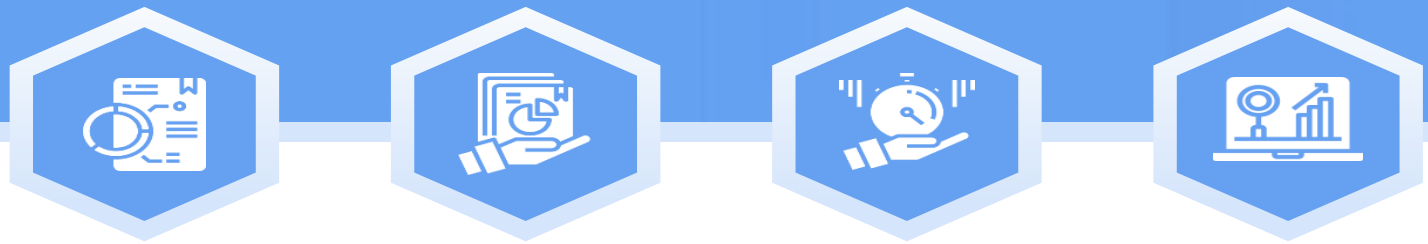
시스템 보안

#2 운영체제의 이해 - 2



KOREA UNIVERSITY
SEJONG CAMPUS

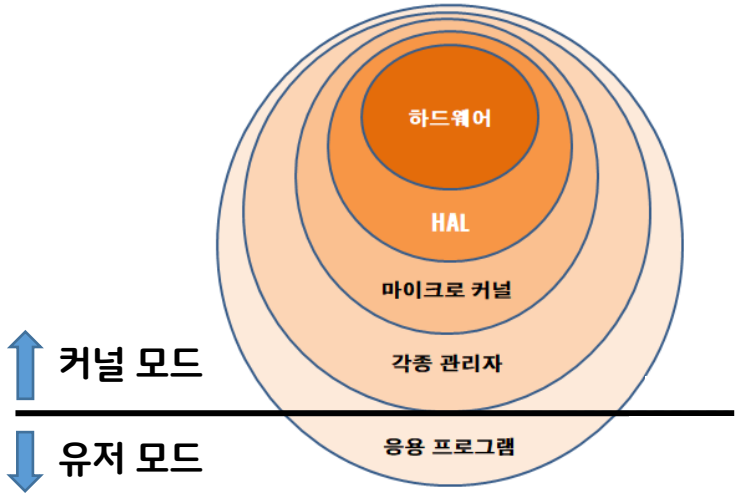
- 목차

- 윈도우 시스템
- 리눅스 시스템
- 보안 운영체제

윈도우 시스템

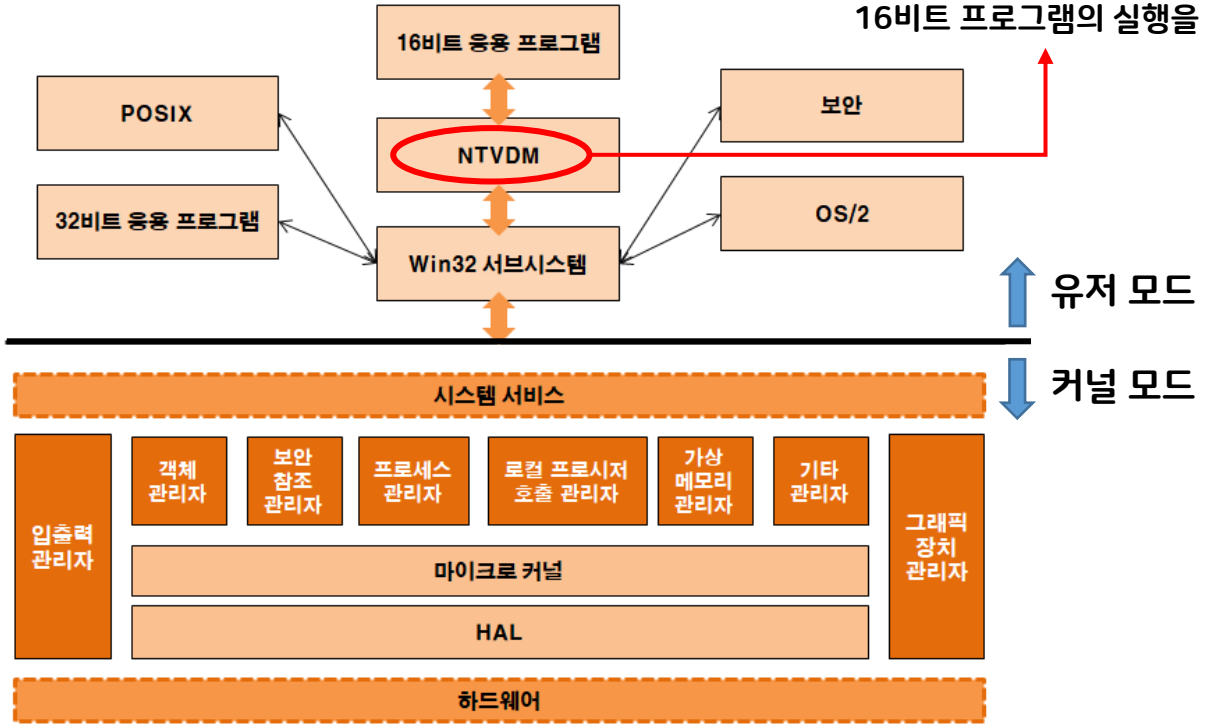
윈도우 시스템

- **윈도우의 구조**
  - 커널은 인터럽트(Interrupt) 처리, 프로세스 관리, 메모리 관리, 파일 시스템 관리, 프로그래밍 인터페이스 제공 등 운영체제의 기본 기능을 제공



윈도우 링 구조

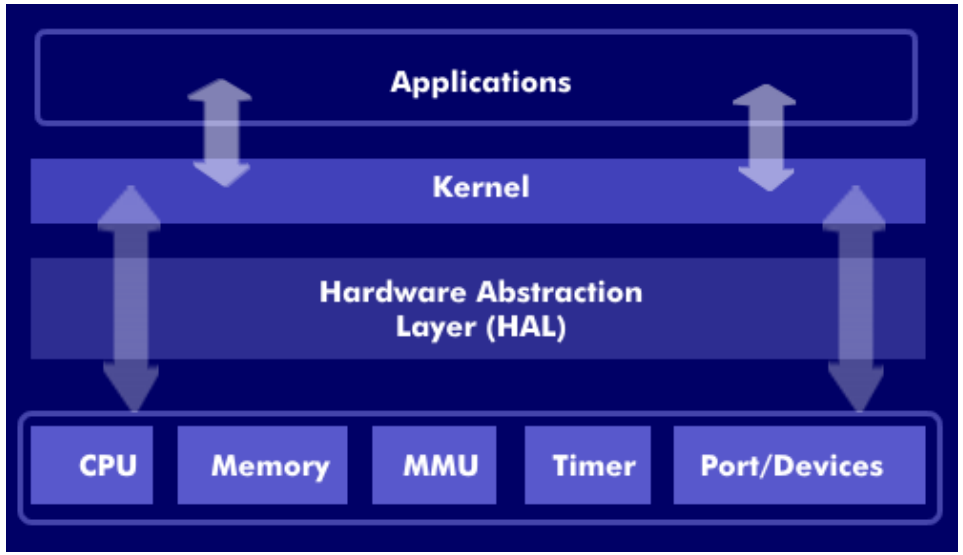
- NTVDM은 32비트 환경에서 16비트 프로그램의 실행을 지원



윈도우 NT 시스템 구조

윈도우 시스템

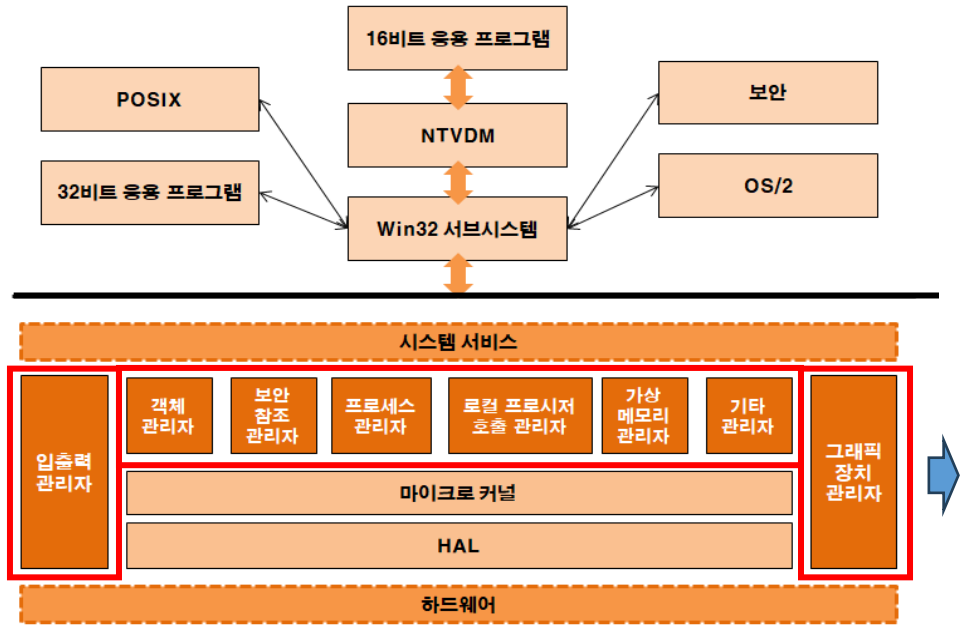
- **윈도우의 구조 - HAL(Hardware Abstract Layer) : 하드웨어 추상화 계층**
  - HAL은 컴퓨터의 물리적인 하드웨어와 컴퓨터에서 실행되는 소프트웨어 사이의 추상화 계층
  - 소프트웨어가 수많은 종류의 하드웨어 상에서 별 차이 없도록 동작할 수 있도록 하는 역할을 하며, OS의 커널 또는 장치 드라이버에서 호출



