Optimizing Client-side Resource Utilization in Public Clouds

Swapnil Haria, Mihir Patil, Haseeb Tariq, Anup Rathi

Outline

- Motivation
- Solution
- Implementation
- Evaluation
- Conclusion

Outline

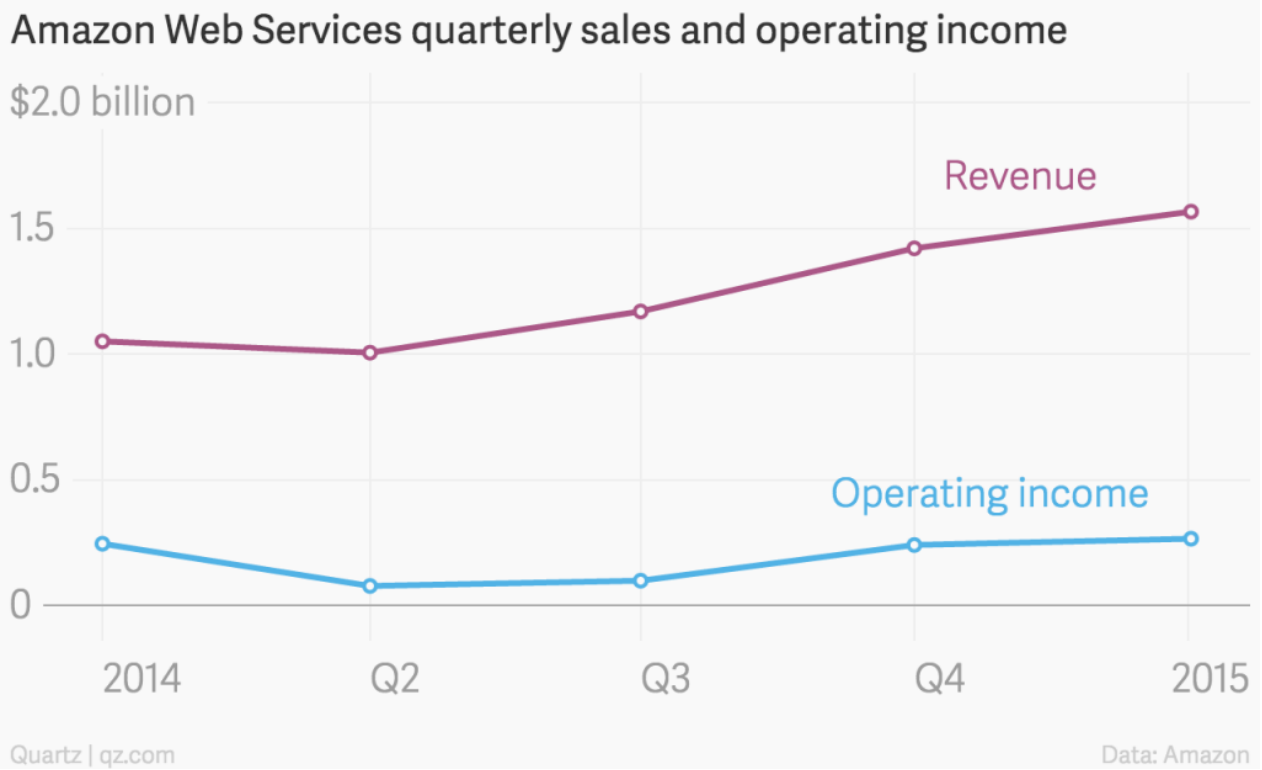
- Motivation
- Solution
- Implementation
- Evaluation
- Conclusion

Cloud Services
(Not a distraction anymore[1])

[1] Jeff Bezos' Risky Bet, November 2006, http://www.bloomberg.com/bw/stories/2006-11-12/jeff-bezos-risky-bet

Cloud Services
(Not a distraction anymore[1])

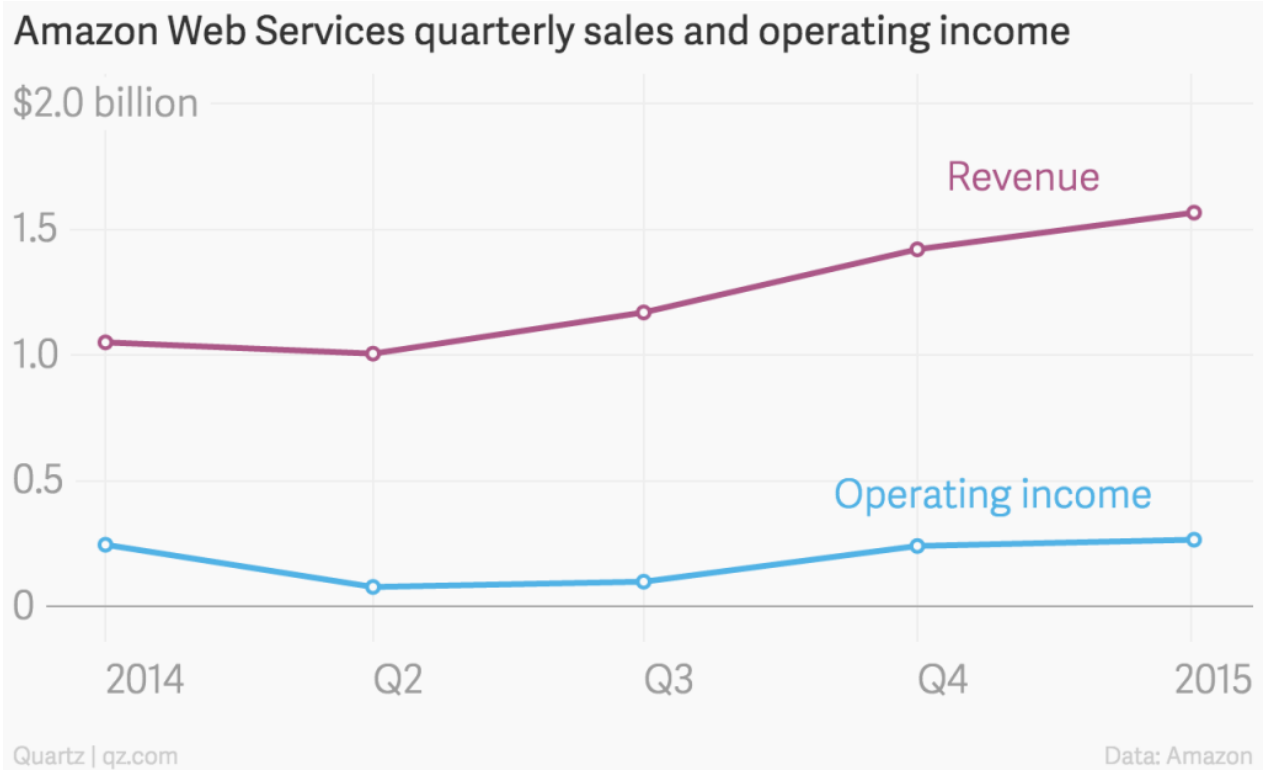
- 30 % of total cloud revenue
- Annual revenues crossed $5 Billion



[1] Jeff Bezos' Risky Bet, November 2006, http://www.bloomberg.com/bw/stories/2006-11-12/jeff-bezos-risky-bet

Cloud Services
(Not a distraction anymore[1])

- 30 % of total cloud revenue
- Annual revenues crossed $5 Billion



Other Players :

Google Compute Engine

[1] Jeff Bezos' Risky Bet, November 2006, http://www.bloomberg.com/bw/stories/2006-11-12/jeff-bezos-risky-bet

Popularity

- ZERO up-front capital expenses
- On-demand hardware availability
- Flexible pricing options

Popularity

- ZERO up-front capital expenses
- On-demand hardware availability
- Flexible pricing options

"Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use."

Popularity

- ZERO up-front capital expenses
- On-demand hardware availability
- Flexible pricing options

"Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use."

Elastic Cloud Compute

Popularity

- ZERO up-front capital expenses
- On-demand hardware availability
- ~~Flexible pricing options~~

"Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you actually use."

Elastic Cloud Compute

Limitations

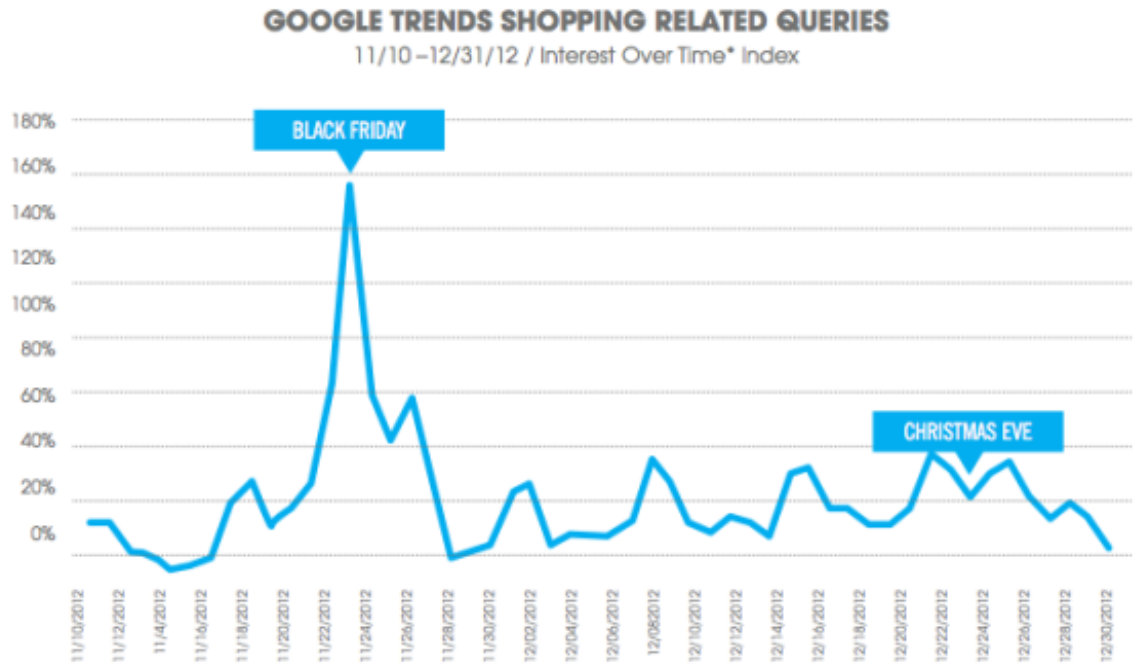
- Allocate resources in fixed sized chunks (EC2 Instances)
  - 1 core , 1GB RAM -> 36 core, 244 GB RAM

