Project Design Document

Shubham Rawat
2013CS10258
Anupam Khandelwal
2013CS10212

April 2016

1 Overview

- We want to update the JOS into a virtual machine monitor
- Boot a guest JOS on the updated JOS.
- This will be a software based virtualisation and no hardware support is taken.

2 Design Points

- We will create a new system call for creating an environment for the guest JOS and a new environment type for differentiating it with normal user type.
- We will load the bootloader and the kernel images using filesystem.
- Then , we will execute the images as user program.

3 Components

3.1 File System

- File system is needed to load the Bootloader and the Guest Kernel image files at execution time . These images are compiled along with that of the host JOS and be added to the image of the filesystem.
- The main methods required from the filesystem are :
  - open()
  - read()
  - seek()
- Code reference : https://github.com/benwei/MIT-JOS

3.2 Host JOS

- Same as JOS as implemented till lab4.
- We will introduce a new environment type ENV_TYPE_GUEST. This is to ensure that traps for an environment of type ENV_TYPE_GUEST are handled differently from the type ENV_TYPE_USER.
- Any Guest environment is considered as user program , i.e. , it is not a part of the kernel.
- The virtual machine monitor is run as a user program, which creates a child environment to run the guest OS.

3.3 Guest JOS

- Same as the JOS as implemented till lab1.

4 Implementation

- Kernel and Bootloader images are loaded at the appropriate physical address as required by the BIOS.
- eip of guest JOS is set at the first instruction of the bootloader.
- When privileged instructions are executed by the guest JOS , a general protection fault will be generated as it is running in user mode.
- We will check if the environment type is ENV_TYPE_GUEST . If that is the case , then we will use **Trap and Emulate** strategy . In this , using trap-frame , we will find out the opcode that trapped and emulate the instruction accordingly.
- As the current privilege level of guest JOS is same as user privilege level , so , normal instructions will run in the same way as user programs do it.
- Thus , guest JOS will be run using only software virtualization , without any hardware support. This task is made simpler since the guest JOS (completed till lab1) is very basic and runs in kernel mode only.