- HAL은 소프트웨어와 하드웨어 사이에서 소프트웨어가 PC의 시스템 메모리, CPU, 또는 기타 하드웨어 장치에 직접적으로 접근하는 것을 막아주며, 소프트웨어는 HAL을 통해 하드웨어의 종류에 관계없이 컴퓨터 자원을 사용하여 일관된 작업을 수행할 수 있게 됨

윈도우 시스템

- 윈도우의 구조 – 윈도우 관리자

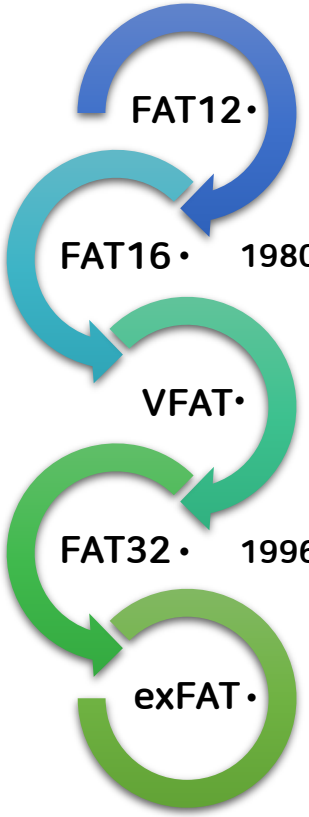


- **입출력 관리자(I/O Manager)**
  시스템의 입출력을 제어
- **객체 관리자(Object Manager)**
  객체(파일, 포트, 프로세스, 스레드 등)들에 대한 정보 제공
- **보안 참조 관리자(Security Reference Monitor Manager)**
  데이터나 시스템 자원의 제거, 허가, 거부함으로써 시스템의 보안을 설정
- **프로세스 관리자(Process Manager)**
  프로세스와 스레드를 생성하고 요청에 따라 처리
- **로컬 프로시저 호출 관리자(Local Procedure Call Manager)**
  프로세스는 서로의 메모리 공간을 침범하지 못하기 때문에
  프로세스 간의 통신이 필요할 때 대신해서 수행
- **가상 메모리 관리자(Virtual Memory Manager)**
  RAM의 메모리를 할당하고, 가상 메모리의 페이징(paging)을 제어
- **그래픽 장치 관리자(Graphics Device Interface Manager)**
  화면에 선이나 곡선을 그리거나 폰트 등을 관리
- **기타 관리자**
  캐시(Cache) 관리자, PNP(Plug and Play) 관리자, 전원 관리자 등

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : FAT(File Allocation Table)

  - 일반적으로 FAT이라고 하면 FAT16에서 긴 파일명 처리가 가능하도록 개선된 VFAT(Virtual FAT)을 의미
  - 2006년 대용량 고속 플래시 메모리를 위한 exFAT(FAT64)를 개발



FAT12 • 1980년대 초 플로피디스크용으로 처음 개발

FAT16 • 1980년대 말 하드디스크를 지원하기 위해 개발

VFAT • 1995년 FAT의 성능 향상, 긴 파일 이름 사용

FAT32 • 1996년 VFAT을 확장, 고용량 하드디스크 지원

exFAT • 2006년 대용량, 고속의 플래시 메모리를 효율적으로 다루기 위해 개발 (FAT64)

FAT 파일시스템 차이점

구분	FAT12	FAT16	FAT32	exFAT
사용 용도	플로피디스크	하드디스크	대용량 하드디스크	대용량 플래시 메모리
클러스터 비트 수	12bit	16bit	32bit(28bit)	64bit
최대 클러스터 수	4,084	65,524	268,435,456	2,294,967,295
최대 볼륨 크기	16MB	2GB	2TB(32GB)	512TB
최대 파일 크기	볼륨 크기	볼륨 크기	4GB	볼륨 크기

윈도우에서 포맷할 경우 최대 볼륨 크기는 32GB로 제한

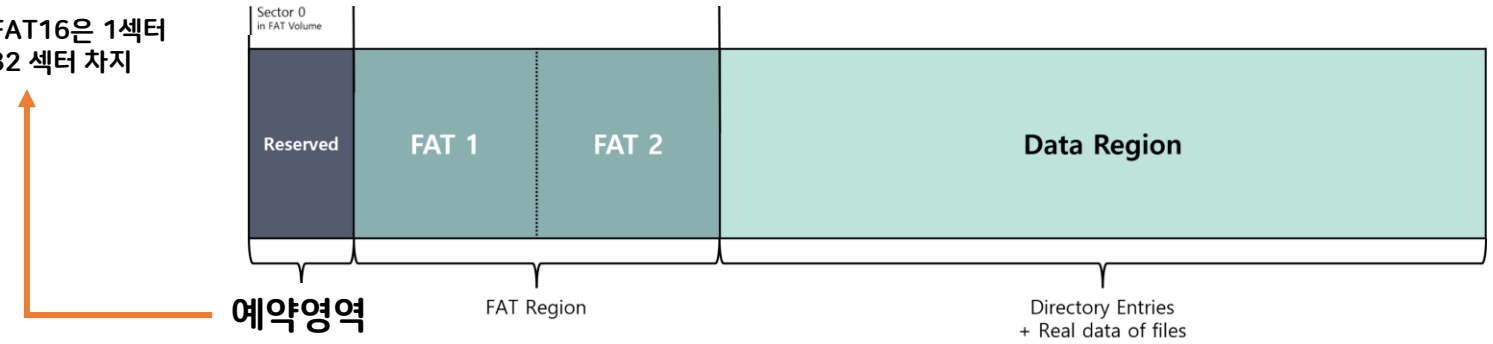
→ FAT32의 한계를 극복하고 NTFS의 오버헤드와 호환성 문제를 극복하기 위해서 개발된 경량 파일 시스템
(안정성이 NTFS에 비해서 낮기 때문에, USB 안전제거 등을 하지 않으면 파일이 삭제되기도 함)

윈도우 시스템

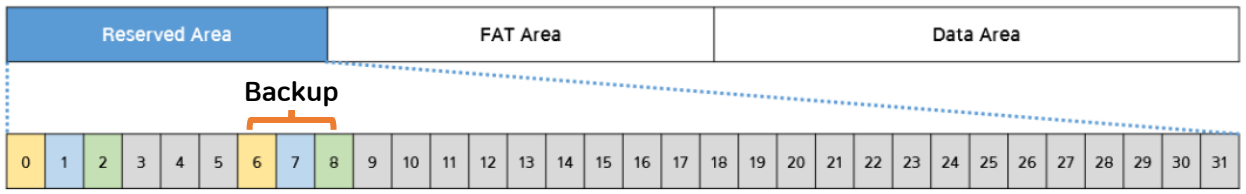
- ## 윈도우 구조 - 파일 시스템 : FAT(File Allocation Table)

  - FAT는 부트 섹터 영역에 부트 코드와 Bios Parameter Block(BPB), 패딩 영역을 포함한 Reserved 영역과 클러스터 할당 내역을 저장, 관리하는 File Allocate Table(FAT) 영역, 그리고 파일 테이블인 Directory Entry와 실제 파일 데이터가 저장되는 데이터 영역으로 구분

FAT12와 FAT16은 1섹터
FAT32는 32 섹터 차지



Layout of FAT Volume



Layout of Reserved Area

FAT File System	Size (bytes)	Offset (hex)	Description
FAT32	3	0x0000 ~ 0x0002	Jump command to Boot Code
	57	0x0003 ~ 0x0059	BPB (BIOs Parameter Block)
	420	0x005A ~ 0x01FD	Boot Code / Error msg
	2	0x01FE ~ 0X01FF	Signature (0x55AA)

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)

  - NTFS는 MS-DOS부터 쓰인 FAT를 대체하기 위해 1993년 Windows NT 3.1과 함께 발표
  - 향상된 안정성 : 파일의 변경 내용을 기록한 이후에 시스템 오류 발생으로 재부팅 될 경우 해당 파일에서 롤백을 진행
  - 보안 강화 : 파일 및 폴더에 대한 ACL(Access Control 목록) 기반 보안, BitLocker 드라이브 암호화 지원
  - 대규모 볼륨 지원 : Windows Server 2019 이상 및 Windows 10 버전 1709 이상에서 최대 8PB 볼륨을 지원
    MBR에서는 최대 파티션 크기가 2TB 제한이 있음 (GPT는 8ZB(Zettabyte)까지 허용하며 128개의 주 파티션을 생성 가능)

버전	OS
1.0	Windows NT 3.1
1.1	Windows NT 3.5
1.2	Windows NT 4.0
3.0	Windows 2000
3.1	Windows XP

NTFS 버전
(NTFS.sys 버전 번호와 혼동해서는 안됨)

NTFS 버전 확인하는 방법 (관리자 권한으로 cmd.exe 실행 후 다음 명령어 입력)
fsutil fsinfo ntfsinfo c:

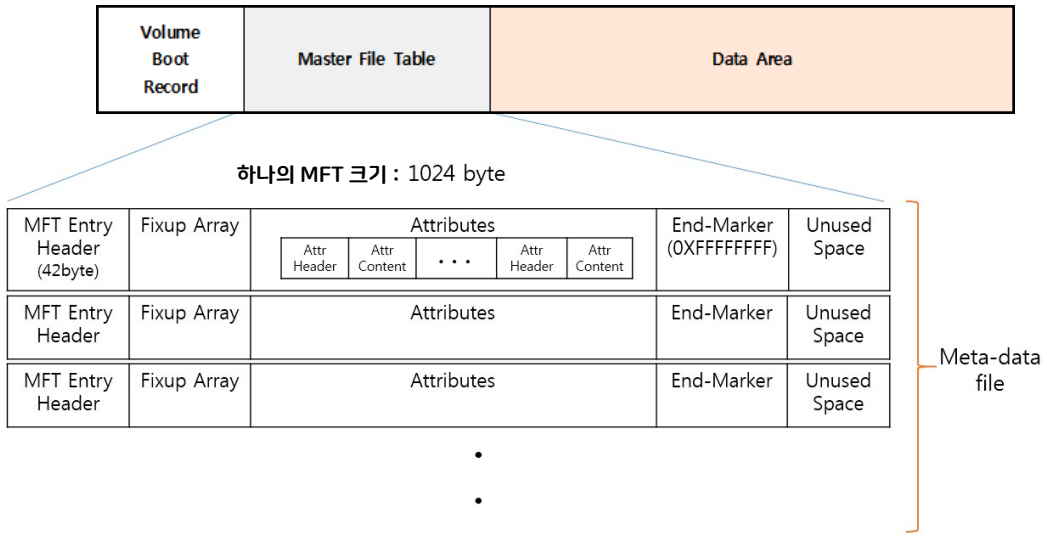
클러스터 크기	최대 볼륨 및 파일
4KB(기본 크기)	16TB
8KB	32TB
16KB	64TB
32KB	128TB
64KB(이전 최대)	256TB
128KB	512TB
256 KB	1PB
512KB	2PB
1024KB	4PB
2048KB(최대 크기)	8PB

NTFS 최대 볼륨 및 최대 파일 크기

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)

  - MFT(Master File Table) 엔트리는 NTFS 파일시스템에서 하나의 파일마다(디렉터리도 파일로 간주) 하나(하나 이상이 될 수도 있음)의 MFT 엔트리가 할당되며, 하나의 MFT 엔트리 크기는 1024 바이트



MFT 엔트리 구조 (MFT Entry Structure)

Entry Number	Entry Name (Meta File)	설명
0	$MFT	NTFS 상의 모든 파일의 MFT Entry 정보
1	$MFTMirr	$MFT 파일의 일부 백업 본
2	$LogFile	메타데이터의 트랜잭션 저널 정보
3	$Volume	볼륨의 레이블, 식별자, 버전 등의 정보
4	$AttrDef	볼륨의 이름, 크기 등의 정보
5	.	볼륨의 루트 디렉토리
6	$Bitmap	볼륨의 클러스터 할당 정보
7	$Boot	볼륨이 부팅 가능할 경우, 부트 섹터 정보
8	$BadClus	배드 섹터를 가지는 클러스터 정보
9	$Secure	파일의 보안, 접근 제어와 관련 된 정보
10	$UpCase	모든 유니코드 문자의 대문자
11	$Extend	24~26 까지의 추가적인 파일의 정보를 기록하기 위해서 사용
12-23		$MFT 확장 엔트리를 위해 예약된 영역
-	$Extend₩$Quota	사용량 정보(Windows 2000 ~)
-	$Extend₩$ObjId	파일 고유의 ID 정보(Windows 2000 ~)
-	$Extend₩$Reparse	Reparse Point에 대한 정보(Windows 2000 ~)
-	$Extend₩$UsnJrnl	파일, 디렉터리의 변경 정보(Windows 2000 ~)

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)

  - NTFS의 특징 : 보안 - 권한

① NTFS 사용권한은 파티션 또는 볼륨이 NTFS 파일 시스템으로 포맷되어 있어야만 부여할 수 있음

② NTFS 사용권한은 공유 폴더 사용권한과 달리 파일과 폴더 각각에 대해서 부여할 수 있고, 부여된 권한은 개별적으로 적용

③ NTFS 사용권한은 사용자가 로컬로 자원을 액세스하거나, 네트워크를 통해 액세스하는 모든 경우에 적용

※ NTFS 파티션 또는 볼륨 상에 있는 모든 폴더와 파일들은 자신의 ACL (Access Control List)를 가지고 있으며 이 ACL이 NTFS 사용권한
(ACL은 액세스 가능한 사용자들의 목록)

윈도우 시스템

- **윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)**

  - NTFS의 특징 : 보안 – 권한의 종류(그룹 또는 개별 사용자에 대해 설정할 수 있는 권한의 6가지 종류)

① 모든 권한: 디렉터리에 대한 접근 권한과 소유권을 변경하고 하위에 있는 디렉터리와 파일을 삭제할 수 있음

② 수정: 디렉터리 삭제 가능. 읽기, 실행, 쓰기 권한이 주어진 것과 같음

③ 읽기 및 실행: 읽기를 수행하고 디렉터리나 파일을 옮길 수 있음

④ 디렉터리 내용 보기: 디렉터리 내의 파일이나 디렉터리의 이름을 볼 수 있음

⑤ 읽기: 디렉터리의 내용을 읽을 수만 있음

⑥ 쓰기: 해당 디렉터리에 하위 디렉터리와 파일을 생성하고 소유권이나 접근 권한의 설정 내용을 확인할 수 있음

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)

  - NTFS의 특징 : 보안 - 사용자 별 권한

① Administrators : 모든 권한이 부여되어 있음

② CREATE OWNER : 자신의 폴더에 대한 모든 권한을 가지고 있음

③ SYSTEM : 모든 권한이 부여되어 있음

④ Users : 읽기 및 실행, 폴더 내용 보기, 읽기, 특정 권한이 부여되어 있음.
특정 권한에는 폴더 생성 및 파일 생성 권한이 설정되어 있고 특정 사용자만이 액세스 하도록 하기 위해서는 Users를 제거하고
특정 사용자나 그룹에게 NTFS 사용권한을 부여해야 함

※ 폴더 NTFS 사용권한 위해 계정 및 그룹추가 : 읽기 및 실행, 폴더 내용보기, 읽기 사용 권한
파일 NTFS 사용권한 위해 계정 및 그룹 추가 : 읽기 및 실행, 읽기 사용권한

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)

  - NTFS의 특징 : 보안 – 권한에 적용되는 3가지 규칙

① 규칙1 : 접근 권한이 누적

개별 사용자가 여러 그룹에 속하면 특정 파일이나 디렉터리에 대한 접근 권한이 누적되어 속해 있는 그룹에서 받은 권한을 모두 가짐

② 규칙2 : 파일 접근 권한이 디렉터리 접근 권한보다 우선

파일을 포함하고 있는 디렉터리에 대한 접근 권한보다 파일에 대한 접근 권한이 우선함

③ 규칙3 : '허용'보다 '거부'가 우선

윈도우에서는 허용 권한 없음이 거부를 의미하지 않으며,

허용과 거부 중 반드시 하나만 선택할 필요가 없고 권한이 중첩되어 적용되므로 하나가 거부 설정이 되어있으면 허용보다 거부를 우선함

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)

  - NTFS의 특징 : 기타

① USN Journal (Update Sequence Number Journal 또는 Change Journal) :
파일의 변경 내용을 기록한 이후에 시스템 오류 발생으로 재부팅 될 경우 해당 파일에서 롤백을 진행

① ADS(Alternate Data Stream) : 다중 데이터 스트림을 지원

② Sparse 파일 : 파일 데이터가 대부분은 0일 경우에 실제 데이터 기록 없이 정보만 기록

③ 파일 압축 : LZ77의 변형된 알고리즘을 사용하여 파일 데이터 압축 지원

④ VSS(Volume Shadow Copy) Service : Windows 2003 부터 지원된 기능
새로 덮여 씌워진 파일 및 디렉터리의 백업본을 유지해서 복구를 지원

① EFS(Encrypting File System) : FEK(File Encryption Key)를 이용한 대칭 키 방식의 파일 데이터 암호화 지원

② Quotas : 사용자 별 디스크 사용량 제한

③ 유니코드 지원 : 다국어 사용 가능

④ 동적 배드 클러스터 할당 : 배드 섹터가 발생 했을 때 자동으로 클러스터를 재할당

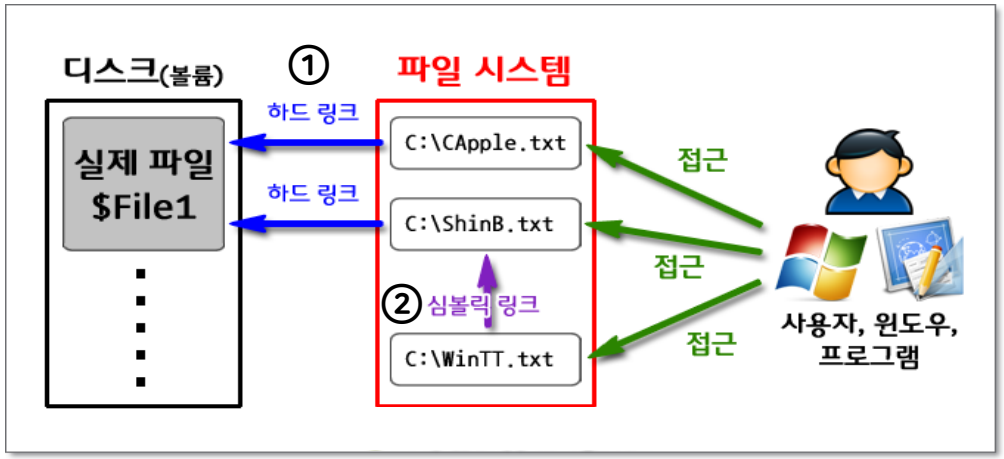
⑤ 대용량 지원 : Windows Server 2019 이상 및 Windows 10 버전 1709 이상에서 최대 8PB 볼륨을 지원

