운영체제 과제4

proc 파일 시스템 정보 출력

|  |  |
| --- | --- |
| 교수명 | 양승민 |
| 분반 | 공통 |
| 이름 | 박호정 |
| 학번 | 20131847 |
| 제출일 | 20191122 |

목차

1. 관련 연구
   1. 파일시스템
   2. Proc 파일 시스템
2. 문제 해결 방법
3. 추가 및 개선할 부분과 결론

1.관련 연구

1.1파일 시스템

파일시스템은 파일이나 자료를 쉽게 발견 및 접근할 수 있도록 보관 또는 조직하는 체계를 가리키는 말이다. 컴퓨터에서 파일 시스템은 운영체제에 구현되어 있다. 중요한 역할 중 하나는 사용자가 파일을 생성, 삭제하고 읽고 쓰는 것을 지원하는 것이다. 이를 위해 파일 시스템 내부에서 파일을 관리하는 규칙을 정하고 이에 따라 파일을 관리한다.

파일 시스템은 FAT, NTFS, HFS, ext2, ext3, ex4, UFS 등 많은 종류가 존재한다.

교재에서 배운 유닉스 파일 시스템을 예를 들어 설명하도록 한다.

유닉스 파일 시스템은 디스크를 네 가지 파티션으로 나누어 관리한다. Boot block, super block, Inode table(FCB), Data block이다. 각 블락을 간략하게 설명하면 다음과 같다.

Boot block은 os를 로딩하기 위한 프로그램이 저장되어 있다.

Super block은 file system에 대한 data가 들어가는 부분이다.

Inode table은 각 data about file를 저장하는 부분이다. 파일 하나 당 inode가 하나 필요하다.

Data block은 실제 file의 내용을 저장하는 부분이다.

구체적으로 inode정보와 data block정보를 설명하면 아래와 같다.

Inode table에 각 inode는 다음과 같은 정보를 가진다.

File mode: 16 bit를 사용하여 file type, execution flags, permission 정보를 저장한다. file type은 총 7 종류가 있다. 일반 파일, 디렉토리 , character device, block device, named pipe, socket, symbolic link가 그것이다. permission은read-write-execution permission을 세 개의 그룹 마다 정의한다.

Link count: 파일시스템의 디렉토리에서 이 inode를 참조하는 숫자

Owner id: 파일 소유자

Group id: 이 파일과 관계된 그룹 소유자

File size: 파일의 바이트 수

File address: 주소 정보(39바이트). Direct addressing과 indirect addressing 영역으로 나누어져서 파일의 모든 블락 정보를 가지고 있는다.

Last access: 마지막 파일 접근 시각

Last modified: 마지막 파일 수정 시각

Inode modified: 마지막 inode 수정 시각

Data block은 파일의 종류에 따라 다른 내용이 저장된다. UFS에서는 디렉토리도 파일로 관리되어 진다. 디렉토리 파일의 각 구조체는 다음과 같은 필드들로 이루어 진다. 필드는 Inode 번호, 디렉토리 엔트리 크기, 이름 길이, 파일 타입, 파일 이름이다. 기본적으로 디렉토리 파일에는 해당 디렉토리인 ‘.’과 상위 디렉토리인 ‘..’이 포함된다. 이후 해당 디렉토리의 파일들의 정보가 있다. 파일 들에는 디렉토리 파일이 포함될 수 있다.

1.2 Proc 파일 시스템

Proc 파일 시스템은 커널이 가지고 있는 여러 데이터 구조체를 사용자에게 제공해준다. 사용자는 각종 시스템 정보를 /proc 디렉토리를 통해 접근할 수 있다. /proc은 mount point의 역할을 하며 다양한 시스템 정보를 포함한다. #Ls -l /proc을 통해 확인해보면 현재 실행 중인 프로세스 별로 폴더와 20여 개가 넘는 파일들이 있음을 확인할 수 있다. 또한, 운영체제 종류, 커널 버전 마다 proc 파일시스템에서 제공하는 정보와 형식이 다를 수 있음도 파악하였다. 본 과제에서는 리눅스 16.04.5 버전을 기준으로 설명하도록 한다.

프로세스 별 pid폴더의 내용과 그 외의 파일들을 설명하면 다음과 같다.

현재 실행 중인 프로세스 폴더에는 cmdline, cpu, cwd, exe, fd, maps 등 프로세스와 관련된 커널 정보들이 제공된다. 프로세스 폴더의 파일 내용을 설명하면 다음과 같다.

cmdline: 명령 행 옵션

cpu: smp 시스템인 경우 실행된 cpu

cwd: 작업 디렉토리 링크

exe: 프로세스를 실행시킨 명령어의 링크

fd: 모든 file descriptor

maps: 실행어와 라이브러리 파일의 메모리 맵

mem: 이 프로세스에 의해서 사용되는 메모리

root: 이 프로세스의 루트 디렉토리

stat: 프로세스 상태

statm: 프로세스 메모리 상태

status: 인간이 보기 쉬운 포맷으로 된 프로세스 상태

각 프로세스 별 정보 외에도 시스템과 관련된 정보들이 존재한다. 이는 각 파일 별로 제공이 된다. Cpuinfo, meminfo를 비롯해 기본적인 시스템 정보부터loadavg, locks 등 로드와 kernel lock 정보들도 제공된다. 이름을 통해 어떤 정보가 제공되는지 대략적으로 파악할 수 있다. 본 구현에 사용된 파일들을 대상으로 설명하도록 한다. 또한, 구체적인 정보보단 대략적인 내용을 설명하도록 한다.

