Производительность программных RAID-массивов из разных дисков в Linux

Rygoravich, Четверг, 04 Сентябрь 2008, 00:00

Для начала — пару слов о RAID вообще. Не буду углубляться в описание этой технологии, т.к. заинтересованные могут легко найти множество описаний этой технологии в Интернете — как вводных, дающих общее представление, так и достаточно подробных. Если вы не знакомы с принципами работы RAID-массивов, то рекомендую перед прочтением данной статьи ознакомиться хотя бы в общих чертах. Здесь скажу лишь, что RAID позволяет при наличии нескольких жестких дисков организовать их пространство так, что они будут видны пользователю как одно устройство и при этом обеспечить повышение производительности и (или) отказоустойчивость. Под отказоустойчивостью здесь следует понимать сохранность пользовательских данных при физическом выходе из строя одного из составлющих RAID-массив дисков. RAID-массивы бывают аппаратные (реализуемые непосредственно оборудованием) и программные (реализуемые на уровне операционной системы).

Итак, первый вопрос — выбрать аппаратный или программный RAID? Преимущества аппаратного массива — скорость, которая по утверждению большинства интернет-источников выше, чем у программного, а также меньшая нагрузка на центральный процессор. Однако, существует также много недостатков. Во-первых, для этого нужно иметь аппаратный контроллер. Он присутствует на некоторых материнских платах, если же на материнской плате его нет — можно купить отдельную плату под PCI или PCIe шину (стоимость начинается примерно от 20 у.е.). Нужно заметить, что недорогие контроллеры (в том числе и все встроенные в материнские платы для настольных компьютеров) являются смешанными программно-аппаратными и не имеют собственного процессора, а это значит, что и в скорости отстают от дорогих серверных контроллеров (и даже программных RAID — об этом еще будет сказано ниже) и центральный процессор все же нагружают. Кроме того, для таких контроллеров нужен специальный драйвер, коего под Linux может и не существовать. Настоящие же аппаратные RAID-контроллеры выпускаются, как правило, для серверов и стоят на порядок дороже. Далее — для аппаратного контроллера небходимо, чтобы все диски были одной марки (некоторые источники даже рекомендуют использовать только диски из одной серии). А ведь если даже имеющиеся диски действительно одной марки, то в случае выхода одного из них из строя заменить его будет скорее всего уже нечем. И самый главный минус — при выходе из строя RAID-контроллера восстановить данные будет проблематично, т.к. нужно будет искать точно такой же контроллер (или материнскую плату — в случае, если контроллер встроенный). В условиях предприятия, где компьютеры закупаются партиями, последний аргумент зачастую можно не учитывать, а для домашнего пользователя он может быть очень критичным.

С учетом вышесказанного можно сказать, что в большинстве случаев создание программного RAID будет более предпочтительным. Однако встает вопрос производительности — насколько быстрой будет система с программным RAID-массивом? Начнем с теории. Существует множество реализаций (уровней) RAID, однако на практике обычно используются только четыре. Это JBOD (последовательная запись на диски массива, в программных массивах Linux носит название linear), RAID0 (запись на диски с чередованием), RAID1 (зеркалирование) и RAID5 (запись с чередованием и контролем четности). Рассмотрим их производительность по отдельности, приняв, что диски, из которых состоит RAID-массив, одинаковы.

JBOD (linear) — производительность равна производительности одного диска (иногда можно наблюдать прирост производительности, если работа идет одновременно с данными, расположенными на разных физических дисках), объем — сумме объемов дисков, отказоустойчивости нет (при выходе из строя одного из дисков есть некоторые шансы спасти часть информации). Программный массив linear вряд ли имеет смысл использовать, т.к. LVM (Logical Volume Manager) обеспечивает ту же функцию, однако гораздо более гибкий.

RAID0 — производительность в n раз выше, чем у одного диска (здесь и далее n — количество дисков в массиве). Заметим, что это только теоретически, практически же даже на лучших аппаратных RAID-массивах n-кратного прироста скорости достичь не удается. Объем равен произведению n на объем одного диска. Отказоустойчивости нет, при выходе из строя одного диска гарантированно и безвозвратно теряется вся информация.

