RHEL 퍼포먼스 튜닝 가이드

- Chapter 2 System Tuning Tools for Linux Servers
    - 디스크 i/o 벤치마킹 프로그램 bonnie
    - 기타 디스크 벤치마킹 프로그램 : dbench, IOzone, tiobench 등
- Chapter 4.2 Disk Tuning
- Chapter 5.2 Adding a Hardware RAID

http://people.redhat.com/mbehm/RHEL_Tuning_Guide.pdf

Chapter 2 System Tuning Tools for Linux Servers

디스크 i/o 벤치마킹 프로그램 bonnie

Block reads/writes : 한번에 8KB 블락을 읽어들이고 순차적으로 다음 블락을 읽음. OS에서 순차적인 디스크 읽기 성능을 측정함.
파일시스템을 통해 정규파일을 읽어들이는 테스팅으로 async i/o 나 raw disk i/o 테스팅이 아니라는 것을 생각해야한다. IOMeter 프로그램의 경우는 raw i/o와 async i/o 를 사용하여 속도를 측정하며 파일시스템 레이어를 제외하고 디스크과 컨트롤러의 속도를 측정한다.파일 시스템을 통해야 실제 환경과 비슷한 결과를 판단할 수 있다. 대부분의 애플리케이션은 async i/o, raw i/o를 사용하지 않기 때문에 IOMeter 에서 나타는 수치와 같을 수가 없다. (http://www.iometer.org/)

Character reads/writes : getc, putc 를 이용하여 파일에서 1바이트를 읽고 쓰는 테스팅을 함. libc 라이브러리 의 속도만을 테스팅하는 것이다. 이번 테스팅과는 관련이 없다.

rewrites : 8KB chunks of data를 읽고 1바이트를 변경한후 다시 파일을 쓰는 테스트. 파일시스템에 심한 부하를 주는 작업. 계속 디스크의 데이타 블록을 업데이트하거나 디스크의 블록을 재할당하는 작업. OS에서 데이타 캐쉬를 얼마나 잘하는지 테스팅을 한다. 작은 SQL DB에 update를 계속 주는 상황과 비슷.

random seeks : os에서 i/o 요청받은 것에 대해서 얼마나 잘 처리하는지 테스팅. 임의로 파일을 찾고 쓰며 작은 데이타 블락을 읽는 테스팅임. 초당 처리량. 하드디스크에서 발생하는 가장 큰 정체현상은 실제 디스크 처리량(throughput)이 아니라 디스크 탐색 시간이다.

