***README ASSIGNMENT\_4(OS)***

***Git-Hub -*** [*https://github.com/A-WASIF/OS*](https://github.com/A-WASIF/OS)

***Explanation :***

Loader loads an ElF (executable and linkable format)binary into memory and finds the program entry point and executes the program .When a segmentation fault occurs the program needs to identify the memory address where the fault happened and once the faulty address is identified the program needs to determine which segment contains that address.After the identifying the segment the program should allocate physical pages for that segment.

***Global Variables :***

**ElF32\_EHDR-:** Pointer to the ELF header structure .

**ELF32\_phdr \*phdr-:** pointer to the program header table .

**Int fd-:** File descriptor for the opened file.

**Address:-** A function pointer used to store the entry point address of the program .

**Totalpages:-** Tracks the total number of pages allocated .

**Page Fault:-** Tracks the total number of page faults .

**Entry\_addresses:-** Stores the entry address of the program.

**Total Memory Size:-** Tracks the total allocated memory size .

**Entry offset:-** Stores the entry offset from the elf header.

**Internal Fragmentation :-** Tracks the internal fragmentation in memory allocations .

**Virtual memory :-** A pointer to the allocated virtual memory.

updated addresses. A pointer used to track the updated memory address during memory allocation .

***Functions:***

1.load and run elf: It opens the specified ELF executable file and allocates the memory for elf header and program header table after that reads the ELF header and program header table from the file

and if there is any error then check the error during allocation of memory.Iterates through the program header table to find the segment containing the entry point and allocated the memory for the segment ,copies segment content and calculates the entry point address.Executes the program by calling the entry point function.

2.callEntryAddress: Function starts by iterate through program headers using the loop variable and for each program header it initializes phdr table to the corresponding program header and after starting the function now it turns to calculating entry address in this part entry point offset from the elf header stored in the variable entry offset and the entry address is typecasted to a function pointer and stored in global variable \_address.

3.run\_program: Calls the program entry point function (\_start) and prints the return value .

4.Loader\_cleanup: Frees allocated and closes the file

descriptor .

5.Segfault handler: Signal handler for handling segmentation faults and allocating memory on demand.

6.Hex to decimal : Converts a hexadecimal string to a decimal integer .

***Main:***

It checks the number of command line arguments to ensure that there is exactly one argument provided which should be the path to the ELF executable and if the argument count is not equal to two indicating that no ELF executables was provided as an argument ,it displays a usage the message with name of the program and exits with a non zero status code.

If a valid ELF executable argument is provided ,it proceeds to call the load and run elf function ,passing the command line arguments and after executing the elf program and cleaning up resources.Finally it returns 0 to indicate successful execution of the loader.

***Error Handling:***

This loader provides the error handling for the opening ,memory allocation and reading operations.Errors are displayed with relevant error messages before the program exits.

**PhdrTable :** It is a two dimensional array used in the loader code to keep track of memory sizes of segments in the program Header Table (phdr table) of the ELF executable.

The phdr table array has 1000 rows and 2 columns each row represents a segment in the program header table and columns are used for storing information related to that segment.

Data Storage : 1. Zeroth index is used as a flag when the loader is looping through the segment in the phdr table during a segmentation fault handler ,it sets the value in this column to 0 if the segment has been accessed.

2.This column stores the memory size of the respective segment from the phdr table .It indicates how much memory is required for the segment.

***Contributions:***

**1.Wasif Ali :** Segfault\_handler, Ceil, main.

**2.Vipul Verma :** loader\_cleanup, hex\_to\_decimal, callEntryAddress, Error Checking.