# CS 460
# Operating Systems

T TH 1-2:35pm

Chadd Williams

## Read

- Read Chapter 1 and Chapter 2
  - Chapter 1: preview of the semester
    - we won’t discuss all of this chapter in class
  - Chapter 2: OS basics
    - prepares you for your labs on Feb 8 and Feb 13
- Assignment Zero
  - posted now. Due Feb 6
  - Linux practice

# Overview

- Practical introduction to Operating Systems
- Topics
  - Purpose
  - History
  - Design Issues/Structure
  - Devices
  - System (Kernel) vs User mode
  - Concurrency/Deadlock
  - Processes/Threads
  - Multi-Core CPUs
  - Memory Management
  - Security

## Syllabus

- *Operating System Concepts* (8th), Silberschatz, et al.
- Grades:

| | | | |
|---|---|---|---|
| Midterm 1 | 15% | Mar | 08 |
| Midterm 2 | 15% | April | 19 |
| Final | 15% | May | 12 |
| Homework/Quizzes/Labs | 15% | | |
| Programming Projects | 40% | | |

- Pop Quizzes: frequent, unannounced, open-note quizzes will be given
- Projects/Labs: Many projects and labs will have a followup Quiz!
- Late Policy: No late assignments accepted
- Grade Complaints: one paragraph summary of why the grade is wrong, **within one week of receiving the graded material**
- All projects are ***individual*** projects unless otherwise stated
- http://zeus.cs.pacificu.edu/chadd/cs460s18

## Great Expectations

- Read the book
  - theory
  - older material
  - **bring questions to class**
- Class lecture
  - more practical
  - more up-to-date
  - ask questions
- Assignments/Labs/Homeworks
  - practical
- Office Hours
  - bring questions!

## Some of what I expect you to know

- C programming
  - arrays / pointers
  - structs
  - dynamic memory / Valgrind
  - Makefiles
  - argv/argc
- Linux command line
  - ls, cd, mkdir, cp, scp, mv, rm, &, vi/emacs/nano/pico/vim , >, <
- Eclipse
  - debugger
- Subversion or Git

- But I know you don’t remember all of that…..

- **Assignment Zero: Linux**

- Assignment 1: C/Eclipse

## Introduction to Operating Systems

- Read Chapter 1!
  - Definition of an Operating System:
  - Kernel:
  - What is not part of the OS?
  - Linux vs GNU/Linux?
- Computers that need an OS:
  - How are their needs different?

# OS Kernel

- Some jobs of the OS

- Not the job of the OS

# OS Interface

- Kernel

- System Call API

- Not a system call:

## Goals of the OS

- Perspectives:
  - User View:
    - Who is the user?
  - System View:
    - Who is the system?

# The Computer

- What does a computer really look like?
  - components of the system

## Boot

- Startup Sequence

- BIOS vs UEFI

## We booted!

- Now what?
- Interrupts:
  - Characteristics:
  - Hardware:
  - Software:
    - Trap
  - Interrupt vector:

# Memory System

- Random Access Memory
- CPU
    - Registers
        - Instruction register
        - data registers
        - load
        - store
- Caches

# Disk Storage

- Magnetic Tape
- Magnetic Disks
  - Spinning Disks
  - SSD (Solid State Drives)
  - RAM spills over to disk
  - Virtual memory
- USB drives
  - Flash memory

## Devices

- Device controller
  - specialized chip
  - buffer
- Device driver

# System Architecture

- Single Processor System
- Multiprocessor System
  - Increased throughput
    - Speed up approaches *N* for *N* processors (Ahmdal's Law)
  - Economy of scale
  - Increased reliability
  - Asymmetric MP
  - SMP
- Multi core System
  - dual-core
  - quad-core

## Software OS Pieces

- Multiprogramming
  - Job
  - Switching
- Time sharing/multitasking
  - Response time
  - Pre-emptive MT
- Process
- Scheduling
  - Job
  - CPU

