PROJECT 4 REPORT
CS444 FALL17

DECEMBER 1, 2017

THE SLOB SLAB

PREPARED BY

COREY HEMPHILL

JASON YE

GROUP 46

Abstract

This project report is a summary of Homework 4 for Operating Systems II at Oregon State University, Fall 2017. This document includes the approach we used to implement a encrypted block device, a version control log for homework files, a comprehensive team member work log/history, and a report that addresses what we think the main point of this assignment was, how we approached the problem, design decisions relating to our algorithm, how we ensured our solution was correct including testing details, what we learned from the assignment, and how the TA should test our patch.

CONTENTS

1 Question Responses

1.1 What do you think the main point of this assignment is?

The main point of this assignment was to compare the fragmentation ratios of two different memory management algorithms, in this case, the first-fit and best-fit algorithms within the slob.c kernel file. In a lot of cases, there are many solutions to a given problem, but these solutions often have their own trade-offs. In this instance, we observed that the best-fit algorithm actually had better fragmentation ratios than that of the first-fit algorithm. The best-fit algorithm attempts to minimize wasted space by taking extra time to find a better sized fit for allocation. The first-fit algorithm is more efficient in that it is faster, however, it always allocates the next available block rather than worrying about fit, therefore is more prone to wasting space and fragmentation. The secondary point of this assignment was to implement and use system calls to retrieve live data for testing purposes.We got to practice on writing system calls inside the kernel.

1.1.1 best-fit Test Output

********** TEST 1 **********

Fragmentation: 0.079371

Free Memory: 415636

Total Memory: 5228544

********** TEST 2 **********

Fragmentation: 0.079494

Free Memory: 415636

Total Memory: 5228544

********** TEST 3 **********

Fragmentation: 0.079494

Free Memory: 415570

Total Memory: 5228544

1.1.2 first-fit Test Output

********** TEST 1 **********

Fragmentation: 0.087984

Free Memory: 459696

Total Memory: 5218304

********** TEST 2 **********

Fragmentation: 0.088093

Free Memory: 459696

Total Memory: 5218304

********** TEST 3 **********

Fragmentation: 0.088093

Free Memory: 461802

Total Memory: 5218304

1.2 How did you personally approach the problem? Design decisions, algorithm, etc.

We approached the problem by analyzing the kernel's slob.c implementation and tried to gain a working understanding of the SLOB allocator. We copied the original slob_page_alloc function and created simple logic for switching to and from both first-fit and best-fit algorithms. We tweaked the existing logic to implement the best-fit algorithm, which has to be over all allocated pages, not just the current page.We added logic to the SLOB algorithm to search for the best fitting bin to allocate, rather than the first bin available.

1.3 How did you ensure your solution was correct? Testing details, for instance.

For testing, we have added system call functions within the slob.c that returned both free and total memory values for testing the best-fit algorithm implementation in slob.c. System call declarations also had to be added to three other files—unistd_32.h, syscall_32.tbl, and syscalls.h. We created a test program that monitors memory allocation and outputs fragmentation test data to the console. We observed that our solution was correct by comparing the ratios of the two fitting algorithms, and as predicted, the best-fit algorithm was able to decrease fragmentation ratios. To test both algorithms, we would change the value of a constant (1 for best-fit, 0 for first-fit) in order to control which algorithm we wanted to test, and then we would recompile the kernel and run our test program on the virtual machine.

1.4 What did you learn?

For this assignment we gained experience working with system calls within the kernel, as well as how to approach testing and comparing inherent attributes of algorithms to determine trade-offs. We also learned how to determine memory fragmentation.

1.5 How should the TA test your patch?

The TA should install the patch by navigating to the linux-yocto-3.19/mm directory, copy slob.patch into the directory, and then run the following command:

slob.patch -p1 < ../linux-yocto-3.19

Once the patch is installed, run the virtual machine from where the linux root folder lives with the following:

qemu-system-i386 -redir tcp:5546::22 -nographic -kernel linux-yocto-3.19/arch/x86/boot/bzImage -drive file=core-image-lsb-sdk-qemux86.ext4 -enable-kvm -usb -localtime –no-reboot –append ”root=/dev/hda rwconsole=ttyS0 debug”

Once the vm is running, scp the frag_test.c file into the vm root folder with the following command:

scp -P 5546 frag_test.c root@localhost:~

Make the frag_test.c file:

gcc -std=c99 frag_test.c -o frag_test

Observe the output. Compare as needed.

