CSE 451
Autumn 2003

October 9 Section

Mike Swift

What is section for?

- Clarify ideas from lecture
- Discuss homework
- Discuss projects
- Learn extra cool stuff

Announcements

- Homework 2 delayed until Monday
- Please pick project groups of 3 people for the remainder of the term
  - Email me or give me paper in class on Monday

Questions from lecture

- Any questions yet?

Homework questions?

- What problems are introduced when multiple users use a system?
- What are caches good for?
- What are system calls for?
- Do operating systems need to be efficient?

Project questions

- Turnin
  - Hand in your writeup either in lecture tomorrow or with your source code
  - You don't need to include makefiles this time
- C and makefiles
- Calling system calls from usermode code
- Understanding the Linux kernel

C header files

- Header files contain shared information
  - Prototypes:
    - int sys_execcounts(int flags, int * values);
  - Global variables
    - extern int counts[];
    - Need separate declaration:
      - int counts[4];
- Gcc -I~/linux-2.4.20/include
- #include "~/linux-2.4.20/include/asm/unistd.h"

Understanding System Calls

- Linux style:
  - in <asm/unistd.h>

```
#define __NR_foo 292
static inline _syscall1(long, foo, int,
  param)
```

- BSD style

```
in shell.c
  #define __NR_foo 292
  ret = syscall(__nr_foo, param);
```

System Calls in the Kernel

- in entry.S:

```
ENTRY(system_call)
  pushl %eax             # save orig_eax
  SAVE_ALL
  cmpl $(NR_syscalls),%eax
  jae badsys
  call
    *SYMBOL_NAME(sys_call_table)(,%eax,4)
  movl %eax,EAX(%esp)    # save the
  return value
  RESTORE_ALL
```

Operating System Code Quality

- Can malloc() fail?
- What happens if it does?

```
char * buffer;
buffer = malloc(100);
strcpy(buffer, param);
```

System Call Parameters

- What's wrong with this code:

```
long sys_foo(char * param) {
  char * array[100], *temp;
  int i = 0;
  temp = strtok(param, " ");
  while (temp != NULL) {
    array[i++] = temp;
    temp = strtok(NULL, " ");
  }
}
```

Copying data to/from Kernel

- Unsafe to directly access user pointers!

```
long sys_gettimeofday(struct timeval *tv)
{
  if (tv) {
    struct timeval ktv;
    do_gettimeofday(&ktv);
    if (copy_to_user(tv, &ktv, sizeof(ktv)))
      return -EFAULT;
    }
  return 0;
}
```