리눅스 기본 개념

- description : 리눅스 기본 개념
- author : 치킨5분컷
- email : hgkim@repia.com
- firstupdate : 2022-12-15 Thu
- lastupdate : 2022-01-09 Mon

리눅스 개념 정리

Linux란

- 대표적인 오픈소스 운영체제(OS)이고, CPU, 메모리, 스토리지처럼 시스템의 하드웨어와 리소스를 직접 관리하는 소프트웨어. OS는 애플리케이션과 하드웨어 사이에서 모든 소프트웨어와 작업을 수행하는 물리적 리소스를 연결함

리눅스의 구조

- 응용프로그램에서 사용가 명령을 내리면 명령어 해석기인 셸이 해석된 사용자 명령어를 커널에 전달함. 커널은 하드웨어를 제어하는 코드를 통해 소프트웨어와 커뮤니케이션을 하며 역할을 수행함

리눅스의 특징과 종류

- 유닉스 운영체제를 기반으로 높은 안정성과 보안성, 높은 신뢰성과 성능이 특징임
- 시스템의 자원을 효율적으로 관리하고 사용가능
- 멀티 유저와 멀티 태스킹을 지원함
- 대부분의 리눅스는 CLI(명령어창)와 GUI(그래픽)를 모두 지원
- 다양하고 강력한 네트워킹 기능 덕분에 서버 OS로 적합
- 성능이 낮은 PC에서도 작동
- 파일의 확장자는 시스템에 아무런 상관이 없음(더 보기 쉽게 하기 위함)

활용 분야

- 라우터나 AP와 같은 곳에서 사용하는 네트워크 서버장비, 라즈베리파이와 같은 임베딩시스템과 IoT, TV 셋톱박스, 정밀 의료기기, 리눅스 개발 서버 등

권한(sudo)

- 패키지 설치와 같은 활동에서 나타나는 permission denied라는 메시지가 나타나며 활동이 제한 될 때가 있음. 이럴때 sudo를 써서 관리자 권한 획득 가능

리눅스 파일 시스템

- 파일 시스템이란, 저장 장치 내에서 데이터를 읽고 쓰기 위해 미리 정한 약속을 말함
- 대부분의 파일 시스템은 디렉토리와 파일 형태로 구성
- 리눅스의 파일 시스템은 root 파일 아래에 계층적으로 모든 파일과 디렉토리가 만들어짐

파일시스템의 종류

- FAT(File Allocate Table) : 파일 할당 테이블이라고 말하며 카메라의 메모리카드나 수많은 컴퓨터 시스템에 널리 쓰이는 파일 시스템의 종류
- 장점으로는 높은 호환성
- 단점으로는 단순한 자료구조때문에 작은 파일이 여러개 있을 경우 공간 활용을 제대로 하지 못한다는 단점이 있음
- NTFS(New Technology File System) : FAT구조를 대체하기 위해 만들어짐
- 시스템 고장 및 손상시, 디스크 볼륨을 재구성하여 일관성있는 상태로 복구가 가능하여 안정성이 높고, 보안성도 FAT보다 향상된 파일 시스템
- EXT(EXTended file system) : 확장 파일 시스템의 준말로 리눅스의 기본 파일 시스템

mount

- 물리적인 저장장치를 디렉토리에 연결시켜주는것 ex) 하드,USB등의 보조기억장치를 연결하면 자동으로 디렉토리에 연결
- USB를 꽂자마자 사용할 수 있게되는 것을 PnP라고 하며, **리눅스**의 경우 PnP 기능이 작동하지 않음
- 때문에 보조기억장치를 설치했을때 mount작업을 수행 필요
    - mount된 디스크 정보 출력 - df
    - mount 해제 - remount [device] [directory]

리눅스 프로세스

- 프로세스란, 시스템에서 메모리에 적재되어 실행되고 있는 모든 프로그램을 말함

프로세스의 특징

- 모든 프로그램은 실행될 때 하나 이상의 프로세스를 갖음
- 병행적으로 실행이 가능
- 부모(PPID), 자식(fork를 통해 복사된 것) 프로세스가 있게 됨
- 커널(kernel)에 의해 관리
- 모든 프로세스에는 소유자(리눅스 계정)가 있음
- 프로세스마다 식별을 위한 ID(PID)가 부여

대소문자 구별

- Windows에서는 파일/디렉터리의 대소문자를 구별하지 않음.(test.txt=TEST.txt)
- Linux에서는 대소문자를 구별함. (test != TEST)

