сотнями работающих приложений. С единым представлением приложений и их "связыванием" с серверными платформами NGC обеспечивает законченное управление многоуровневыми приложениями, распределенными на гетерогенных платформах.

Рассмотрим некоторые новые особенности NGC: клиент/серверную архитектуру, а также расширенное управление рабочей нагрузкой.

Клиент/серверная архитектура

NGC значительно отличается от существующего VCS. В первую очередь это переход от RSM-архитектуры (Replicated State Machine), где все узлы выполняют идентичную копию кластерного узла, к клиент/серверной архитектуре, основными компонентами которой являются: мастер политик – PM (Policy Master), "тонкие клиенты" и база данных конфигурации/состояния (Config/Status Database – CSD).

Мастер политик

Это логика, или "мозг" полного NGC кластера. В отличие от традиционного VCS, где на всех узлах имеется одинаковая процедура, запускаемая при возникновении сбоев, в NGC – единственный "бегающий" мастер политик, ответственный за все политики на всех узлах и 1 или более клиентов. Хотя может показаться, что это вносит дополнительный риск в общую схему высокой доступности, встроенная в мастер политик достаточная избыточность обеспечит необходимую отказоустойчивость.

При использовании одного PM появляется возможность более оптимально использовать процессорную мощность для решения конкретной задачи, продолжая принимать корректные решения в любой ситуации. Для отказоустойчивости планируется ввести 1 или несколько дополнительных "резервных мастеров политик", которые при сбое основного возьмут на себя его обязанности. Соответственно сам PM и его резерв будут работать в кластере, построенном на основе жестких соединений GAB/LLT, для обмена

информацией между узлами. Для предотвращения ситуации “сплит-брейн” в кластере мастеров политик будет использоваться технология SCSI-III I/O fencin.

База данных конфигурации/статуса

CSD обеспечивает репозиторий как всей первичной информации о конфигурациях всего кластера, так и вторичного статуса всех ресурсов на всех узлах. Это обеспечивает множество расширений для существующей архитектуры. Основное ее назначение – фиксация состояния приложений клиентов на сервере при их остановке для всех последующих их рестартов. CSD также обеспечивает долгосрочную хронологию всей конфигурации, команд оператора и состояния приложений для использования в ComandCentral Availability.

Тонкие клиенты

NGC избавится от RSM-концепции и будет двигаться к архитектуре, управляемой мастером политик и “тонкими клиентами”. “Тонкий клиент” – кластерный узел, связанный через IP-сеть с мастером политик. Тонкий клиент состоит из существующих VCS агентов с дополняющим ПО и клиентским HAD, или “HAD proxy”. Тонкий клиент будет работать так же, как VCS узел делает это уже сегодня. Отличие – в том, что тонкий клиент выполняет только приложения и агенты, а все политики фактически выполняются на PM. Концепция тонкого клиента обеспечивает обратную совместимость буквально с тысячами существующих агентов, т.к. HAD-клиент “видится” так же, как агент фрейма, откомпилированного в каждый агент (как и у существующего VCS).

Чтобы защитить от повреждения ядро, клиент/серверная модель будет использовать SCSI-III постоянное резервирование, чтобы блокировать доступ к хранению для неподключенных узлов. В этом случае, если теряется связь с узлом, узел, занимая любые разделенные ресурсы данных, блокирует доступ к подозреваемому мертвому узлу, используя SCSI-III PR. И даже если узел был в реальности “не мертв”, он не способен сделать что-либо “плохое”. Другие формы защиты данных на стороне клиента будут также реализовываться по мере того, как это будет необходимо.

Первая реализация NGC будет поддерживать до 256 клиентских узлов на Solaris и Linux. В последующих реализациях будет добавлена поддержка Windows и всех других Unix-вариантов, а также увеличено число поддерживаемых клиентов до 512 узлов и более.

Расширенное управление рабочей нагрузкой

Еще одним важным новым свойством, включенным в NGC, является политикоориентированный метод для полного управления большим количеством приложений, названный Advanced Workload Management (AWM) – расширенное управление рабочей нагрузкой. AWM предназначен для решения существующих проблем при большой консолидации серверов или блэйд-серверов. AWM представляет большую коллекцию новых возможностей, обеспечивающих лучшую автоматизацию больших сред и намного бóльшую их предсказуемость поведения в случае множественных отказов.