RAID1 — производительность при записи равна производительности одного диска, при чтении — в n раз выше (обычно в два, так как зеркалирование более чем на двух дисках применяется исключительно редко) за счет одновременного считывания разных участков файла с разных дисков. Опять же, практически и при чтении и при записи производительность оказывается ниже теоретической. Объем равен объему одного диска. Есть отказоустойчивость — при выходе из строя одного диска вся информация сохраняется. Замечу, что производительность RAID1 при использовании недорогих программно-аппаратных обычно уступает (иногда — значительно) одиночному диску как в записи, так и в чтении.

RAID5 — производительность в n-1 раз выше, чем у одного диска (как и в предыдущих двух случаях — на практике ниже этого значения), объем равен произведению объема одного диска на n-1, есть отказоустойчивость, выход из строя любого диска не влечет за собой потерю информации. За счет необходимости контроля четности загружает центальный процессор (только в случае программного или программно-аппаратного RAID, в случае аппаратного массива эту нагрузку берет на себя процессор контроллера).

Итак, на десктопах возможно использование RAID0 — для обеспечения лучшей производительности (хотя этот параметр редко является критичным, в отличие от серверов, но все же при копировании больших объемов данных по гигабитной сети или же в некоторых мультимедийных операциях может быть полезен), RAID1 — для обеспечения надежности хранения важных данных, RAID5 — как сочетание обоих преимуществ.

Теперь протестируем указанные варианты в реальной системе с разными дисками. Тестовая конфигурация: процессор Duron-1300 на материнской плате Elitegroup K7S5A (чипсет SIS-735), 1Gb памяти DDR-266, остальные устройства не так важны. Тестирование проводилось в Slackware Linux 12.0 с установленным по умолчанию для этого дистрибутива ядром 2.6.21.5-smp.

В качестве дисков были протестированы три IDE-устройства:

1. 160Gb Seagate-Maxtor STM3160215A, 7200RPM, cache 2Mb подключенный как primary master и соответственно определяемый системой как /dev/hda (далее по тексту — диск a)

2. 40Gb Seagate ST340015A, 5400RPM, 2Mb (primary slave, /dev/hdb, диск b)
3. 160Gb Samsung SP1614N, 7200RPM, 8Mb (secondary master, /dev/hdc, диск c)

Все диски работали в режиме UltraATA-100 (udma5). Как secondary slave был подключен привод DVD, во время тестов он не использовался. Заметим, что использование IDE-дисков в RAID не рекомендуется при расположении входящих в массив устройств на одном шлейфе. Вызвано это тем, что из подключенных на один шлейф дисков только один из них может использоваться одновременно, а работа с RAID обычно подразумевает одновременный доступ ко всем устройствам массива. Однако мы будем рассматривать и такие массивы для того, чтобы определить, насколько существеннен этот параметр.

Для теста на каждом из перечисленных устройств был создан раздел объемом 10Gb. Кроме того, для проверки конфигураций с RAID-массивами, состоящими из разделов на одном физическом диске, был создан еще один дополнительный раздел того же объема на диске c. Замечу, что такие конфигурации носят чисто академический характер и практического значения не имеют. Разметка производилась с учетом уже существующих разделов, т.к. диски, напомню, разные и целью тестирования не ставилось измерение производительности конкретных моделей дисков. Итак, в итоге получилось четыре раздела объемом по 10Gb и определяемые системой как /dev/hda3, /dev/hdb1, /dev/hdc5 и /dev/hdc6. Во всех экспериментах на диске с использовался раздел /dev/hdc5, кроме конфигураций с двумя разделами на одном физическом диске, в которых оба раздела находились на диске c.

В качестве файловой системы использовалась ext3. Применение нежурналируемой системы ext2 дало бы более правильные результаты с точки зрения теории, однако поскольку цель опыта практическая, то взята была наиболее распространенная в настоящий момент общеупотребительная файловая система в Linux. Создание файловой системы производилось с настройками по умолчанию. Тестирование производилось утилитой bonnie++ (http://www.coker.com.au/bonnie++/).

