Running an SME on Debian
or “Managing Debian across the whole fleet”

Apollon Oikonomopoulos
apoikos@debian.org

DebConf16 - Cape Town
2016-07-04
Outline

Introduction

Installation

Configuration management

Package management

People
About me

▶ apoikos@d.o
▶ Head of Infrastructure at skroutz.gr
▶ Linux user since 1999, Debian user/admin since 2006
▶ xmobar\textsubscript{2009} \rightarrow (more packages) \rightarrow DM\textsubscript{2013} \rightarrow DD\textsubscript{2014}
▶ Mostly packaging work, mostly server stuff
▶ Local DSA contact for the GRNET machines
Debian across the fleet: a success story

- scrøøge skroutz.gr
- Product search/comparison engine
- The most visited Greek webpage
- 600k visitors daily, 5.5M unique visitors/month
- 150 employees in Greece
skroutz.gr infrastructure

- 85 physical servers
- 280 KVM VMs managed by Ganeti
- 3 physical locations (collocated)
- Redundancy/HA
- 4 sysadmins doing infrastructure/operations
- 1 office IT admin
What skroutz.gr uses Debian for

(Almost) everything that can run it
- Servers (physical and virtual)
- Routers
- Developers’ workstations/laptops
- Non-tech staff workstations
- Pi’s connected to TVs
What skroutz.gr uses Debian for

(ALmost) everything that can run it
- Servers (physical and virtual)
- Routers
- Developers’ workstations/laptops
- Non-tech staff workstations
- Pi’s connected to TVs

We don’t run Debian on our switches (yet?)
Servers

- Full HTTP stack: HAProxy → Varnish → Nginx → Unicorn → Rails
- Ganeti for virtualization cluster management (KVM)
- Full core infrastructure
  - DNS (auth/rec)
  - SMTP/IMAP
  - LDAP, RADIUS
  - Monitoring (Icinga, Munin, ELK, …)
- Managed using Puppet
- Debian packages for everything (sometimes updated/patched/rebuilt)
Routers

Routers? Routers!

- Pairs of redundant routers 1U servers with ≥8 GbE interfaces
- BIRD for BGP + OSPF
- `keepalived` for VRRP/HA on the client side
- Stateful dual-stack firewall with `ferm`
- `conntrackd` for state replication
- ≈1 Gbps routed traffic
- 5 different uplinks, 2 upstream providers + 1 IX
- Routing config managed by Puppet, BGP peers in Hiera
- Get rid of SNMP, use `check-mk` local checks!
Workstations

- Different uses, both tech/non-tech users
- Laptops with full-disk encryption
- Mostly desktops for non-tech users
- Desktops managed using Puppet
- GNOME as DE, puppetized gconf/dconf settings
Bootstrapping

- d-i preseeding across the fleet
  - PXE boot for servers/workstations
  - USB boot for laptops
  - ganeti-os-di for Ganeti VMs (ITP)
- Completely unattended installation for most classes of systems
- Brings the system to a state where it can run puppet
- partman recipes could be better though :)}
Why use d-i for VMs?

- Full VM images need to be kept up-to-date (point releases, security updates)
- Care must be taken to strip sensitive data (keys, UUIDs etc)
- d-i solves all of the above
- **ganeti-os-di:**
  - Boot an ephemeral KVM instance running d-i w/ preseeding config
  - Capture and log d-i output
  - Abort if a prompt appears
  - Use writeback caching to speedup the installation
  - Install time down to 2 min using a local APT cache
Managing configuration

- Puppet across the fleet
- Essential for maintaining anything more than a handful machines
- … but can be easily abused
- Config management must augment the package manager, not override/replace it
Puppet manifests that play nice with Debian

1. Drop config files in configuration directories if possible
   - /etc/apt/sources.list.d/

2. Create exclusively managed snippet directories wherever supported
   - /etc/rsyslog.d/ + /etc/rsyslog.puppet.d/
   - /etc/ferm/manual.d/ + /etc/ferm/puppet.d/

3. Don’t ship whole config files, use augeas to modify defaults

4. Use dpkg-divert and dpkg-statoverride to play nice with dpkg
Puppet-friendly packaging

- include configuration from directories by default
- Split out sane defaults from sample values
  - Debian-specific defaults can be left untouched: safer/easier upgrades
Should Debian provide a batteries-included `debian` Puppet module?