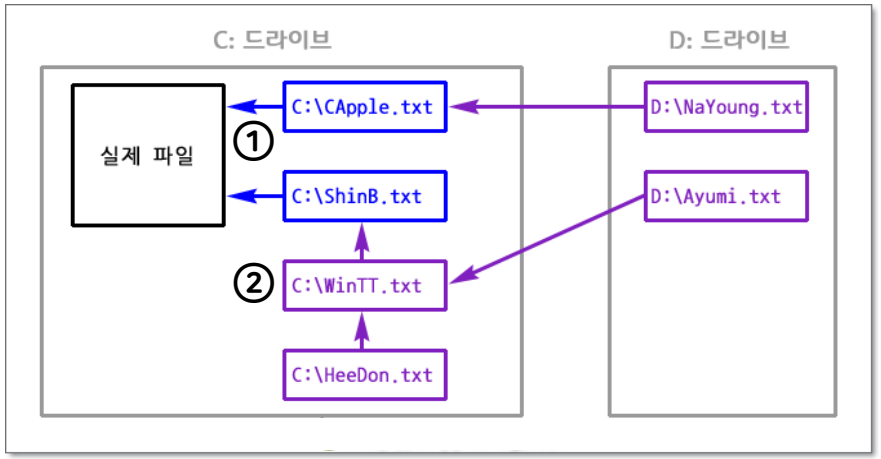
⑥ 하드 링크(Hard Link), 심볼릭 링크(Symbolic Link), 정션(Junction) 지원

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)

  - NTFS의 특징 : 하드 링크(Hard Link), 심볼릭 링크(Symbolic Link)
  - 윈도우에서 보는 모든 파일은 실제 파일에 대한 파일 시스템의 링크(Link)
  - 실제 파일과 직접 연결된 링크를 하드 링크(Hard Link)라고 부르며, 링크에 연결된 링크는 소프트 링크(Soft Link)라고 표현,
    소프트 링크의 정확한 명칭은 심볼릭 링크(Symbolic Link)
  - 하드 링크는 실제 파일에 직접 연결해야 하기 때문에 실제 파일이 위치한 동일 드라이브에서만 가능하며,
    심볼릭 링크는 대상만 설정하면 되는 일종의 바로 가기 연결이기 때문에 동일 드라이브는 물론 다른 드라이브로도 연결이 가능





① 하드 링크

② 심볼릭 링크

윈도우 시스템

- ## 윈도우 구조 - 파일 시스템 : NTFS(New Technology File System)

  - NTFS의 특징 : 바로 가기, 정션(Junction)

  - 하드 링크(Hard Link), 심볼릭 링크(Symbolic Link)가 바로가기와 다른 점

    바로 가기는 .lnk 라는 완전히 별개의 파일(해당 .lnk 의 실제 파일도 가지고 있는)이 따로 존재하고 윈도우의 익스플로러에서 지정된 파일로 이동을 시켜주는
    하나의 파일 형식을 의미하지만, 하드 링크나 심볼릭 링크는 파일 시스템 차원에서 지원하는 기능으로 생성된 파일은 그 자체로 개별적인 파일처럼 취급

  - 정션(Junction)과 심볼릭 링크(Symbolic Link)가 다른 점

    폴더에서는 윈도우 XP 시절 부터 정션(Junction)이라는 개념이 사용되고 있었고 현재까지도 계속 사용되고 있음
    정션이 가지고 있는 특성이나 개념은 심볼릭 링크와 크게 다르지 않지만, 심볼릭 링크는 상대 경로의 사용이 가능하지만 정션은 지정할 수 없으며
    심볼릭 링크는 네트워크 드라이브로의 링크가 가능하지만 정션은 불가능하다는 것 정도가 다름
    정션을 폴더에 대한 구형 소프트 링크로 보고, 심볼 링크를 폴더에 대한 신형 소프트 링크로도 볼 수 있음

윈도우 시스템

- ## 윈도우 구조 - 윈도우(XP, 윈도우 서버200/2003)의 부팅 순서

- 1단계 : POST(Power On Self Test)의 실행

  BIOS(Basic Input/Output system)에서 POST를 실행

- 2 단계 : 기본 부팅 관련 설정 사항 로드

  BIOS는 CMOS에 설정되어 있는 시스템 설정 사항 및 부팅과 관련된 여러 가지 정보를 읽어 시스템에 적용

- 3단계 : MBR(Master Boot Record, 마스터 부트 레코드) 로드

  저장 매체의 첫 번째 섹터(LBA 0)에 위치하는 512바이트의 영역, 'Missing operating system'은 운영체제 미설치나 CMOS 설정 오류 시 표시

- 4 단계 : NTLDR(NT Loader) 실행

  윈도우가 부팅될 수 있도록 간단한 파일 시스템을 실행하고 boot.ini 파일의 내용을 읽어 가능한 부팅 옵션을 표시

- 5 단계 : NTDETECT.com 실행

  하드웨어 검사(PC의CPU 유형, 버스 유형, 비디오 보드 유형, 키보드와 마우스 종류, 직렬 포트와 병렬 포트, 플로피 등)

윈도우 시스템

- ## 윈도우 구조 - 윈도우(XP, 윈도우 서버200/2003)의 부팅 순서

- 6단계 : ntoskrnl.exe(NT OS Kernel) 실행, HAL.DLL(Hardware Abstraction Layer) 로드 단계

① 커널 로드 : 시스템 설정을 로드 하여 HKEY_LOCAL_MACHINE₩ System₩CurrentControlset₩Services 에 저장하고
이 정보를 확인하여 로드 할 드라이브와 그 순서를 결정

② 커널 초기화 : 드라이브에 대한 현재의 제어 설정을 검사하고 작업을 시작

③ 서비스 로드 : 세션 관리자 서브 시스템(smss.exe)과 Win32 서브 시스템을 로드

④ 서브 시스템 시작 : 윈도우 서브 시스템 초기화
Win32 서브 시스템은 로그인을 처리하고 Winlogon.exe를 시작
로그인 창을 통해서 계정과 패스워드 입력 받아 로컬 보안 인증 서버(Local Security Authentication Server, Lsass.exe)에 전달
로컬 보안 인증 서버는 보안 계정 관리자 (Security Accounts Manager, SAM)에 저장된 정보와 비교하여 일치 여부 확인
Userinit.exe 프로세스가 HKEY_LOCAL_MACHINE₩SOFTWARE₩Microsoft₩Windows NT₩CurrentVersion₩Winlogon에 등록된 쉘을 실행

윈도우 시스템

- **윈도우 구조 - 윈도우(Vista 이후)의 변경된 부팅 순서**

- 1~3단계 : POST 실행/기본 부팅 관련 설정 사항 로드/MBR 로드

- 4단계 : Window Boot Manager 실행

  MBR에서 NTLDR을 실행하지 않고 Window Boot Manager(bootmgr.exe)를 실행

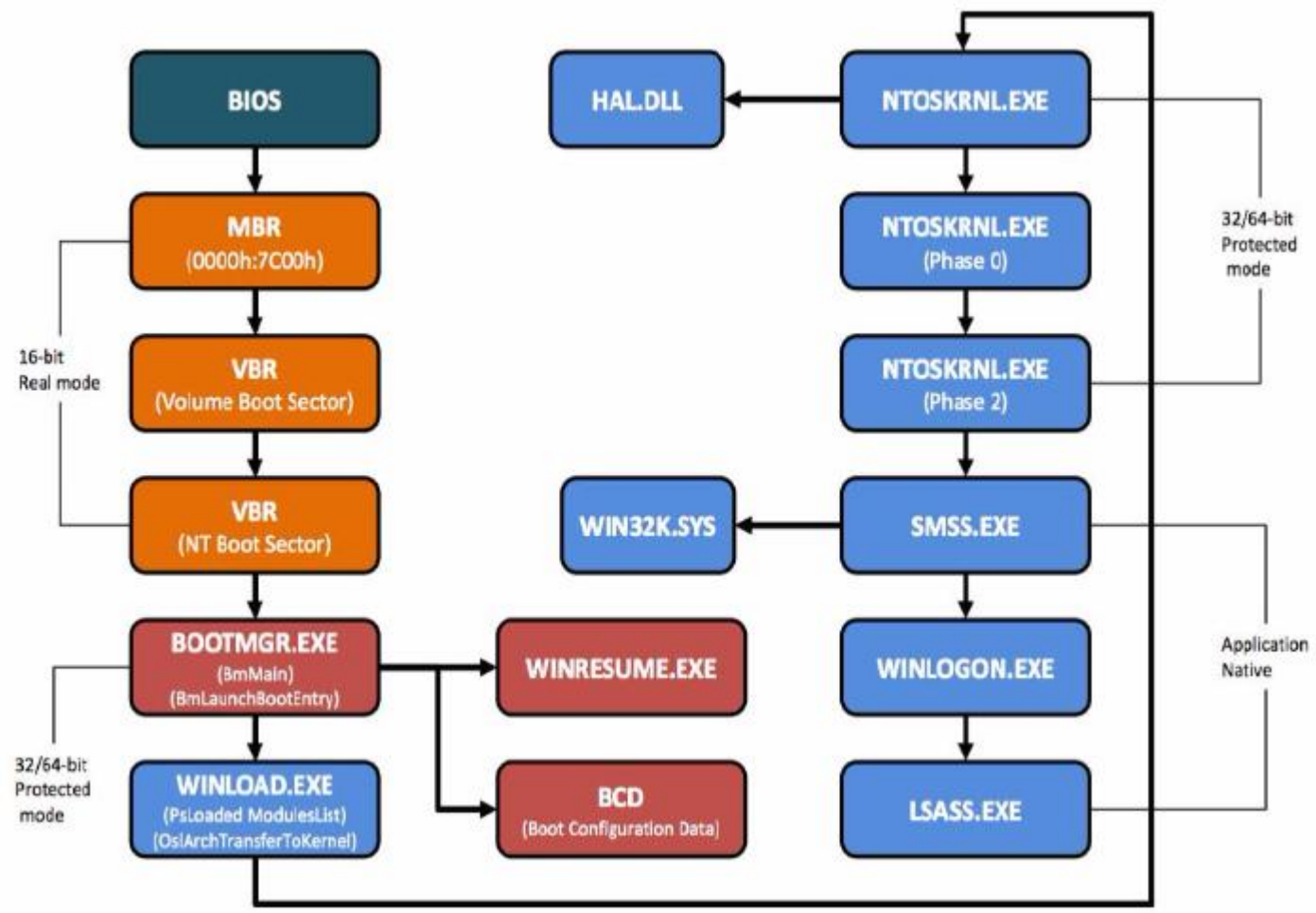
  부트 설정 데이터(BCD, Boot Configuration Data)를 읽어 실행 가능한 운영체제 목록 보여줌(bcdedit.exe 이용 편집 가능)