Командная строка выглядела так:
bonnie++ -s 4096 -d md0/
где ключ -s задает размер записываемового/считываемого объема данных (4Gb, был выбран для обеспечения не очень продолжительного времени теста и одновременно для минимизации влияния кэширования диска операционной системой на результаты тестов). Ключ -d задает каталог, в который производилось монтирование тестируемых устройств.

Замеры производительности осуществлялись для семнадцати возможных конфигураций. Во-первых, все четыре диска были протестированы отдельно. Во-вторых были испытаны конфигурации с RAID1 для дисков a+c, b+c, a+b (обратите внимание — в этой конфигурации разделы находились на дисках, подключенных к одному шлейфу) и c+c (разделы на одном физическом диске). Размер чанка принимался по умолчанию (64kb). Далее тестировался RAID0 — диски a+c, b+c, a+b, c+c и конфигурация с тремя дисками a+b+c. И наконец RAID5 — диски a+b+c, а также конфигурации b+c, a+c, a+b с одним удаленным командой
mdadm --manage /dev/md0 --fail /dev/hdxN --stop /dev/hdxN
разделом для тестирования нештатных режимов, которые могут возникнуть при выходе из строя одного из дисков массива.

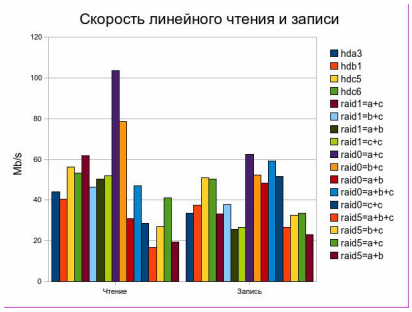
Производительность оценивалась по следующим критериям: скорость линейного чтения и записи (везде выражена в Mb/s), поиск случайного участка данных (случайный доступ, выражена в секундах в минус первой степени, т.е. как отношение количества найденных

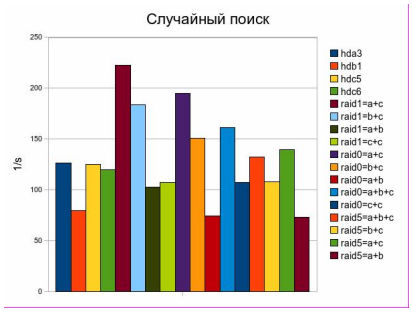
участков ко времени поиска). Кроме того, была определена загрузка процессора. Т.к. загрузка процессора напрямую зависит от скорости чтения/записи, для объективной оценки введем понятие удельной загрузки процессора, т.е. отношения загрузки процессора к скорости чтения/записи.

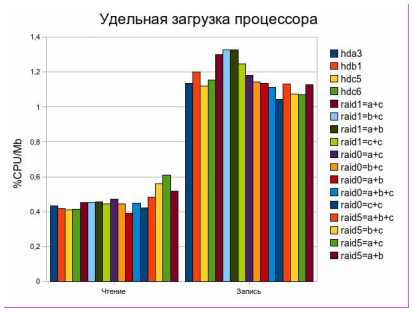
Перейдем к результатам. Сведем их в таблицу:

Конфигурация	Чтение			Запись			Случайный доступ
	Скорость	Загрузка процессора	Удельная загрузка	Скорость	Загрузка процессора	Удельная загрузка	
hda3	44	19	0,43	33,47	38	1,14	126,5
hdb1	40,56	17	0,42	37,45	45	1,2	79,6
hdc5	56,19	23	0,41	50,9	57	1,12	125
hdc6	53,17	22	0,41	50,25	58	1,15	119,8
raid1=a+c	61,76	28	0,45	33,08	43	1,3	222,6
raid1=b+c	46,45	21	0,45	37,69	50	1,33	184,1
raid1=a+b	50,27	23	0,46	25,62	34	1,33	102,7
raid1=c+c	51,93	23	0,44	26,46	33	1,25	107,1
raid0=a+c	103,76	49	0,47	62,51	74	1,18	194,9
raid0=b+c	78,46	35	0,45	52,36	60	1,15	150,9
raid0=a+b	30,77	12	0,39	48,38	55	1,14	74,2
raid0=a+b+c	46,98	21	0,45	59,29	66	1,11	161,1
raid0=c+c	28,55	12	0,42	51,65	54	1,05	107,7
raid5=a+b+c	16,59	8	0,48	26,52	30	1,13	132,3
raid5=b+c	26,77	15	0,56	32,58	35	1,07	108,1
raid5=a+c	40,95	25	0,61	33,58	36	1,07	139,4
raid5=a+b	19,31	10	0,52	23,04	26	1,13	73,3

