What is an Operating System?

January 18th, 2017
Prof. Ion Stoica
http://cs162.eecs.Berkeley.edu

Who Am I?
• Ion Stoica
  – 465 Soda Hall (AMP Lab)
  – Web: http://www.cs.berkeley.edu/~istoica/
  – Office hours: Mondays and Tuesdays 11-12 in 465F Soda

• Research areas:
  – Big Data Systems (Apache Spark, Tachyon)
  – Previous: Cloud computing (Apache Mesos), Peer-to-Peer networking (Chord), Networking QoS

This Week
• Sections start Friday – attend any section you want
  – We’ll assign permanent sections after forming project groups
  – This week will help us determine the section balance

• This is an Early Drop Deadline course (January 27)
  – If you are not serious about taking, please drop early
  – Dept will continue to admit students as other students drop

• On the waitlist ???
  – Unfortunately, we maxed out sections and this room capacity

“The Magnificent Seven” - CS162 TAs

Joaao Carreira  Cory Cheung  Josh Don  Apurv Gandhi

Melissa Huang  Nathan Pemberton  Justin Yum
Example: What’s in a Search Query?

- Complex interaction of multiple components in multiple administrative domains
  - Systems, services, protocols, ...

Goals for Today

- What is an Operating System?
  - And – what is it not?
- Examples of Operating Systems Design
- What Makes Operating Systems So Exciting?
- Oh, and “How does this class operate?”

Interactive is important!
Ask Questions!

What is an operating system?

- Special layer of software that provides application software access to hardware resources
  - Convenient abstraction of complex hardware devices
  - Protected access to shared resources
  - Security and authentication
  - Communication amongst logical entities

What Does an OS do?

- Manage resources:
  - Memory, CPU, storage, …
- Provide abstractions to apps
  - File systems
  - Processes, threads
  - VM, containers
  - Naming system
  - …
- Achieves the above by implementing specific algos and techniques:
  - Scheduling
  - Concurrency
  - Transactions,
  - Security
  - …
What makes Operating Systems Exciting and Challenging?

Gordon Moore (co-founder of Intel) predicted in 1965 that the transistor density of semiconductor chips would double roughly every 18 months. Called “Moore’s Law.”

Microprocessors have become smaller, denser, and more powerful.

New Challenge: Slowdown in Joy’s law of Performance

- VAX: 25%/year 1978 to 1986
- RISC + x86: 52%/year 1986 to 2002
- RISC + x86: 77%/year 2002 to present

⇒ Sea change in chip design: multiple “cores” or processors per chip
Another Challenge: Power Density

- Moore’s Law Extrapolation
  - Potential power density reaching amazing levels!
- Flip side: Battery life very important
  - Moore’s law can yield more functionality at equivalent (or less) total energy consumption

![Power Density Graph]

ManyCore Chips: The future is here

- Amazon X1 instances
  - 128 virtual cores, 2 TB RAM
- How to program these?
  - Use 2 CPUs for video/audio
  - Use 1 for word processor, 1 for browser
  - 76 for virus checking???
- Parallelism must be exploited at all levels
  - Intel 80-core multicore chip (Feb 2007)
    - 80 simple cores
    - Two FP-engines / core
    - Mesh-like network
    - 100 million transistors
    - 65nm feature size
    - 24 “tiles” with two cores/tile
    - 24-router mesh network
    - 4 DDR3 memory controllers
    - Hardware support for message-passing

The End of Moore’s Law…

- Moore’s Law has (officially) ended -- Feb 2016
  - No longer getting 2 x transistors/chip every 18 months…
  - or even every 24 months
- May have only 2-3 smallest geometry fabrication plants left:
  - Intel and Samsung and/or TSMC
- Vendors moving to 3D stacked chips
  - More layers in old geometries

Storage Capacity

- Retail hard disk capacity in GB
Network Capacity


Internet Scale: Over 3.6 Billion Users!

