Recitation 7
February 24\textsuperscript{th} & 26\textsuperscript{th}, 2015
Overview

- **Administrative issues**
  Office Hours, Piazza guidelines
- **Last week’s reflection**
  Project 2.3, OLI unit 3 module 8
- **This week’s schedule**
  – Unit 3 - Modules 9, 10, 11
  – Project 3.1 - March 1st
- **Twitter Analytics: The 15619 Project**
- **Demo**
Announcements

• Monitor AWS expenses regularly
  - Check your bill (Cost Explorer > filter by tags).

• Terminate your resources when not in use
  - Stop still costs EBS money ($0.1/GB/Month)

• Use spot instances
  - And **tag them** at launch time
  - Tag 15619 resources or they are charged to your weekly projects
This Week: Content

- UNIT 3: Virtualizing Resources for the Cloud
  - Module 6: Introduction and Motivation
  - Module 7: Virtualization
  - Module 8: Resource Virtualization - CPU
  - Module 9: Resource Virtualization - Memory
  - Module 10: Resource Virtualization – I/O
  - Module 11: Case Study
Unit 3 : Module 8

- Popek/Goldberg (1974)
  - VMM Properties
    - Equivalence / Fidelity
    - Resource control / Safety
    - Efficiency / Performance
  - ISA Instruction Classification:
    - Privileged instructions
    - Control sensitive instructions
    - Behavior sensitive instructions
  - Theorem:
    - A hypervisor can only be constructed if the set of sensitive instructions is a subset of the set of privileged instructions (i.e., sensitive instructions always trap in the user mode).

- Full- v/s Para-virtualization ; Emulation
Unit 3 : Module 9, 10, 11

● Memory Virtualization
  ○ Two-level mapping
  ○ Overcommitment and reclamation
    ■ Ballooning

● I/O Virtualization
  ○ Device Sharing (cross-OS)
    ■ Privileged Instruction vs Memory-mapped
  ○ Intercepting I/O requests

● Case Studies and Comparison
Virtualization Black-belt?

- Chen, Peter M., and Brian D. Noble. "When virtual is better than real [operating system relocation to virtual machines]." Hot Topics in Operating Systems, 2001

**Research Areas:** Trust, Security, Patching, Scheduling, Live Migration, Monitoring, Nesting, Networks, Energy Efficiency
Diversion: Containers

- Radically changing software deployment
- Encapsulate application and all dependencies

Why Containers (not VMs)?
- Improved utilization
- Faster provisioning
- Easier management
- Microservices

Why not Containers?
- Reduced Isolation
Quiz 3

- Quiz 3 will be open for 24 hours, Friday, Feb 27
  - Quiz 3 becomes available on **Feb 27, 00:01 AM EST.**
  - Deadline for submission is **Feb 27, 11:59 PM EST.**
  - Once open, you have **180 min** to complete the quiz.
  - Late submissions are NOT accepted.
  - You may not start the quiz after the deadline has passed.
  - **Maintain your own timer from when you start the quiz.**
  - **Click submit before deadline passes. No Exceptions!**

<table>
<thead>
<tr>
<th>Location</th>
<th>Silicon Valley</th>
<th>Pittsburgh</th>
<th>Rwanda</th>
<th>Adelaide</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>Feb 26, 09:01 PM</td>
<td>Feb 27, 00:01 AM</td>
<td>Feb 27 07:01 AM</td>
<td>Feb 27 03:31 PM</td>
</tr>
<tr>
<td>Deadline</td>
<td>Feb 27, 08:59 PM</td>
<td>Feb 27, 11:59 PM</td>
<td>Feb 28 06:59 AM</td>
<td>Feb 28 03:29 PM</td>
</tr>
</tbody>
</table>
Quiz 3

• 5% of your Overall Grade
• You only have 1 attempt
• You can save your Quiz answers
  • Highly recommended
  • Save prompt every 15 minutes
• What can I expect from the Quiz?
  • Questions similar to the activities in the Units
  • multiple choice, fill-in-the-blanks, numeric questions, ...
• Feedback for Quiz 3 is released after the deadline passes
Project 2 Reflection

- AWS APIs
- Load Balancing
- AutoScaling
- Monitoring and Debugging
- Multi-tiered applications
- Control Plane
Last Week : A Reflection

● Implementing a Caching Tier
  ○ Pre-warming of backend caches
  ○ Traffic division strategies
  ○ Multiple concurrent caches
  ○ Demand-filled v/s Predictive
  ○ Full-Range v/s Sparse
  ○ Rule-based v/s Adaptive
  ○ Async Fetching
  ○ What else matters?
  ○ Where else can these techniques be used?