- `debian::apt::source`
- `debian::apt::multiarch`
- `debian::systemd::unit`
- `debian::systemd::service`
- `debian::alternative`
- `debian::dpkg::divert`
- `debian::dpkg::statoverride`
2 or 3 roles?

- FHS and conffile handling assume two roles
  1. Vendor/Distribution
  2. Local system administrator
- Should we assume a third one: config management system (or “site administrator”)?
- CMS should be able to override the Distribution
- Local admin should be able to override the CMS
- Should the CMS ship things under `/usr/local/`?
- Should the CMS place systemd units in `/etc/systemd/system/`?
Managing packages

- 99% Debian packages
- 1% either:
  - not in Debian
  - too old in Debian
  - site-specific
- squid-deb-proxy for the 99%
- reprepro for the 1%
- Try to minimize the delta by contributing :)

17/26
Managing packages

- Unlike the Debian archive we need multiple versions of the same package for each distribution. Examples:
  - Mongo
  - Elasticsearch
  - ...
- We also need thin, partial distributions for certain needs:
  - Ruby + cURL rebuilt against OpenSSL 1.0.2 (alternate path checking)
  - Nginx/HAProxy rebuilt against OpenSSL 1.0.2 (ALPN - HTTP/2)
- Solved with heavy use of components (e.g. profile/appserver, profile/lb) + apt_preferences magic
Managing packages

- Deploying a package to prod $\Leftrightarrow$ SRM
- Two main distributions
  - jessie-skroutz
  - jessie-skroutz-proposed-updates
- Configured on all machines
- Different APT priorities (940 vs -1)
- Prefer profile/* packages over main
- Packages enter p-u and are copied afterwards
Building packages

- Too small/few packages to setup a buildd infrastructure
- Run pbuilder on our workstations
- pbuilder-skroutz package shipping config, hooks and scripts
  - pbuilder-skroutz-create, pbuilder-skroutz-update: manage chroots
  - Hooks ensure that packages built for a profile/* component will use the correct B-D’s
- pdebuild-skroutz: build packages with correct Distribution (p-u) and an X-Component field in .changes
- Wrapper around reprepro process incoming, respecting X-Component in .changes
Deploying security updates

- Keeping 300+ machines up to date is difficult
- Workstations use unattended-upgrades
- Servers are a different story…
  - Gradual roll-out
  - No unwanted service restarts!
Deploying security updates

- Custom solution based on Puppet, servermon and Redis
- On every Puppet run, available updates are POSTed to servermon
- Central dashboard offering fleet-wide overview
- Available updates can be "staged" (= key in Redis) using the manage_updates CLI
  - manage_updates add *php* # Install all available PHP updates
  - manage_updates add -s '*' # Install all security updates
- On the next Puppet run, every "staged" update turns into `apt-get install`
- A Puppet report processor deletes successfully installed updates from Redis and notifies for potential errors.
Getting your sysadmins involved

- Involvement = benefit both ways
- Relatively high barrier, even for experienced sysadmins
- Reluctant to report bugs
- Build environments are non-trivial to set up; most people will use `debuild`
- Policy and New Maintainer’s Guide? TL;DR
Getting your sysadmins involved

- Lead by example
  - File bug reports but keep your sysadmins in the loop
  - Explain severities, tags, policy issues
  - Get them to install `how-can-i-help` :)

- Things we could do in Debian:
  - Improve BTS search & interface
  - Add an MTA-less mode to `reportbug` and `bts`
Links

- **servermon**: [https://github.com/servermon/servermon](https://github.com/servermon/servermon)
- “Local corporate APT repositories” by bernat@d.o
Thank you!

Q&A

CC BY-SA 4.0