**Implementation:**

Our SimpleLoader is a simple implementation of an ELF loader in C. it is designed to load a 32 bit ELF executable file into the memory and execute it.

The loader first reads the elf file and verifies its validity. It then loads the necessary segments into the memory, locate and jump to the entry point to execute the program. After the execution it cleans up any allocated resources.

The loader ensures the elf files validity and its compatibility with the loader , by checking its ELF header’s magic number and verifying whether the file is 32 bit and uses little endian encoding.

The loader then iterates through the program header and identifies segment marked as PT\_LOAD. The marked segments is then mapped into the memory using mmap() and the segment data from the file is loaded into its allocated memory space.

The loader calculates the program's entry point by checking if it is located within the loaded segment. Firstly, P\_vaddr gives us the virtual address of the segment. The loader then starts by going to the starting address of the allocated virtual memory and then finds the offset of the entry point into the segment by subtracting p\_vaddr from entry point. The address of the starting point of the function is then given by summing the offset to the physical address. It then casts this entry point to a function pointer, which is then called to execute the loaded program.

After the program is executed, the loader releases the allocated memory using munmap() and closes the file descriptor.

**Contributions:**

**Varun srivastava:**

Error checks

Cleaning up memory

Loading segment memory

Arithmetic calculation of entry point

**Dhruv Jain:**

Error checks

Loader functionalities

Arithmetic calculation of entry point

**Github repository:** https://github.com/dhruv230605/OS\_Assignment\_1