● Hierarchical Caching
Students’ Schedule

![Graph showing submissions per day for different dates.
- 16-Feb: Low submissions
- 17-Feb: Very low submissions
- 18-Feb: More submissions than 16-Feb
- 19-Feb: Similar to 18-Feb
- 20-Feb: More submissions than 16-Feb
- 21-Feb: Highest submissions
- 22-Feb: Very high submissions]
# Hall Of Fame

| Username    | 10th Ave (s (s)) | 20th Ave (s (s)) | 30th Ave (s (s)) | 40th Ave (s (s)) | 50th Ave (s (s)) | 60th Ave (s (s)) | 70th Ave (s (s)) | 80th Ave (s (s)) | 90th Ave (s (s)) | 100th Ave (s (s)) | 110th Ave (s (s)) | 120th Ave (s (s)) | 130th Ave (s (s)) | 140th Ave (s (s)) | 150th Ave (s (s)) | 160th Ave (s (s)) | 170th Ave (s (s)) |
|-------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| CrazyAccounting | 1782.5/s (0) | 25.5873 | 1471.6/s (0) | 25.0673 | 2146.6/s (0) | 21.9468 | 2691.0/s (0) | 21.8188 | 2029.9/s (0) | 23.4536 | 118 |
| Ultralisk  | 1683.8/s (0) | 24.4285 | 1510.1/s (0) | 25.5923 | 2028.6/s (0) | 21.0153 | 2463.1/s (0) | 20.3944 | 1831.9/s (0) | 21.6536 | 113 |
| PinkyPiggy | 1569.1/s (0) | 23.105 | 1368.9/s (0) | 23.6888 | 2089.2/s (0) | 21.4937 | 2501.6/s (0) | 20.635 | 1970.9/s (0) | 22.9173 | 112 |
| Toing | 1630.6/s (0) | 22.6808 | 1352.4/s (0) | 23.4418 | 2000.0/s (0) | 21.5 | 2662.9/s (0) | 21.6431 | 1794.2/s (0) | 21.3109 | 111 |
| DragonForce | 1557.8/s (0) | 22.9746 | 1396.3/s (0) | 24.0268 | 2066.2/s (0) | 21.3121 | 2546.2/s (0) | 20.9137 | 1882.1/s (0) | 22.11 | 111 |
| Kobe | 1435.3/s (0) | 21.5812 | 1304.9/s (0) | 22.7941 | 2133.5/s (0) | 21.8434 | 2603.8/s (0) | 21.2738 | 1878.6/s (0) | 22.0732 | 110 |
Project 2.3

- Manual Grading: 20 Points are for the code
  - Follow a Style Guide. AutoFormat. Comment!!!

<table>
<thead>
<tr>
<th>Violation</th>
<th>Penalty of the project grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spending more than $10 for this project phase</td>
<td>-10%</td>
</tr>
<tr>
<td>Spending more than $20 for this project phase</td>
<td>-100%</td>
</tr>
<tr>
<td>Failing to tag all your resources for this project</td>
<td>-10%</td>
</tr>
<tr>
<td>Submitting your AWS credentials in your code for grading</td>
<td>-10%</td>
</tr>
<tr>
<td>Using instances other than the ones specified</td>
<td>-100%</td>
</tr>
<tr>
<td>Caching more than 1000 records in the front end</td>
<td>-100%</td>
</tr>
</tbody>
</table>
Project 3

- Storage in the cloud (It’s Hot!!!)
This Week: Project

- P3.1: Files v/s Databases
  - Data Analysis
    - using bash scripts
    - using MySQL
      - Indexing
      - Joins
  
- Vertical Scaling
  - Instance Size
  - Disk Type / IOPS
Upcoming Deadlines

• Quiz 3: Unit 3 - Virtualizing Resources
  Due: 02/27/2015 11:59PM Pittsburgh

• Project 3.1: Files v/s Databases
  Due: 03/01/2015 11:59PM Pittsburgh

• Project 15619: Phase 1 Part 1
  Due: 03/04/2015 11:59PM Pittsburgh
TWITTER ANALYTICS: THE 15619PROJECT
Architecture

[Diagram showing a web service with back end and front end, with query and response arrows between them]
What is the 15619Project?

- Do MapReduce jobs to extract tabular data from raw data set
  - JSON (more than 1 TB)

- Load the data into HBase and MySQL

- Create a web service that handles HTTP requests responds with data from the backend

- Faster response time = More points
- Winner gets grades, fame (?), job (?)
Motivations and End-Goals

- The C10k/C1M Challenge
- Scalable System Design
- Building 1-click clouds
- Resource Allocation
- Distributed and NoSQL DBs [Tradeoff Eval]
- Data Wrangling / Schema design
- Security
Architecture

- Writeup and Queries will be released on Thu, Feb 26
- We can have more discussions in subsequent recitations
- For now, read about web server architectures, clustering, HBase and MySQL optimization ...
### 15619 Project Time Table

