Sane Operations

“The amount of effort required to manage a complex system can be directly correlated to the consistency enforced on said system.”

Zachary Schneider @sigil66

boundary

What is DevOps, really?

Marketers have hijacked a concept that once had meaning and turned into something that is now used to sell product.

boundary

DevOps is …

- Chef and Puppet?
- NOPE!
- This is: CONFIGURATION MANAGEMENT
- **It’s been around for awhile**

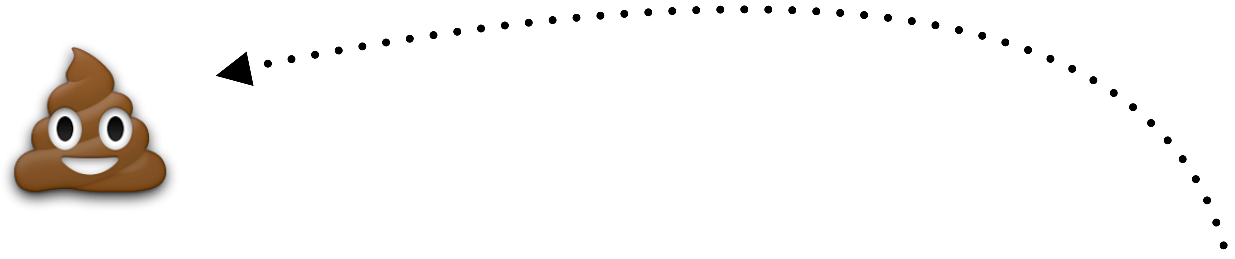
DevOps is …

- About deployment automation?
- About build automation?
- About continuous integration?
- About your operational tooling?
- **About responsibility, consistency, and contribution.**

A DevOps Engineer is …

- The guy who writes the chef and puppet?
- The guy who packages my software?
- The guy who runs the 💩 I write and is on call?
- **A marketing construct.**

The reality dictated by marketing



OPS + TOOLS

DEVS

Moving Beyond …

Fix your organization

- Eliminate divisions of leadership
  - You can’t prioritize for the benefit of the product when you are prioritizing for the benefit of your own team.
  - Conway’s law: “organizations which design systems ... are constrained to produce designs which are copies of the communication structures of these organizations”
- Create an environment that fosters responsibility for the product and it’s customers

Develop understanding …

- Understand when a contribution adds value
- Understand that software is a liability
- Understand that a liability, that does not add value must be eliminated
- Understand that systems must be designed to be deployed and operated
- Understand technological and organizational scale

Develop your thinking …

- Develop the desire to work for/toward simple solutions (minimalism)
- Develop and master incremental change management
- Develop the desire to create things that work for individuals that are less skilled or knowledgeable than yourself
- Abolish the software consumer mentality
    - **Wrong:** What can I install to solve this problem (DevOp)
    - **Right:** What can I contribute/change to solve this problem (Engineer)

Design concepts …

- Separation of Concerns
- Single Responsibility Principle
- Principle of Least Knowledge
- Composition

A few basic best practices …

- Keep your dependencies, tooling, and deployment platform up to date.
- Use task based build frameworks for your projects, a build server should not be required to build artifacts and run tests
- Deliver software via repositories and packages
- Avoid hipster tech
- Don’t diverge from exemplars
- Finish your work

Boundary :: A Case Study

Organization

Deployment and CM

Architecture (saving for another time)

Organization :: Then

- Divided
  - DEV / QA - Engineering Director
  - OPS - Ops Director
- OPS and DEV on call
- Anyone can contribute to anything

Organization :: Now

- Unified
  - Engineering - Director of Engineering
    - Ops
    - QA
    - UI
    - Backend
    - Architecture

Deployment & CM :: Then

- Chef Server 10
  - System software and updates
  - Cron every 15 minutes
- Small_Wonder
  - scp + ssh + sudo + ruby templates + chef api
    - custom software deployments and updates

How do we fix this?

