|  |
| --- |
| * Ch\_1 \_ 커널 컴파일 * Ch\_2 \_ SATA 장치와 SCSI 장치 * Ch\_3 \_ 하드디스크 추가 개념 , 장착 실습 * Ch\_4 \_ RAID 정의와 개념 * Ch\_5 \_ 여러 개의 하드디스크 장착 및 파티션 생성 * 모듈의 개념과 커널 컴파일의 필요성 * 모듈: 필요할 때마다 호출하여 사용되는 코드      * 커널 컴파일 * 커널 컴파일 순서      * 커널 업그레이드 방법      * 현 커널 버전 확인 (uname -r 명령어)      * 버전 확인 후 커널 소스 다운로드 * /user/src 폴더에 다운로드 (본인은 다운로드 폴더에서 mv 명령어로 이동함)\      * 커널 소스 압축 풀기 * unxz [xz로압축된파일] 명령어 사용하여 압축 풀기 * tar xvf [tar파일] 명령어 사용하여 tar아카이브 파일을 해제 * 폴더 생성      * 추가 패키지 설치      * 커널 설정 초기화 (make mrproper 명령어)        * 커널 환경 설정 * # make xconfig 명령어로 커널 환경 설정 열기 * processor family에서 cpu종류 선택가능 // Generic-x86-64 선택 * windows MTFS 쓰기 설정 (읽기는 기본적으로 적용되어 있는 상태)      * 이전 정보 삭제 후 커널 컴파일 및 설치 진행 * # make clean 명령어로 이전 정보 삭제 * # make #make modules\_install #make install 각자 설치 명령어들이지만 ; (세미콜론) 을 붙임으로써 이어서 할 수 있음   (예: make ; make modules\_install 경우 make 다음 make modules\_install 진행하는 식)   * 시간이 걸리기 때문에 ; 사용 권장 * SATA 장치와 SCSI 장치의 구성 (1) * Server의 하드웨어 구성도        * CD/DVD가 SATA에 장착되어 있는 모습 * Virtual Machine Settings에서 변경 가능 * SATA 0:1을 리눅스에서는 /dev/sr0 로 부름      * Hard Disk 가 SCSI에 장착되어 있는 모습 * SCSI 중간에 Reserved는 예약이 되어 사용 불가 * SCSI0:0 리눅스에서는 /dev/sda 로 부름 (순서에 따라 맨 뒤 변경)   (ex) SCSI 0:1 = /dev/sdb SCSI 0:2 = /dev/sdc   * 파티션으로 구분가능 (4개까지) (맨 뒤 넘버 부여)   디스크 파티션이 나눠진 것을 논리적으로는 /dev/sda1 , /dev/sda2 ,  /dev/sda3 … 형식으로 부름   * 하드디스크 추가하기 - 1개 * 하드디스크 1개 추가 하드웨어 구성      * 장착된 디스크의 이름은 /dev/sdb * 논리적인 파티션의 이름은 /dev/sdb1 * 파티션을 그냥 사용할 수 없으며 반드시 특정한 디렉터리에 마운트 시켜야만 사용이 가능 * 하드디스크 1개 장착 실습 * 실습목표 * 하드디스크를 추가 장착해서 사용한다. * 디스크 파티셔닝과 관련된 fdisk, mkfs, mount 명령을 익힌다. * 부팅시 자동으로 읽히는 /etc/fstab 파일을 편집한다. * 실습 흐름도        * Server에 새로운 하드디스크를 추가 (SCSI 0:1 로 생성된 것을 확인 가능)      * fdisk명령으로 이동 * # fdisk [장치이름] || ex) # fdisk /dev/sdb * m 입력으로 서브 명령어 확인 가능      * 파티션 (sdb1) 생성 * 서브 명령어 n 입력 (add a new partition 새로운 파티션 만들기) * partition type 은 primary 선택 (p 입력) * partition number 1~4까지 설정 가능 (4개파티션만 만들 수 있기 때문)   1 설정 (default 값이 1)   * First sector와 Last sector 설정 (본인은 전부 사용 “2048-2097151”)   원하는 용량 선택 가능 (K,M,G,T,P 등)     * 생성된 파티션 확인 * 확인 (p) 후 적용(w)      * mkfs.ext4 /dev/sdb1 명령어로 포맷 진행 (파일시스템 생성) * 파티션이름을 지정해야 