Bringing Kubernetes into Salesforce

Steve Sandke
Principal Architect
Salesforce

ssandke@salesforce.com

Forward-Looking Statements

Statement under the Private Securities Litigation Reform Act of 1995:

This presentation may contain forward-looking statements that involve risks, uncertainties, and assumptions. If any such uncertainties materialize or if any of the assumptions proves incorrect, the results of salesforce.com, inc. could differ materially from the results expressed or implied by the forward-looking statements we make. All statements other than statements of historical fact could be deemed forward-looking, including any projections of product or service availability, subscriber growth, earnings, revenues, or other financial items and any statements regarding strategies or plans of management for future operations, statements of belief, any statements concerning new, planned, or upgraded services or technology developments and customer contracts or use of our services.

The risks and uncertainties referred to above include – but are not limited to – risks associated with developing and delivering new functionality for our service, new products and services, our new business model, our past operating losses, possible fluctuations in our operating results and rate of growth, interruptions or delays in our Web hosting, breach of our security measures, the outcome of any litigation, risks associated with completed and any possible mergers and acquisitions, the immature market in which we operate, our relatively limited operating history, our ability to expand, retain, and motivate our employees and manage our growth, new releases of our service and successful customer deployment, our limited history reselling non-salesforce.com products, and utilization and selling to larger enterprise customers. Further information on potential factors that could affect the financial results of salesforce.com, inc. is included in our annual report on Form 10-K for the most recent fiscal year and in our quarterly report on Form 10-Q for the most recent fiscal quarter. These documents and others containing important disclosures are available on the SEC Filings section of the Investor Information section of our Web site.

Any unreleased services or features referenced in this or other presentations, press releases or public statements are not currently available and may not be delivered on time or at all. Customers who purchase our services should make the purchase decisions based upon features that are currently available. Salesforce.com, inc. assumes no obligation and does not intend to update these forward-looking statements.

Agenda

Motivations

Lining It Up

Decisions, Decisions

What We Built

Where We Are

Motivations

Enabling Phenomenal Customer Success

Growth Across The Clouds

2014	2016	
99.97% Availability	**99.98%** Availability	
50 Production Instances	**109** Production Instances	118% ↑
490 Billion Transactions • 230ms average latency	**1.1 Trillion** Transactions • 210ms average latency	124% ↑
10 Data Centers • 1st in EMEA – London	20 Data centers • 3 in EMEA – London, Paris, Frankfurt	100% ↑
194 MC Customer Databases	**395** MC Customers Databases	104% ↑
247 Billion emails sent	**478 Billion** emails sent	94% ↑

Infrastructure Engineering Principles

Integrating and Scaling

One Salesforce

- Leverage best practices and common processes

Design Principles

- Service Ownership: Software Engineers operate the services they create
- Recovery-oriented software architectures
- Prefer scale-out architectures
- Simple, consistent hardware
- Measure and improve key metrics such as availability through continuous application and platform changes

salesforce

Evolving for Scale

From Manual & Automated to Autonomous Operations

Software Defined Everything

- Compute
- Storage
- Security
- Networks

Task	Manual ("Ops")	Automated ("DevOps")	Autonomous ("No Ops")
Sets the goal	Human	Human	Human
Decides when to start the work	Human	Human	Machine
Adjudicates work priorities	Human	Human	Machine
Does the work	Human	Machine	Machine
Generates the validation report	Human	Machine	Machine
Interprets the validation report	Human	Human	Machine
Handles failures	Human	Human	Machine
Handles exceptions	Human	Human	Human

Looking for a New Way

We wanted:

Easy Onboarding for new services

High Fidelity between production and nonproduction environments

Declarative rather than imperative

Secure by Default to enhance our security profile

Fully Automated to cut human involvement and error

Usable across public cloud and private data centers

Simple. Secure. Automated.

A Simple Model

Deployment artifacts are containers

Provide a health probe

Declare your desired deployment state

Automation gets you safely to your desired state and keeps you there.

We call it the Salesforce Application Model (**SAM**)

Lining It Up

We Chose Kubernetes

Open source, container based

High development velocity

Opportunity to affect direction

Opportunity to help build it

Broad-based, welcoming community

Project vision aligned with our own

salesforce

From Decision to Delivery

We needed **Exec Support**

We needed **Cross Company Collaboration**

We needed **Launch Partners**

The Right Launch Partners Are Key

Internal Services to prove things out.

Comfortable with **ambiguity**

Interested in **defining a new platform**

Covering a spectrum within our organization

Willing to **wait to ship**

Stateless

Decisions, Decisions

A Light Abstraction Over Kubernetes

Guard rails are important

Exposing new features is easy

Taking features away is hard

We needed infrastructure related extensions anyway.