2 VERSION CONTROL LOG

Detail SHA	Author	Description
6d1ed42e513ce6ba58bc6d691a910293f4e9bde0	hemphilc	Add files via upload
107bab1b4ee404c32466993c380f55b31c7ff786	hemphilc	Add files via upload
d4e5ee819399c91951304f0c51c166c819e7ee14	hemphilc	Added assignment head comments
300a9ec2984cac704abf87241400f8e49c004e94	hemphilc	Update slob.c
0dcfc037c04daf2d3def90dbea666857f062c6fa	hemphilc	Update slob.c
5bd19d49dea7012fff235ce683287de648818a4a	hemphilc	Update slob.c
45e6dbfce60140b95d9d4cfecefef130ed469c58	hemphilc	Update slob.c
c0cbbc6006f29914993a4397e84bbc1f2c735918	hemphilc	Update slob.c
ccc90e1b9dc18c5e6f466f1bd592bdfcf2529af6	hemphilc	Update slob.c
eb6169bf50a7194f06547a932b546b44c97afa1a	hemphilc	Update slob.c
b2afc97c26d6f8b09ae5b6b96065fa020bb45904	hemphilc	Update slob.c
3b1acfb3dde89b41641dcfa81f362463c174bd38	hemphilc	Update slob.c
7f8cb34bd5065fd48b764127e48a1c629353bd2f	hemphilc	Update slob.c
b516e5dd0e7f2674bc13b3d84ed9b584b21e1624	hemphilc	Create fragtest.c
68a10ee1d270f8cdcd717ba361717ec334aad2d7	hemphilc	Rename CS444project4-46/fragtest.c to CS444project4-46/FragTest/f
784166c5ea6d8929063eddc9b924c0684025bcec	hemphilc	Add files via upload
650c368fe33b7f3f3c1f33766525fbc0fc0f9a15	hemphilc	Update fragtest.c
d8d7b1d83f54a391b5797986041a18f34d559874	hemphilc	Update fragtest.c
8fbafa25367c7594b52cbd163ef191607dda8be4	hemphilc	Update fragtest.c
546a79c5095c35fb4b14474a848f7f168dc59df7	hemphilc	Update fragtest.c
844407ea36b3eae9792c4e39ec98ae2e6f15815c	hemphilc	Update fragtest.c
e4a2951e5825ab51a5b67e92901d3458e477a634	hemphilc	Update fragtest.c
b89d33c37c0614a609ccdfd0aedd56e53ca50a6f	hemphilc	Update fragtest.c
991378e6cf9b1cf483b78b985a2f2bce489f76d4	hemphilc	Update fragtest.c
2a699ee72605bed4db9c46d00b3d53e4942df074	hemphilc	Update CS444project4-46.tex
4f40c3fa135e0020eebf0bf16ddc96769b204bd0	hemphilc	Update fragtest.c
5190687f10bd5bb6dfc59c7cf4c0195c19ff0bfa	yeja94	Updated Laxtex
69f4f8f0c3c851af1a6e6beb6afbecca244ab0cb	hemphilc	Update fragtest.c
5ecae2175cc7db34497bcf39d219267650a0a1b6	hemphilc	Add files via upload
2cd8685219aeea7699a02e42283f86a1f1dbb037	hemphilc	Update Update Syscalls
a1953fc767b849867fb1b3df16ad2e4cd31b5b9b	hemphilc	Update Fixed syscall list bug
ab5bc1fac33bf279dfc8424ea212ebcfbe8971ac	hemphilc	Fine tuning
efc0e375e12a8054b1a60a039359ee8064e2ec32	hemphilc	Add file via upload
99c22330a753684f779d8de4229f5c5e34e87802	hemphilc	Fixed function defs for patch
ea94f22f0210bbcc6b654e04cde3852cca1cc6cc	hemphilc	typo
29f152e130774900abb66394647ba7a8762088db	hemphilc	fixed function names
fbc826484bf6a77ebc314670c231b413e602cfe8	hemphilc	Update slob.patch

3 Work Log

- *11/25/2017*

  Corey created a project 4 folder in the repo, moved all of the necessary dependencies, and started on the assignment.
- *11/25/2017*

  We begin implementing the best-fit algorithm in slob.c.

- *11/26/2017*

  Corey continued researching and implementing the best-fit algorithm. We created a testing folder for a test program and output files.
- *11/27/2017*

  Jason has been working on the last concurrency assignment.

- *11/27/2017*

  Jason started to do deep research and testing of the algorithms based on Corey's work.

- *11/29/2017*

  Corey updated the testing program and all files inside the test folder.

- *11/30/2017*

  Jason updated the Latex report for this assignment.
- *12/01/2017*

  Jason ran the test files to verify program. Corey ran a couples of tests and tuned up the program.