- Accurately predict application requirements
  - Undersized VM - Performance degradation
  - Oversized VM - Extra costs

Multiple applications, multiple VMs, no peace

Challenges

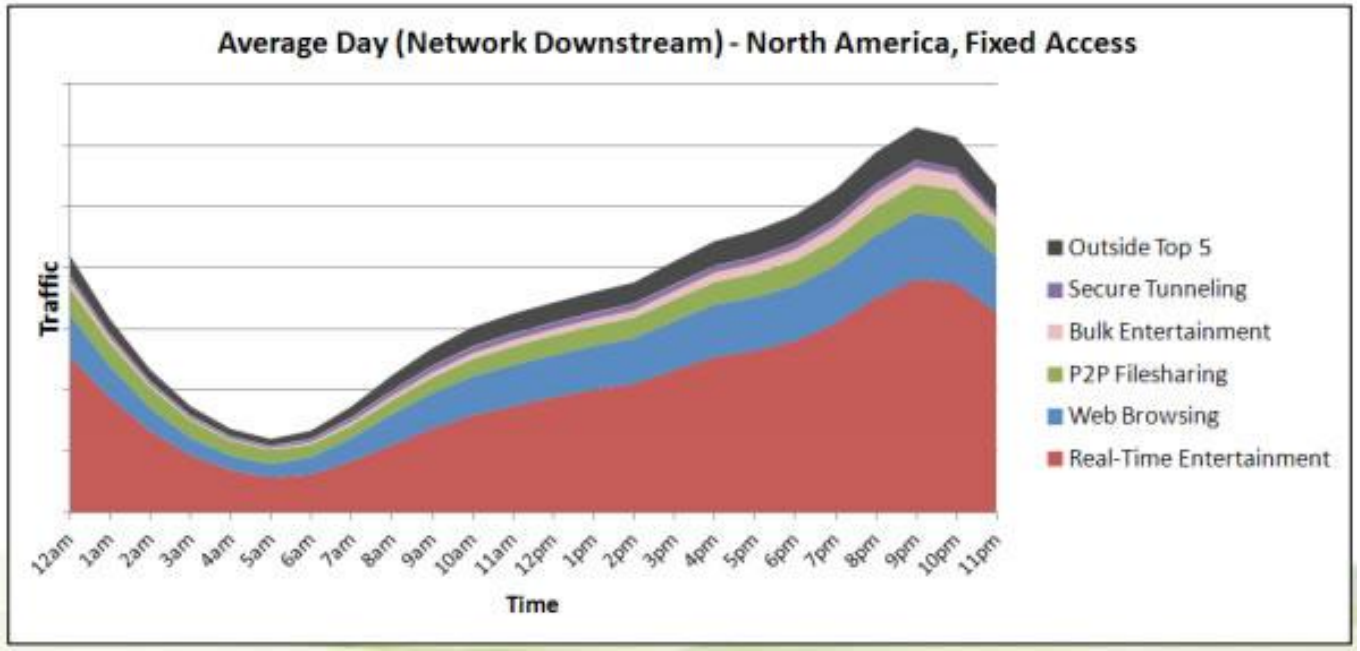
- Application requirements vary widely
  - Black Friday for e-commerce websites



http://www.xad.com/media-mentions/mobile-activity-on-xmas-eve-24-pct-higher-than-black-friday/

Challenges

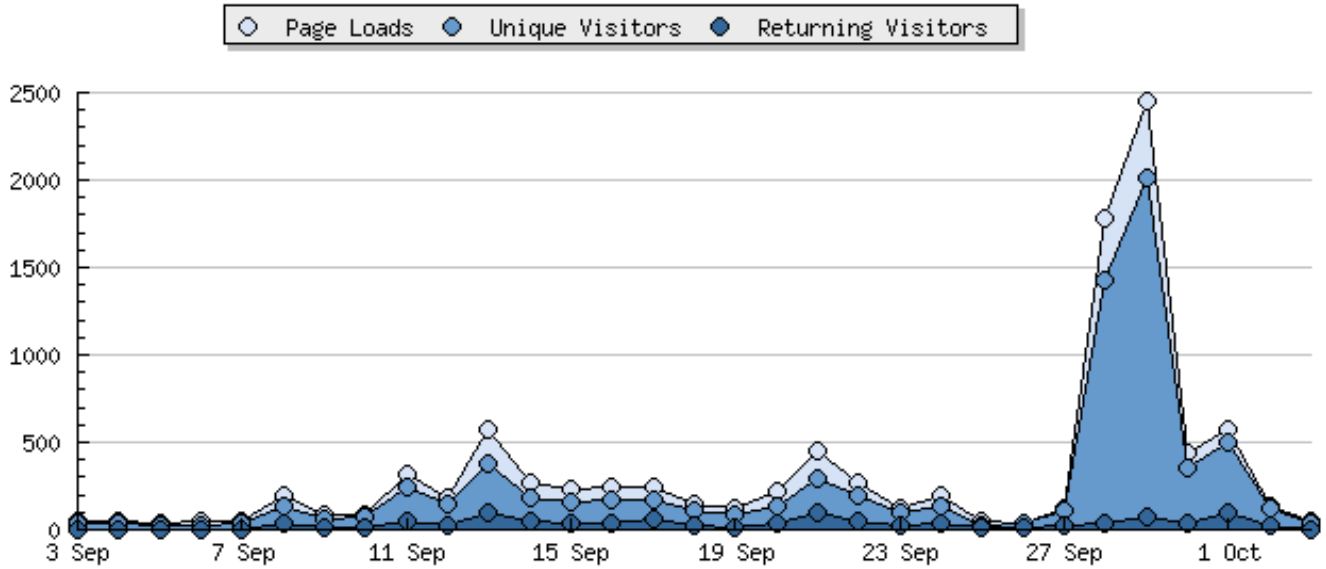
- Application requirements vary widely
  - Black Friday for e-commerce websites
  - Evenings and late nights for Netflix



http://www.techspot.com/news/46048-netflix-represents-327-of-north-americas-peak-web-traffic.html

Challenges

- Application requirements vary widely
  - Black Friday for e-commerce websites
  - Evenings and late nights for Netflix
  - Slashdot effect!



Challenges

- Humans are ~~bad~~ terrible at estimating workload requirements[2]
- Study of developers at Twitter submitting jobs to datacenter
  - 70% overestimated by 10x
  - 20% underestimated by 5x

[2]**Quasar: Resource-Efficient and QoS-Aware Cluster Management**. Christina Delimitrou and Christos Kozyrakis. ASPLOS 2014.

Outline

- Motivation
- Solutions
- Implementation
- Evaluation
- Conclusion

Resource as a Service[3]

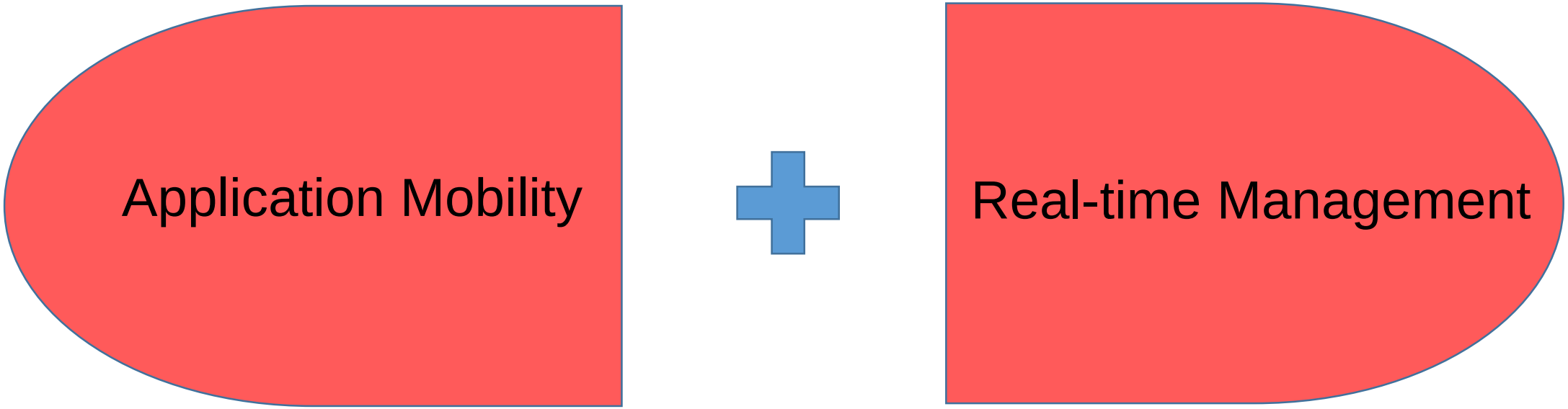
1. Fine grained cloud reservations
2. CPU (cycles), memory (pages), I/O (bandwidth), Time (seconds)

- Where does it stop?
- Reduces wasted costs, but difficult to reason about
- Hardware feasibility issues for service providers

[3] The rise of RaaS: the resource-as-a-service cloud. Orna Agmon Ben-Yehuda et al. Commun. ACM 2014

Proposal

Tell me more!



Application Mobility

- On-demand application migration across machines
- Conventional issues -
  - Application state stored in kernel (file descriptors, sockets)
  - Residual dependencies left on source machine
  - Execution Continuity

We need

- Process Isolation (even from kernel)
- Minimal state in kernel

Now where did I see that before?