- 5단계 : Windows OS Loader(Winload.exe) 실행

  NTDETECT와 같이 각종 장치 드라이버를 로드하고, ntoskrnl.exe 실행

윈도우 시스템

- 윈도우 구조 – 윈도우(Vista 이후)의 변경된 부팅 순서



- MBR

  Master Boot Record
- VBR

  Volume Boot Record
- SMSS

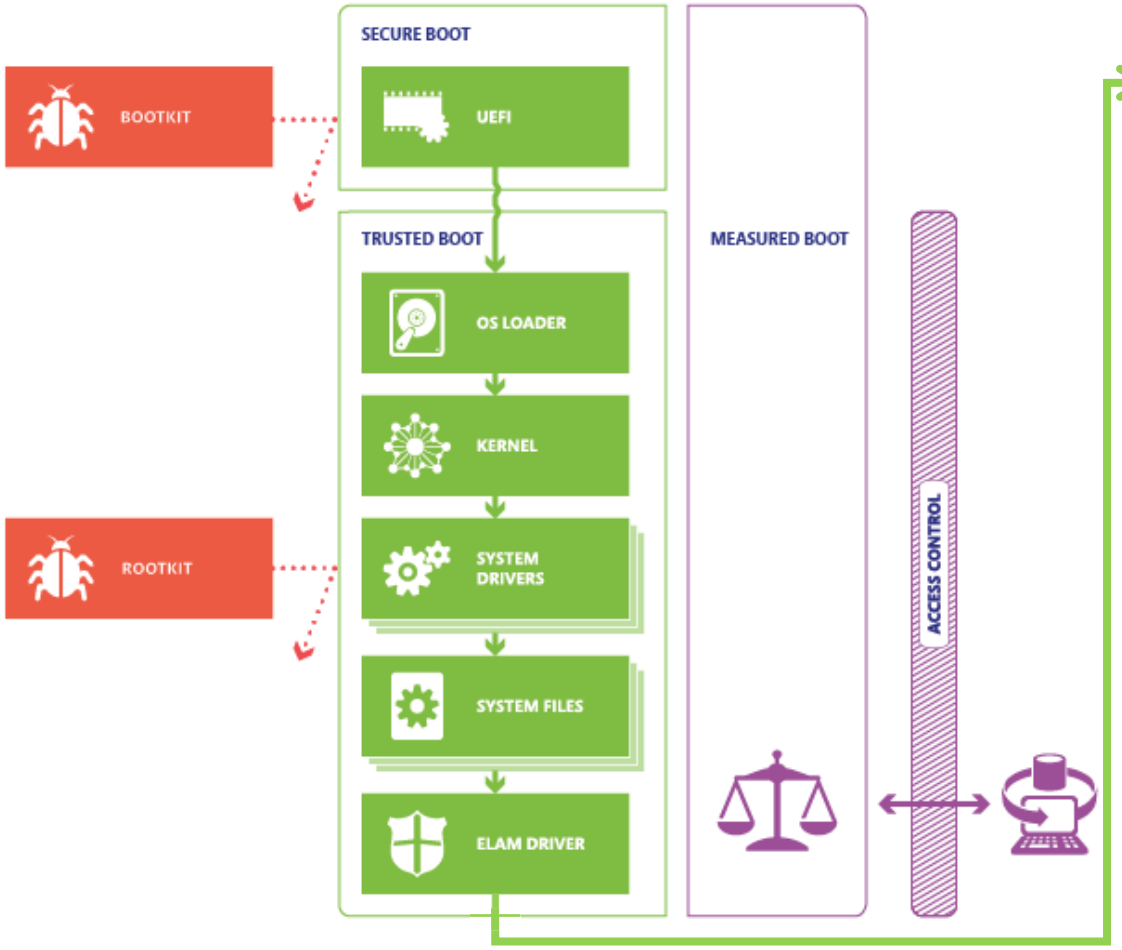
  Session Manager Subsystem
- LASS

  Local Security Authentication Server

윈도우 시스템

- **윈도우 구조 - 윈도우 부팅 프로세스 보호(Windows 8.1, 10, 11)**

  - 윈도우는 시작 프로세스 중에 루트킷 및 부트킷이 로드되지 않도록 하는 네 가지 기능을 지원



① **보안 부팅 :**
UEFI 펌웨어 및 TPM(Trusted Platform Module, 신뢰할 수 있는 플랫폼 모듈)이 있는 컴퓨터는 신뢰할 수 있는 OS 부팅 로더 만 로드 하도록 구성할 수 있음

② **신뢰할 수 있는 부팅 :**
윈도우는 로드하기 전 시작 프로세스의 모든 구성 요소의 무결성 확인

③ **ELAM(멀웨어 방지 조기 실행, Early Launch AntiMalware) :**
로드하기 전에 모든 드라이버를 테스트하여 승인되지 않은 드라이버 로드 방지

④ **계획 부팅 :**
PC의 펌웨어는 부팅 프로세스를 기록(PC의 UEFI 펌웨어는 TPM에 펌웨어, 부팅 로더, 부팅 드라이버 및 멀웨어 방지 앱 전에 로드되는 모든 해시를 저장)하며 윈도우는 이 로그와 다른 보안 정보를 신뢰할 수 있는 다른 서버로 전송하여 PC가 정상인지 검증

윈도우 시스템

- ## 윈도우 구조 - TPM(Trusted Platform Module, ISO/IEC 11889)

  - TPM(신뢰할 수 있는 플랫폼 모듈)은 보안 암호화 프로세서를 위한 국제 표준

- TPM은 기본적으로 통합 암호화 키를 사용하여 컴퓨터에서 하드웨어 기반 보안 기능을 구현할 수 있는 보안 칩으로 마더보드 또는 CPU에 추가됨
  인텔은 PTT(플랫폼 신뢰 기술), AMD는 fTPM(펌웨어 TPM)이라는 이름으로 이를 제공

- 최신 TPM 버전은 2.0으로 Windows 11은 TPM 2.0을 요구

- TPM 칩은 용도가 광범위하며 주로 장치 식별, 인증, 암호화 및 장치 무결성 검증에 사용할 수 있음
  플랫폼 무결성 : TPM의 주요 범위는 운영 체제에 관계없이 모든 컴퓨터 장치의 무결성을 보장하는 것
  TPM 사용의 무결성을 보장하는 책임은 펌웨어(Unified Extensible Firmware Interface) 및 운영 체제
  하드 디스크의 모든 파티션 암호화 : TPM 기술로 모든 하드 디스크 파티션을 암호화할 수 있는데, BitLocker에서도 이를 사용
  비밀번호 저장 및 관리 : 운영 체제는 일반적으로 키, 데이터 또는 시스템을 보호하기 위해 인증(암호 또는 기타 방법 포함)이 필요한데,
  키가 칩에 고정된 메모리 셀에 저장되며 전원이 차단되어도 정보가 손실되지 않음
  BIOS 관리 암호와 비교할 때 TPM 보안 칩은 훨씬 더 안전

리눅스 시스템

리눅스 시스템

- **유닉스의 역사**

  - 유닉스의 시초는 멀틱스(Multics)라는 시분할 운영체제이며,
    리눅스는 1983년부터 리처드 스톨만이 개발한 Unix-like 오픈소스(GNU) 운영체제로 정식 버전은 1991년 10월에 'Copy Left' 원칙에 의해 배포

  - 1963년부터 3년 동안 MIT, GE(General Electric), 벨 연구소가 공동으로 멀틱스(Multics) 개발.
    국방성(ARPA)의 지원으로 추진된 프로젝트로 GE가 하드웨어를 만들고 PL/1(Programming Language 1) 언어로 제작

  - 데니스 리치(Dennis Ritchie)는 파견지에서 돌아온 뒤 켄 톰슨(Ken Thomson)과 함께 PDP- 7을 만들기 시작하며 멀틱스의 여러 개념 구현
    PDP-7에는 파일 시스템 구성의 개념, 사용자가 명령을 내려 바로 실행하는 명령어 인터프리터(command interpreter)의 개념,
    각 명령이 새로운 프로세스를 형성해서 실행하도록 하는 개념이 모두 포함

