Brownout: Building More Robust Cloud Infrastructures

Cristian Klein
Post-doc
Cloud Computing Group
Umeå University, Sweden

2014-03-26, Research Day
Umeå, Sweden
Cloud Computing

- Amazon Web Services
- Dropbox
- Windows Azure
- Salesforce
- OpenStack
Cloud Computing Definition

- **Rapid provisioning (and release) from a shared pool of resources**
- **3 deployment models**
  - Public cloud ("our stuff")
  - Private cloud ("my stuff")
  - Hybrid cloud (combine the above two)
- **3 service models**
  - Infrastructure-as-a-Service (IaaS)
    - Data centers
    - Lease CPU, memory, disk
    - **Virtual Machines**
Problem: Unexpected Events

- 82% of customers give up on a lost payment transaction*
- 25% of users leave if load time > 4 s**
- 1% reduced sale per 100 ms load time**
- 20% reduced income if 0.5 s longer load time***

* JupiterResearch  ** Amazon  ***Google

Cristian Klein: Brownout  2014-03-26
Idea

Disable optional computations
E.g., recommender engine

Questions

– **When** to disable such code per application?
– **How** to coordinate multiple applications?
Cloud Application Architecture

Data center

Application 1

Client → Load balancer → Replica

Application 2

Client → Load balancer → Replica

Resource Manager
Brownout: Inside a Replica

Client → Logic → Database

Replica Controller

\( t \) = response times
\( \theta \) = probability of serving optional content (dimmer)

Replica Controller

- The devil is in the details
- Need to adapt to changes
  - Number of users
  - Resource capacity available
- Not all requests take the same time
  - E.g., cached in memory, disk
- Need to reject noise
  - E.g., NTP daemon firing up
- Adaptive PI controller

\[ \Theta^{k+1} = \Theta^k + \frac{1 - p_1}{\alpha} \cdot e^{k+1} \]

M. Maggio, C. Klein, K-E Årzen, “Control strategies for predictable brownouts in cloud computing”, IFAC, 2014 (accepted)
Results: flash-crowd

RUBiS: benchmark eBay-like cloud application

RUBiS: no control  RUBiS: with control
Other Brownout Contributions

• Load-balancer
  – Goal: direct load based on dimmer
  – Result: optional content up by 5%

• Resource manager
  – Goal: give resources to application that “struggles”
  – Result: fairly balances resources among applications

J. Dürango et al. “Control-theoretical load-balancing for cloud applications with brownout”, (submitted)
Conclusions

• Cloud applications need to tolerate
  – Flash-crowds (sudden increase in users)
  – Hardware failures

• Brownout
  – Minimally intrusive method
  – Renders cloud applications more robust

• Perspectives
  – Improve load-balancing algorithms
  – More experiments on faulty hardware
Acknowledgments

- Manfred Dellkrantz, Lund University
- Jonas Dürango, Lund University
- Erik Elmroth, Umeå University
- Francisco Hernández, Umeå University
- Martina Maggio, Lund University
- Alessandro Papadopoulos, Lund University
- Luis Tomás, Umeå University
- Johan Tordsson, Umeå University
- Karl-Erik Årzén, Lund University
Appendix
Zoom: Inside a Physical Machine

![Diagram of a physical machine with components labeled: RM, app_1, app_2, LC_1, LC_2, VM Dom_0, VM Guest_1, VM Guest_2, Hypervisor. Connections are indicated with arrows and symbols such as f^k, Θ^k, t^k.]
Results: performance interference

RUBiS: no control

RUBiS: with control

2014-03-26  Cristian Klein: Brownout
Results: 2 Applications

The graph shows the performance metrics for two applications: RUBiS and RUBBoS. The metrics include:

- **max latency [s]**: The graph plots the maximum latency over time for different user counts (50,10, 50,100, 50,200, 10,200, 100,10).
- **dimmer**: This metric is shown for both RUBiS and RUBBoS, indicating the dimmer's performance over time.
- **matching value**: The graph illustrates the matching value for different user counts, showing fluctuations over time.
- **CPU [%]**: The CPU usage is depicted for RUBiS and RUBBoS, showing how the system is utilized over time.

The time axis is labeled as [s] ranging from 0 to 1000 seconds. The graph provides insights into the performance and scalability of the applications under varying user loads.
Results: 4 Applications

- $RUBiS_1$
- $RUBBoS_2$
- $RUBiS_3$
- $RUBBoS_4$

Users: $1, 2, 3, 4$

50, 10, 10, 10
50, 100, 10, 10
50, 100, 50, 10
50, 100, 50, 100
10, 100, 50, 100

Max dimmer latency [s]

CPU [%] matching value

Time [s]