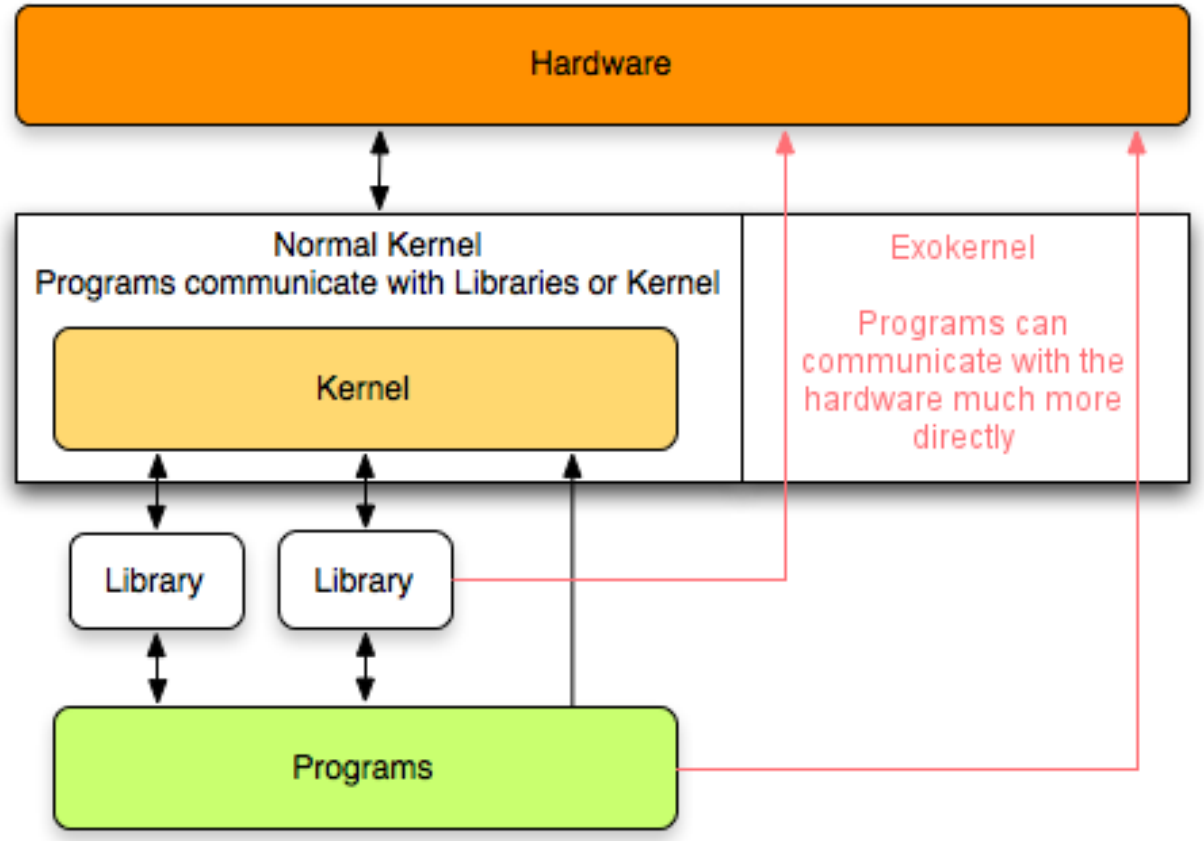


Image Source - Wikipedia

Where do I find one of these?

Old idea, but making a comeback in Cloud OS

- Drawbridge from Microsoft Research
- MirageOS from University of Cambridge

Both (claim to) support application-migration!

Real-time Management

- Monitor application requirements in real-time

- Use application migration to organize processes on VMs

Real-time Management

- Monitor application requirements in real-time
  - Relatively easy
  - Working set sizes, idle cycles

- Use application migration to organize processes on VMs
  - Complex
  - Varying configurations and prices of VMs
  - Identifying processes to migrate
  - Downtime / Budgets!

Policies

Steps

- Determine migration events
- Identify process(es) for migration
- Choose target from existing VMs, if possible
- Figure out instance types for creating new VMs

Policies

Metrics (in order of priority)

- Maximize VM utilization
- Satisfy performance guarantees
- Minimize costs

User-Defined Parameters

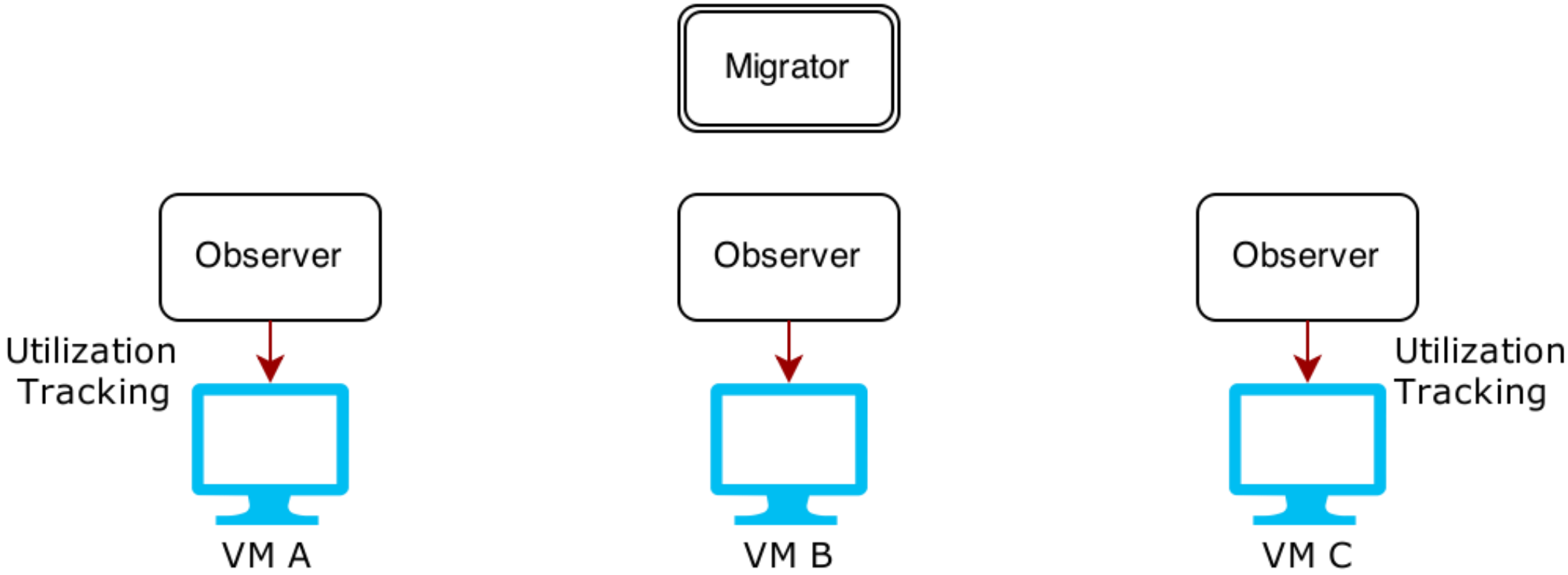
- Upper limit on cost
- Max downtime per process

Policies

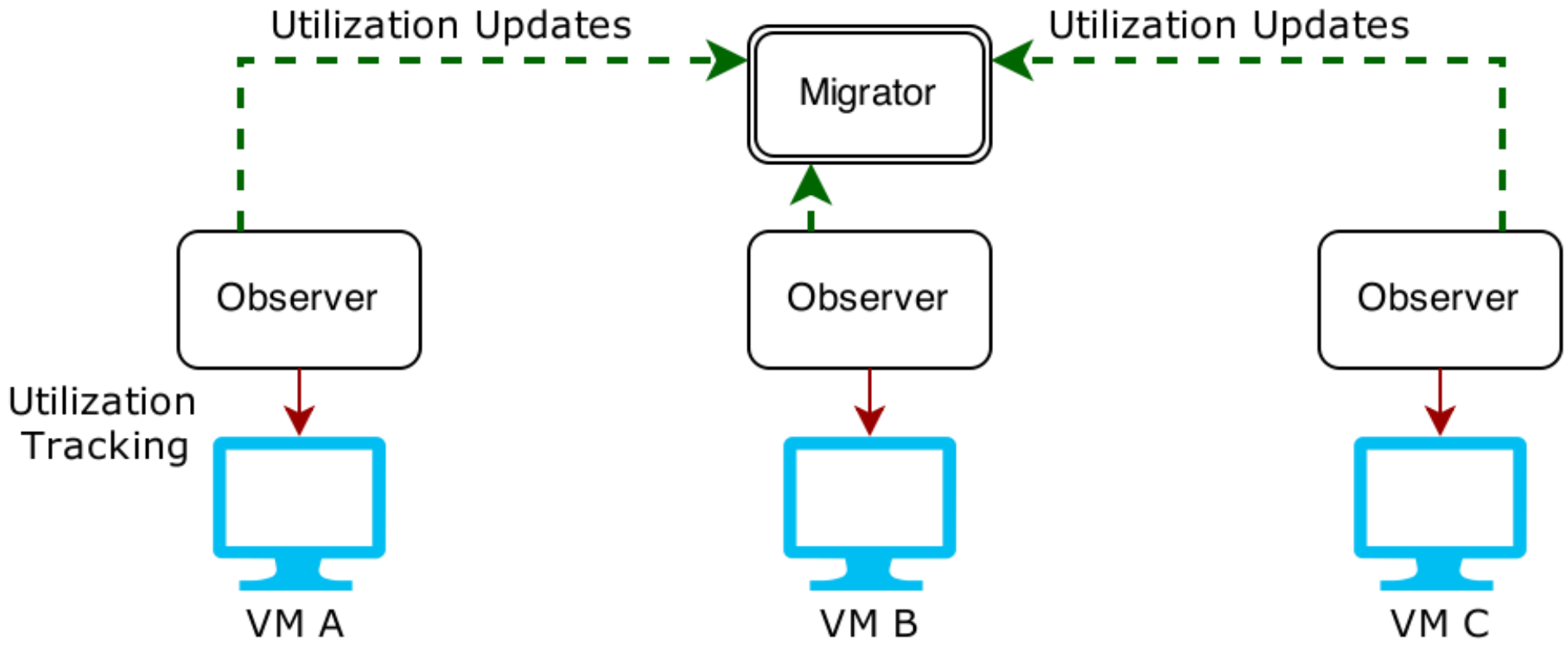
- **Single Application per VM**
  - Easy to reason about
  - Use naive best fit model to find target VMs

- **Multiple Applications per VM**
  - Highly complex optimization problem (NP-Hard)
  - Use Heuristics!
  - Use best fit and explore nearby options to find target VMs