Следующие особенности определяют основную функциональность AWM:

- задание приоритетов сервисной группы (Service group priorities – SGP);
- установление совместимости сервисной группы (Service group Compatibility);
- определение вместимости сервера (Server Capacity) и загрузки группой (Group Load);
- интеграция с менеджером ресурсов (Resource Manager Integration);
- интеграция с технологиями виртуальных машин (Virtual Machine Technology Integration);
- динамическое управление производительностью.

Приоритеты сервисной группы

SGP обеспечивают информацией для управления перераспределением приложений в случае отказа при отсутствии адекватной мощности в кластере. Когда происходят множественные отказы резервной мощности в кластере, может быть не достаточно, чтобы обработать все сервисные группы и поддержать указанную производительность для критических приложений соответственно. Установление SGP позволяет кластеру принять меры для защиты работоспособности приложений с наиболее высоким приоритетом, перемещая или останавливая приложения с меньшими приоритетами.

AWM обеспечивает 4 приоритетных уровня для сервисных групп.

- *Приоритет 1(Pri 1) – самый критический.* Приложения с данным приоритетом обрабатываются как “золотые”. Кластер не будет останавливать или перемещать эти группы до тех пор, пока в самой группе не будут идентифицированы ошибки или не вмешается оператор. Это приложения, где допускается абсолютно минимальное время простоя. Pri 1 группа не может быть выведена в “off-line” чтобы освободить место для другой Pri 1 группы.
- *Приоритет 2 (Pri 2) – бизнес-критический.* Данный приоритет сервисных групп менее важен, чем Pri 1. Различие – в том, что кластер может выполнить переключение (перевести в “off-line” или на менее производительные серверы) приложений с данным приоритетом, чтобы поддержать кластерные характеристики загрузки приложений с более высоким приоритетом (Pri 1) . Группа с Pri 2 не может заставить другую группу с Pri 2 переключиться.
- *Приоритет 3 (Pri 3) – критическая задача.* Приложения с Pri 3 сервисных групп могут перемещаться по желанию, чтобы поддержать загрузку кластера, и могут быть остановлены, если никакой другой возможный метод не существует и нет никакой другой возможности для поддержания требуемой загрузки в кластере. Приложения с Pri 3 не могут быть прерваны приложением с таким же приоритетом.
- *Приоритет 4 (Pri 4) – некритическая задача.* Приложения с данным приоритетом – несущественные приложения, например, тестовые приложения или различные внутренние служебные программы. Эти сервисные группы могут быть остановлены по желанию, чтобы поддержать кластерную нагрузку.

Совместимость сервисной группы

AWM осуществляет совместимость сервисной группы как дополнение/расширение приоритетов сервисной группы. Совместимость сервисной группы позволяет оператору точно определять, что другие сервисные группы были проверены для совместного использования с рассматриваемой группой. Например, большинство пользователей никогда не запустило бы две различные системы управления базой данных на одном и том же сервере. Пользователь может определить, что сервисная группа, содержащая базу данных Oracle, допускает работу с другими группами Oracle-приложений, но никогда не должна работать на одном сервере с DB2- или Sybase-приложениями. AWM будет использовать информацию совместимости сервисной группы при создании отказоустойчивых решений. Когда определяется, где будет выполняться приложение, механизм политик выберет сервер с адекватной способностью и с отсутствием возможных проблем по совместимости. Если необходимо, в зависимости от приоритета, другие группы могут или будут перенесены для создания такого сервера.

Загрузка группой и вместимость сервера

Загрузка и вместимость предназначены для того, чтобы позволить администратору устанавливать фиксированную вместимость для всех серверов в группе. Каждому приложению назначается показатель загрузки, основанный на том, сколько из стандартного модуля (сервера) вместимости приложение, как ожидается, будет использовать.

Например, в кластере с 4 узлами, состоящем из 2 серверов с 16 процессорами и 2 серверов с 8 процессорами, администратор мог бы установить вместимость на системах с 16 центральными процессорами показатель 1600 и на системах с 8 центральными процессорами – 800 (предполагается, что стандартный объем оперативной памяти на один CPU – 100 Мбайт). Вместимость, по существу, стандартизированная мера мощности сервера, основанная на знании числа CPU, его производительности и доступной оперативной памяти.