Deployment Manifest: a unified set of Deployment and Service specifications.

Git is the Master

Deployment Manifests live in git

Pro: super easy to review proposed deployments

Pro: full history of all production deployments

Con: git can take a while to learn

Deployment Request = Pull Request (PR)

Deployment Approval = PR Approval

Two factor authentication for PR approvals

One Unified Deployment Manifest Repository

All deployment manifests for all of production live in one git repository

Pro: easy to audit/view all production changes

Pro: easier for our tooling to consume

Con: repository is pretty noisy

Requires fine grained access control; we added it

Isolation Across Security Profiles

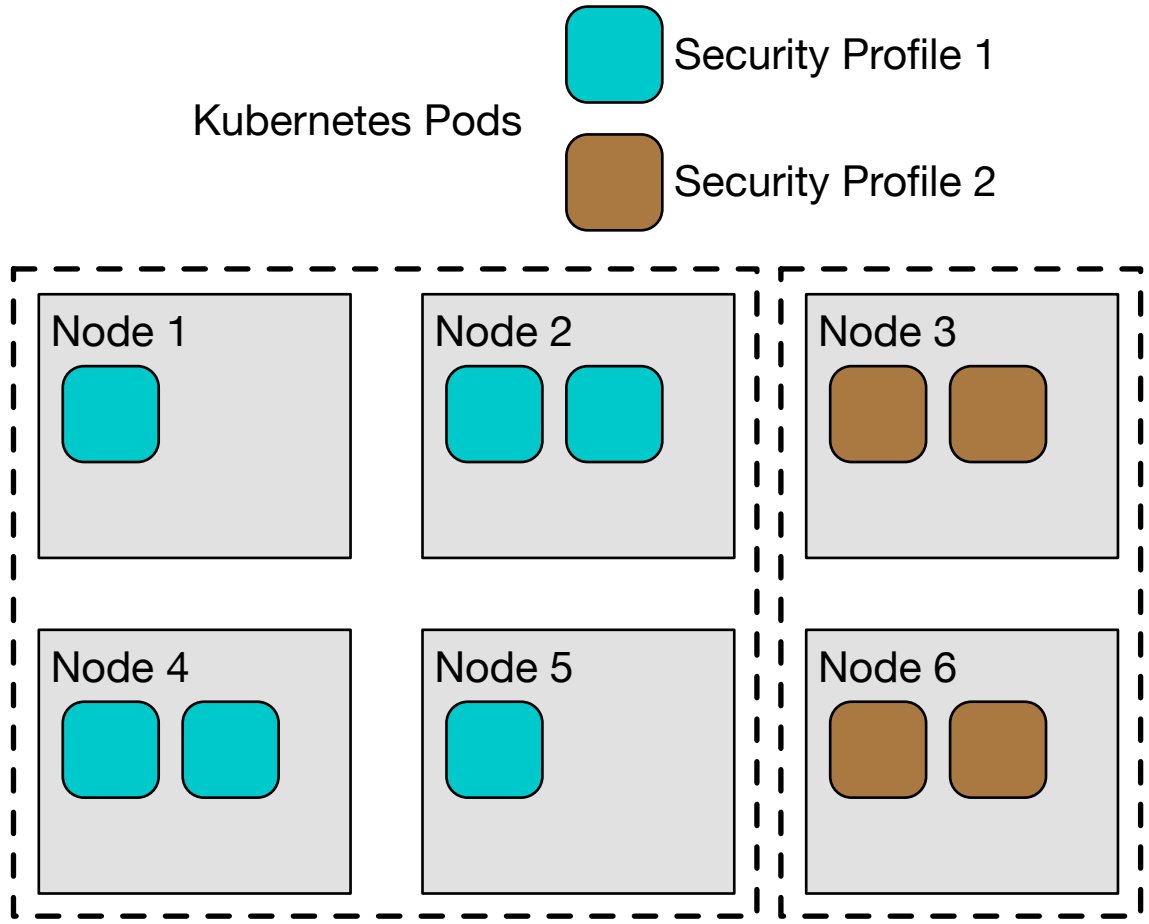
Security Profile:

- Inbound/outbound communications
- Data sensitivity

Container Isolation is nascent, so we don't fully rely on it (yet)