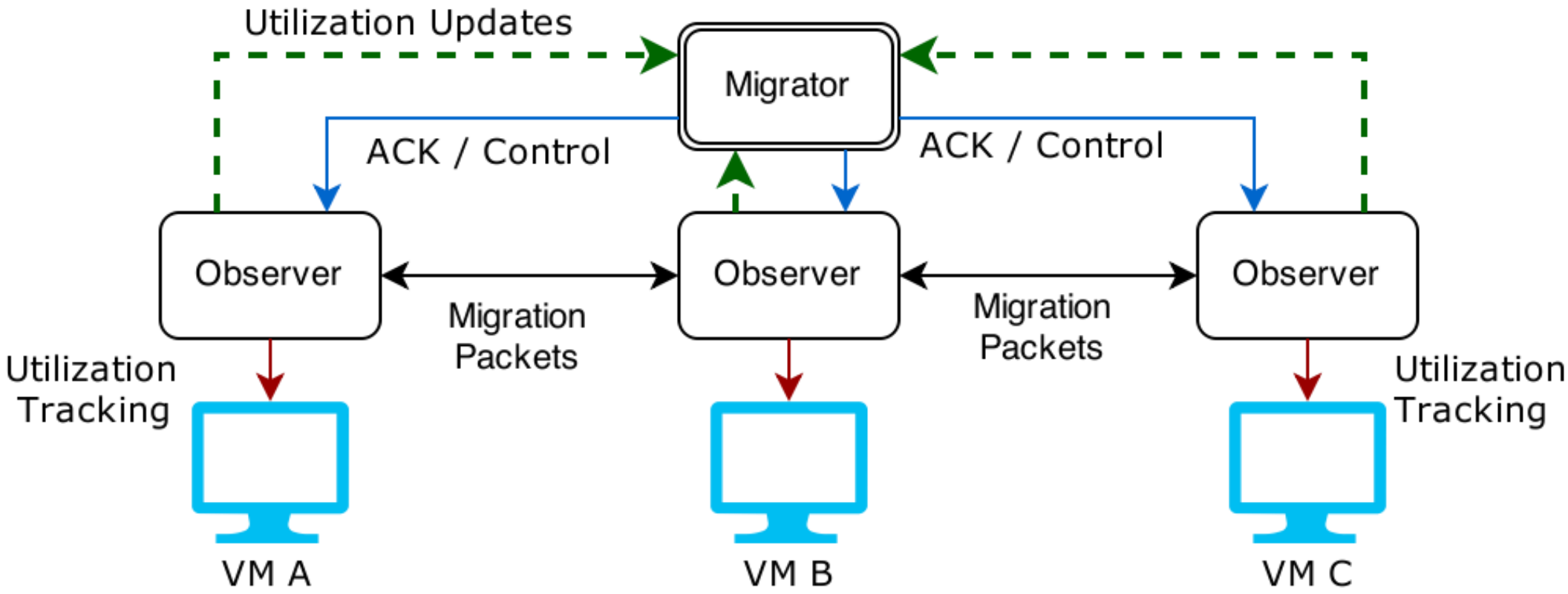
Software Architecture



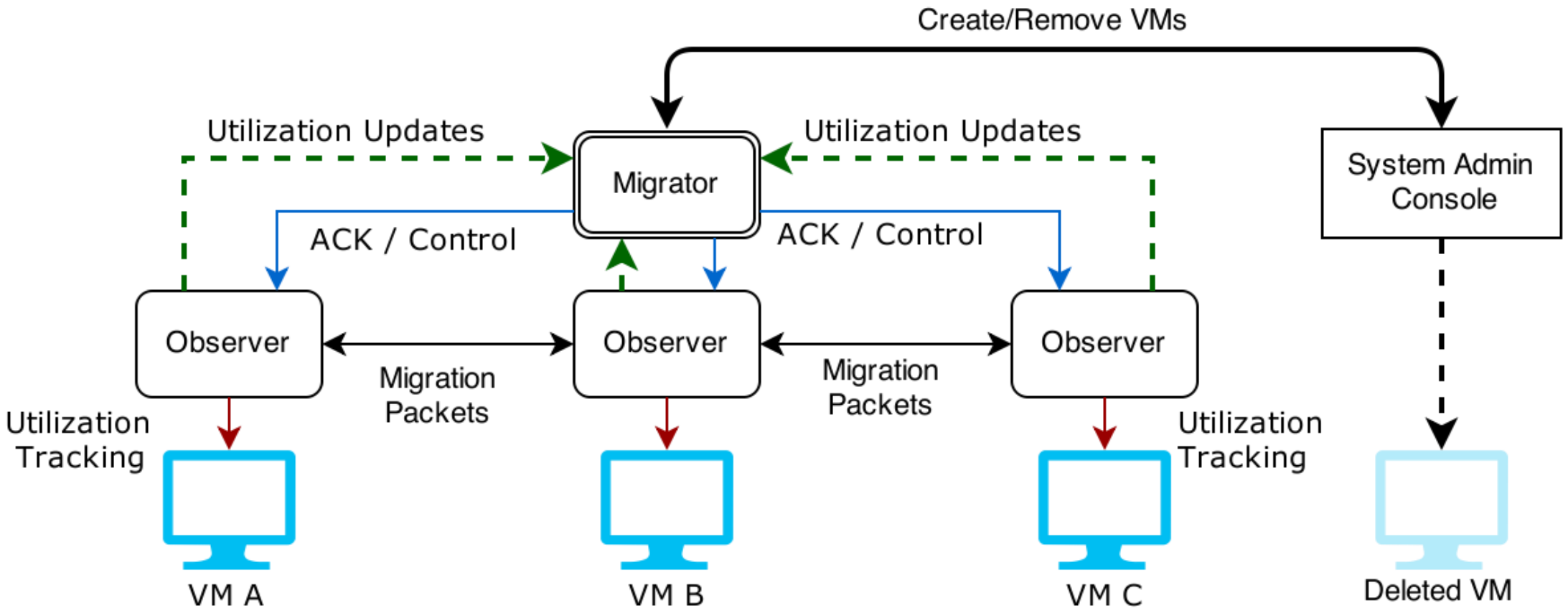
Software Architecture



Software Architecture



Software Architecture



Outline

- Motivation
- Solutions
- Implementation
- Evaluation
- Conclusion

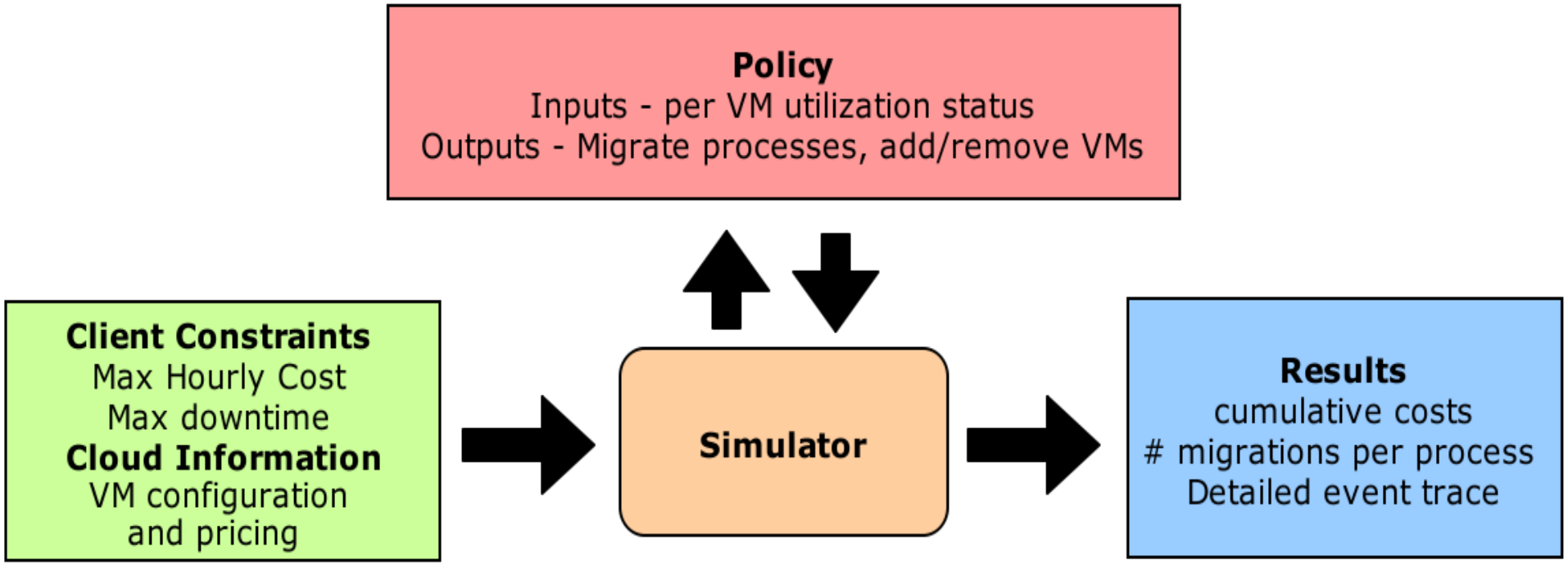
Proof of Concept Model

- Linux Containers (lxc)
  - Emulate isolated processes on Drawbridge/MirageOS

- Checkpoint/Restore in Userspace (CRIU)
  - Checkpoint containers on VM A
  - Migrate files to VM B
  - Restore on VM B

Simulator

- Rapidly validate migration policies
- Evaluate the influence of policy parameters on results
- Written in about 2000 lines of Java code



Outline

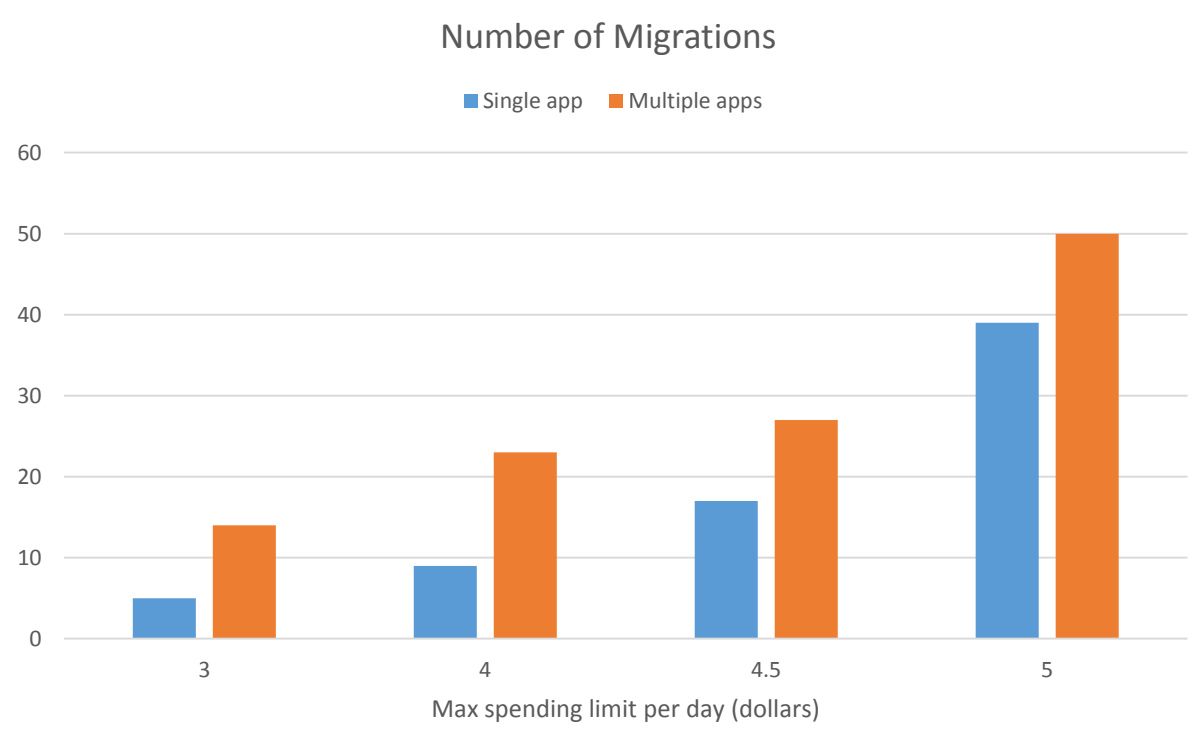
- Motivation
- Solutions
- Implementation
- Evaluation
- Conclusion