cpuinfo 에서는 사용하는 cpu의 정보가 있다. Physical id를 통해 cpu를 구분하며 cpu cores는 해당 cpu의 코어 개수를 나타낸다. Hyperthreading 가능 여부는 sibling/cpu cores 를 통해 구할 수 있다. 이 외에는 vendor\_id, cache size, cpu Mhz 등 다양한 cpu 정보를 확인할 수 있다.

Meminfo 에서는 사용하는 메모리의 정보가 있다. 총 메모리와 미사용 메모리, 프로세스에게 할당할 수 있는 메모리 크기 정보가 제공되고 버퍼 사용량, 캐시 사용량 등 30개가 넘는 정보가 제공된다.

그 외의 loadavg, uptime, stat 등은 다음 절에서 설명하도록 한다.

2.문제 해결 방법

본 과제에서는 몇 가지 시스템 정보들을 출력하는 과정이 필요하다. 이를 위해 필요한 정보가 위치하는 파일을 파악하고 해당 파일의 어떤 내용을 봐야하는지 파악하는 것이 중요했다. 또한 그 내용을 프로그램 상으로 얻기 위해서 파싱하는 작업이 필요했다. 이를 위해 서브스트링 검사 함수인 strstr과 토크나이져 함수인 strtok을 사용하였다.

총 7개의 출력 정보를 얻기 위해서 어떤 과정을 거쳤는지 각 정보마다 설명하도록 한다.

-리눅스 커널 버전

/proc/version 에 Linux version x.x 와 같은 형태로 시작한다. 따라서 3 번째 토큰을 출력하면 리눅스 커널 버전이 출력되어 진다.

-시스템 cpu개수, 클럭 속도, 코어 개수

/proc/cpuinfo에 정보가 제공된다. Cpu 개수는 출력된 logical cpu의 개수에서 sibling을 나누어서 계산하였다. Sibling의 값은 코어의 개수와 하이퍼쓰레드 여부에 따라 결정되는데 cpu가 1개인 듀얼코어에 HT라면 sibling의 값은 4이다. 따라서 전체 logical cpu의 개수 4개에서 4를 나누면 1개로 physical cpu 개수가 계산되어 진다. Physical id의 종류를 통해 파악하는 것도 가능하다.

클럭 속도는 cpu MHz에 제공되어 진다.

코어 개수는 각 cpu 별로 cpu cores 값에 나와있다. 따라서 cpu가 2개 이상인 시스템의 경우 physical id에 해당하는 cpu cores 값을 사용하면 된다.

-시스템의 사용 중인 메모리 및 사용 가능한 메모리

/proc/meminfo에 정보가 제공된다. memFree은 사용 중이지 않은 메모리 크기이고 memTotal이 전체 메모리 크기이다. 따라서 사용 중인 메모리는 memTotal-memFree로 구하면 된다. memAvailable은 프로세스에 할당할 수 있는 메모리 크기이다.

-시스템이 마지막으로 부팅된 시간(day:hr:min:sec)

/proc/uptime 에 첫 번째 토큰이 부팅되고 나서 경과한 시간(단위: sec)이다. 현재 시각에서 경과한 시간을 빼면 부팅이 일어난 시간이 된다. Sec단위로 먼저 계산한 후 date 포맷으로 바꿔주면 된다.

-문맥 교환 개수

/proc/stat에 정보가 제공된다. Stat파일의 ctxt에 해당하는 내용이 문맥 교환 개수를 나타낸다.

-인터럽트 개수

/proc/interrupt에 정보가 제공된다. 첫 줄을 제외하고는 각 줄이 인터럽트 하나를 의미하므로 해당 파일의 전체 줄 수에서 1을 빼주면 인터럽트 개수가 구해진다.

-최근 15분 동안 부하 평균

/proc/loadavg에 정보가 제공된다. 첫 번째 토큰부터 1분 5분 15분 동안의 부하이므로 3번째 토큰 정보를 사용하면 된다.

3.추가 및 개선할 부분과 결론

Linux kernel 16.04.5버전에서 제공하는 proc 파일시스템을 기준으로는 출력 결과를 제대로 가져올 수 있다. 하지만 형식이 조금만 변경되어도 프로그램을 다시 작성해야한다. 따라서 모든 리눅스의 버전 별 출력 정보를 분석하여 일반화된 프로그램으로 개선할 필요가 있다.

본 과제 구현과 같은 방식으로 Pid별 정보와 cpuinfo, mem info를 통해top, ps와 같은 프로그램을 구현할 수 있을 것이다. 커널의 정보를 손쉽게 가져와서 프로그램 작성시 참고하여 코딩할 수 있음을 파악하였다.

출처: 위키백과, joinc블로그, geeksforgeeks