커널에 의해 수행되는 task

- 프로세스 스케쥴링 - 여러 프로세스가 동시에 메모리에 존재하며 어떤 프로세스가 CPU 자원을 얼마나 할당받을지를 관리하는것을 프로세스 스케쥴링이라고 함
- 메모리 관리 - 메모리는 커널에 의해 여러 프로세스에서 공유하며 사용되는 한정적 자원
- 리눅스도 가상 메모리를 통해 메모리를 관리하고 독립성을 보장
- 프로세스의 일부분만 메모리에 보관하여 메모리에 좀 더 많은 프로세스들이 상주할 수 있도록 한다는 장점을 제공
- 파일시스템 제공 - 커널은 디스크에 대한 파일시스템을 제공하며 파일이 생성, 추출, 업데이트, 삭제 될 수 있도록 함
- 프로세스 생성과 종료 - 새로운 프로그램을 메모리에 로드하고 실행에 필요한 리소스를 제공
- 실행되는 프로그램은 프로세스에 해당되는데, 프로세스가 실행을 끝내면, 커널이 리소스를 거둬들여 다른 프로세스가 사용할 수 있도록함
- 디바이스에 대한 접근 - 커널은 인터페이스가 포함된 프로그램을 제공하여 디바이스에 대한 접근을 단순화하고 표준화
- 네트워킹 - 사용자의 프로세스를 대신해 네트워크 메시지의 송수신을 담당

리눅스 디렉토리 구조

- Unix 디렉토리처럼 Tree 모양의 계층 구조를 가지고 있으며 이를 FHS(Filesystem Hierarchy System)라고 함
- 최상위 디렉토리 - /
- 모든 디렉토리들의 시작점으로 일반적인 데이터를 저장하지 않음
- 기본 명령어가 있는 바이너리 디렉토리 - bin
    - binaryse의 약어로 리눅스의 기본 명령어(binary)들이 들어있는 디렉토리
    - 시스템을 운영하기 위한 기본적인 명령어들이 모여있음 (ls, cp, mv, rm 등과 같은 명령어들의 실행파일이 바로 이곳이 존재)
- 부트 디렉토리 - boot
    - 부팅에 핵심적인 커널 이미지와 부팅 정보 파일을 담고 있는 디렉토리
- 시스템 설정 파일 디렉토리 - etc
    - 시스템 환경 설정 파일이 있는 디렉토리
    - 시스템 및 설치한 프로그램의 환경설정 또는 스크립트 파일이 저장
    - 네트워크 관련 설정 파일, 사용자 정보 및 암호정보, 파일 시스템 정보, 보안 파일 등등
- 디바이스 디렉토리 - dev
    - 하드디스크, ODD, USB 등 물리적 장치들이 파일 형태로 매핑되어 있는 디렉토리
    - 연결되어 있는 장치의 정보를 확인 가능
- 사용자 계정 디렉토리 - home
    - 리눅스 사용자의 홈 디렉토리(사용자 계정명과 동일)가 만들어지는 디렉토리
- 공유 라이브러리 디렉토리 - lib
    - 프로그램을 실행할 때 동적으로 로드되는 라이브러리 파일들이 저장
    - **주의사항** - lib 정말 왠만하면 건들지 말것. 잘못 건들면 서버날아감
- 시스템 정보 디렉토리 - proc
    - 현재 동작중인 프로세스들의 정보가 저장

    - 현재 메모리에 존재하는 모든 작업들이 파일 형태로 존재
    - 프로그램 정보 그리고 하드웨어적인 정보들이 저장
    - 디스크에 실제 존재하는 것이 아니고 메모리상에 존재하므로 “ 가상 파일 시스템 ” 이라고 불림.
    - 실제 물리적인 디스크에 존재하는 것이 아니라 메모리에 저장
- 시스템 관리에 관련된 바이너리 파일이 있는 디렉토리 - sbin
    - bin 디렉토리와 유사하지만 오직 루트유저만 실행할 수 있는 프로그램들이 있는 디렉토리
    - 부팅, 복원, 복구 및 수리를 위한 중요한 명령어 디렉토리
- 가변 자료 저장 디렉토리 - var
    - 로그파일, 데이터베이스 캐싱파일, 웹서버 이미지 파일등이 위치
    - 기타 모든 다용도로 사용될 수 있는 파일들이 저장
    - 파일의 크기가 추후 계속 확장될 수 있을경우 더욱 적합
- 사용자 디렉토리 - usr
    - 기본 실행파일과 라이브러리 파일, 헤더 파일등의 파일이 저장되어있는 디렉토리
    - 대부분의 응용프로그램과 파일이 저장되어 있음
    - GCC, Apache, MySQL, PHP등의 응용프로그램이 이곳에 설치
    - 시스템이 아닌 일반 사용자들이 주로 사용하는 디렉터리
- 시스템 디렉토리 - sys
    - 리눅스 커널관련 정보가 있는 디렉토리