А также представим также в виде графиков:







А теперь непосредственно анализ. Сначала уделим внимание последнему из графиков, поскольку он получился достаточно ровным и проще всего поддается анализу. В сравнении с однодисковыми конфигурациями удельная загрузка процессора несколько повышается при чтении с RAID5 (особенно, если избыточность нарушена) и при записи на RAID1, в остальных случаях этот параметр практически не зависит от применяемых массивов, а значит, потерей вычислительных ресурсов на обслуживание RAID-массивов можно пренебречь (напомню,

тестирование производилось на устаревшем на сегодняшний день процессоре Duron-1300).

Однодисковые конфигурации. В общем и целом, все ожидаемо, за исключением скорости записи на Seagate-Maxtor STM3160215A (hda3) — она оказалась меньше, чем аналогичный показатель гораздо более старого Seagate ST340015A, несмотря на большую плотность записи и более низкую скоростью вращения шпинделя. Сам факт, конечно, интересен, однако не будем на нем останавливаться, поскольку цель тестирования иная.

Далее — конфигурации с RAID1. Скорость записи для винчестеров, расположенных на разных шлейфах приблизительно (в пределах допустимой погрешности) равна скорости записи на более медленный из них . Если же разделы расположены на одном винчестере, либо винчестеры находятся на одном шлейфе — скорость записи серьезно снижается. Скорость чтения для конфигураций дисков a+c и a+b заметно превышает скорость чтения обоих дисков. Здесь видно, что прирост производительности при чтении в RAID1 имеет место даже в случае расположения винчестеров на одном шлейфе. Массив с разделами на одном винчестере ожидаемо несколько меньше, чем производительность этого же диска без RAID. А вот пара дисков b+c показала неожиданно низкую производительность. И наконец скорость случайного поиска. Для разделов на одном диске, как и скорость чтения она оказалась несколько ниже, чем для одного раздела того же винчестера. Для дисков на одном IDE-канале она приблизительно равна среднему арифметическому от аналогичных показателей входящих в массив дисков. А вот конфигурации с дисками на разных шлейфах показали очень высокую скорость поиска — забегая вперед, скажу, что даже выше, чем аналогичный показатель для массивов RAID0.

Общий вывод — если не жалко многих гигабайтов потраченных на избыточность во имя надежного сохранения многих гигабайтов имеющейся информации (напомню, что половина дискового пространства в RAID1 для использования теряется), то программный RAID1 вполне можно использовать. Хорошая скорость при чтении, средняя скорость записи и очень высокая скорость поиска в сочетании с отказоустойчивостью — для многих задач такая организация дискового пространства может показаться вполне приемлемой. Если с устройства будет производиться преимущественно считывание данных (как, например, происходит в FTP или HTTP-серверах), то можно даже использовать для RAID1 два IDE-устройства на одном канале, поскольку серьезная потеря производительности будет проявляться лишь для записи. Конфигурации с RAID1 подойдут тем пользователям, которые хотят гарантированно сохранить свои данные при отказе одного из винчестеров, но не имеют возможности построить массив RAID5.

Массив RAID0 в общем и целом подтвердил свою репутацию сверскоростного накопителя. Конфигурации с дисками на разных каналах IDE показали высокую скорость и при чтении и при записи, уступив, правда RAID1 в скорости поиска. Отмечу парадокс — скорость чтения конфигурации RAID1=a+c по результатам теста превысила сумму скоростей чтения составляющих ее дисков, что логически необъяснимо. Впрочем, разница невелика и вполне может быть объяснена погрешностью измерений, однако прирост скорости все равно можно отметить как чрезвычайно высокий. Массив с двумя дисками на одном шлейфе показал среднюю скорость записи, низкую скорость чтения (замечу, что чтение оказалось намного медленнее записи) и низкую скорость поиска, а поскольку RAID0 не только не обеспечивает отказоустойчивость, но даже снижает ее (в сравнении с одиночным диском), то вряд ли кто-нибудь сумеет найти выгоду от подобного использования дискового пространства.

