Xikolo
Social · Massive · Online
Scalable & Extensible Teaching System

Xikolo Infrastructure
openHPI, openSAP and beyond

Linux Containers Seminar

Malte Swart
Hasso-Plattner Institute

Potsdam, Germany, May 6, 2015

Table of Contents

Tracing A Web Request

External LoadBalencer

- terminates all external communication
- does HTTPS offloading (internal net is protected)

External LoadBalencer

- terminates all external communication
- does HTTPS offloading (internal net is protected)
- responsible for all here hosted domains: `open.hpi.de`, `open.sap.com`, `codeocean.openhpi.de`, `blog.openhpi.de` ...
- based on domain forwarding to different sets of backend services (up to 10 per set/domain)

External LoadBalencer

- terminates all external communication
- does HTTPS offloading (internal net is protected)
- responsible for all here hosted domains: `open.hpi.de`, `open.sap.com`, `codeocean.openhpi.de`, `blog.openhpi.de` ...
- based on domain forwarding to different sets of backend services (up to 10 per set/domain)
- (software) LoadBalencer for failover

web-VM (Backend-VM)

By wiki.nginx.org

- User facing web server: NGinx

web-VM (Backend-VM)

By wiki.nginx.org

- User facing web server: NGinx
- forwards special services like piwik (provided on subpath) to special VMs

web-VM (Backend-VM)

By wiki.nginx.org

- User facing web server: NGinx
- forwards special services like piwik (provided on subpath) to special VMs
- serves application assets (images, javascript ...) directly
- provides basic cache header and compression options
- forwards remaining requests to application

Frontend-App

From wikipedia

From rails bootstrap app

- each worker processes requests serial (unicorn)
- current frontends: web, API and some very special purpose frontends
- main tasks of frontends: session management, authentication and authorization, template rendering + frontend stuff (html ...)
- data are requested from services via HTTP/JSON

HTTP-Request Cache

By Peter Gilg

- stored within central Redis (key-value storage)
- cache responses for statically objects like user, course
- cache is invalided on data change → should be up to date at every time

Internal LoadBalencer

By haproxy.org

By wiki.nginx.org

- HAProxy for large instances like openHPI and openSAP
- monitors list of possible upstream continuously via health check
- redirects internal HTTP requests to one available backend
- for single instance: Nginx sites instead of load balancing

Service

From wikipedia

From rails bootstrap app

- again Ruby on Rails
- currently each backend server runs every service
- provides JSON api for frontends and other services

Database

PostgreSQL
the world's most advanced open source database

Provided by The PostgreSQL Community Association

- every service has individual database, could use their matching database tools
- mostly PostgreSQL 9.1
- but e.g. ElasticSearch, too

RabbitMQ

RabbitMQ
Messaging that just works
Provided by Alexis Richardson

- Asynchronous communication between services
- Services publishing messages like new user created
- other services subscribe to all events they need

Sidekiq

Sidekiq

Simple, efficient background processing for Ruby.

From sidekiq.org

- Background processing

Tracing A Commit / A Release

A New Commit

GitLab.com

Provided by Gitlab

- pushed into central gitlab

Testing it: TeamCity

TeamCity

Provided by JetBrains

It gets a little bit complicated!

In general: Teamcity monitors Gitlab for new changes

Testing it: TeamCity (2)

Unit Tests

- Independent per service: all external communication is stubbed / disabled
- Ensure that service fulfils request specification

Testing it: TeamCity (3)

Integration Tests

- Goal: Testing all together
- checking out all services in its version
- running all services in productive-ish configuration
- executes selenium tests to check whether the interaction between services is working

Testing it: TeamCity (4)

Packaging

- Again independent per service
- building deb packages from source code
- skips optional things like test directory
- embedding all needed gems for service
- compiles assets (if needed)
- generates upstart scripts for all components (unicorn, sidekiq ...)

Testing it: TeamCity (5)

Release

- chooses a revision combination that resulted in green integration
- is executed automatically for opensap staging / manual for all other instances
- publishes packages into instance specific Debian repository (packages are reused)

Puppet

Provided by PuppetLabs Media Kit

- runs on every productive VM
- upgrades Xikolo packages with new ones from repository
- executes database migrations
- injects production specific configurations (database credentials, logging settings ...)
- manages Xikolo configuration and service endpoint mapping

Anything Else?

What's left?

User Content

- Beside database records primary files
- shared between nodes via central NFS

E-Mail

Central e-mail relay

Logging

Central Syslog Server

All Together

Basic Structure

- Frontend-LoadBalencer
- Compute nodes with VMs
- database server with PostgreSQL, Redis, RabbitMQ, ElasticSearch
- one infrastructure server one with syslog, exim, nfs-server

What’s next?

Mid/long-term improvements

- Increase reliability (current 3 server must not fail)

Mid/long-term improvements

- Increase reliability (current 3 server must not fail)
- Increase maintainability (current 3 server can not be upgraded/adjusted without downtime)

Mid/long-term improvements

- Increase reliability (current 3 server must not fail)
- Increase maintainability (current 3 server can not be upgraded/adjusted without downtime)
- Convert pull component of deployment process (puppet) into push process

Mid/long-term improvements

- Increase reliability (current 3 server must not fail)
- Increase maintainability (current 3 server can not be upgraded/adjusted without downtime)
- Convert pull component of deployment process (puppet) into push process
- Make instances more independent with regard of more special configuration / service list

Mid/long-term improvements

- Increase reliability (current 3 server must not fail)
- Increase maintainability (current 3 server can not be upgraded/adjusted without downtime)
- Convert pull component of deployment process (puppet) into push process
- Make instances more independent with regard of more special configuration / service list
- More precise scaling (more specific per service or service process like unicorn, msgr, sidekiq)

Mid/long-term improvements

- Increase reliability (current 3 server must not fail)
- Increase maintainability (current 3 server can not be upgraded/adjusted without downtime)
- Convert pull component of deployment process (puppet) into push process
- Make instances more independent with regard of more special configuration / service list
- More precise scaling (more specific per service or service process like unicorn, msgr, sidekiq)
- Replace central file system (NFS) with distributed service approach