A node is associated with one security profile at a time

Nodes are isolated across security profiles



What We Built

Deployment Manifests

Deployments and Services in one artifact

Functions → Deployments

LoadBalancers → Services

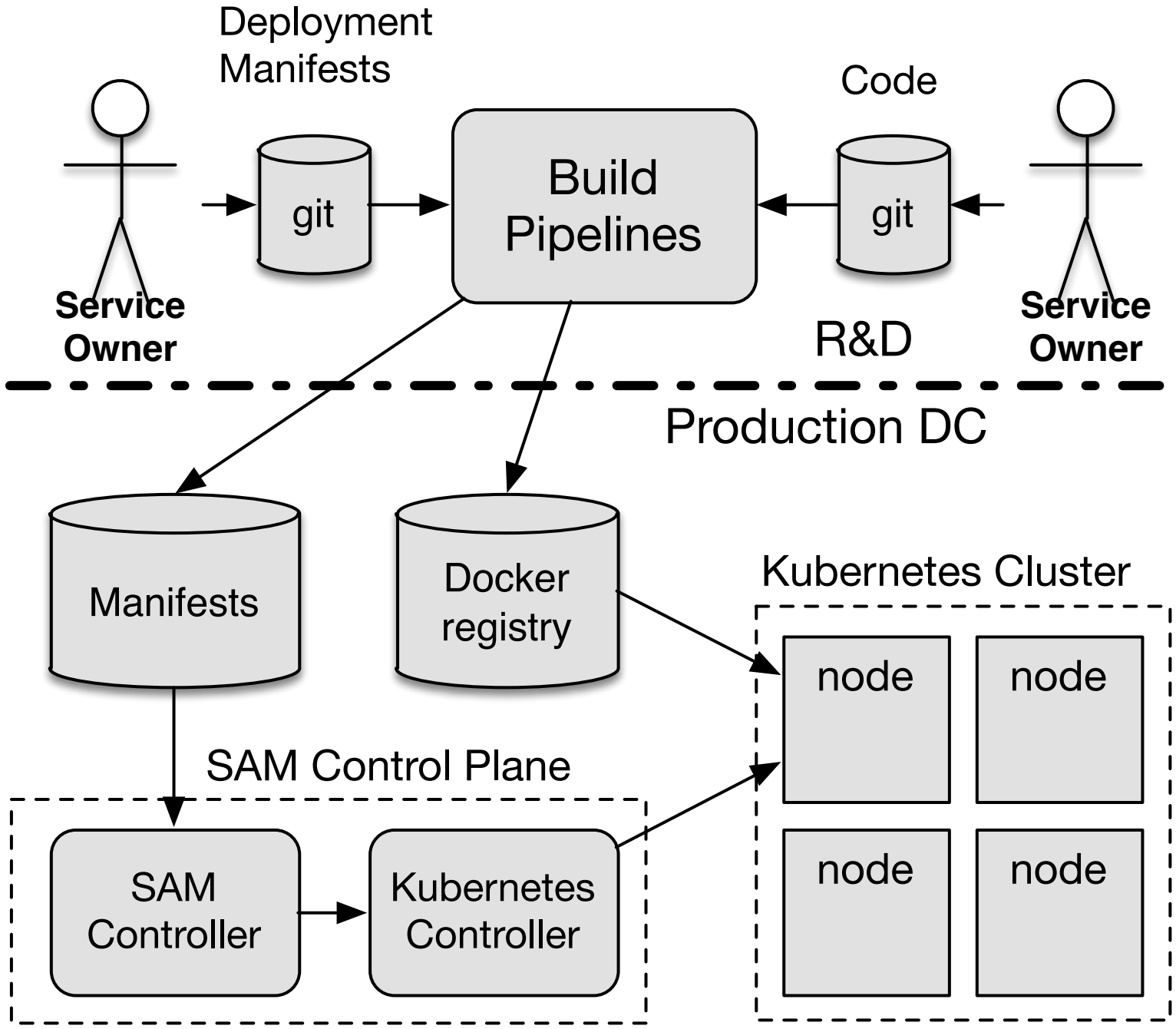
(Eventually) Salesforce Infrastructure Controls

