Moving Large Workloads from a Public Cloud to an OpenStack Private Cloud: Is It Really Worth It?

April 7th, 2016

Nicolas Brousse | Sr. Director Of Operations Engineering | nicolas@tubemogul.com

Who are we?

An **enterprise software company** for **digital branding**

- Filtered over **12.6 Trillion** Ad Auctions in 2015
- Served over **3 Billion** Ad Impressions on linear TV via our **PTV** solution
- Process bids in less than **50 ms**
- Serve bids in less than **80 ms** (includes network round-trip)
- Serve **5 PB** of monthly video traffic

Who are we?

A team of **Operations Engineers**

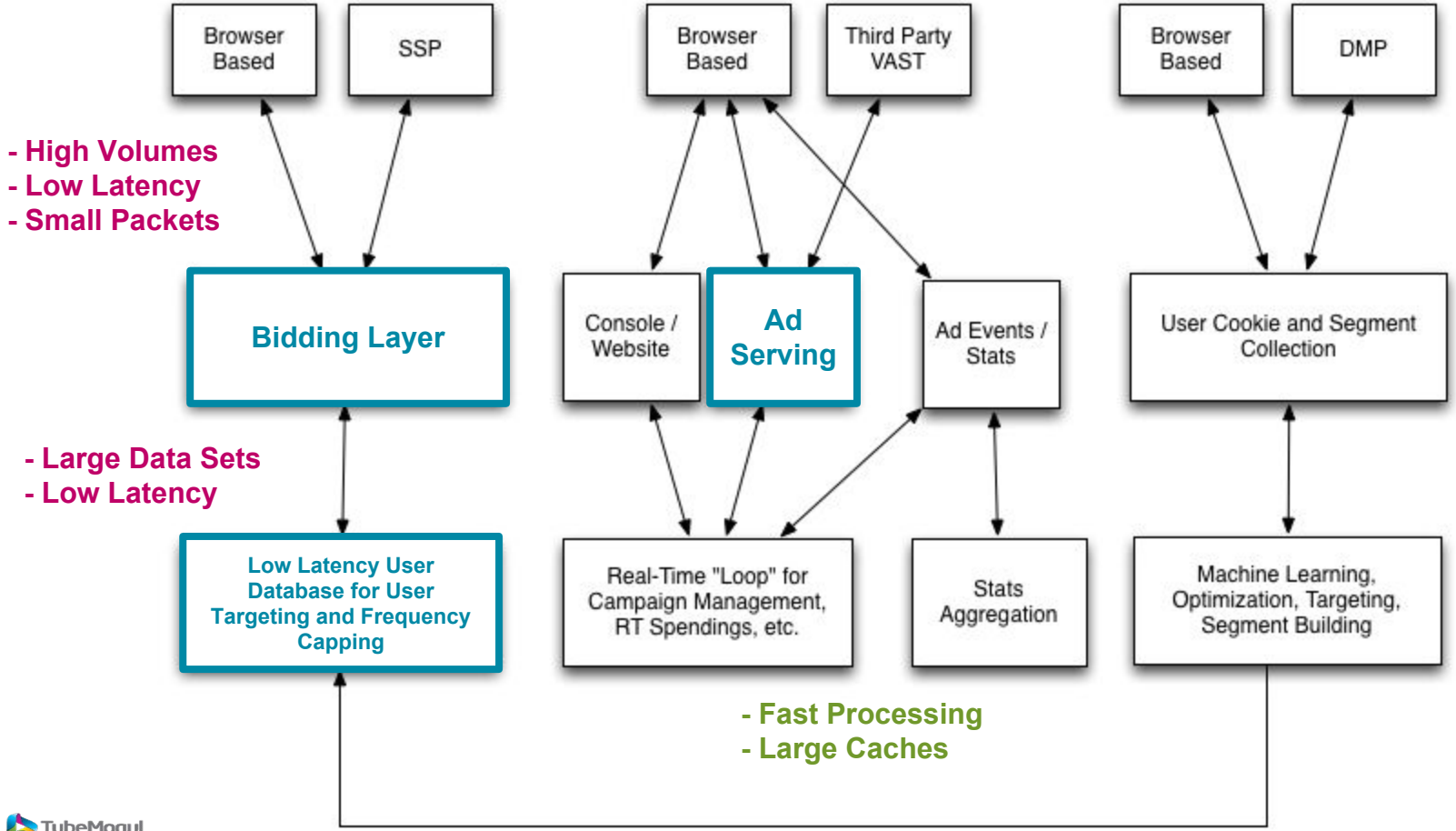
- Comprised of **SRE**s, **SE**s and **DBA**s
- Ensure the **smooth day-to-day** operation of the platform infrastructure
- Provide a **cost-effective** and **cutting edge** infrastructure
- Manage over **2,500 servers** (virtual and physical)

Mixed Infrastructure

TubeMogul

Public Cloud On Premises 2016 Private Cloud Deployment

High Level Technical Overview



Technology Hoarders

- Java (a lot!)
- MySQL (Percona, MariaDB)
- Memcached, Couchbase
- Aerospike, Vertica, Druid
- Kafka
- Storm
- Zookeeper, Exhibitor
- Hadoop, HBase, Hive
- Terracotta
- ElasticSearch, Logstash, Kibana
- Varnish
- PHP, Python, Ruby, Go...
- Apache httpd
- Nagios, Sensu
- Ganglia, Graphite, Grafana
- Puppet
- HAproxy
- OpenStack
- Git and Gerrit
- Gor
- ActiveMQ, RabbitMQ
- OpenLDAP
- Redis
- Blackbox
- Jenkins, Sonar
- RunDeck
- Tomcat, Jetty, Netty
- Qubole
- Snowflake
- AWS DynamoDB, EC2, S3, SWF...