<table>
<thead>
<tr>
<th>World Regions</th>
<th>Population (2015 Est.)</th>
<th>Population % of World</th>
<th>Internet Users 30 June 2016</th>
<th>Penetration (% Population)</th>
<th>Growth 2005-2016</th>
<th>Users % of Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Africa</td>
<td>1,155,029,476</td>
<td>16.2 %</td>
<td>310,393,342</td>
<td>28.6 %</td>
<td>7,415.6%</td>
<td>8.4 %</td>
</tr>
<tr>
<td>Asia</td>
<td>4,092,675,886</td>
<td>55.2 %</td>
<td>1,792,163,654</td>
<td>44.2 %</td>
<td>1,467.9%</td>
<td>49.6 %</td>
</tr>
<tr>
<td>Europe</td>
<td>832,073,224</td>
<td>11.3 %</td>
<td>614,975,903</td>
<td>73.0 %</td>
<td>485.2%</td>
<td>17.0 %</td>
</tr>
<tr>
<td>Latin America</td>
<td>626,054,702</td>
<td>8.5 %</td>
<td>384,751,302</td>
<td>61.5 %</td>
<td>2,020.4%</td>
<td>16.7 %</td>
</tr>
<tr>
<td>Middle East</td>
<td>246,755,459</td>
<td>3.4 %</td>
<td>112,599,768</td>
<td>52.7 %</td>
<td>3,006.5%</td>
<td>2.7 %</td>
</tr>
<tr>
<td>North America</td>
<td>359,482,263</td>
<td>4.9 %</td>
<td>320,047,193</td>
<td>98.0 %</td>
<td>190.1%</td>
<td>6.8 %</td>
</tr>
<tr>
<td>Oceania/Australia</td>
<td>37,080,704</td>
<td>0.5 %</td>
<td>27,546,654</td>
<td>92.3 %</td>
<td>201.4%</td>
<td>0.8 %</td>
</tr>
<tr>
<td>WORLD TOTAL</td>
<td>7,344,903,986</td>
<td>100.0 %</td>
<td>5,611,375,813</td>
<td>79.3 %</td>
<td>907.5%</td>
<td>100.0 %</td>
</tr>
</tbody>
</table>

NOTES: (1) Internet Usage and World Population Statistics updated as of June 30, 2016. (2) Source for each world region name for detailed regional usage information. (3) Demographic (Population) numbers are based on data from the US Census Bureau, Eurostat, and from local surveys agencies. (4) Internet usage information comes from data published by Nielsen Online, by the International Telecommunications Union, by the (5) ENISA by local ICT Regulators and other reliable sources. (6) For definitions, disclaimers, navigation help and methodology, please refer to the Site Survey Guide. (7) Information in this site may be cited, giving the due credit and placing a link to www.internetworldstats.com. Copyright © 2001 - 2016, Netindex Marketing Group. All rights reserved worldwide.

(source: http://www.internetworldstats.com/stats.htm )

Not Only PCs connected to the Internet

- Smartphone shipments exceed PC shipments!

  - 2011 shipments:
    - 487M smartphones
    - 414M PC clients
      - 21OM notebooks
      - 112M desktops
      - 63M tablets
    - 25M smart TVs
  - 355.2 million in Q3 2015
  - 73.7 million in Q3 2015
  - 48.7 million in Q3 2015
  - 97.4 million in 2014

- 4 billion phones in the world → smartphone over next decade

Societal Scale Information Systems
(Or the “Internet of Things”?)

- The world is a large distributed system
  - Microprocessors in everything
  - Vast infrastructure behind them

  Internet Connectivity
  Scalable, Reliable, Secure Services
  Databases
  Information Collection
  Remote Storage
  Online Games
  Commerce
  …

MEMS for Sensor Nets
Infrastructure, Textbook & Readings

- **Infrastructure**
  - Website: [http://cs162.eecs.berkeley.edu](http://cs162.eecs.berkeley.edu)
  - Piazza: [https://piazza.com/berkeley/2017/cs162](https://piazza.com/berkeley/2017/cs162)
  - Webcast: Cal Central - [https://calcentral.berkeley.edu/academics/teaching-semester/spring-2017/class/compsci-162](https://calcentral.berkeley.edu/academics/teaching-semester/spring-2017/class/compsci-162)

  - Copies in Bechtel

- **Online supplements**
  - See course website
  - Includes Appendices, sample problems, etc.
  - Networking, Databases, Software Eng, Security
  - Some Research Papers!

Syllabus

- **OS Concepts** How to Navigate as a Systems Programmer!
  - Process, I/O, Networks and Virtual Machines

- **Concurrency**
  - Threads, scheduling, locks, deadlock, scalability, fairness

- **Address Space**
  - Virtual memory, address translation, protection, sharing

- **File Systems**
  - I/O devices, file objects, storage, naming, caching, performance, paging, transactions, databases

- **Distributed Systems**
  - Protocols, N-Tiers, RPC, NFS, DHTs, Consistency, Scalability, multicast

- **Reliability & Security**
  - Fault tolerance, protection, security

- **Cloud Infrastructure**

Learning by Doing

- **Individual Homeworks**: Learn Systems Programming
  - 0. Tools, Autograding, recall C, executable
  - 1. Simple Shell
  - 2. Web server
  - 3. Memory allocation

- **Three Group Projects (Pintos in C)**
  - 1. Threads & Scheduling
  - 2. User-programs
  - 3. File Systems

Group Project Simulates Industrial Environment

- **Project teams** have 4 members (try really hard to find 4 members – 3 members requires serious justification)
  - Must work in groups in “the real world”
  - Same section much preferred

- **Communicate with colleagues (team members)**
  - Communication problems are natural
  - What have you done?
  - What answers you need from others?
  - You must document your work!!!