Deployment & CM :: Fix

- Understand our service interactions and properties
- Identify our requirements
  - Chef can be used to update the entire system
  - At least one node must be online during deployment per service
- Develop the implementation plan

Deployment & CM :: Fix

- Interim Solution
  - Software deployed as packages
  - Chef run in order via ssh tool of choice (disable chef via cron)
- Result?
  - Consistency
  - Reliability
  - Lowered liability (we deleted custom tooling that did not add value)

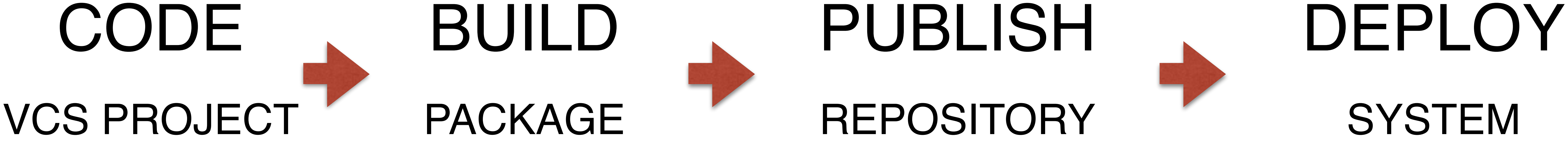
boundary

Packages: Why?

Packages :: Required

- Package systems are a secure integration point between environments
- Package contents are audit-able and can be signed and verified before and after installation (security++)
- Packages are versioned, versions have meaning, package metadata can be linked to VCS revisions
- Packages can be rolled back
- Packages and repos are a units of composition

Package :: Workflow



CODE

VCS PROJECT

BUILD

PACKAGE

PUBLISH

REPOSITORY

DEPLOY

SYSTEM

Packages :: The Ugly

- Distribution tooling is 💩
    - Why?
        - Build system is commingled with the packaging infrastructure
        - Designed for intermittent releases of groups of packages

Packages :: Tools

- **FPM** https://github.com/jordansissel/fpm
  - Single command package build tool from Jordan Sissel
- **PRM** https://github.com/dnbert/prm
  - APT repository tool in the same vein
- **APTLY** http://www.aptly.info
  - more advanced apt repository tool

So where are we?

Welp, it is time to make Config Management work for us.

Chef :: The Bad

- Problems
    - Chef servers (and Puppet for that matter) are evil
        - They contribute to inconsistency (split commit system)
        - They seek to solve an already solved problem (software distribution)
        - They introduce even more PKI into your environment
        - Single point of failure for deployment
    - Knife is just a plain horrible catch all tool

Chef :: No More Servers

- Solution
  - Distribute Chef artifacts via packages
    - cookbooks
    - databags
    - roles
- chef-cascade (a custom chef client)

Chef :: Reclaim Lost Value

- What have we lost?
  - node inventory
  - node list
  - node role lists
  - pseudo service discovery for configuration

Cascade

An opinionated integration

Cascade

- Consul + Chef + Packages
- Features
    - Ordered Chef runs with role de-duping
    - Service Discovery for CM and applications
    - Roll locking for multiuser environments
    - Per environment attribute injection (no more databags)
    - Our apps can now discover themselves (no glue infra)
- Open sourced soon

Moving Beyond ;) …

What we have just … isn’t … good enough.

ZPS :: The Last Word

- **“The last word in package management”**
- ZPKG
  - Fast random access
  - Encrypted payloads
  - Installation actions
- ZPM
  - Developer oriented package management
  - Decoupled from distro specific build system

ZPS :: The Last Word

- ZREPO
  - Multi-Publisher (multi-tenant)
  - Publish to channels
    - One package many channels
  - Designed for rolling updates
  - Scalable metadata and storage backends (dynamic mode)
  - Snapshots for HTTP (static mode)

Thank You!

http://www.boundary.com