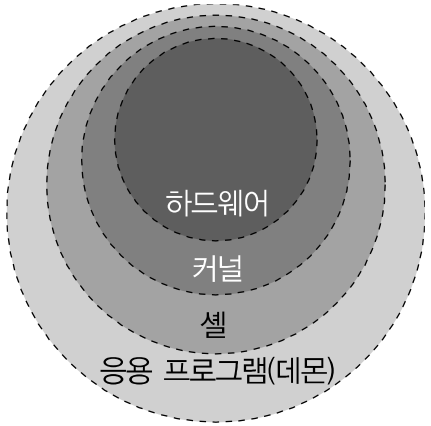
  - AT&T 연구소로부터 유닉스 소스 코드를 400달러에 구입
    빌 조이(Bill Joy)와 척 핼리(Chuck Haley)는 BSD(Berkeley Software Distribution)라는 이름을 붙여 당시 50달러 가격으로 판매

  - 유닉스의 POSIX라는 인터페이스를 이용하여 표준화(특정 유닉스에서 개발한 소프트웨어가 다른 유닉스에서도 잘 작동하도록 하기 위함)
    POSIX.1(핵심부, 운영체제 인터페이스), POSIX.2(셸과 툴, 사용자 명령 등), POSIX.3(표준 규격의 적합성 검증 법), POSIX.4(실시간 POSIX 확장),
    POSIX.5(에이다 언어), POSIX.6(보안 기능) 등이 있음

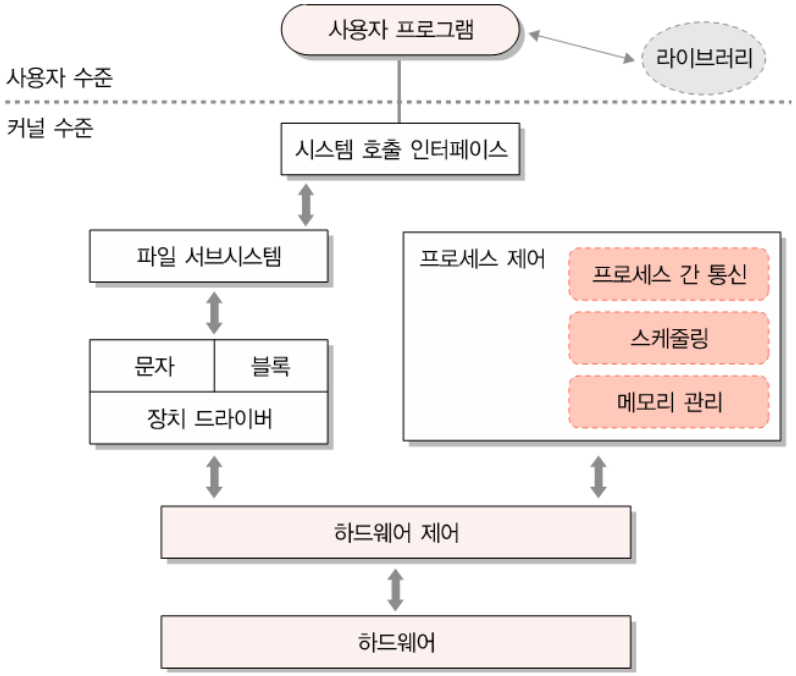
리눅스 시스템

- **리눅스의 구조**

  - 리눅스는 윈도우보다 훨씬 단순한데, 크게 파일 서브 시스템, 장치 드라이버, 프로세스 제어로 나뉘며, 커널의 크기 또한 윈도우의 1/3 정도



리눅스 링 구조



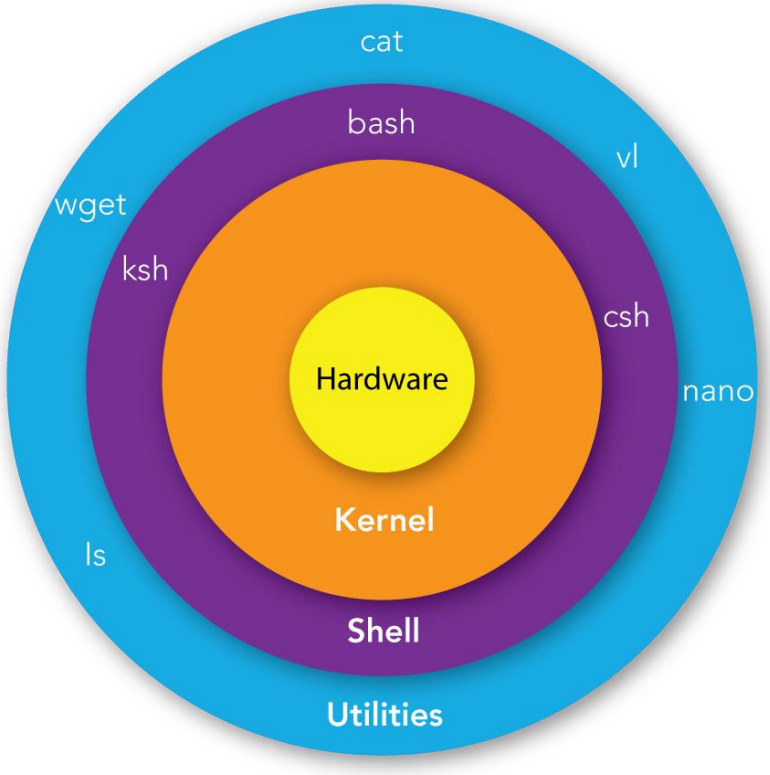
리눅스 시스템 구조

- 프로세스 제어에서는 전체 프로세스 간 통신, 스케줄링, 메모리 관리를 구현
- 장치 드라이버는 윈도우에서처럼 하드웨어와 소프트웨어를 연결해주는 인터페이스를 제공
- 파일 서브 시스템은 하드 디스크와 같은 저장 공간에 파일을 저장하고 읽는 역할

리눅스 시스템

- **리눅스의 구조 – 쉘(Shell)**

  - 응용 프로그램에서 명령을 받아 커널에 전송하는 역할, 사용자의 키보드 입력 인식 해당 프로그램을 수행



- **쉘(Shell)의 종류**

  - 본 쉘(Bourne Shell), 콘 쉘(Korn Shell), C 쉘(C Shell)

- **셸이 제공하는 주요 기능**

  - 자체의 내장 명령어 제공
  - 입력/출력/오류의 방향 변경(redirection)
  - 와일드카드(wildcard)
  - 파이프라인
  - 조건부/무조건부 명령 열 작성
  - 서브 셸 생성
  - 백그라운드 처리(Background processing)
  - 셸 스크립트(프로그램) 작성

리눅스 시스템

- 리눅스의 구조 – 파일(File)

  - 리눅스의 파일은 크게 아래 네 가지로 구분

  - 일반 파일 : 데이터 파일이나 실행

  - 디렉터리 : 유닉스에서는 디렉터리도 파일에 해당, 디렉터리가 담고 있는 여러 파일과 하위 디렉터리 정보를 포함

  - 특수 파일 : 프린터나 터미널, 테이프 드라이버 같은 물리적인 장치를 특수 파일을 통해 접근, 특수 파일은 /dev(device)에 있음

  - 파이프 파일 : | 문자를 말하며, 2개의 명령을 연결 시 사용, 임시 파일이 생성 되었다가 명령 수행을 마치면 사라지는 것으로,
    이 파일을 파이프 파일이라고 함 (ex: ps –ef | grep hugo)

리눅스 시스템

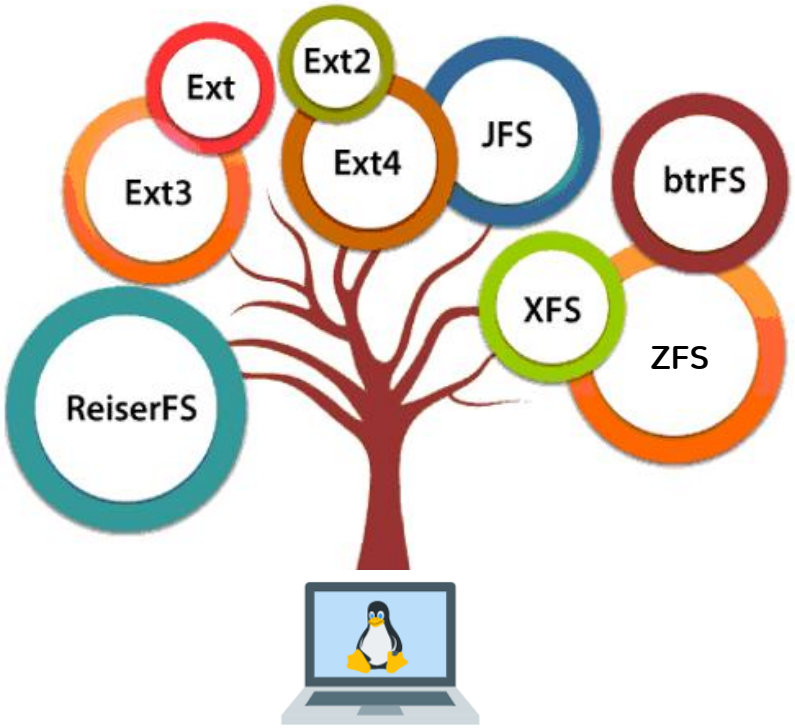
- 리눅스의 구조 – 디렉토리(Directory)