Multi-User

- 여러사용자가 동시에 하나의 시스템에 접근할 수 있음

Multi-Tasking

- 여러 개의 task(작업)를 동시에 실행하고, 교대로 컴퓨터의 자원을 사용할 수 있는 기능

리눅스 배포판

- 리눅스를 설치하기 위해 필요한 설치 소프트웨어 패키지. 리눅스 커널과 자유 소프트웨어(GUN)등으로 구성되었으며, 대략 300 여종 정도가 존재함

init 프로세스

- 시스템 부팅 시, 커널에 의해서 init이라는 특별한 프로세스를 생성
- init프로세스는 /sbin/init에 존재하는 프로그램으로 부터 얻고, 모든 프로세스들의 부모 프로세스에 해당
- 시스템의 모든 프로세스는 init에 의해 생성되거나, init의 자손 중에 하나에 의해 생성
- init은 슈퍼유저 권한을 가지고 실행
- killed가 불가능하고 시스템이 종료될때만 끝남
- init의 주요 일은 실행되는 시스템에 의해 요구되는 여러 프로세스들을 생성하고 모니터링하는것

데몬 프로세스

- 특별한 목적의 프로세스로 시스템에 의해 다른 프로세스와 동일하게 생성됨

데몬 프로세스의 특징

- 오랜 기간 실행됨. 보통 시스템 부팅 시 실행되어 시스템 종료 시까지 실행되게 됨
- 백그라운드에서 실행됨. input을 읽거나 output을 write할 컨트롤 터미널을 가지지 않음
- 예로, 시스템 로그를 기록하는 syslogd나 웹 서버인 httpd 등이 해당

셸이란

- 인터페이스를 통해 사용자의 명령어를 커널에 전달하여 시스템 자원들을 관리를 도와주는 시스템(사용자 ←– 셸 —> 리눅스 커널)
- 리눅스 Bash는 CLI방식의 장치이므로, 리눅스 명령어를 익힌다는 것은 결국 bash가 제공하는 명령어를 배우는 것

리눅스 커널을 조작하기 위한 셸의 종류

셸이름	실행 명령	설명
Bourne Shell	sh	많은 셸 스크립트의 기반이 되는 셸
C Shell	csh	C언어 구문과 유사, Bourne Shell 을 확장하여 히스토리, 작업제어, 엘리어스 등 기능 추가 개발자들에게 유용한 기능들을 제공
TC Schell	tcsh	C Shell 에 명령 행 완성 과 명령 행 편집 기능을 추가
Korn Shell	ksh	Bourne Shell 가 호환되며 C Shell 의 많은 기능을 포함, Unix 계열에서 많이 사용
Bourne Again Shell (bash)	bash	리눅스에서 가장많이 사용되는 셸로 Bourne 셸을 토대로 C셸과 Korn Shell 의 기능들을 통합시켜 개발

셸 확인 및 변경

- 셸 변경
  - grep [사용자계정ID] /etc/passwd

```
$ chsh                                  #명령어 입력(Changing shell)
Password:                               #사용자계정ID 비밀번호 입력
New shell [/bin/bash]: /bin/csh         #새로 사용할 셸
Shell changed.                          #변경완료.
```

- 셸 목록 확인
  - etc 디렉터리는 시스템 설정 파일이 존재하는 디렉터리 이며 리눅스 시스템에서 가장 중요한 디렉터리

```
$ cat /etc/shells
/bin/sh
/bin/bash
```

```
/sbin/nologin
/bin/dash
/bin/tcsh
/bin/csh
```

셸 스크립트란

- Shell(쉘)에서 사용할 수 있는 명령어들의 조합을 모아서 만든 배치(batch)파일. 즉, 운영체제의 Shell을 이용하여 한줄씩 순차적으로 읽으면서 명령어들을 실행시켜주는 인터프리터 방식의 프로그램으로 Script를 활용하여 묶어진 명령어 조합을 수행하거나 반복적인 명령어를 단일 명령으로 쉽게 사용가능

셸 스크립트의 장단점

- 장점
  1. 셸 스크립트를 기록하는 것은 다른 프로그래밍 언어의 같은 코드로 쓰인 것보다 훨씬 더 빠른 경우가 많음
  2. 다른 해석 언어에 비해, 셸 스크립트는 컴파일 단계가 없기 때문에 스크립트는 디버깅을 하는 동안 빠르게 실행 가능
- 단점
  1. 실행되는 각 명령에 대한 잠재적으로 새로운 하부 프로세스의 수많은 필요에 따라 속도가 느려질 수 있음
  2. 단순 sh 스크립트는 다양한 종류의 유닉스, 리눅스, BSD 운영 체제, therof 버전, 시스템 유틸리티와 호환가능
  3. 이 있지만 더 복잡한 셸 스크립트는 셸, 유틸리티, 다른 필수 요소 간의 약간의 차이가 많은 경우 실패할 가능성이 있음

치킨5분컷, 리눅스 개본 개념, 셸, shell

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다. - 2023.12**

Permanent link:
http://125.132.25.164/dokuwiki/doku.php?id=wiki:os:linux:%EB%A6%AC%EB%88%85%EC%8A%A4_%EA%B8%B0%EB%B3%B8_%EA%B0%9C%EB%85%90&rev=1673238569

Last update: **2023/01/09 13:29**