Ganeti in Debian, Skroutz, and other stories

Apollon Oikonomopoulos

apoikos@debian.org

mini GanetiCon 2017
15 Dec 2017 — Leipzig, DE

About me

- System Administrator/Head of Infrastructure at Skroutz
- Debian Developer, maintainer of the Ganeti Debian package
- Long-time Ganeti user (Skroutz, GRNET) and contributor
- Original Ganeti Manager author
- Free Software enthusiast

Outline

- Ganeti & Debian
- Ganeti @ Skroutz
- Where do we go next?

Ganeti & Debian

- A long relationship
  - Two of the initial authors are Debian Developers
- First appeared in Debian in 2007 (1.2 beta3)
- Part of every stable release since Lenny
- Debian has historically been the best tested/supported platform for running Ganeti

Ganeti in Debian

- Team-maintained: `pkg-ganeti-devel@lists.alioth.debian.org`
- 3 team members (1 active, help appreciated :)
- Package source tracked in `git` `http://anonscm.debian.org/gitweb/?p=pkg-ganeti/ganeti.git`
- Generally in good shape, with automated integration tests

Current status

- 2.15.2 in stretch/buster/unstable
  - Heavily patched (20 patches!) to keep up with current dependency versions
  - Backported non-DSA SSH key support to Stretch (2.15.2-7+deb9u1)
- 2.16.0~rc1 in experimental
  - Built with GHC 7.10, outdated
- Versioned packages allow cluster-wide upgrades via `gnt-cluster upgrade`
- Despite development slowing down, `popcon` shows increasing popularity

Interlude: Ganeti @ Skroutz

skroutz

Skroutz — what we do

Main product: price comparison engine

- ∼ 2.5k e-shops
- 6M products
- ∼ 1M visits / day
- 3 countries (GR, TR & UK)

skroutz

What we use

- Debian
- Ganeti
- Ruby on Rails
- MariaDB
- H2O
- HAProxy
- Varnish
- Elasticsearch
- MongoDB
- Redis
- Kafka
- …

skroutz

Infrastructure

- 100 physical servers
- 300+ virtual machines
- 3 physical locations
  - production site
  - DR site
  - HQ

skroutz

Ganeti at Skroutz

Runs production, development and staging instances. Production:

- Elasticsearch clusters
- alve.com and scrooge.co.uk appservers
- Redis
- Side-project servers
- Analytics infrastructure
- Core services (LDAP, mail, DNS, monitoring) …

skroutz

Ganeti at Skroutz (2)

A *single* Ganeti cluster with…

- 17 nodes
- 3 nodegroups (one per location)
- DRBD (using secondary IPs)
- ganeti-os-d-i (more later)

skroutz

Staging instances

- Staging ”cluster”: 3-4 instances (app server, ES server, DB server), with iSCSI-backed disks
- Access to fresh snapshots of production data
- Automated instance creation and cleanup via RAPI
  - CLI tool giving control to developer teams
- Modified Ganeti Manager to allow cluster creation using a multi-alloc RAPI call
- Wrapped RAPI to restrict instance operations via Ganeti Manager’s interface to specific domain suffixes only

skroutz

systemd integration

- All nodes run Jessie or Stretch with systemd
- Normally KVM instances appear in the `ganeti.service` (or `cron.service`…) cgroup
  - Not especially pretty
  - Does not allow setting individual process limits easily, e.g. using `systemctl set-property`
  - Risks killing KVM instances on service stop when `KillMode` is `mixed` or `control-group`

skroutz

systemd integration (2)