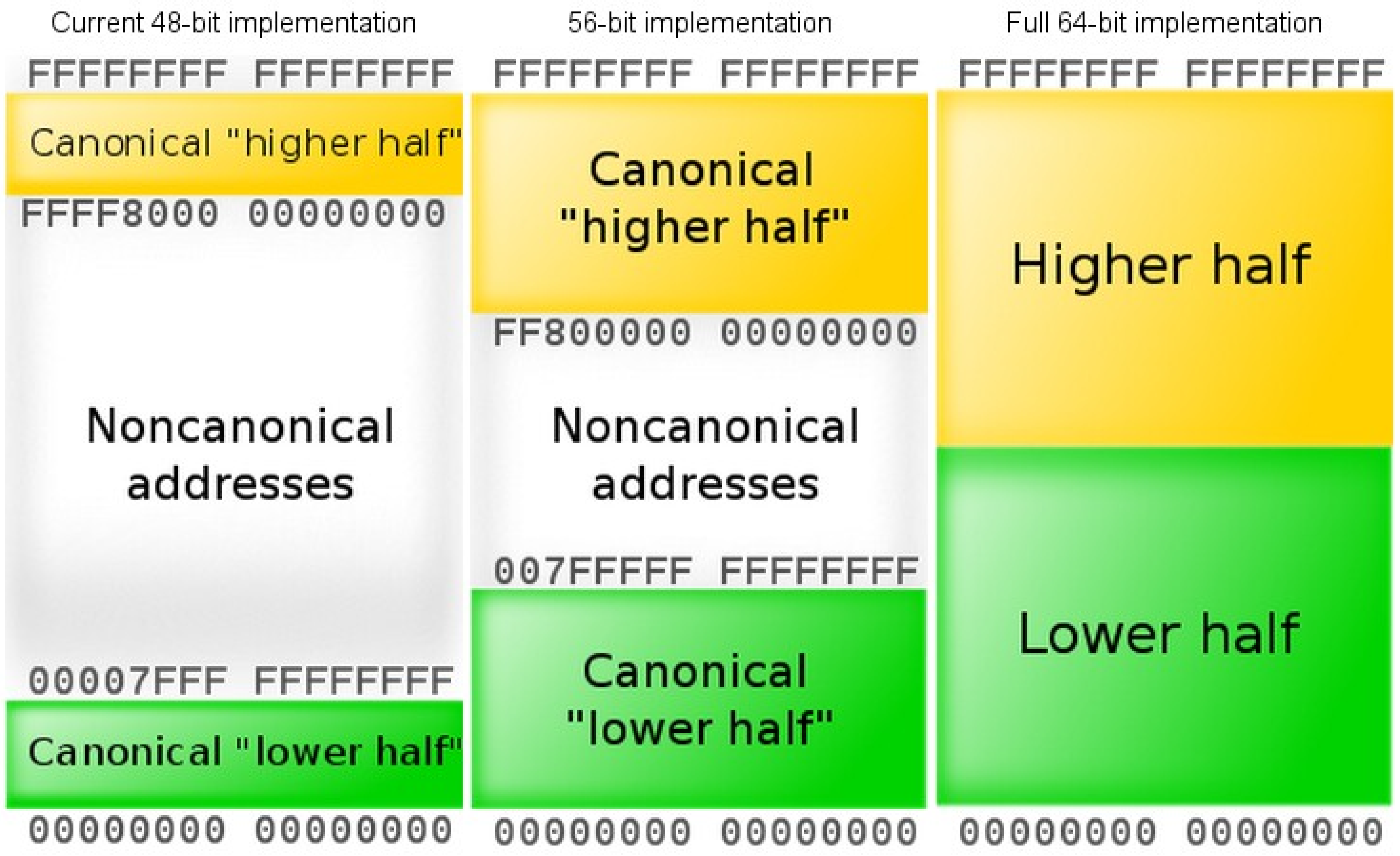
## OS Pieces, cont.

- Virtual Memory
- Physical Memory
- Security

# Virtual Memory, AMD64

- AMD64 is not currently implemented to use all 64 bits of **virtual** memory addressing.