Public Cloud: Technical Challenges

- Our **high volume** and **low latency** traffic makes our proximity to some partners matter.
- **Huge datasets** used for decisioning require high performance infrastructure, which **costs a lot**. Even with reserved capacity.
- Instances' **packet per second limitations** lead us to large public footprint and poor backend performances, especially for load balancers.
- **Network disruptions** with no root causes.

Our Strategy

Go in-house in 4 locations and 3 continents, in less than 6 months, using a ready-to-go cloud solution and two part-time engineers. Then, go celebrate in Vegas.

EASY!

Our Strategy (revised)

Go in-house in 4 locations and 3 continents, in ~~less than 6~~ 3 years ~~months~~, using a ready-to-go cloud solution and two part-time engineers. Then, go celebrate in Vegas.

EASY!

Our Strategy (revised)

Go in-house in 4 locations and 3 continents, in ~~less than 6~~ 3 years ~~months~~, using ~~a ready-to-go cloud solution~~ OpenStack and Bare Metal and two part-time engineers. Then, go celebrate in Vegas.

EASY!

Our Strategy (revised)

Go in-house in 4 locations and 3 continents, in ~~less than 6~~ 3 years ~~months~~, using ~~a ready to go cloud solution~~ OpenStack and Bare Metal and ~~two part-time~~ 3 dedicated engineers. Then, go celebrate in Vegas.

EASY!

Our Strategy (revised)

Go in-house in 4 locations and 3 continents, in ~~less than 6~~ 3 years ~~months~~, using ~~a ready to go cloud solution~~ OpenStack and Bare Metal and ~~two part-time~~ 3 dedicated engineers. Then, ~~go celebrate in Vegas~~ keep it up and running.

~~EASY!~~ YEAH!

TCO analysis: what to consider?

- Which infrastructure is being moved?
- How do you compare apples to apples?
- Do you plan to **overcommit**?
- Do you cost properly for engineering resources, software maintenances, and various support?
- Does your design make **trade-offs** on High Availability?
- Do you plan for a **test environment and R&D**?
- Are you using your public cloud at its best?
- Are you building a public cloud or **optimizing** your environment?
- Do you plan for **growth** and how does it impact your cost models?
- Which locations are you deploying to? What is the impact on bandwidth and data center costs?

Three simple TCO rules

- **Be Fair**: Challenge your Public Cloud partners and share your TCO with them.
- **Keep It Simple**: Limit the scope of your TCO to a clear and well known subset.
- **Make Clear Assumptions**: Have a defined list of feature sets on what you are building.

How did we start?

We built a test environment with **Eucalyptus** to move our integration environment on it.

How did we start (again)?

We built a test lab environment with **a vendor using CloudStack** to move our integration environment on it.

How did we start (really)?

We built a test lab environment and first data center location **ourselves using OpenStack on Gentoo** and shared the lab for our software engineers' integration environment.

First Failures and Lessons Learned

- **Do not share your lab**: Your lab is meant to fail and be destroyed. Don't assume people will be OK to work with something unreliable.
- Don't mess up your block storage strategy. **No last minute changes**.
- Starting a first data center location may require a lot of paperwork time, executive approval, and hardware mistakes. **Plan ahead**.
- **OpenStack is complex**. Don't make it more complex.

How did we reboot?

We **(re)**built our lab and prod environment with **a vendor using OpenStack on Ubuntu** to move our QA environment into one region.

How did we reboot (again)?

We (re)built our lab and prod environment **ourselves by using OpenStack on Ubuntu** to **move our production** environment into **one region**.

In Production...

- First traffic switch went smoothly and allowed us to **decrease our footprint** by **40%** and our load balancer footprint by **95%**.
- Progressive **traffic migration is not easy**. Consider the impact of multiple environments to maintain and all application dependencies.
- Load Balancers, and core data services run on **bare metal** and leverage **VLANs**.
- **Fully automated** bare metal and OpenStack provisioning.
- We are deploying **three new on-premise locations** in Q2 2016.
- **Limit scope** to our high volume and low latency infrastructure.

Lessons Learned

- OpenStack requires a **long learning curve and design phase**. Account for it, in terms of cost, skill-set, and time.
- We **are not** building a Public Cloud. **Be very clear on your feature set and business case**.
- You don't know your application as well as you think. Be ready to **adapt quickly** and don't overlook the impact of network traffic that **switches from private to public**.
- Really, you don't know your application as well as you think. Be ready to deal with **ip conntrack table full**.

So, is it worth it?

- Moving in-house led to an estimated **30% cost savings** and reduced our server footprint.
- The **improved visibility** on our network traffic and our full application stack greatly helped for **troubleshooting and performance improvements**.
- Have a **strong technical need** for it. **Cost shouldn't be the only driving factor**.

THANK YOU

www.tubemogul.com | @tubemogul

Nicolas Brousse @orieg