Рейтинг загрузки сервисной группы основан на том, насколько группа может (ожидается) загрузить стандартный сервер. Если пользователь предполагает, что сервисная группа загрузит упомянутый сервер с 8 CPU свыше 50%, они позволят установить сервисное значение загрузки группы – 400 единиц. Тогда можно предположить, что таже самая группа загрузит сервер с показателем 1600 на 25%. В таком случае, AWM будет тогда гарантировать, что никогда не будет запущено более двух таких загрузок на 800-

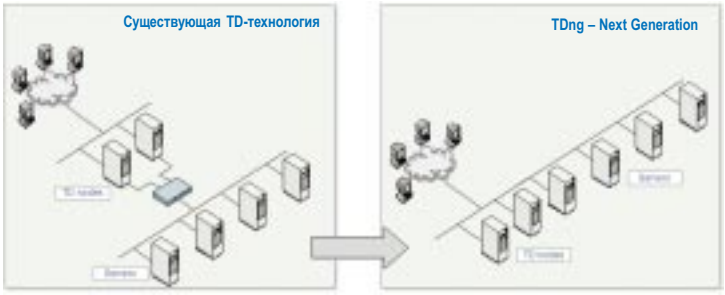

Рис. 4. "Встраивание" в кластер существующей TD-технологии и "TDng – Next Generation" в интеграции с NGC.

сервере или четырех таких загрузок на 1600-сервере.

Чтобы точнее определить вместимость каждого сервера, а также потенциальные требования по загрузке каждого приложения, VERITAS будет поставлять соответствующие инструментальные средства и мастера.

Интеграция с менеджером ресурсов сервера

Для того чтобы модель, представляемая "загрузкой группы и вместимостью сервера", реально работала, она должна быть интегрирована с ПО управления загрузкой сервера (типа Aurema, Solaris Resource manager и др.) на уровне операционной системы, которое будет производить квантование ресурсов сервера на основании значений загрузки группы и вместимости сервера. Использование такого подхода значительно уменьшает административную сложность и соответственно накладные затраты (в ресурсном измерении) в среде консолидации серверов.

Интеграция AWM с Next Generation Traffic Director

Traffic Director (TD) – компонента, оптимизирующая трафик между клиентами и кластером. В существующей реализации TD имеет ряд особенностей, снижающих его эффективность (рис. 4), в частности:

- TD-узлы и серверы должны быть в различных подсетях, что требует дополнительных усилий и средств по коммутации;
- клиентские и пакеты сервера используют TD-узлы, что снижает общую производительность при высокой нагрузке;
- трудности по интеграции TD в VCS.

Интеграция TD в NGC 2.0 (или (TDng – Next Generation Traffic Director) будет иметь ряд преимуществ:

- TD-узлы и узлы серверов могут быть в одной сети (см. рис. 4). TDng может быть установлен в существующей топологии по принципу "plug and play";
- значительно более высокая производительность;
- сильная интеграция TD с NGC.

Интеграция с технологиями виртуальных машин

Архитектура NGC идеально подходит для использования в среде, поддерживающей различные технологии виртуальных машин, охватывающих диапазон решений от различных продавцов, включая Sun Solaris 10 "Containers", VMware Virtual Machines и Microsoft Virtual Server. В каждой технологии множественные виртуальные машины выполняются на вершине "главной ОС" (host OS). В кластере NCG каждая host OS будет работать, как NGC тонкий клиент. Индивидуальные виртуальные машины создаются как сервисные группы, которые могут работать на определенных узлах. Когда сервисная группа переведена в "on-line", запускается виртуальная машина, в пределах которой выполняются различные NGC агенты для управления и мониторинга процессов в ней.

Представление виртуальных машин как сервисных групп в NGC позволяет полностью использовать функциональные возможности AWM для управления виртуальных машин и соответствующих приложений, которые на них выполняются, а также поддерживать загрузку сервера в соответствии с его вместимостью и установленными приоритетами.

Как работает AWM?

AWM работает по двум разным алгоритмам. Первый ориентирован на первичную операцию запуска сервера и операцию обработки отказа/сбоя. Второй – на динамическое управление производительностью NGC.