Virtual Addresses:



http://en.wikipedia.org/wiki/File:AMD64-canonical--48-bit.svg

```
/usr/sbin/hwinfo | grep bits
address sizes : 48 bits physical, 48 bits virtual
```

# Setup Minimal Linux VM

- Download http://zeus.cs.pacificu.edu/chadd/Minimal-CS460-Arch-SSD.ova
- File | Import Appliance
- Right click VM, Show in Explorer
- Right click VM, Remove , Remove Only
- In Explorer, copy entire Minimal-CS460-Arch-SSD directory to your SSD.
- Machine | Add | Open Minimal-CS460-Arch-SSD.vbox in the directory on the SSD.

## Launch VM

- Login as student

- Launch a terminal

```
sudo useradd -m -G wheel -s /bin/bash punetid
sudo passwd punetid
```

- Logout
- Login as punetid
  - choose **Default Panel**

## Hands on

```
uname -a
ls   /sys/devices/system/cpu/vulnerabilities/
tail /sys/devices/system/cpu/vulnerabilities/*

cat /proc/version
cat /proc/cmdline
cat /proc/devices
cat /proc/cpuinfo | less
vendor:
model name:
address size:
cat /proc/1/cmdline
cat /proc/1/cmdline | xargs --null
```

## Hands on

- Launch Firefox

```
ps aux |grep firefox # get pid for firefox
cd /proc/PID
ls -al
What is cmd?
What is exe?
cat cmdline
cat limits
cat environ
ls -al fd
ls -al task
ls -al task/###
```

## BONUS: Hands On

```
df -h
```

```
lsmod | less
```

- What is i2c_piix4 ?

```
lspci -v | less
```

- What Ethernet kernel driver is in use?
- Can you find the source code for the **Ethernet network driver** here:
- http://elixir.free-electrons.com/linux/v4.11.5/source
- The lspci gives you lots of useful information
  - who produced the hardware?

```
strings /usr/lib/libc.a | grep printf
```

## Read Chapter 2

## CS 300 Leftovers

# argc/argv

- "The C Programming Language" section 5.10, page 114

int main(int argc, char** argv)