디렉터리	내용
/bin	기본적으로 실행 가능한 파일을 담고 있음(echo, mv, copy, pwd, who 등)
/etc	시스템의 환경 설정 및 주요 설정 파일을 담고 있음(passwd, hosts, xined.conf 등)
/tmp	프로그램 실행 및 설치 시 생성되는 임시 파일을 담고 있으며, 재부팅 시 임의로 삭제될 수 있음
/lib	기본 프로그램의 모듈을 담고 있음
/boot	커널을 위한 프로그램 파일을 담고 있으며, 부팅할 때 수행됨
/dev	프린터나 터미널 같은 물리적인 장치를 다루기 위한 특수 파일을 담고 있음
/home	각 사용자의 작업 디렉토리를 담고 있으며, 각 계정으로 로그인할 때 이 디렉토리 하위에 자신의 작업 디렉터리가 시작 디렉토리가 됨
/usr	사용자가 직접 쓰는 파일을 담고 있으며, 다른 디렉토리에 있는 파일이 똑같이 위치하고 있는 것이 많지만 이는 링크되어 있는 것임
/usr/lib	C언어나 포트란의 라이브러리를 담고 있음
/usr/include	C언어에 사용되는 헤더 파일을 담고 있음

시스템에서의
각 디렉토리의 역할

리눅스 시스템

- 리눅스의 구조 – 파일 시스템(File System)



리눅스에서 사용하는 다양한 파일 시스템들

리눅스 시스템

- **리눅스의 구조 – 파일 시스템(File System) : EXT(EXTended file system, 확장 파일 시스템)**

  - EXT(EXTended file system)의 특징

    호환성 : 유닉스의 파일 시스템인 UFS(Unix File System)의 영향을 받아 만들어졌으며,
    리눅스 초기에 사용되던 시스템으로 EXT2의 원형, 현재는 사용되지 않음

    제한 : 2G의 데이터와 255자의 파일명 사용 가능

  - EXT2(EXTended file system 2)의 특징

    호환성 : 고용량 디스크 사용을 염두하고 설계된 파일 시스템으로 호환과 업그레이드가 쉬움

    제한 : 2TB 파일 크기까지 지원하고 최대 볼륨 크기는 32TB

  - EXT3(EXTended file system 3)의 특징

    호환성 : EXT2에 있었던 고질적인 문제 자료 손실 문제를 저널링을 통해 해결
    EXT2보다 파일시스템의 보안 기능이 크게 향상(ACL을 통한 접근 제어 지원)

    제한 : 파일 최대 크기는 2TB이고 최대 볼륨 크기는 32TB

  - EXT4(EXTended file system 4)의 특징

    호환성 : 64비트로 공간 제한을 없애고 EXT3의 성능을 향상시킨 대형 파일 시스템
    (EXT4의 중요한 새로운 특징인 extents를 사용한다면 EXT3로 마운트는 불가능)

    제한 : 16TB 파일 크기까지 지원하고 최대 볼륨 크기는 1EB(엑사바이트, 10^18)

구분	ext	ext2	ext3	ext3cow	ext4
도입연도	1992년	1993년	1999년	2003년	2006년
최대 볼륨크기	?	32 TB	32 TB	32 TB	1 EB
최대 파일명 길이	?	255 바이트	255 바이트	255 바이트	255 바이트
최대 파일크기	?	2 TB	2 TB	2 TB	16 TB
최대 파일 수	?	10^{18}	다양함	다양함	40억
타임스탬프 단위	?	1초	1초	1초	1나노초
타임스탬프 범위[6]	?	2038-01-18	2038-01-18	2038-01-18	2514-04-25
저널링	·	·	O	O	O

EXT 파일 시스템 간의 비교

리눅스 시스템

- 리눅스의 구조 – 파일 시스템(File System) : 기타

  - JFS(Journaling File System)의 특징

    IBM사의 독자적인 저널링 파일 시스템으로 GPL로 공개하여 현재 리눅스용으로 개발

    파일 시스템의 규모에 비하면 부하가 적다는 장점이 있어서 예전엔 사용되었지만, 현재는 많이 사용되지 않음

  - XFS(eXetended File System)의 특징

    SGI가 개발하여 자사 운영체제에 사용하다가 2000년에 소스를 공개하면서 리눅스에 이식된 파일 시스템

    64bit 파일 시스템으로 8EB까지의 대용량 파일도 다룰 수 있음

  - ReiserFS의 특징

    독일의 한스 라이저가 개발한 파일 시스템으로 범용 목적의 저널링 파일 시스템

    모든 파일 객체들을 B트리에 저장, 간결한 색인화 된 디렉터리 지원

리눅스 시스템

- 리눅스의 구조 – 파일 시스템(File System) : 기타

  - btrFS(B-tree file system 또는 Butter file system)의 특징

    오라클, 후지쯔, 레드햇에서 개발하는 유닉스/리눅스용 파일 시스템

    압축기능을 제공하고 실시간 오류 정정 기능과 스냅샷을 이용하여 볼륨 복원이 가능하여 장애 복원성이 좋음

    최대 파일 크기는 16 EB이고 최대 볼륨 크기도 16 EB

  - ZFS(Zettabyte File System)의 특징

    썬마이크로시스템즈에서 개발한 파일 시스템으로 솔라리스에 탑재되어 사용되었고,

    썬마이크로시스템즈가 오라클에 인수되면서 오라클이 ZFS를 클로즈드 소스로 전환 (OpenZFS라는 오픈소스 프로젝트로 분리)

    파일시스템들 가운데 최초로 128비트 파일 시스템을 적용하여 거의 무한대의 용량을 제공

    파일시스템 자체에서 볼륨 매니저 기능을 포함하여 시스템 내에 있는 하드 디스크들을 구성하거나 스토리지 풀로 통합하여 사용

리눅스 시스템

- **리눅스의 구조 – 리눅스 부팅 순서**

  - 1단계 : POST(Power On Self Test) 수행

  - 2단계 : 기본 부팅 관련 설정사항 로드

  - 3단계 : MBR(Master Boot Record, 마스터 부트 레코드) 로드

  - 4단계 : 부트 로더(Boot Loader) 실행

    ① LILO(Linux Loader)와 GRUB(Grand Unified Bootloader, GNU 프로젝트의 부트 로더) 사용

    ② LILO는 /etc/lilo.conf에 설정

    ③ GRUB은 /etc/grub.conf (=/boot/grub/grub.conf)에 설정

  - 5단계 : 실행 레벨에 따른 서비스 실행

    ① 부트 로더는 스와퍼(Swapper)라는 pid 0번 프로세스를 실행, 스와퍼는 다시 pid 1번 init(/sbin/init) 프로세스를 실행