Для первого алгоритма в течение любой операции сервисной группы, типа запуска или обработки отказа, AWM находит наилучший подходящий сервер, основываясь на AWM политиках. Если *существует сервер с адекватной вместимостью и без нарушения совместимости,* сервисная группа будет запущена на нем.

Если *существует множество подходящих серверов*, AWM выберет тот, который будет по вместимости наиболее близко подходить прикладной загрузке, чтобы предотвратить фрагментацию ресурса.

Если *не существует сервер*, соответствующий загрузке и правилам совместимости, механизм политик AWM оценит каждый сервер в кластере для определения более низких по приоритету приложений, которые могут быть переключены или завершены. Каждой группе, которая может быть остановлена, назначается числовое значение, основанное на ее приоритете. Механизм политик выбирает систему с самым низким "фактором сбоя" и останавливает их. Далее аналогично ищутся группы с более низким приоритетом для повторения процесса с целью перезапуска остановленных.

Если возникает ситуация, где просто *нет адекватных серверных ресурсов*, чтобы обработать определенные приложения, NGC может связаться с помощью интерфейса к пакетам обеспечения/выделения ресурсов типа VERITAS Opforce, чтобы запросить новый сервер для его распределения в кластер.

В случае, когда внешняя программа, отслеживающая производительность приложений, устанавливает, что приложение нуждается в бóльших CPU-ресурсах, AWM будет самостоятельно (или с помощью оператора) увеличивать предустановленное значение загрузки для соответствующей сервисной группы. Это заставит механизм политик AWM предпринимать корректирующие действия. AWM сначала будет делать попытку на завершение/переключение сервисных групп с более низким приоритетом, также выполняющихся на данном сервере. Если это не приводит к необходимому результату, AWM может пытаться делать фактический сервер бóльшим, изменяя размер виртуальной машины (или размер физической партиции, измеряемой в количестве CPU – Rel. 2). Если это невозможно, AWM будет пытаться перенести сервисную группу на бóльший сервер.

Последние две операции будут выполняться средствами UpScale или т.н. технологии расширенной прикладной виртуализации (Advanced Application Virtualization – AAV) – механизма, разработанного компанией Ejasent и приобретенного VERITAS в начале 2004 г.). AAV позволяет осуществлять миграцию онлайновых приложений между платформами без прерывания или разъединения пользователей в течение 1-2 сек. AAV делает мгновенный снимок работающего приложения, переносит его на другой сервер и восстанавливает его точное состояние до переноса и все открытые сетевые соединения. Тестовые испытания функционирования AAV, по заявлениям представителей VERITAS (в частности, В. Критова на состоявшейся конференции VISION-2004), прошли успешно и первые поставки AVV ожидаются в первой половине 2005 г.

Доступность и реализация NGC

В настоящее время у нескольких заказчиков в США успешно работает бета-версия NGC 1.0. Выпуск NGC 1.0 планируется на конец 2005 г. с поддержкой Linux и Solaris. В следующих реализациях будут добавлены Windows 2003, AIX и HP UX. Начиная с версии 2.0 NGC будет поддерживать связанные кластеры в целях обеспечения функции DR (развиваемой с VCS 4.0). Все возможности GCO в VCS 4.0 будут перенесены в NGC. NGC будет также поддерживать удаленный кластер как "offsite" репозиторий базы данных с config/status-информацией серверов в целях обеспечения бóльшей защиты мастера политик.

Вместо заключения

Реализация NGC обеспечит значительное развитие автоматизации управления кластерной архитектуры с точки зрения повышения доступности, надежности, гетерогенности, управляемости, используемости, масштабируемости при общем снижении затрат. NGC позволит организовывать крупные центры данных с кластерным гетерогенным пулом серверов до многих сотен, с большим количеством различного типа приложений, каждый со своими уровнями доступности, требованиями восстановления после катастроф, а также требованиями производительности. Однако NGC не отменяет использования, например VCS, для небольших и средних компаний с числом не самых мощных серверов до нескольких десятков, к которым требования гораздо ниже, чем к NGC-кластерам.

Дальнейшее развитие архитектуры кластеров лежит в рамках направления VERITAS H^2A, которое развивает платформу для более быстрого восстановления после сбоев/отказов и параллельных приложений.