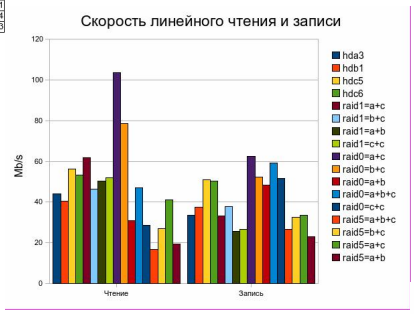
Тестирование двух разделов одного диска дало примерно те же результаты, опередив в скорости поиска. И наконец RAID0 с тремя дисками оставил двоякое впечатление — хорошая скорость записи при низкой скорости чтения и среднем времени поиска. Все-таки практичнее будет исключить из массива один из дисков, подключенных на один шлейф и использовать его независимо.

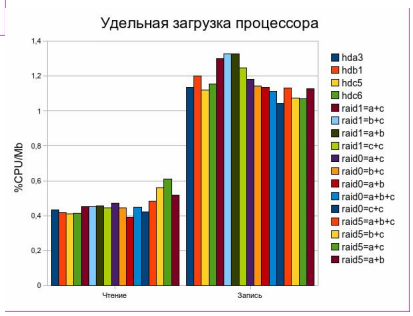
И, наконец, RAID5. Увы, эти массивы не порадовали высокой производительностью — наоборот, скорость чтения и записи были очень низкими при средней скорости поиска. Причина, вероятно, все в том же расположении двух винчестеров на одном шлейфе — подтверждение тому повышение производительности при отключении от массива дисков a или b.

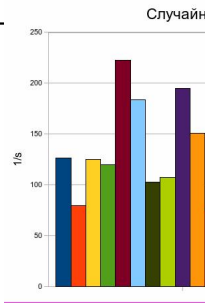
Выводы:

1. Во многих конфигурациях использование программных RAID-массивов в Linux способно повысить быстродействие дисковых операций.
2. Использование двух IDE-дисков, подключенных к одному контроллеру зачастую ведет не к повышению, а наоборот — к снижению быстродействия.

Конфигурация	Чтение			Запись			Случайный доступ
	Скорость	Загрузка процессора	Удельная загрузка	Скорость	Загрузка процессора	Удельная загрузка	
hda3	44	19	0,43	33,47	38	1,14	126,5
hdb1	40,56	17	0,42	37,45	45	1,2	79,6
hdc5	56,19	23	0,41	50,9	57	1,12	125
hdc6	53,17	22	0,41	50,25	58	1,15	119,8
raid1=a+c	61,76	28	0,45	33,08	43	1,3	222,6
raid1=b+c	46,45	21	0,45	37,69	50	1,33	184,1
raid1=a+b	50,27	23	0,46	25,62	34	1,33	102,7
raid1=c+c	51,93	23	0,44	26,46	33	1,25	107,1
raid0=a+c	103,78	49	0,47	62,51	74	1,18	194,9
raid0=b+c	78,46	35	0,45	52,36	60	1,15	150,9
raid0=a+b	30,77	12	0,39	48,38	55	1,14	74,2
raid0=a+b+c	46,98	21	0,45	59,29	66	1,11	161,1
raid0=c+c	28,55	12	0,42	51,65	54	1,05	107,7
raid5=a+b+c	16,59	8	0,48	26,52	30	1,13	132,3
raid5=b+c	26,77	15	0,56	32,58	35	1,07	108,1
raid5=a+c	40,95	25	0,61	33,58	36	1,07	139,4
raid5=a+b	19,31	10	0,52	23,04	26	1,13	73,3






Случайн
250
200
150
100
50
0
1/s