Deploying large payloads at scale

Ramon van Alteren
Hyves

• 9.7M dutch members (16.7M population)
• ~7M unique visitors / month (Comscore 09/2011)
• ~2.3M unique visitors / day
• 800.000 photo uploads / day
• 7M chat messages / day
Hyves - Operational environment

- 3500 node serverpark in 3 datacenters
- 6Gbps daily outgoing traffic
- System Engineering team: 12
- Development team: 33
Weekend project
Weekend project

Result: 4.5 x speed/throughput increase
Weekend -> Company project

HA HA!

BUSINESS
A Few Minor problems
A Few Minor problems

- compilation took ~40-60 minutes
A Few Minor problems

• compilation took ~40-60 minutes
• resulting binary was 750MB

Wednesday, November 9, 2011
A Few Minor problems

- compilation took ~40-60 minutes
- resulting binary was 750MB
- Code issues
A Few Minor problems

- compilation took ~40-60 minutes
- resulting binary was 750MB
- Code issues
- gcc 4.5.2 required + added deps
Part I - Solving the build problem

Jenkins to the rescue
Part I - Solving the build problem

Jenkins to the rescue

Add distCC to speed up compilation times
Part I - Solving the build problem

Add some serious hardware
Part I - Solving the build problem

Add some serious hardware

Compile / build in < 6 mins
Part II: Deploying Sequential
Part II: *Deploying*

Sequential?

- $500\text{MB} @ 1\text{Gb/s} = 4\text{ seconds}$
- $500\text{MB} @ 500\text{Mb/s} = 8\text{ seconds}$
- $500\text{MB} @ 200\text{Mb/s} = 20\text{ seconds}$
Part II: Deploying

Sequential?

500MB @ 1Gb/s = 4 seconds
500MB @ 500Mb/s = 8 seconds
500MB @ 200Mb/s = 20 seconds

450 servers * 8 seconds = 3600 seconds == 1 hour
450 servers * 20 seconds = 9000 seconds == 2.5 hour
Part II: *Deploying*

**Sequential?**

500MB @ 1Gb/s = 4 seconds
500MB @ 500Mb/s = 8 seconds
500MB @ 200Mb/s = 20 seconds

450 servers * 8 seconds = 3600 seconds == 1 hour
450 servers * 20 seconds = 9000 seconds == 2.5 hour

**Diffs?**
Part II: Deploying

Sequential?

- 500MB @ 1Gb/s = 4 seconds
- 500MB @ 500Mb/s = 8 seconds
- 500MB @ 200Mb/s = 20 seconds

450 servers * 8 seconds = 3600 seconds == 1 hour
450 servers * 20 seconds = 9000 seconds == 2.5 hour

Diffs?

- binary diff would be between 10KB - 400MB
- Even on consecutive runs without
Part II: Deploying - Bittorrent
Bittorrent - Previous experiences

Naive run using bittorrent to transport 300MB throughout our serverpark

• Near-complete network outage due to bandwidth starvation
• Several crucial subsystems delayed or unreachable due to network bandwidth shortage
Bittorrent - The Problem

Every server has 1Gb/s link to every other server
Bittorrent - The Problem

Every server has 1Gb/s link to every other server

they don’t
Bittorrent - Actual bandwidth available

1-4Gb/s

Core Network

Wednesday, November 9, 2011
Bittorrent - Actual bandwidth available

Core Network

Production traffic
Administration traffic
Murder - Why not?

Murder uses two tricks:

- Clients (including the seeder) capped to 1 upload peer
- Every client receives every peer from the tracker
- No download bandwidth cap (easy to add though)
Murder - Why not?

Murder uses two tricks:

- Clients (including the seeder) capped to 1 upload peer
- Every client receives every peer from the tracker
- No download bandwidth cap (easy to add though)

Peers will still connect all over the place
Murder - Why not?

Murder uses two tricks:

- Clients (including the seeder) capped to 1 upload peer
- Every client receives every peer from the tracker
- No download bandwidth cap (easy to add though)

Peers will still connect all over the place

It's slow, timing run over 25 peers took 7 mins
DIY: Location aware tracker

We can rebuild him.