    ② init 프로세스는 다시 /etc/inittab 파일을 읽음

    ③ inittab 파일은 부팅할 기본 모드를 선택하여 그에 따른 환경을 제공하는 분기점

리눅스 시스템

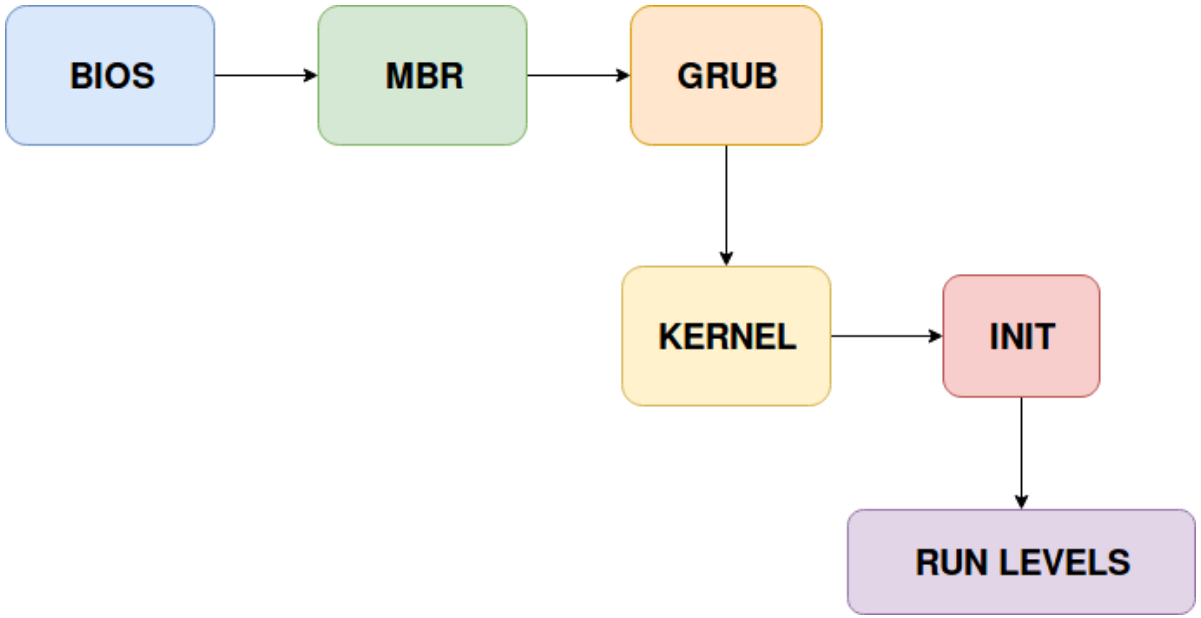
- 리눅스의 구조 – 리눅스 부팅 순서

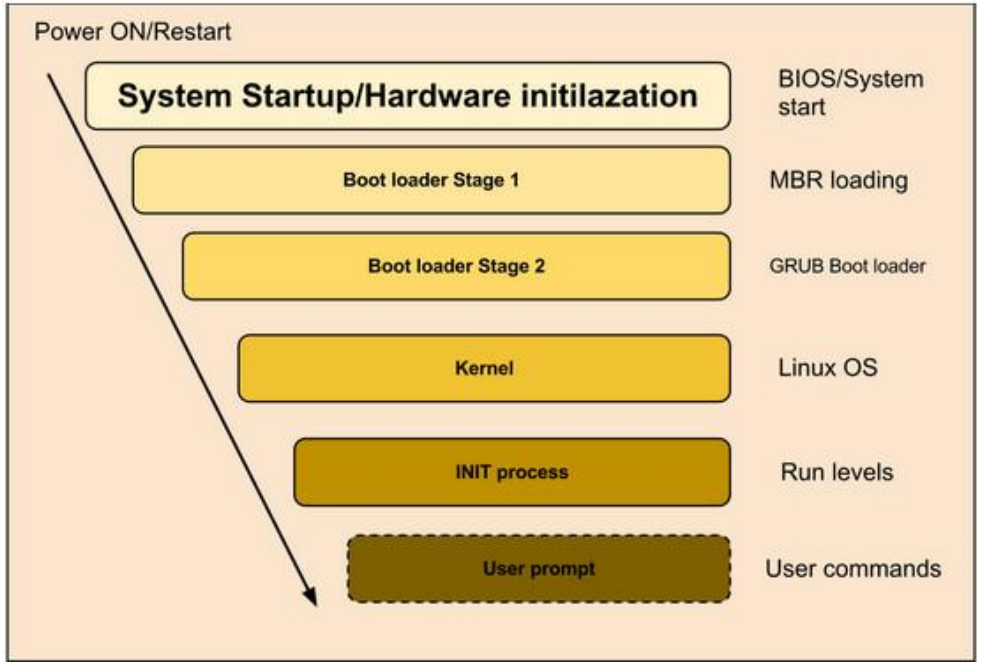
- initttab 파일 : 실행 레벨(Run Level), 7개의 실행 레벨이 있음

실행 레벨 0 : 시스템을 종료할 때 사용한다.
실행 레벨 1 : 단일 사용자 모드(Single User Mode)로, 기본적으로 관리자 권한의 셸을 얻는다.
그러나 대부분 데몬이 실행되지 않으므로 기능은 제약되어 있다.
실행 레벨 2 : NFS(Network File System)를 지원하지 않는 다중 사용자 모드다.
실행 레벨 3 : 일반 쉘 기반의 인터페이스를 가진 다중 사용자 모드다.
실행 레벨 4 : 기본적으로 사용되지 않지만 사용자가 임의로 정의하여 사용할 수 있다.
실행 레벨 5 : 기본은 실행 레벨 3과 같으나 GUI 환경을 지원한다.
실행 레벨 6 : 재부팅

리눅스 시스템

- 리눅스의 구조 – 리눅스 부팅 순서





- GRUB(Grand Unified Bootloader)
  GRUB은 GNU 프로젝트의 부트 로더로 대부분 리눅스 배포판에서 부트 로더로 사용

보안 운영체제

보안 운영체제

- **보안 운영체제(Secure OS)**
  - 신뢰성 운영체제(Trusted OS)로도 불리며 컴퓨터 운영체제 상에 내재된 보안상의 결함으로 인하여 발생할 수 있는 각종 해킹으로부터 시스템을 보호하기 위하여 기존의 운영체제 내에 보안 기능을 추가한 운영체제

  - **필요성**

    컴퓨터 운영체제 상에 내제된 보안 취약점에 대한 대응 및 보완, 내외부의 침입에 대한 서버 운영체제 자체 보호 기능 필요

  - **구조**

    보안 커널이 운영체제 내부에 위치, 운영체제가 제공하지 못하는 보안 기능을 커널 자체적으로 제공

    설치 시 성능이 30%까지 저하되므로 추가 메모리 및 CPU 확보 필요

  - **1970년대 미 공군이 컴퓨터를 이용한 군사 비밀 처리 시 강화된 접근제어 기능(다중등급 보안 기능)이 부가된 보안 커널을 연구한 것이 시초**

    국내에서는 ETRI(한국전자통신연구원)에서 1987년부터 연구를 시작

    국산 보안 운영체제 : Secuve TOS(시큐브), RedOwl(티에스온넷 > 하우리), RedCastle(SGA)

보안 운영체제

- **보안 운영체제(Secure OS)**
  - 신뢰성 운영체제(Trusted OS)로도 불리며 컴퓨터 운영체제 상에 내재된 보안상의 결함으로 인하여 발생할 수 있는 각종 해킹으로부터 시스템을 보호하기 위하여 기존의 운영체제 내에 보안 기능을 추가한 운영체제

  - 보안계층을 파일 시스템과 디바이스, 프로세스에 대한 접근권한 결정이 이루어지는 커널 레벨에 통합한 운영체제

  - 사용자에 대한 식별 및 인증, 강제적 접근 통제, 임의적 접근 통제, 재사용 방지, 침입 탐지 등의 보안 기능 요소를 갖춘 운영체제

  - 보안 운영체제의 목적

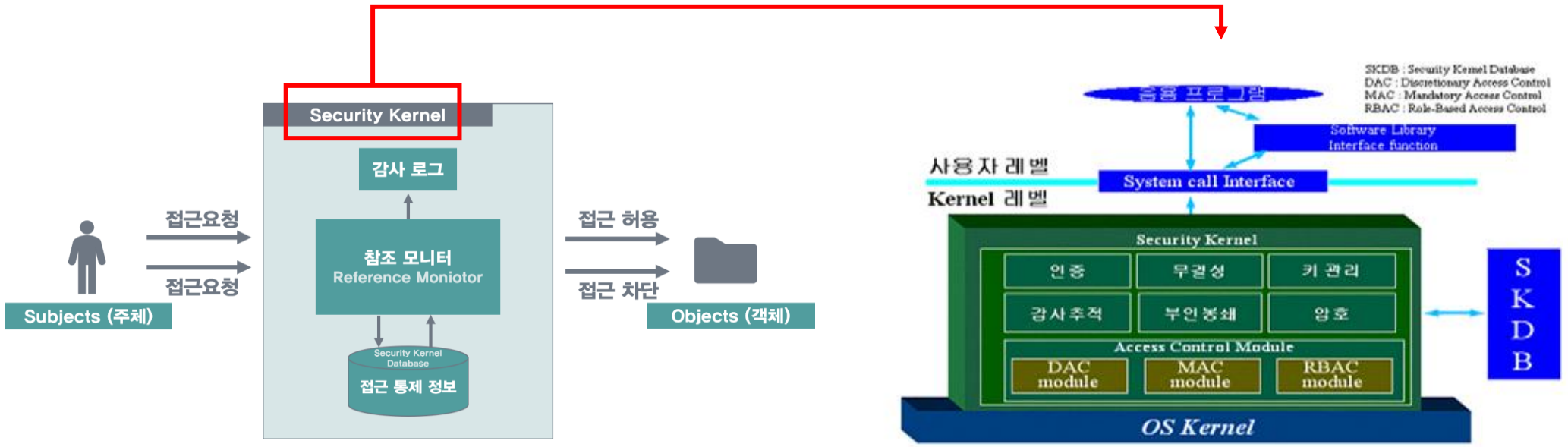
    ① 안정성 : 중단 없는 안정적인 서비스를 지원

    ② 신뢰성 : 중요 정보의 안전한 보호를 통한 신뢰성 확보

    ③ 보안성 : 주요 핵심 서버에 대한 침입차단 및 통합 보안 관리, 안전한 운영체제 기반 서버보안 보호대책 마련, 버퍼오버 플로우, 인터넷 웜 등 다양해지는 해킹 공격을 효과적으로 방어할 수 있는 서버 운영환경 구축

보안 운영체제

- 보안 운영체제(Secure OS)



- 보안 운영체제(Secure OS)에서 중요하게 알아 두어야 할 핵심 구성요소는 보안 커널과 참조모니터

① 보안 커널(Security Kernel) : 접근통제를 위한 기본적 보안 절차를 구현한 하드웨어 펌웨어 or 소프트웨어 요소

② 참조모니터(Reference Monitor) : 객체 접근통제 기능을 수행하는 커널의 핵심

보안 운영체제

- **보안 운영체제(Secure OS) – 주요 기능**

  - 식별 및 인증, 계정 관리 : 고유한 사용자 신분에 대한 인증 및 검증, Root의 기능 제한 및 보안관리자와 권한 분리
  - 접근 통제(강제적 접근통제, 임의적 접근통제) : 사용자의 접근 권한 통제
  - 완전한 중재 및 조정 : 모든 접근 경로에 대한 완전한 통제
  - 메모리 재사용 방지 : 메모리 사용 후 기억장치 공간 초기화
  - 안전한 경로 : 보안, 작업의 안전한 수행 경로 제공
  - 감사 및 감사기록 축소 : 보안관련 사건 기록의 유지 및 감사기록의 보호, 막대한 양의 감사기록에 대한 분석 및 축소
  - 보안 커널 변경방지 : 보안 커널의 관리기능과 시스템의 분리, 시스템의 루트 권한으로 접근하더라도 보안 커널의 변경 방지
  - 해킹 방지(Anti-Hacking) :

    커널의 참조 모니터를 통해 알고리즘에 의해서 해킹을 근원적 탐지 및 막을 수 있음

    BOF, Format String, Race Condition, Process Trace, Root Shell 등의 해킹 기법에 대한 직접적 대응

    Remote/Local Attack 대응, 알려지지 않은 Worm 대응

    해킹의 즉각적 탐지/실시간 차단/실시간 경보

Q A