Introduction: From Another Perspective

Yajin Zhou (http://yajin.org)

Zhejiang University

What happens when a program runs?

- Execute instructions (obviously)
    - fetch, decode, and execute
- Others things are happening in the backend
    - make the program to run
    - allow many programs to use/share memory
    - allow may programs to interact with devices

无名英雄

wú míng yīng xióng

All about Virtualization

- Virtualization
  - OS transforms the physical resources into easy-to-use virtual form
  - Interaction: system calls - interfaces between program and OS
- Managing: resources manager

Virtualizing The CPU

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <assert.h>
#include "common.h"

int
main(int argc, char *argv[])
{
    if (argc != 2)
    {
        fprintf(stderr, "usage: cpu <string>\n");
        exit(1);
    }
    char *str = argv[1];
    while (1)
    {
        Spin(1);
        printf("%s\n", str);
    }
    return 0;
}
```

Virtualizing Memory

```c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include "common.h"

int
main(int argc, char *argv[])
{
    int *p = malloc(sizeof(int));
    assert(p != NULL);
    printf("(%d) memory address of p: %08x\n", getpid(), (unsigned) p);

    *p = 0;
    while(1) {
        Spin(1);
        *p = *p + 1;
        printf("(%d) p: %d\n", getpid(), *p);
    }

    return 0;
}
```

Concurrency

```
#include <stdio.h>
#include <stdlib.h>
#include "common.h"

volatile int counter = 0;
int loops;

void *worker(void *arg) {
    int i;
    for(i = 0; i < loops; i++) {
        counter++;
    }
    return NULL;
}

int
main(int argc, char *argv[])
{
    if (argc != 2) {
        fprintf(stderr, "usage: threads <value>\n");
        exit(1);
    }

    loops = atoi(argv[1]);
    pthread_t p1, p2;
    printf("Initial value : %d\n", counter);

    Pthread_create(&p1, NULL, worker, NULL);
    Pthread_create(&p2, NULL, worker, NULL);
    Pthread_join(p1, NULL);
    Pthread_join(p2, NULL);
    printf("Final value : %d\n", counter);
    return 0;
}
```

I/O

```c
#include <stdio.h>
#include <unistd.h>
#include <assert.h>
#include <fcntl.h>
#include <sys/types.h>

int
main(int argc, char *argv[])
{
    int fd = open("/tmp/file", O_WRONLY | O_CREAT | O_TRUNC, S_IRWXU);
    assert(fd > -1);
    int rc = write(fd, "hello world\n", 13);
    assert(rc == 13);
    close(fd);
    return 0;
}
```

Details

- File System: Where the data will reside on the disk
  - /tmp/file: directory path, file path
- Device driver: Issue I/O requests to underlying physical devices

Design Goals

- OS
  - It virtualizes resources: CPU, memory, or disk
  - It handles related issues, e.g., concurrency
  - It stores files persistently
- Goals:
  - High performance
  - Protection
  - Reliability
  - Emergency-efficiency
  - Security