운영체제 실습

[Assignment#4]

Class : 목34

Professor : 최상호 교수님

Student ID : 2021202003

Name : 강준우

Introduction

4주차에서는 1.기존의 336pid를 통해서 파일 경로를 확인하는 기능을 구현한다. 해당 기능 구현에서는 코드와 데이터 주소 메모리의 기존 경로들에 대한 정보들을 획득 한다. 획득하는데 있어서 task,mm,vm area를 기반으로 획득을 수행한다.
2. OPT,FIFO,LRU,Clock 의 페이지 교체 기능을 구현한다. 교체후 성능을 확인하기 위해서 page fault와 fault rate를 기반으로 수행한다.

결과화면

Assignment 4-1

```
[17926.563551] ##### Loaded files of a process 'test(3374)' in VMA #####
[17926.563602] mem(55c25d1cf000-55c25d1d0000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /root/Assignment4/Assignment4-1/test
[17926.563649] mem(55c25d1d0000-55c25d1d1000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /root/Assignment4/Assignment4-1/test
[17926.563695] mem(55c25d1d1000-55c25d1d2000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /root/Assignment4/Assignment4-1/test
[17926.563744] mem(55c25d1d2000-55c25d1d3000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /root/Assignment4/Assignment4-1/test
[17926.564239] mem(55c25d1d3000-55c25d1d4000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /root/Assignment4/Assignment4-1/test
[17926.564287] mem(7fb16d5ca000-7fb16d5ec000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/libc-2.31.so
[17926.564345] mem(7fb16d5ec000-7fb16d764000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/libc-2.31.so
[17926.564392] mem(7fb16d764000-7fb16d7b2000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/libc-2.31.so
[17926.564441] mem(7fb16d7b2000-7fb16d7b6000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/libc-2.31.so
[17926.564487] mem(7fb16d7b6000-7fb16d7b8000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/libc-2.31.so
[17926.564533] mem(7fb16d7ce000-7fb16d7cf000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/ld-2.31.so
[17926.564580] mem(7fb16d7cf000-7fb16d7f2000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/ld-2.31.so
[17926.564633] mem(7fb16d7f2000-7fb16d7fa000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/ld-2.31.so
[17926.564689] mem(7fb16d7fb000-7fb16d7fc000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/ld-2.31.so
[17926.564737] mem(7fb16d7fc000-7fb16d7fd000) code(55c25d1cf000-55c25d1d0225) data(55c25d1d2db0-55c25d1d3010) heap(55c25ed78000-55c25ed78000) /usr/lib/x86_64-linux-gnu/ld-2.31.so
[17926.564778] ############################################################
root@ubuntu:~/Assignment4/Assignment4-1# cd ..
```

맨 위 헤더 줄에서 이 시스템콜이 어떤 프로세스(test)와 어떤 PID(3374)에 대해서 실행됐는지 바로 알 수 있다. 그 아래에 나오는 각 줄은 하나의 파일이 매핑된 VMA 하나를 의미하고, mem(시작-끝) 주소 범위와 함께 코드·데이터·힙 구간 정보가 같이 출력된다.

동일한 실행 파일 경로가 여러 줄 반복되는 이유는 ELF 실행 파일이 텍스트 영역, 상수 영역, 데이터 영역처럼 여러 구간으로 쪼개져서 각각 별도의 VMA로 매핑되기 때문이다. heap(start=end)처럼 시작과 끝이 같은 건 이 프로그램에서 아직 동적 메모리 할당을 하지 않아서 힙 영역이 실제로 늘어나지 않은 상태이기 때문에 그렇게 보인다.

정리하면, 과제에서 요구한 “프로세스 이름과 PID, VMA 주소 범위, 코드/데이터/힙 주소, 매핑된 파일의 전체 경로”가 모두 커널 로그에 포함되어 있음을 확인할 수 있었다.

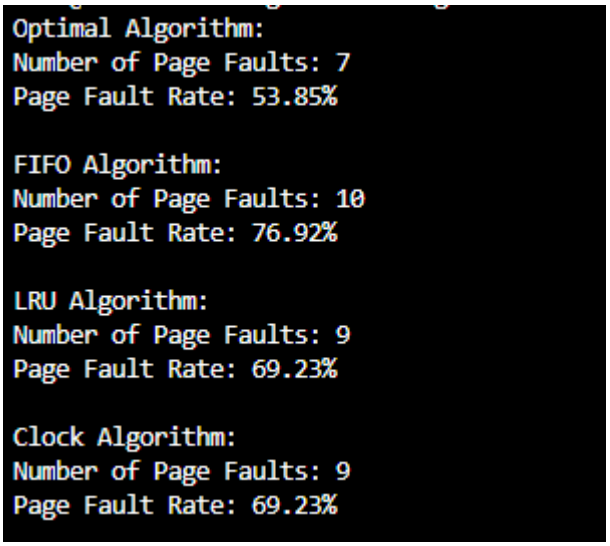
Assignment 4-2

다음은 input 내용을 기술하였다.

3

7 0 1 2 0 3 0 4 2 3 0 3 2

다음은 데이터에 대한 결과이다.



input.1 케이스(프레임 3개, 참조 13번)를 돌려본 결과이다. Optimal은 7번, FIFO는 10번, 그리고 LRU랑 Clock은 둘 다 9번씩 Page Fault가 떴다.

Clock 알고리즘의 경우, Hit가 났을 때는 Hand를 안 움직이고 비트만 1로 다시 찍어주었고, Fault가 났을 때만 Hand를 돌려서 0인 애를 찾아 바꾸는 식으로 위 결과가 나왔다.

비율 계산한 것은 다음과 같다. Optimal은 약 53.85%, FIFO는 76.92%, 나머지는 69.23%

고찰

계속해서 후킹 관련된 과제를 하다보니 운영체제의 작동원리를 모니터링 할 수 있게 되었습니다. 또한 이번 과제에서는 모니터링 단계에서 위치에 대해서도 추가 정보를 얻을 수 있었고, 4-2과제에서는 페이지 교체 기능에 대해서 깊게 공부할 수 있었습니다. 통해서 장단점과 성능에 대해서 비교할 수 있었습니다.

Reference

실습자료 이용했습니다. 감사합니다.