함(sdb1) 물리장치 이름 x (sdb)      * mydata 디렉터리 생성 뒤 해당 디렉터리에 mount 진행 * mkdir /mydata (디렉터리 생성) * mount /dev/sdb1 /mydata (해당 디렉터리에 mount)      * mydata에 성공적으로 mount된 것을 확인 (df 명령어)      * 새로운 파일을 생성하여 mydata에 집어넣기 * 1% > 2% 사용량이 늘어난 것을 확인 가능 (사용 가능)      * 파티션을 umount한 뒤 mydata 디렉터리 확인하니 확인불가      * /etc/fstab에 등록 (본인은 gedit에디터 사용함) * /dev/sdb1 /mydata ex4 defaults 0 0   [sdb1] 장치를 컴퓨터가 실행될 때 마다 [/mydata]에 연결     * 재부팅 후 성공적으로 mount되어 있는 것을 확인 * RAID 정의 및 개념 * RAID 정의 * RAID(Redundant Array of Inexpensive Disks)는 여러 개의 디스크를 하나의 디스크처럼 사용함 * 비용 절감 + 신뢰성 향상 + 성능 향상의 효과를 냄 * 하드웨어 RAID * 하드웨어 제조업체에서 여려 개의 하드디스크를 가지고 장비를 만들어서 그 자체를 공급 * 좀 더 안정적이지만, 상당한 고가임 * 소프트웨어 RAID * 고가의 하드웨어 RAID의 대안 * 운영체제에서 지원하는 방식 * 저렴한 비용으로 좀 더 안전한 데이터의 저장이 가능 * 소프트웨어 RAID 내용을 실습할 예정 * 각 RAID방식의 비교      * Linear RAID, RAID0 * Linear RAID 개요 * 최소 2개의 하드디스크가 필요 * 2개 이상의 하드디스크를 1개의 볼륨으로 사용 * 앞 디스크부터 차례로 저장 * 100%의 공간효율성 (= 비용 저렴) * RAID 0 개요 * 최소 2개의 하드디스크가 필요 * 모든 디스크에 동시에 저장됨 * 100%의 공간효율성 (= 비용 저렴) * 신뢰성 낮음 * ‘빠른 성능을 요구하되, 혹시 전부 잃어버려도 큰 문제가 되지 않는 자료’ 가 적당함 * RAID 1 * RAID 1 개요 * ‘미러링(Mirroring)’이라 부름 * 데이터 저장에 두 배의 용량이 필요 * 결함 허용(Fault-tolerance)을 제공 = 신뢰성 높음 * 두 배의 저장 공간 = 비용이 두배 = 공간효율 나쁨 * 저장속도(성능)은 변함없음 * ‘중요한 데이터’를 저장하기에 적절함 * RAID0와 RAID1 비교      * RAID 5 (1) * RAID 5 개요 * RAID1의 데이터의 안정성 + RAID0처럼 공간 효율성 * 최소한 3개 이상의 하드디스크 * 오류가 발생할 때는 ‘패리티(Parity)’를 이용해서 데이터를 복구 * “000 111 010 011”(12bit) 데이터 RAID5 저장, 복구 예시        * RAID5의 특징 * 어느 정도의 결함 허용을 해 주면서 저장 공간의 효율도 좋음 * ‘디스크의 개수 - 1’의 공간을 사용 * 디스크 2개가 고장 나면 복구 불가 * 기타 RAID * RAID 6 * RAID6 방식은 RAID5 방식이 개선된 것 * 공간 효율은 RAID5 보다 약간 떨어지지만, 2개의 디스크가 동시에 고장이 나도 데이터에는 이상이 없도록 하는 방식 * RAID6의 경우에는 최소 4개의 디스크 필요 * 공간 효율은 RAID5보다 약간 떨어지는 반면에 데이터에 대한 신뢰도는 좀 더 높아지는 효과 * 성능(속도)은 RAID5에 비해 약간 떨어진다 * RAID1+0 = RAID1 + RAID0 * 신뢰성(안전성)과 성능(속도)이 동시에 뛰어난 방법      * 하드 디스크 관리: 디스크 9개 장착 * Linear RAID, RAID0, RAID1, RAID5 구현        * 총 9개의 하드 디스크 생성      * sdb 파티션 생성 뒤 타입 변경 * 각각의 파티션 넘버는 1로 고정 (sdb1, sdf1 …) * 기본적으로 타입이 Linux (id:83) 으로 설정 되어있기 때문에 파티션 타입 변경 ( t 입력 > fd입력[Linux raid auto] ) * 나머지 8개 디스크도 동일하게 파티션 생성      * 9개 파티션 생성 완료 후 스냅샷 적용 * Linear RAID 구축 * 실습 흐름도   ㄴ |