Tune Your Way To Savings!

Sandy Strong (@st5are)
Brian Martin (@brayniac)
About the Speakers

Sandy Strong
@st5are
SRE: Ads Serving

Brian Martin
@brayniac
SRE: Storage
Agenda

- Overview of Ad Server & Our Goals
- Getting Started
- Experimentation & Analysis
- Post-Launch Care & Feeding
- Insights & Takeaways
- Conclusions
Overview of Ad Server & Our Goals
About the Ad Server

- Runs on the JVM

- **Frontend**: routes Ads requests from clients to backend

- **Backend**: calculates Ads to display
What is our environment?

Environment: Aurora / Mesos

- Large deployment
- Abstraction over the DC resources
- Schedules jobs across machines
  - May be mixed platform types
- Shared vs Hybrid Mesos
Basic Controller for CPU Utilization & Latency

- Set point = target success rate (expectation)
- Output = current success rate
- Error = deviation from the expectation

Control variable = f(Error)
Key Metrics:

- QF: adaptive Quality Factor
- Latency (affected by CPU, Network)
- RPMq: Revenue Per Thousand queries
Hypothesis & Goals

Hypothesis:
- Greater control over how resources (CPU, network) are used will enable us to use them more efficiently.

Goal:
- Reduce the cost (resource footprint) of running Ad Server without adversely impacting revenue

Why?
- Small efficiency gains in a large service can result in large savings
Getting Started
Assemble a Team

- Mix of different skillsets
  - Site Reliability
  - Software Engineering
  - Hardware Engineering
Testing Environment

- Get hardware for a hybrid Mesos environment
  - Homogenous hardware platform
  - High-speed networking
  - Isolated from other workloads
  - Ability to tune for the hardware
High-Level Things We Can Change

- **Container Shape**
  - Shrink & shard wider
  - Use taller instances
  - How does this affect QF?

- **Mesos Resource Isolation**
  - Remove CPU throttling
  - Raise or eliminate network egress limits
Low-Level Things We Can Change

● System
  ○ Take control over scheduling
  ○ Enable hugepages

● Hardware
  ○ Enable/Disable Intel Turbo Boost

● Non-exhaustive list, many more experiments possible
Experimentation & Analysis
Areas to Explore

Things we thought of first:

- High-speed networking for all instances
- Disable Mesos network egress limits
- Disable Mesos CPU throttling (CFS)
Disable Egress Limit: Bandwidth (Frontend)
Disable Egress Limit: Latency (Frontend)

~20% improvement
Hugepages

What are hugepages?

- 2MB or 1GB pagesize vs 4KB default
  - architecture dependent
- 1GB pages must allocate at boot time
- Hypothesis: by using hugepages for the java process, we will see improvements to memory access patterns
Hugepages (1GB): Impact on QF

DEPLOYMENT

EXPERIMENT

CONTROL

> 15% improvement
NUMA

What is NUMA?

- Non-Uniform Memory Access
- Some memory is close, some is far
- Hypothesis: if we can control where our process allocates memory and schedules, we can gain efficiency by reducing foreign memory access
NUMA

- **2-socket systems**
  - Half of the RAM is in each zone
  - Memory access across zones is more costly

- **Non-pinned:**
  - Per-node process memory usage (in MBs) for PID 19494 (java)
    - Node 0 | Node 1 | Total
    - Total   | 10666.06 | 8476.95 | 19143.01

- **Pinned:**
  - Per-node process memory usage (in MBs) for PID 26969 (java)
    - Node 0 | Node 1 | Total
    - Total   | 16.84   | 32708.68 | 32725.52
NUMA Results

- Foreign memory access eliminated
  - 10% to 0%
- L1 Cache
  - 60% clockticks to 15% clockticks
- Cycles Per Instruction (CPI)
  - 1.54 to 1.23 (20% improvement)
Ad Server Backend: Instance Packing

- How will changing the process layout impact efficiency?
  - NUMA / CPU Pinning

- How many instances fit on a machine?
  - If we allocate all cores, this determines possible shapes of the Ad Server job
Ad Server Backend: Instance Reshaping

- How is the application affected by differently shaped containers?
  - 4 14-CPU instances/box $\neq$ 2 28-CPU instances/box
  - Hyperthreads: Backend exploits CFS

- What does it look like if we give the instance an entire NUMA zone?
  - More cores/threads and memory per instance
  - Fewer instances means more QPS per instance
  - More memory means larger heap to GC
Iterate and Refine

- Experiment, keeping notes about configurations and results
- Repeat until we have same or better performance
- Analyzing data can be challenging because...
  - Compute workloads vary by time, day, and world events
  - Adaptive nature of system in response to changing SR and latency
- Use load tests and redline tests to validate proposed configurations

<table>
<thead>
<tr>
<th>Exp. #</th>
<th>Summary</th>
<th>Outcome</th>
<th>Start</th>
<th>End</th>
<th>Cluster</th>
<th>Dashboards</th>
<th>Experiment Setup</th>
<th>Notes</th>
</tr>
</thead>
</table>
| 1a     | Run frontend and backend on hybrid mesos with network egress limit disabled and tc disabled | With 1.5x QPS, RPMq is x% lower than the control clusters | 2/10 1:30 PM | 2/12 1:30 PM | 5 RPMq  | - 2 frontends per host  
  - 1 backend per host  
  - tc disabled  
  - Network egress limit disabled  
  - QF target SR set to 99.9  
  - High-speed network enabled  
  - 1.5x QPS | Attribute Changes:  
  Disable CFS  
  Disable network egress  
  Disable tc  
  Expand port range  
  Enable turbo |
Ongoing Operational Work

● Hardware delivery & config automation
  ○ Agree on support model with stakeholder teams

● Fine-tune performance
  ○ Adapting as production workloads change
  ○ Weathering application regressions

● Iterative process
Insights & Takeaways
Experiments will never match reality

- Production != Lab
- Be realistic and practical
- Don’t be afraid to get it wrong
  - But have a plan for when you do
Stay focused on the metrics you’re trying to move

- Don’t get sidetracked by micro-optimizations
- Underweight outlier datapoints (avoid rabbit holes)
- Resist confirmation bias
Data-driven decision making

- Make sure you have the right audience for your data
- Think about how you are presenting your data
- Aim for consensus
- Share what you learn
Think about cost

- Cost of tuning efforts (+ ongoing efforts)

- Payout of tuning efforts
  - Reduce capex%
  - Reduce burden on downstreams and storage
Set realistic deadlines

- Account for the unexpected
- Stick to your deadlines
- Show delivery
Conclusions
• Unique challenges
  ○ Opportunity to learn, hands-on

• Coordinating cross-team
  ○ Think about all layers of the stack

• Resources saved
  ○ 6% reduction in CPU
  ○ >50% reduction in R/W to certain downstreams

• Desired metrics improved
  ○ QF substantially improved
What’s Next?
Upcoming/Ongoing experiments

- More experiments around CPU and NUMA pinning
  - Would 4-pack work better this way?

- Evaluate newer platform, with even more cores
Resources & Further Reading

Resilient Ad Serving at Twitter-Scale (https://t.co/qdmipEyRJy)

Clarifications on Linux's NUMA stats

Systems Performance - Brendan Gregg
Want to learn more?

Twitter is hosting an SRE Open House at HQ on Wednesday, March 15.

RSVP here:

https://t.co/2yLeAFrGcY