기타 디스크 벤치마킹 프로그램 : dbench, IOzone, tiobench 등

Chapter 4.2 Disk Tuning

- 최근의 디스크 컨트롤러는 read, write 성능을 향상시킬 수 있는 캐쉬 기능이 내장되어 있다. 그런데 데이타 손실의 위험때문에 wirte caching 을 사용하지 않도록 하는 제조업체도 있다.
- SCSI 튜닝(SCSI 카드) : Tagged-command queuing(TCO) 는 헤드의 검색 움직임을 줄일 수 있도록 디스크 컨트롤러에서 i/o 요청을 재정렬하는 기능이다. 커널값을 조정하여 변경할 수 있다. 이에 대해서는 특정 SCI 모듈에 대한 해당 드라이버 문서를 참고해야 할 것이다. RHEL4에서는 kernel-doc 문서를 참고한다.
  Read-ahead : Read-ahead 기능은 블락 A를 읽고나서 B,C를 읽으리라 예상을 하는 경우 다량의 데이타 블락을 미리 읽어들이는 기능이다. RHEL4 에서는 blockdev 프로그램을 이용하여 설정을 한다.
- 파일시스템 단편화 : ext2/ext3 파일시스템에서는 일반적인 쟁점사항은 아니다.
- ext2 파일 시스템 튜닝 : ext2 에서 기본 블락사이즈는 1024. 큰 파일을 주로 서비스하면 2048이나 4096으로 변경할 경우 데이타 탐색횟수를 줄일 수 있다.
- ext3 튜닝
  - ext3의 저널링이 하드 드라이트 헤드의 움직임을 최적화하기 때문에 대부분 ext2보다 더 높은 전송량을 보인다. 세가지 저널링 모드에서 스피드 최적화를 선택할 수 있으며 이 경우 데이타 정합성은 미약해질 수 있다.
  - ext3 에서 가능한 최적의 성능을 얻으려면 external journal 파티션을 사용해야한다. 이 파티션은 파일시스템을 담은 디바이스와 비슷하거나 더 좋은 성능에 있어야 한다.또한 저널 파티션은 저널링하는 파일시스템에서 사용하는 동일한 블락 크기로 만들어야 한다.
  - noatime 을 이용하여 access timestamps 를 기록하지 않도록 할 수 있다. 그러나 tmpwatch 프로그램을 사용하는 파티션에는 사용하면 안된다. access time을 기준으로 임시 파일을 삭제하기 때문이다.
- software raid tuning : chunk-size가 중요함. chunk-size는 단일 operation에 디스크에 쓰는 최소 데이타양이다. raid 5 디바이스에서 chunk size는 (the size of a typical i/o operation)/(the number of disks -1) 이다. 일반적인 chunk size는 32KB 에서 128KB이며 4KB(최대 파일시스템 블락크기)로 나누어져야 한다.
  - 큰블락이 빠르지만 작은 파일을 저장할 경우 비효율적이다. 블락이 작은 경우 디스크 공간은 효율적으로 쓰지만 속도는 느리다.
- 파티션 위치 : 생략
- GFS : 생략

Chapter 5.2 Adding a Hardware RAID

- raid level
  - RAID level 0 : strie 사이즈가 너무 크면 하나의 물리적 디바이스에서 불균형한 요청을 처리해야 한다. 그러므로 stripe width

size를 평균 데이타 요청 크기와 맞추어야 한다. raid 0은 sequential transfers, random access 모두에 속도를 늘릴 수 있다. sequentail transfers 에서는 싱글디스크보다 전송량이 크므로 효과가 있고 random transfers 은 특정 드라이브에 대해 더 적은 탐색 요청이 가므로 효과가 있다. 그러나 성능은 선형으로 향상되지는 않는다.
    - RAID level 1 : read 성능은 단일 디스크와 동일하다. write 성능은 write 정책에 따라 다르다. 최적의 parallel write policy 는 싱글 디스크보다 빠르다. 최적의 seiral write policy 는 싱글 디스크의 절반이 될 것이다. 모든 raid 컨트롤러가 parallel write policy 를 지원하지는 않는다.
    - RAID level 5 : 신뢰성과 가격을 고려했을때 최적. raid 0 에서 하나의 디스크를 뺀 성능에 비슷하다. (4 disk raid 5 = 3 disk raid 0) 그러나 write 성능은 패리티 스트립을 업데이트해야 하기 때문에 느리다.
- RAID Trade-offs
    - RAID level
    - 디스크 숫자
    - 디스크 속도
    - 디스크 컨트롤러 속도
    - stripe 크기 (데이타 블락 사이브의 배수가 되어야함)
    - raid 컨트롤러의 캐쉬 사이트(특히 raid 5 시스템에서). 캐쉬는 작은 여러개의 쓰기 요청을 단일한 큰 크기의 요청으로 그룹화할 수 있다.
    - db의 블락 사이즈

주의할것이 있다. stripe width 는 chunk size * 디스크 갯수이다. 위에서 "stripe width size를 평균 데이타 요청 크기와 맞추어야 한다"고 했는데 현재 시스템을 모니터링하여 평균 데이타 요청 크기를 구하고 raid0 의 디스크 갯수로 나눈 값이 적절한 chunk size가 될 것이다.