argc - number of entries in argv

argv - array of character pointers containing the arguments

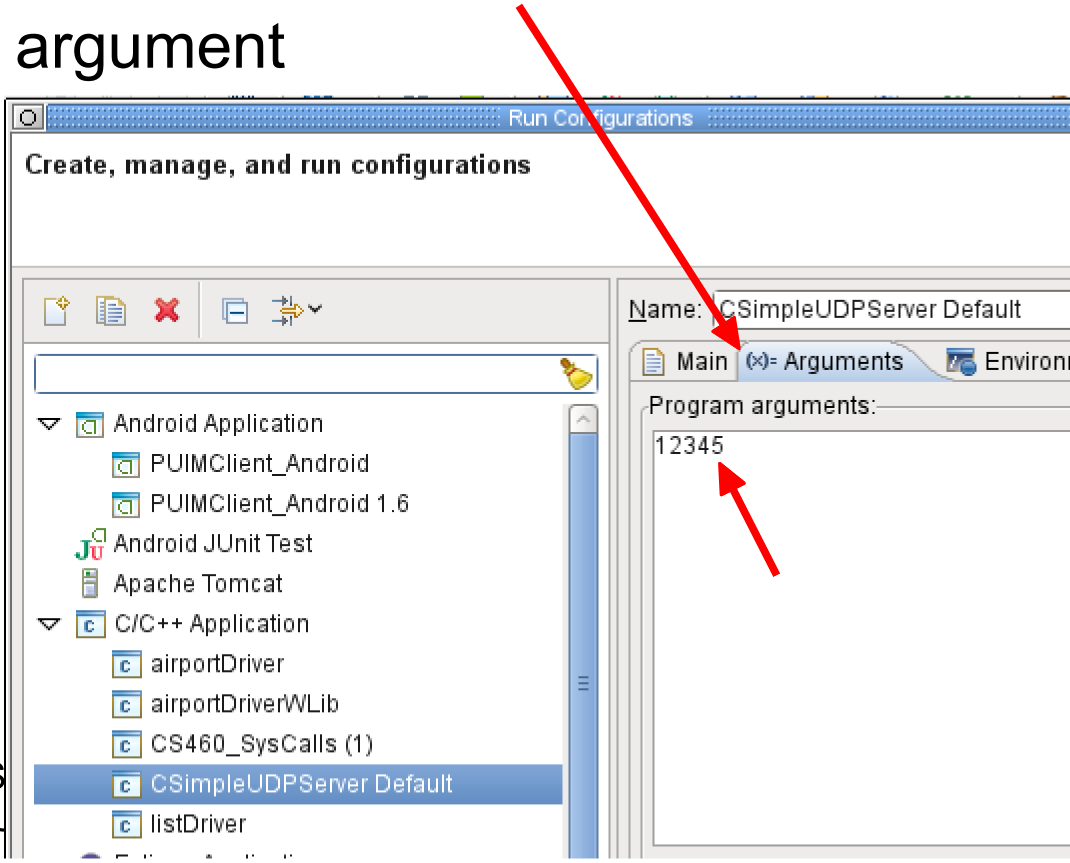
argv[0] - the name of the executable

argv[1] - the first command line argument

**./runMe argument**
**argc = 2**

argv[0] - ./runMe

argv[1] - argument



Create, manage, and run configurations

Android Application
PUIMClient_Android
PUIMClient_Android 1.6
Android JUnit Test
Apache Tomcat
C/C++ Application
airportDriver
airportDriverWLib
CS460_SysCalls (1)
CSimpleUDPServer Default
listDriver