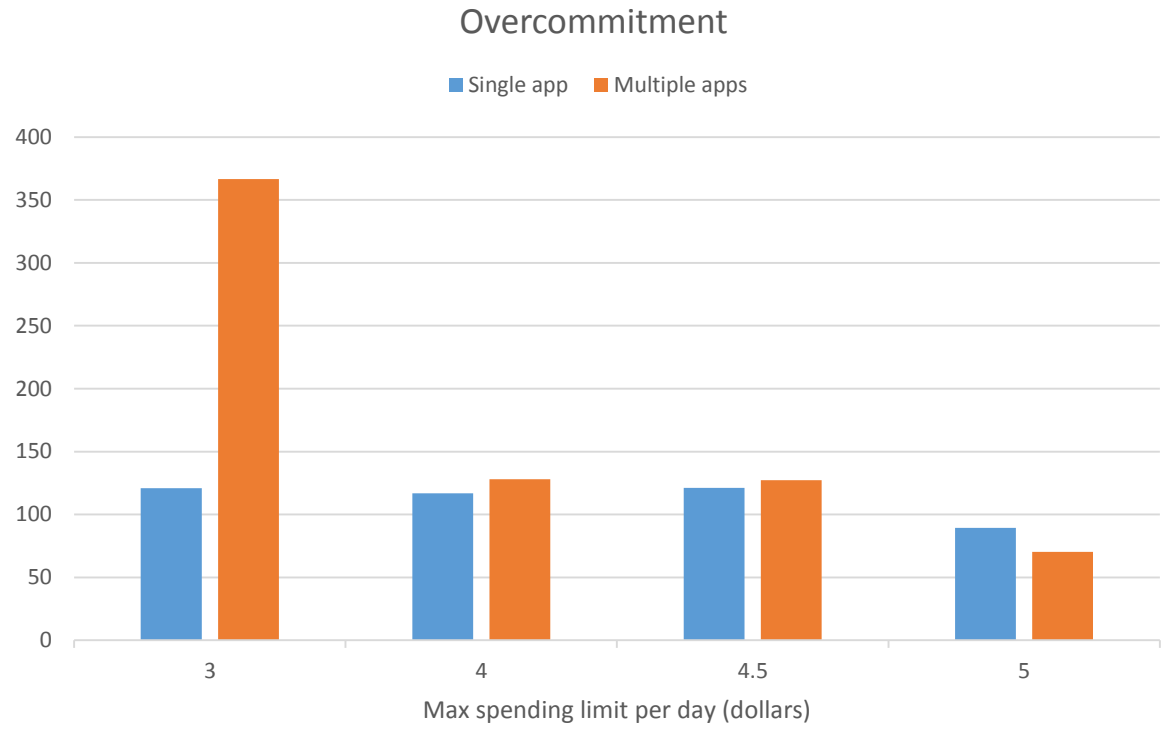
Experimental Setup

- **Proof of concept model(WIP)**
  - Live migrating SPEC benchmarks running in LXC
  - Observed downtime – 30 seconds (depending of process size)

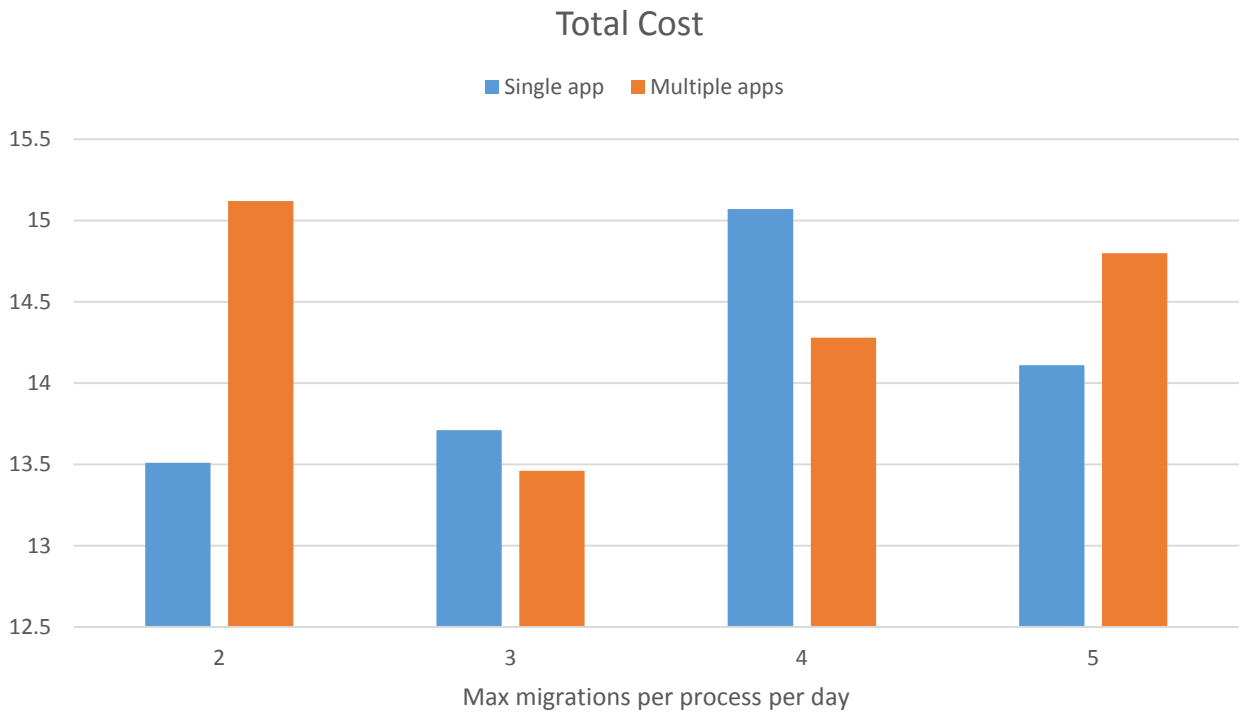
- **Migration Policy Simulations**
  - Used our own random workload generator
  - 2 workloads of each type – static, high variability and low variability