- **Communicate with supervisor (TAs)**
  - What is the team’s plan?
  - What is each member’s responsibility?
  - Short progress reports are required
  - Design Documents: High-level description for a manager!
Getting started

- Start homework 0 Friday
  - Gets cs162-xx@cory.eecs.berkeley.edu (and other inst m/c)
  - Github account
  - Registration survey
  - Vagrant virtualbox – VM environment for the course
    - Consistent, managed environment on your machine
  - Get familiar with all the cs162 tools
  - Submit to autograder via git

- Start forming a project group

Grading

- 45% three midterms
  - Tentative dates: 2/27, 3/21, 4/24
- 35% projects
- 15% homework
- 5% participation
- No final exam

Projects
  - Initial design document, Design review, Code, Final design
  - Submission via git push triggers autograder

CS 162 Collaboration Policy

Explaining a concept to someone in another group
Discussing algorithms/testing strategies with other groups
Helping debug someone else’s code (in another group)
Searching online for generic algorithms (e.g., hash table)

Sharing code or test cases with another group
Copying OR reading another group’s code or test cases
Copying OR reading online code or test cases from prior years

We compare all project submissions against prior year submissions and online solutions and will take actions (described on the course overview page) against offenders

Personal Integrity

- UCB Academic Honor Code: "As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others."

Lecture Goal

Interactive!!!

What is an Operating System?

- **Referee**
  - Manage sharing of resources, Protection, Isolation
    - Resource allocation, isolation, communication
- **Illusionist**
  - Provide clean, easy to use abstractions of physical resources
    - Infinite memory, dedicated machine
    - Higher level objects: files, users, messages
    - Masking limitations, virtualization
- **Glue**
  - Common services
    - Storage, Window system, Networking
    - Sharing, Authorization
    - Look and feel

Challenge: Complexity

- Applications consisting of...
  - ... a variety of software modules that ...
  - ... run on a variety of devices (machines) that
    - ... implement different hardware architectures
    - ... run competing applications
    - ... fail in unexpected ways
    - ... can be under a variety of attacks
- Not feasible to test software for all possible environments and combinations of components and devices
  - The question is not whether there are bugs but how serious are the bugs!
Increasing Software Complexity

How do we tame complexity?

• Every piece of computer hardware different
  – Different CPU
    » Pentium, PowerPC, ColdFire, ARM, MIPS
  – Different amounts of memory, disk, …
  – Different types of devices
    » Mice, Keyboards, Sensors, Cameras, Fingerprint readers
  – Different networking environment
    » Cable, DSL, Wireless, Firewalls,…

• Questions:
  – Does the programmer need to write a single program that performs many independent activities?
  – Does every program have to be altered for every piece of hardware?
  – Does a faulty program crash everything?
  – Does every program have access to all hardware?

OS Tool: Virtual Machine Abstraction

Virtual Machines

• Software emulation of an abstract machine
  – Give programs illusion they own the machine
  – Make it look like hardware has features you want

• Two types of “Virtual Machine”s
  – Process VM: supports the execution of a single program; this functionality typically provided by OS
  – System VM: supports the execution of an entire OS and its applications (e.g., VMWare Fusion, Virtual box, Parallels Desktop, Xen)
Process VMs

- Programming simplicity
  - Each process thinks it has all memory/CPU time
  - Each process thinks it owns all devices
  - Different devices appear to have same high level interface
  - Device interfaces more powerful than raw hardware
    - Bitmapped display ⇒ windowing system
    - Ethernet card ⇒ reliable, ordered, networking (TCP/IP)
- Fault Isolation
  - Processes unable to directly impact other processes
  - Bugs cannot crash whole machine
- Protection and Portability
  - Java interface safe and stable across many platforms

System Virtual Machines: Layers of OSs

- Useful for OS development
  - When OS crashes, restricted to one VM
  - Can aid testing programs on other OSs

What is an Operating System,…. Really?

- Most Likely:
  - Memory Management
  - I/O Management
  - CPU Scheduling
  - Communications? (Does Email belong in OS?)
  - Multitasking/multiprogramming?
- What about?
  - File System?
  - Multimedia Support?
  - User Interface?
  - Internet Browser?
- Is this only interesting to Academics??

Operating System Definition (Cont.)

- No universally accepted definition
- “Everything a vendor ships when you order an operating system” is good approximation
  - But varies wildly
- “The one program running at all times on the computer” is the kernel
  - Everything else is either a system program (ships with the operating system) or an application program
“In conclusion…”

• Operating systems provide a virtual machine abstraction to handle diverse hardware
  – Operating systems simplify application development by providing standard services

• Operating systems coordinate resources and protect users from each other
  – Operating systems can provide an array of fault containment, fault tolerance, and fault recovery

• CS162 combines things from many other areas of computer science:
  – Languages, data structures, hardware, and algorithms