We have the technology.
SMDB - Location metadata

We have bandwidth information available in our server management database

Build two-tier bittorrent swarms:

• 1 swarm with 2 peers / rack (uplink)
• 1 additional swarm per rack (uplink)
• cap every client @ 96mbit/s
DIY: 2-tier swarms
Tracker in python + Flask:

- 1100 lines of code (1900 with tests)
- Stores transfer metadata in redis
- Connects to our SMDB using REST
- Exposes REST interface
DIY - Client

We use a slightly modified rtorrent client

Same things twitter modified:

• Remove features related to operating on the big bad internet.
• Make various timeouts more aggressive
• No DHT, UPNP etc.

Nice bonus: RPC remote API
Results - single deploy ~300 hosts

- Without build
- With build

- Classic deploy
- Bittorrent deploy

Wednesday, November 9, 2011
mainweb Cluster Network last 10min

- **In**: Now: 1.6G, Min: 1.4G, Avg: 3.4G, Max: 11.2G
- **Out**: Now: 954.0M, Min: 828.4M, Avg: 2.8G, Max: 10.6G
Graphs - single node

web1205 Network last 10min

Bits/sec

In: Now: 6.2M Min: 5.0M Avg: 13.5M Max: 71.9M
Out: Now: 3.5M Min: 3.2M Avg: 8.1M Max: 42.9M

Wednesday, November 9, 2011
Graphs - single rack (24 nodes)

ospf3.internal - Traffic - ethernet2/24 - DY05


- Inbound: Current: 102.23 M Average: 107.24 M Maximum: 256.52 M
- Outbound: Current: 154.40 M Average: 154.45 M Maximum: 313.13 M
Bitorrent - Statistics

== release: 101482 expected: 287 actual: 107 seeders: 0 progress: 0.00% start: 12:04:20 last_completed: none failed: 0 ==
== release: 101482 expected: 287 actual: 267 seeders: 0 progress: 0.00% start: 12:04:20 last_completed: none failed: 0 ==
== release: 101482 expected: 287 actual: 286 seeders: 0 progress: 0.00% start: 12:04:20 last_completed: none failed: 0 ==
== release: 101482 expected: 287 actual: 287 seeders: 1 progress: 0.35% start: 12:04:20 last_completed: none failed: 0 ==
== release: 101482 expected: 287 actual: 287 seeders: 2 progress: 42.11% start: 12:04:20 last_completed: 12:06:01 failed: 0 ==
== release: 101482 expected: 287 actual: 287 seeders: 5 progress: 44.48% start: 12:04:20 last_completed: 12:06:05 failed: 0 ==

Wednesday, November 9, 2011
Next Steps

Enable more projects

• currently only three main projects
• one codebase
• Multi-transfer

Move to Continuous Delivery

• currently doing continuous deployment
• 6-12 deploys a day
• want to allow feature teams to deploy
Next Steps

Open Source the tracker:

- Very closely tied to our infra
- Not overly clean code

We will open source it, however some refactoring is needed

contact me if you’re interested.

Watch our github repository:
https://github.com/organizations/hyves-org
Next Steps - Deployment Glue

TODO:

[] prepare stuff
[] transport stuff
[] do some more stuff
[] Activate payload
[] post check
Next Steps - Deployment Glue

Simple from a single host perspective

Complex when executed in parallel, remotely, with failure handling and proper reporting
Thank you, Questions?

Flask: http://flask.pocoo.org/
Rtorrent: http://libtorrent.rakshasa.no/
Twitter - Murder: https://github.com/lg/murder
Boris, Cor, Lorenzo & others at hyves.nl
Michael Tekel: https://github.com/mtekel/
http://wiki.theory.org/BitTorrentSpecification

You ? We’re hiring: http://werkenbijhyves.hyves.nl
email: ramon@hyves.nl
twitter: @ramonvanvanalteren