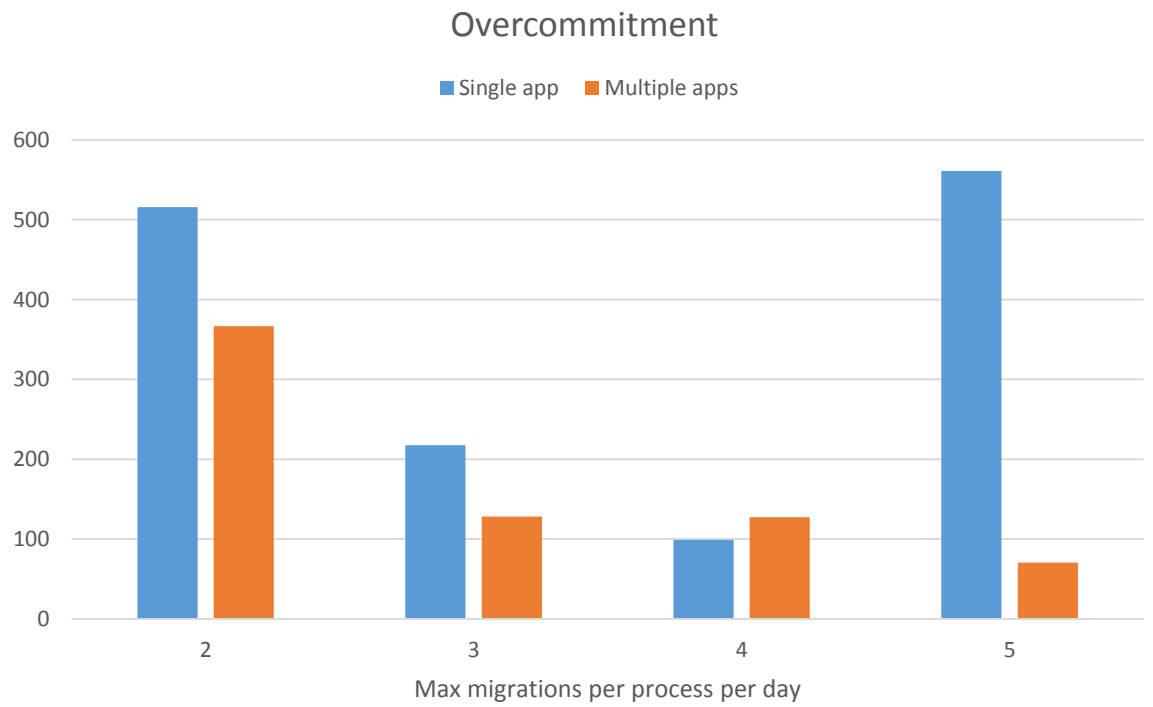
Capping Costs



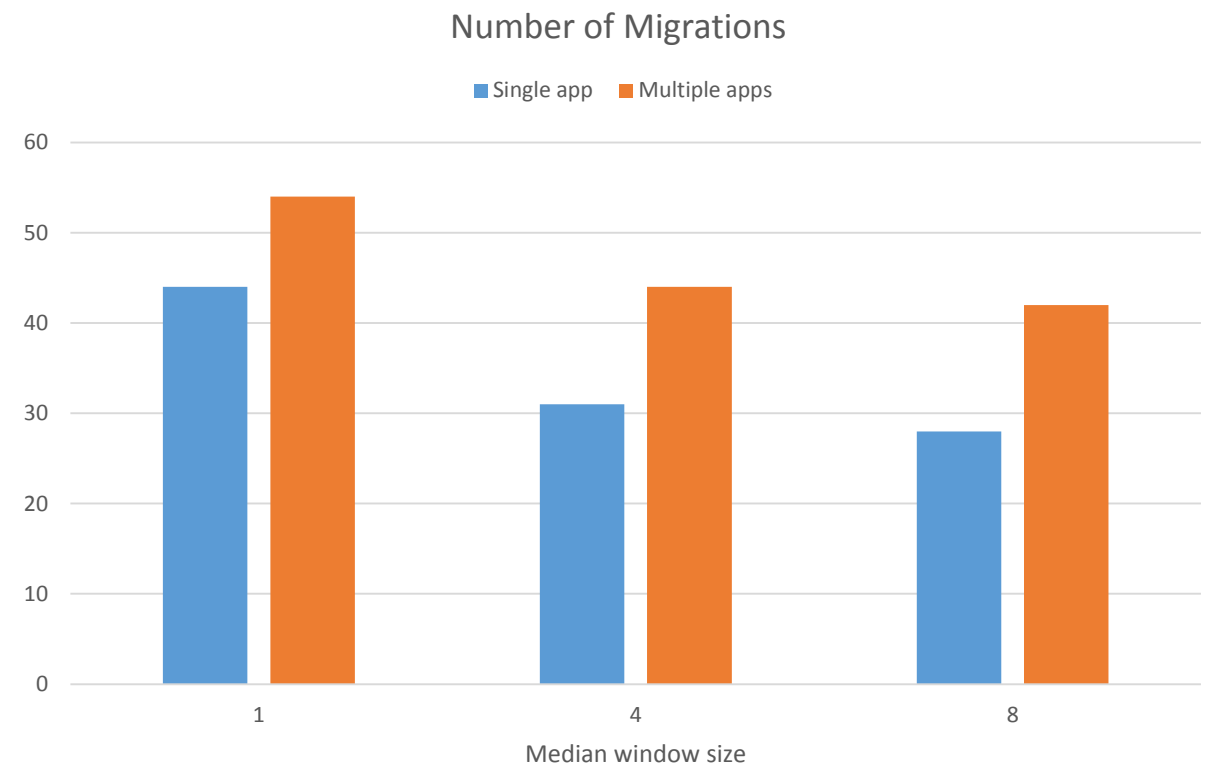


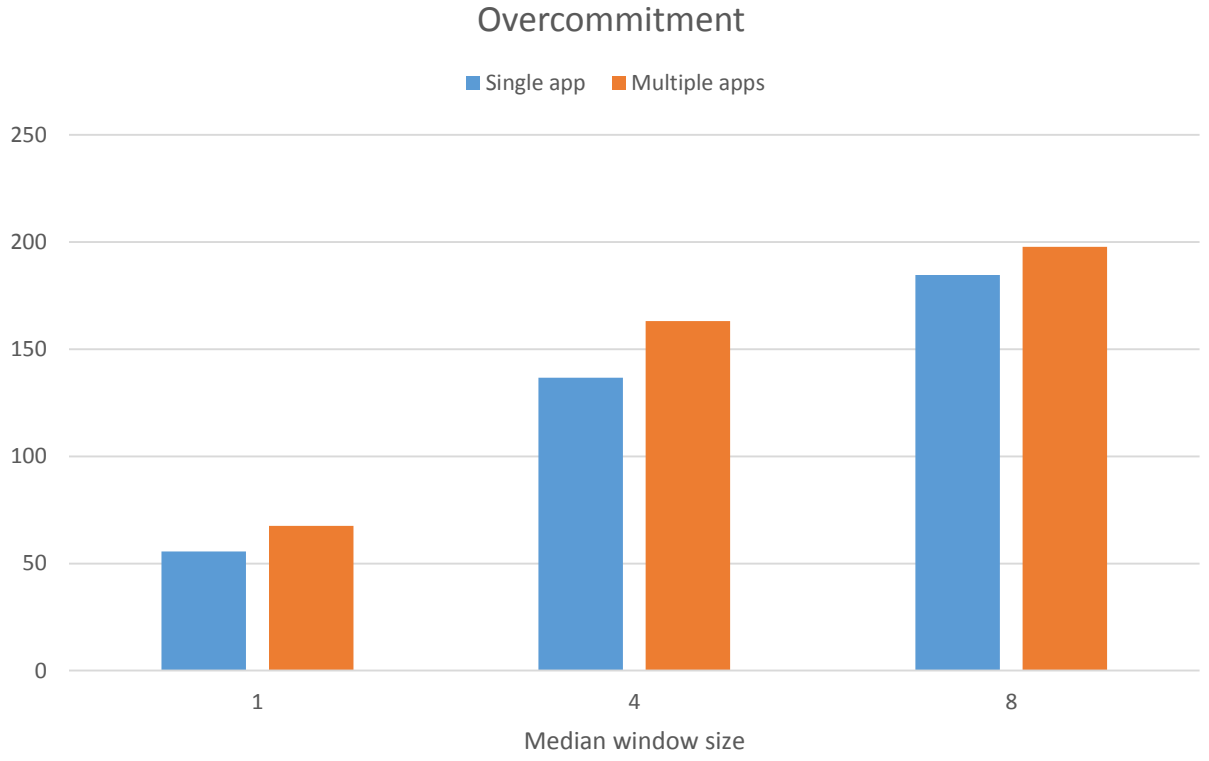
Constraining Downtime





Suppressing Spikes





Show me the money

- **Baseline**
  - Used same workloads as the simulation
  - Picked from available VMs that would best fit the workloads
  - No migrations!
  - Cost for 3 days - $45.36

- **Our solution**
  - No migration policy requires more than $15 for 3 days
  - 66% money saved!

Conclusions

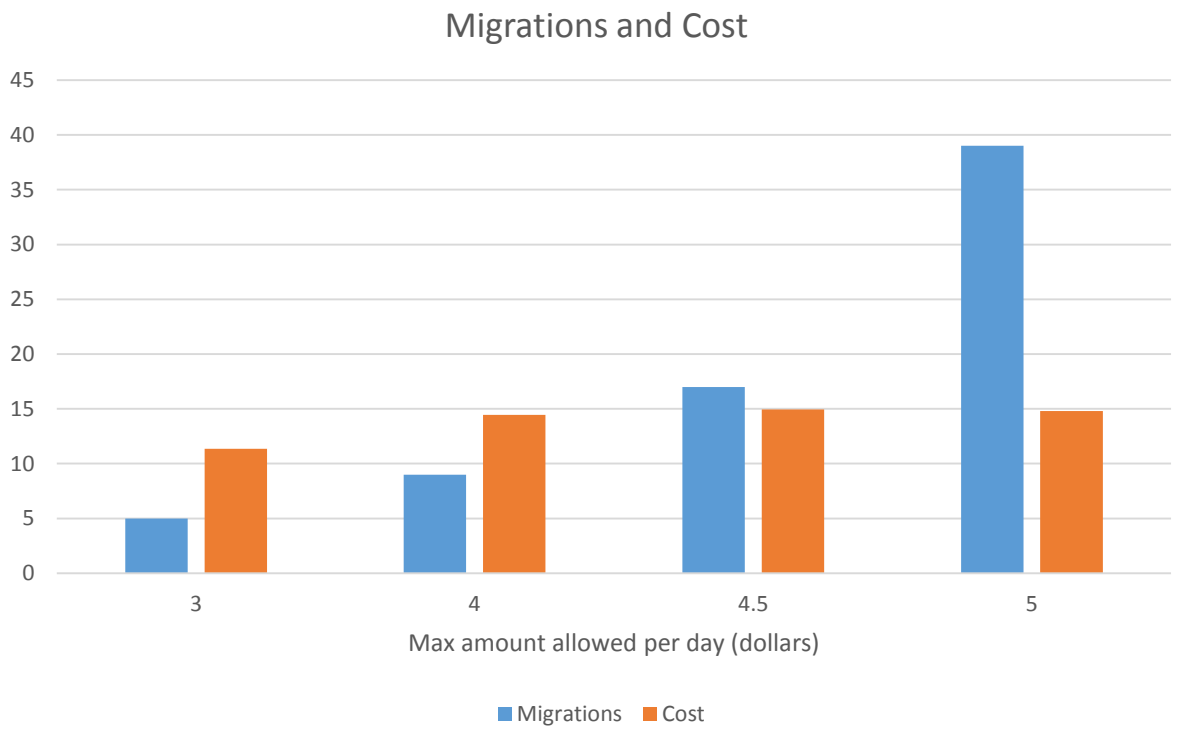
- Streamlining cloud operations important with increasing scale
- Current IaaS reservation models insufficient
- Better support needed from cloud providers
  - Amazon EC2 Container Service
- Migration policies have to optimize in a multi-dimensional space
  - Simple ones offer savings too!

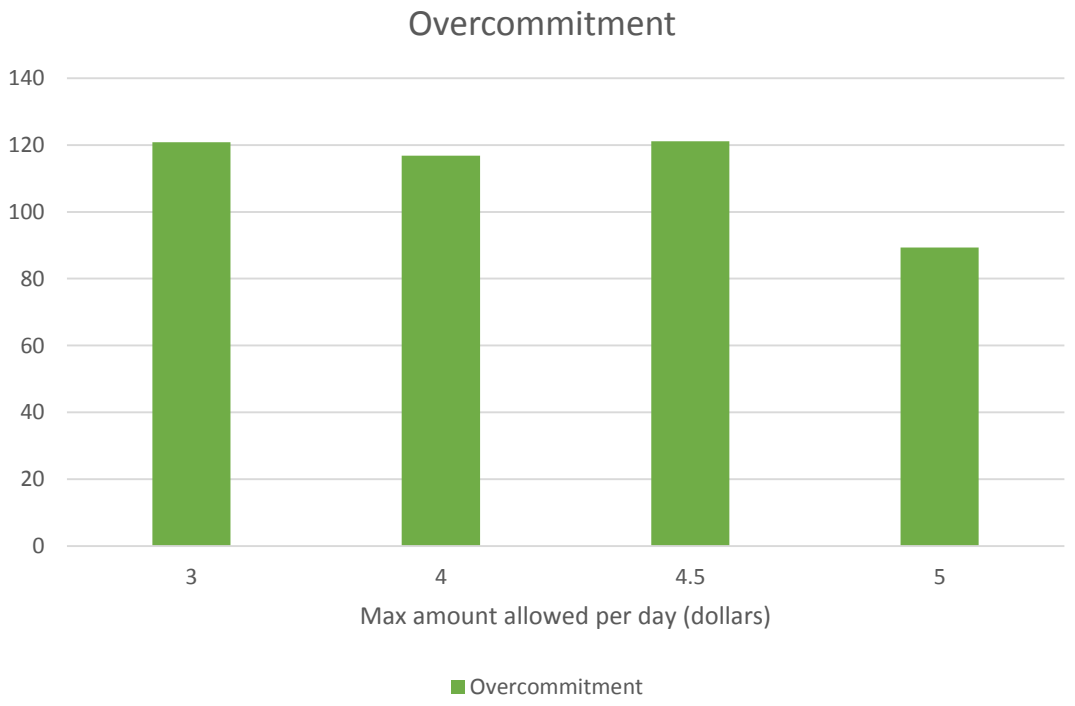
Questions?