<table>
<thead>
<tr>
<th>Phase (and query due)</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phase 1 Part 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Q1 (due), Q2 (not yet due)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Thursday 2/26/2015 00:00:01 EST</td>
<td>Wednesday 3/4/2015 23:59:59 EST</td>
</tr>
<tr>
<td>Phase 1 Part 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Q2 (due)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Thursday 2/26/2015 00:00:01 EST</td>
<td>Wednesday 3/18/2015 23:59:59 EDT</td>
</tr>
<tr>
<td>Phase 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Q1, Q2, Q3, Q4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Thursday 3/19/2015 00:00:01 EDT</td>
<td>Wednesday 4/1/2015 16:59:59 EDT</td>
</tr>
<tr>
<td>Phase 2 Live Test</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Q1, Q2, Q3, Q4 (HBase)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Q1, Q2, Q3, Q4 (MySQL)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wednesday 4/1/2015 18:00:01 EDT</td>
<td>Wednesday 4/1/2015 23:59:59 EDT</td>
</tr>
<tr>
<td>Phase 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Q1, Q2, Q3, Q4, Q5, Q6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Thursday 4/2/2015 00:00:01 EDT</td>
<td>Wednesday 4/2/2015 18:59:59 EDT</td>
</tr>
<tr>
<td>Phase 3 Live Test</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>● Q1, Q2, Q3, Q4, Q5, Q6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wednesday 4/15/2015 20:00:01 EDT</td>
<td>Wednesday 4/15/2015 23:59:59 EDT</td>
</tr>
</tbody>
</table>

There will also be a report due at the end of each phase, where you are expected to discuss optimizations you used to improve your performance.
15619 Project: Upcoming dates

- Thursday Feb 26, 2015
  ○ Release Phase 1

- Wednesday Mar 4, 2015
  Phase 1 Part 1 Due (Q1-- worth 20% of Queries grade)

- Wednesday Mar 18, 2015
  Phase 1 Part 2 Due (Q2-- worth 80% of Queries grade)

- Thursday Mar 19, 2015
  Phase 1 Report Due
Feature Requests

- The Project Zone ([Piazza @ 563](https://example.com/piazza@563))
  - System in progress
    - Continuously updated
    - Still debugging
  - Suggest features
  - Point out bugs

- AWS ([Piazza @ 191](https://example.com/piazza@191))
  - Mature service
    - Even faster update cycle
  - Suggest features
    - Spot, EC2, EMR, Databases
Project 3.1
Project 3.1 Overview

- Run basic Unix commands like grep, awk etc to extract certain data from given datasets
- Use databases to extract similar data
- Vertical scaling in storage technologies (magnetic vs SSD)
Flat Files

  Mrigesh, 15619, A
  Pan, 15319, A
- Accessing data is inconvenient
- Lacking knowledge of file-layout
- Data redundancy
- ……
Databases

- Organized collection of data supporting data structures
- Database management system (DBMS)
  - A software application
  - Interface between user and databases
  - Capture and analyze data
- Relational databases
  - Organized as fixed-length fields in tables: MySQL
- NoSQL Databases
  - Organized as Key-Value pairs: DynamoDB, Cassandra
Databases

- Advantages
  - Logical and physical data independence
  - Concurrent access and transaction support

- Disadvantages
  - Cost
  - Additional expertise
  - Complex, difficult and time consuming to design
  - Damage affects all programs
MySQL Introduction

- Data Definition Language
  - Table definition: column definition (name, length)
  - Data type: INT, CHAR, NUMERIC, DATE etc
- Example: create a table “students”
  - How many columns do we need?
  - What’s the name of the column?
  - What data type? What’s their relationship?

- Data Manipulation Language
  - select, from, where, set operation, ordering, join
MySQL Demo

- Create a table
  - e.g. CREATE TABLE students ( ID int, Name varchar (255), email varchar(255) );
- LOAD DATA INFILE
  - Try copy data to /var/lib/mysql/song_db/, and then just provide "million_songs_metadata.csv" rather than the full path.
- Use MySQL query to answer questions in runner.sh
  - Aggregate functions, inner join
Files vs. Databases

- Compare flat files vs. MySQL
- Answer:
  - What are the advantages and disadvantages using flat files / databases?
  - In what situation would you use a flat file / database?
  - How to build your own databases? How to manipulate it?
Different Types of Storage

- Internal HDD (Hard Disk Drive)
  - Mechanical Disk
  - Usually from 100s Gigabytes to several Terabytes
  - Work best with large files
- Internal SSD (Solid State Drive)
  - Data is stored on chips, much faster access
  - Much faster access
  - Storage capacity is not as high as HDD, but it is slowly catching up
Disk Operations Commands

- mount/umount
  - attach the file system found on some device to the big file tree
- dd
  - Copy and convert file
- mkfs.ext4
  - Create an ext4 file system
- df
  - show your file systems
Performance Benchmarks

- Run sysbench
  - change to mounted directory
  - use prepare option to generate data into storage system
- Experiments
  - run the sysbench with different storage systems and instance types
  - run sysbench multiple times
Reminders

- Tag your instance: Key: Project, Value: 3.1
  - manually tag your spot instance
- Be sure not to terminate the instance before finishing the runner.sh. Make sure to close the instance after finishing the runner.sh.
- You can also take snapshots of the instance if you want to keep the data for later work.