- systemd allows creating *scope* units corresponding to externally managed processes and optionally placing them under a different *slice*
- Idea: use `systemd-machined`'s DBUS interface to create scope units for VMs
- Implemented as a post-{create,startup,migrate,failover} hook, but code is minimal enough to include directly in `hv_kvm`
- All KVM instances happily reside in `ganeti.slice`, `machinectl` shows instances running on the node

skroutz

ganeti-os-di

- Managing OS images is *hard*:
  - needs regular updating (security/point release updates)
  - needs careful cleaning before use (logs, puppet certificates, SSH keys)
  - error prone: e.g. hooks did not clean up ECDSA SSH host keys properly

skroutz

ganeti-os-di

- Managing OS images is *hard*:
  - needs regular updating (security/point release updates)
  - needs careful cleaning before use (logs, puppet certificates, SSH keys)
  - error prone: e.g. hooks did not clean up ECDSA SSH host keys properly
- Idea: bootstrap our instances using the Debian installer in a throwaway KVM instance
  - boots kernel/initramfs from debian-installer-X-netboot-amd64
  - (unsafe) writeback caching for speed
  - integrates well with our preseeding config
  - instances are created up to date, no need for dist-upgrades
  - no installer code runs in the node's context
  - small speed penalty: approx. 2min instead of 1min with g-i-m
  - implemented as a "traditional" OS provider

skroutz

Where do we go next?

General development

- Development has slowed down significantly
  - Last stable release: 2 years ago
  - Last `git` tag: 22 months ago
- Lots of patches floating around (Debian, Skroutz, GRNET, …) and not being consolidated
- Lack of a clear feature/technical roadmap

Python 2

- Python 2 is being deprecated (EOL in 2 years)
- Python 3 is a mature platform; all dependencies should be available
- Python 3 has an (optional) static type system
  - One of the original reasons for choosing Haskell IIRC

GHC

- Haskell is nice and safe etc. but...

GHC

- Haskell is nice and safe etc. but...
- APIs change frequently, and they change a lot

GHC

- Haskell is nice and safe etc. but...
- APIs change frequently, and they change a lot
- Static linking makes rebuilding everything cumbersome
  - Automated rebuilding of *libraries* on Debian, but not leaf packages
  - Last binNMU (binary rebuild) waited 28 days in the queue for the library transitions to complete
    - … and we lost Ubuntu 17.10's merge window

GHC

- Haskell is nice and safe etc. but...
- APIs change frequently, and they change a lot
- Static linking makes rebuilding everything cumbersome
  - Automated rebuilding of *libraries* on Debian, but not leaf packages
  - Last binNMU (binary rebuild) waited 28 days in the queue for the library transitions to complete
    - … and we lost Ubuntu 17.10's merge window
- Rock-solid in general, but awfully hard to debug when something breaks (see e.g. #751886)

GHC

- Haskell is nice and safe etc. but...
- APIs change frequently, and they change a lot
- Static linking makes rebuilding everything cumbersome
  - Automated rebuilding of *libraries* on Debian, but not leaf packages
  - Last binNMU (binary rebuild) waited 28 days in the queue for the library transitions to complete
    - … and we lost Ubuntu 17.10's merge window
- Rock-solid in general, but awfully hard to debug when something breaks (see e.g. #751886)
- Very steep learning curve, not a sysadmin-friendly language; effectively kills external contributions

The way forward

- IMHO Ganeti can survive *only* as a *community project*
  - No single company uses all features
  - Few companies can allocate resources full-time for extended periods of time
- We need to foster community maintenance and lower the barrier for contributions
  - Drop the CLA, adopt Developer's Certificate of Origin instead if needed
  - Grant commit access to trusted external contributors
  - Think twice before re-implementing Python parts in Haskell

The way forward

- IMHO Ganeti can survive *only* as a *community project*
  - No single company uses all features
  - Few companies can allocate resources full-time for extended periods of time
- We need to foster community maintenance and lower the barrier for contributions
  - Drop the CLA, adopt Developer's Certificate of Origin instead if needed
  - Grant commit access to trusted external contributors
  - Think twice before re-implementing Python parts in Haskell
    - Why not re-implement Haskell parts in Python 3?

Thank you!

Q&A