BACKUPS

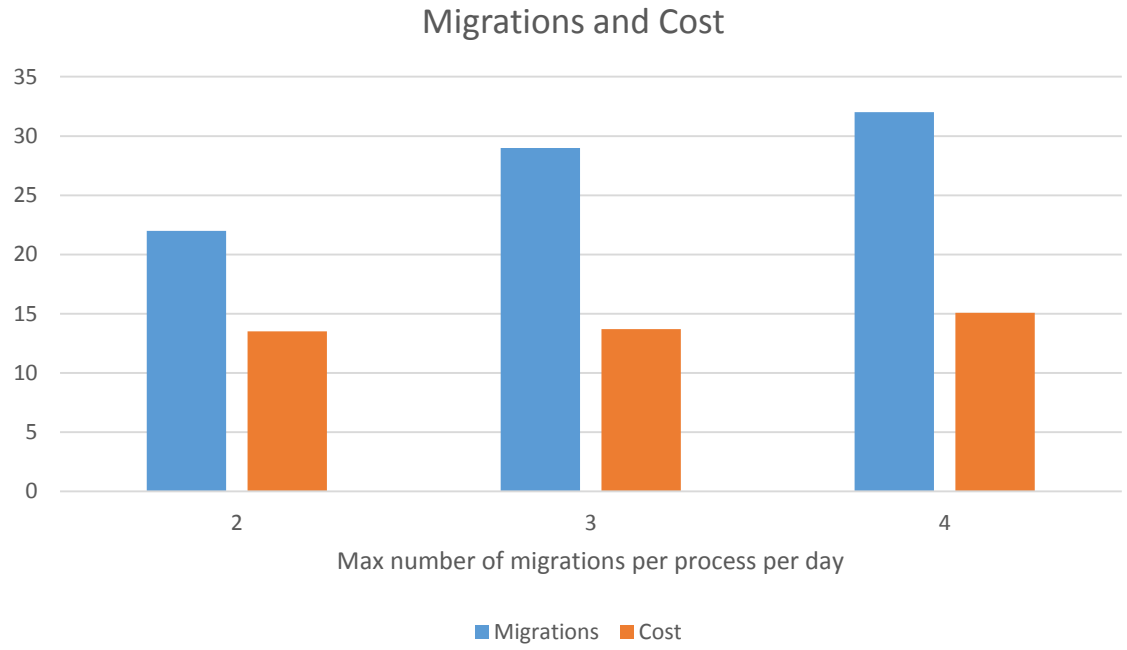
Single application per VM

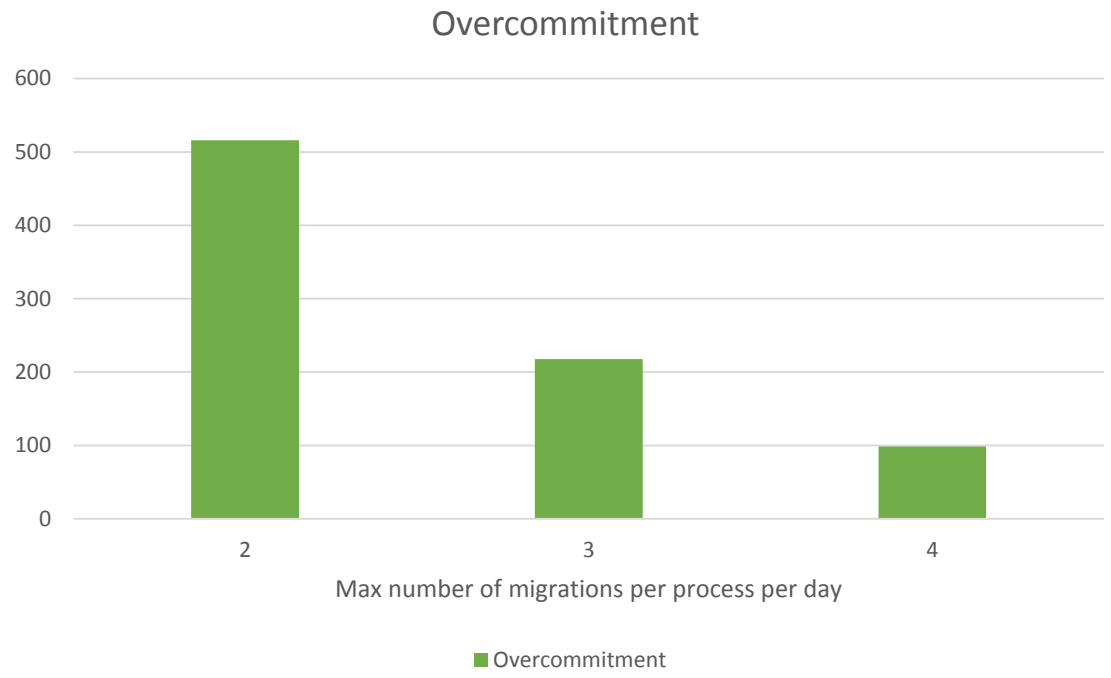
Effect of cost per day



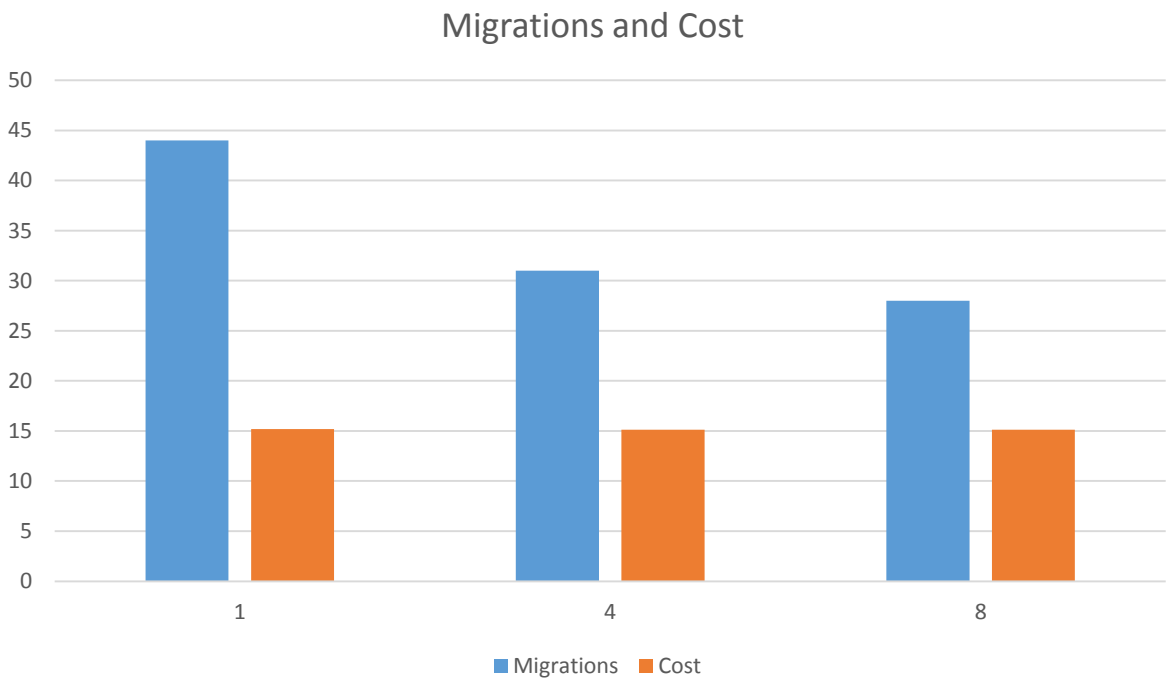


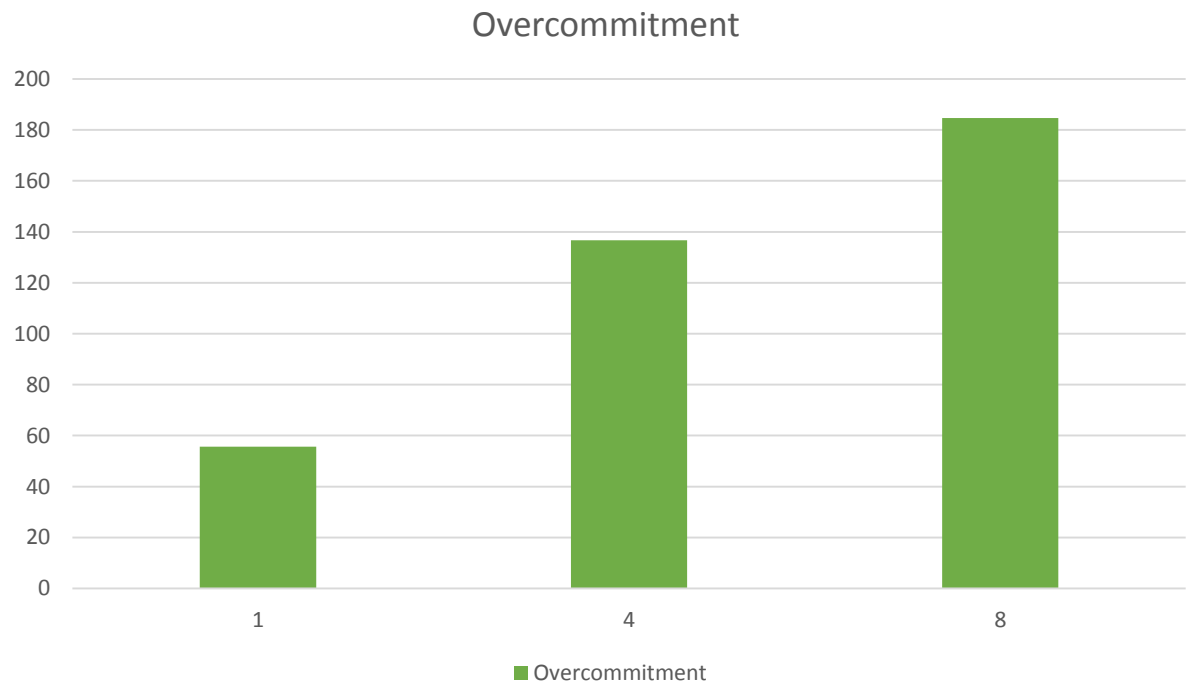
Migrations cap





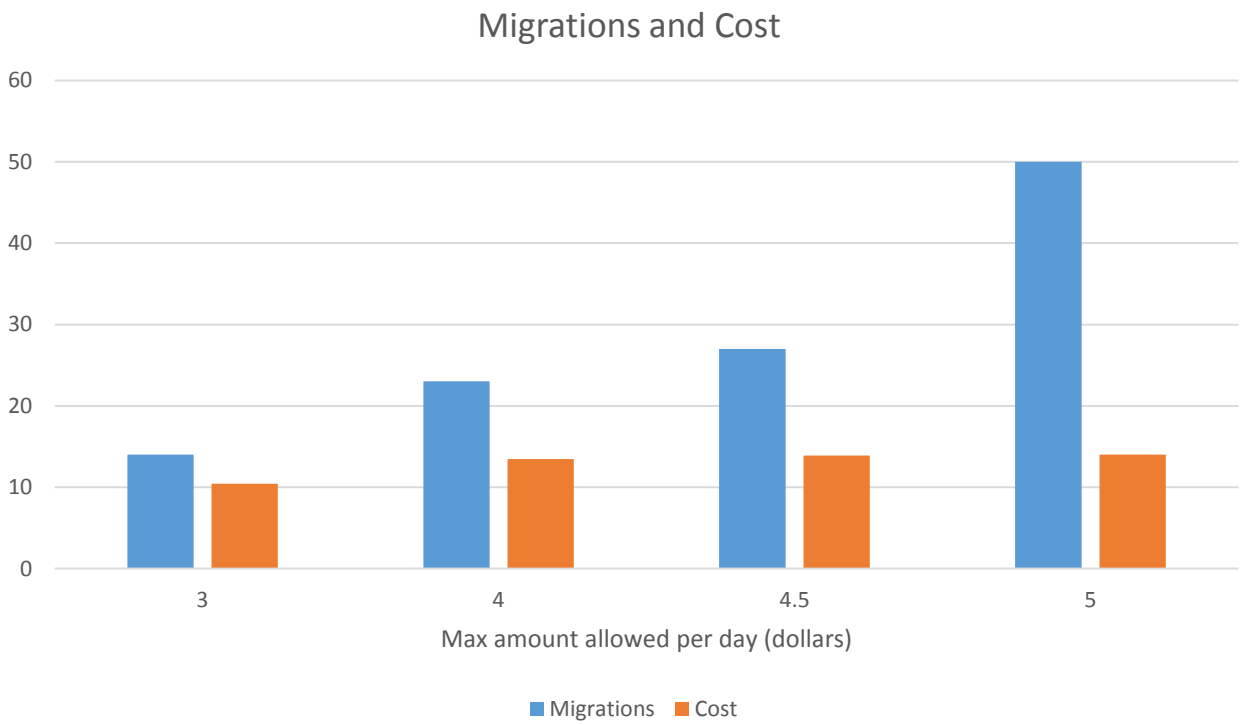