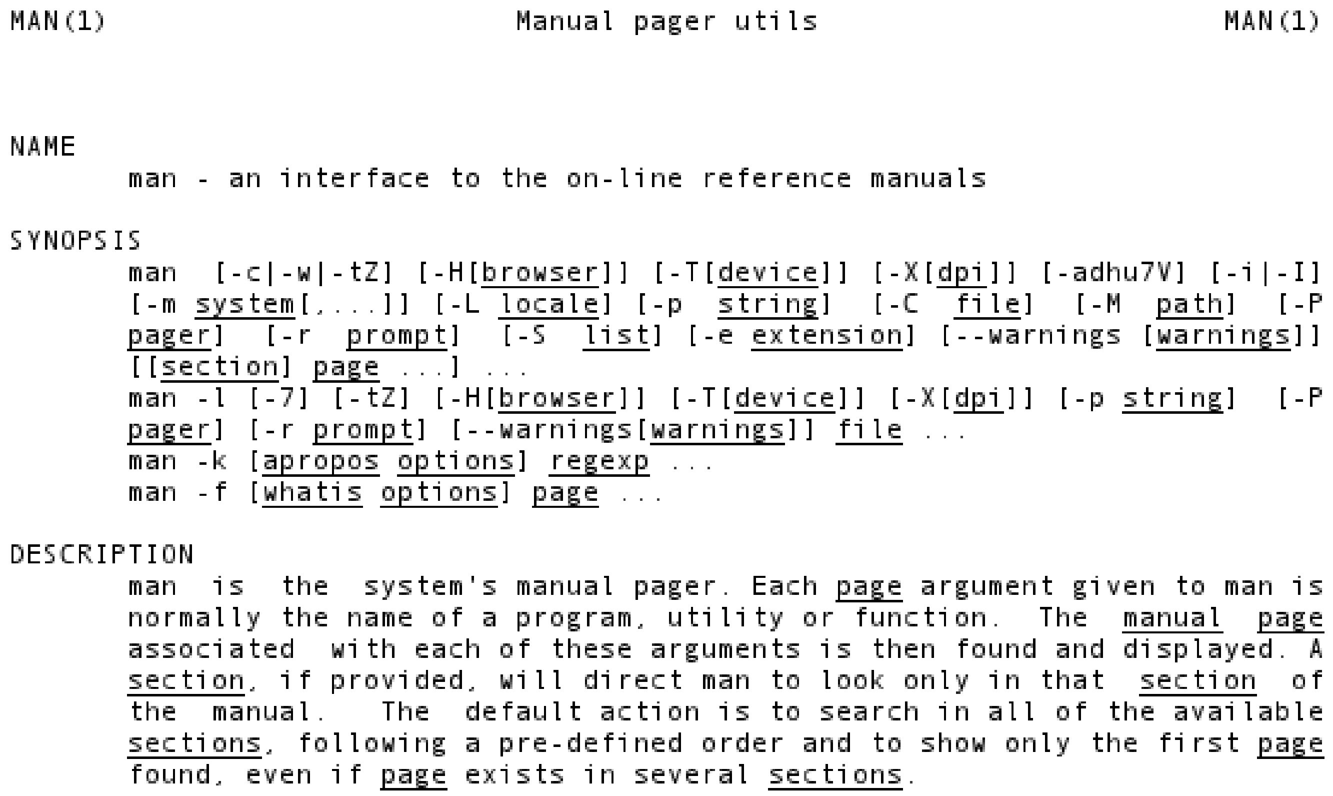
Name: CSimpleUDPServer Default

Program arguments:

```
12345
```

# man pages

- manual pages
  - man bash
  - man man
  - man ls



```
MAN(1)                        Manual pager utils                        MAN(1)


NAME
       man - an interface to the on-line reference manuals

SYNOPSIS
       man  [-c|-w|-tZ] [-H[browser]] [-T[device]] [-X[dpi]] [-adhu7V] [-i|-I]
       [-m system[,...]] [-L locale] [-p  string]  [-C  file]  [-M  path]  [-P
       pager]  [-r  prompt]  [-S  list] [-e extension] [--warnings [warnings]]
       [[section] page ...] ...
       man -l [-7] [-tZ] [-H[browser]] [-T[device]] [-X[dpi]] [-p string]  [-P
       pager] [-r prompt] [--warnings[warnings]] file ...
       man -k [apropos options] regexp ...
       man -f [whatis options] page ...

DESCRIPTION
       man  is  the  system's manual pager. Each page argument given to man is
       normally the name of a program, utility or function.  The  manual  page
       associated  with each of these arguments is then found and displayed. A
       section, if provided, will direct man to look only in that  section  of
       the  manual.  The  default action is to search in all of the available
       sections, following a pre-defined order and to show only the first page
       found, even if page exists in several sections.
```

also available online:
google → man bash
(may be different than what is on your machine)

http://linux.die.net/man/
http://www.kernel.org/doc/man-pages/

## man pages - Library Function

```
FOPEN(3)

Name
   fopen, fdopen, freopen - stream open functions

Synopsis
   #include <stdio.h>
   FILE *fopen(const char *path, const char *mode);

Description

   (arguments or command line options are listed here)

Return Value

Errors

See Also

Referenced By
```

# manual sections

- 0 Header files (usually found in /usr/include)
- 1 Executable programs or shell commands
- 2 System calls (functions provided by the kernel)
- 3 Library calls (functions within program libraries)
- 4 Special files (usually found in /dev)
- 5 File formats and conventions eg /etc/passwd
- 6 Games
- 7 Miscellaneous (including macro packages and conventions), e.g. man(7), groff(7)
- 8 System administration commands (usually only for root)
- 9 Kernel routines [Non standard]

## Don’t do this!

```
chadd@bart:~> man nanosleep
Man: find all matching manual pages (set MAN_POSIXLY_CORRECT to avoid this)
 * nanosleep (2)
   nanosleep (3p)
Man: What manual page do you want?
Man:
```

```
chadd@bart:~>  export set MAN_POSIXLY_CORRECT=1
chadd@bart:~>  man nanosleep

chadd@bart:~>  man -S 3p nanosleep
```

OR
Add that line to the end of **~/.bashrc**

Try
man fork

you get a description of fork in awk!

# Useful header files

- <sys/utsname.h>
    - 
- <unistd.h>
- <sys/sysinfo.h>
- <linux/kernel.h>

## stdout/stdin/stderr

- Three open file streams per process
- stdin
- stdout
- stderr

# Kernighan's Law, p 85

- "Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it."