Median window variations

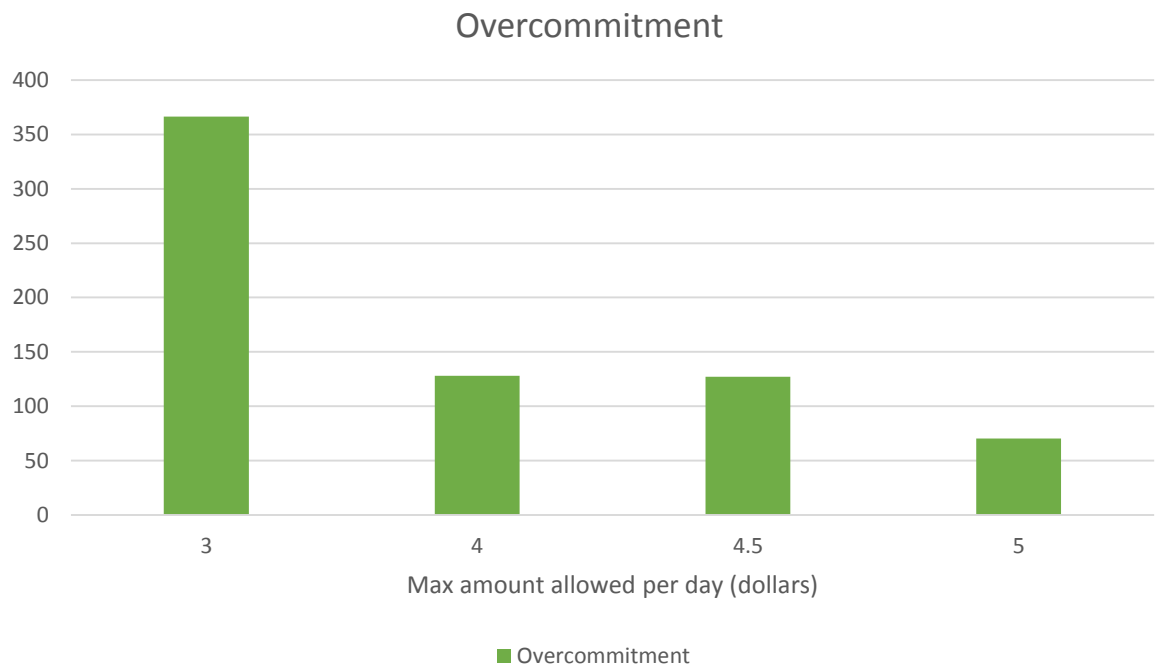




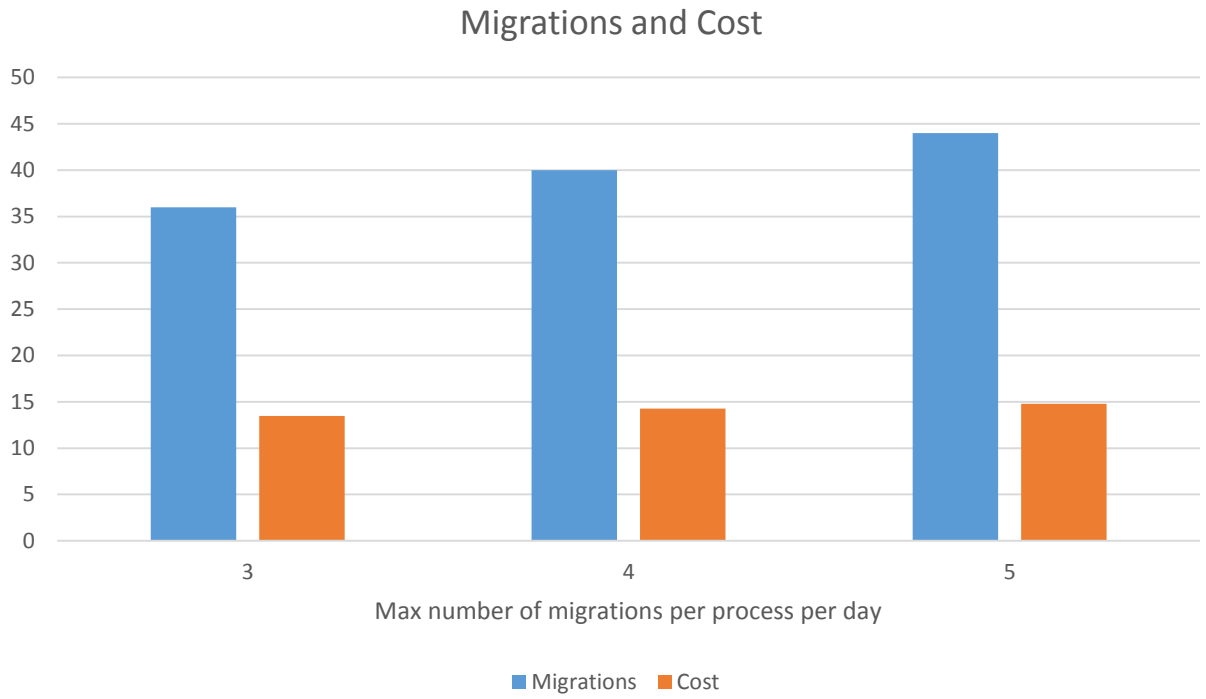
Multiple applications per VM

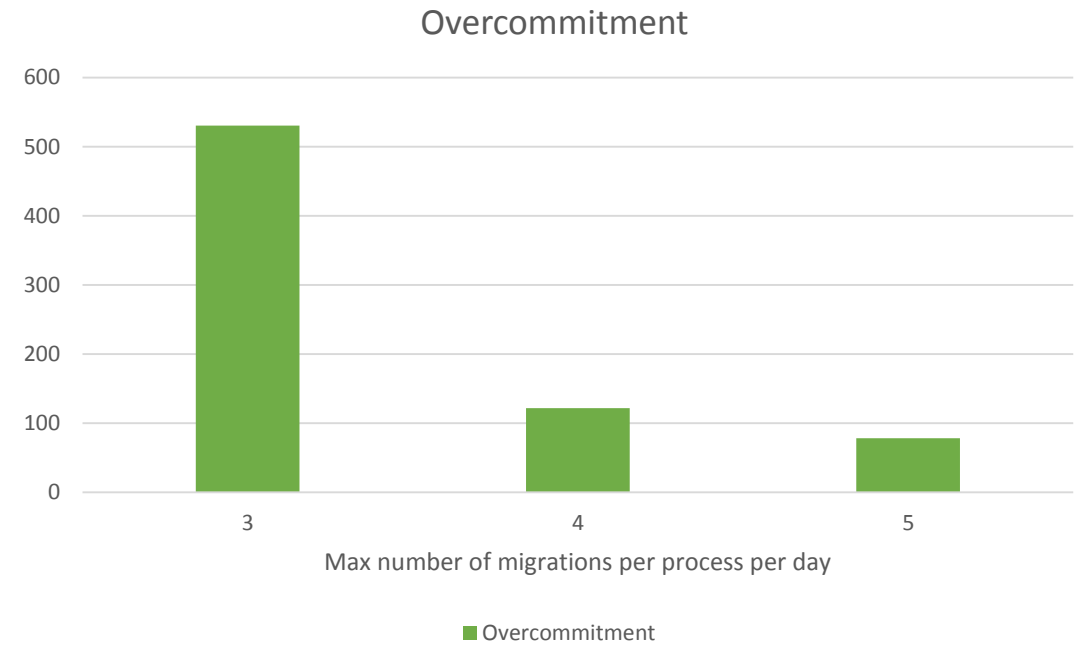
Effect of cost per day





Migrations cap





Median window variations