- Alerting Rules
- Audit Hooks

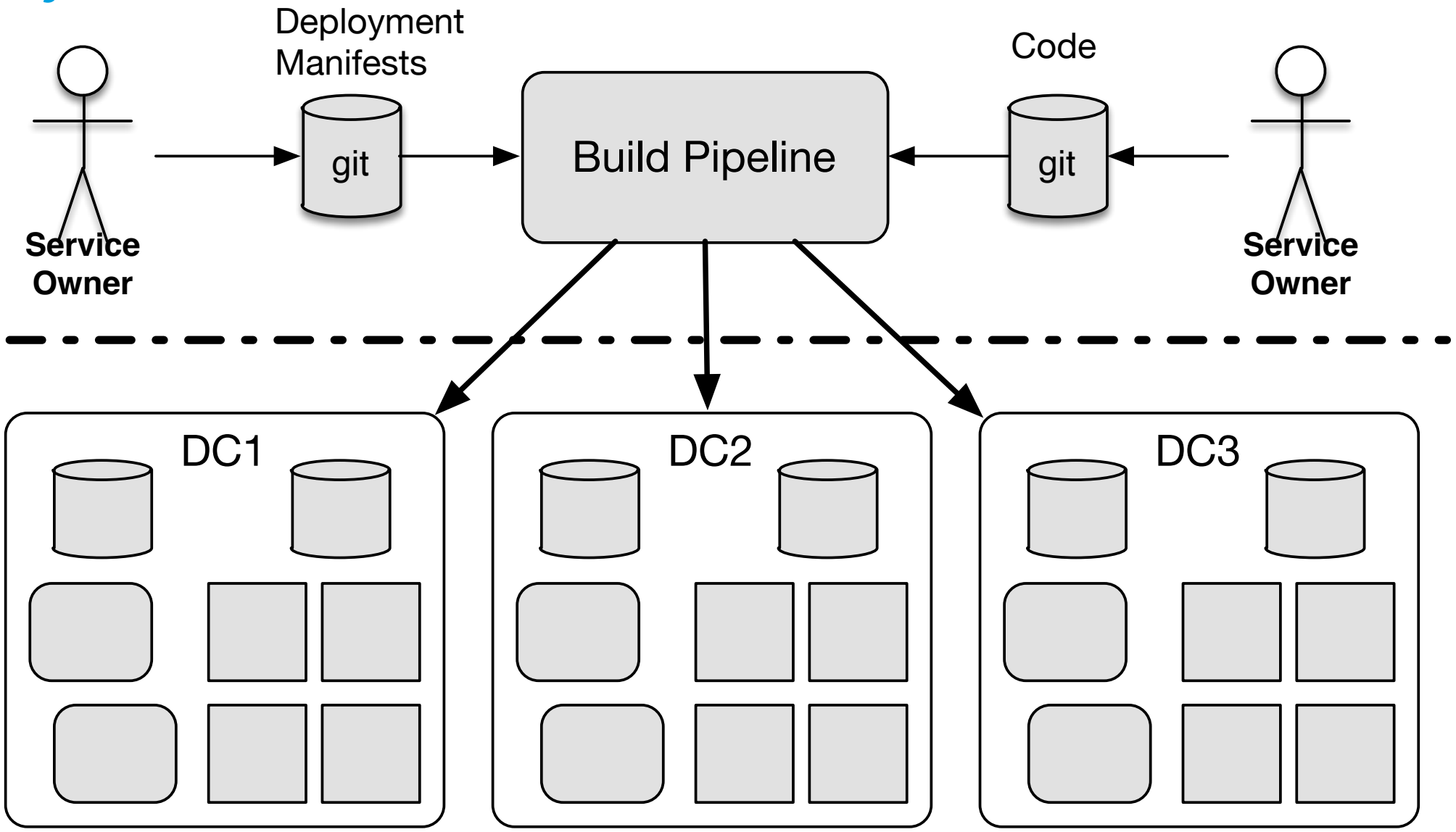
```
functions:
- name: sample
  count: 4
  containers:
  - image: simpletest:1.0.4
    ports:
    - containerPort: 9090
    livenessProbe:
      httpGet:
        path: /
        port: 9090

loadbalancers:
- lbname: samplelb
  function: sample
  ports:
  - port: 8000
    targetPort: 9090
```

Overall Flow



Deployments Go In Parallel



Where We Are

Today and Tomorrow

30 minutes from manifest merge to deployment

Now	End 2017
A Couple Services In Production	**>20** Services In Production
<100 Kubernetes Nodes	**>1000** Kubernetes Nodes
Few Data Centers run Kubernetes	**All** Data Centers run Kubernetes
<10 Kubernetes Clusters	**>20** Kubernetes Clusters
Dozens Daily Service Deployments	**Hundreds** Daily Service Deployments
1 Kubernetes contribution	**Many** Kubernetes contributions

There’s More To Do

Scaling out

Easier onboarding

More infrastructure abstractions

- Logging, cert management, secrets integration, …

More visibility into service state

Support clustered applications

- Redis sentinel, Redis cluster, …

Persistence

Persistence

Static Local Disk Mounts

- Work for many apps
- Great for lightweight caching capability
- We do this ourselves with a local agent.

Persistent Volumes

- StatefulSets + Automatic Upgrades
- Requires a persistent backend

Dynamic Local Disk Mounts

- Locally mounted disks
- Strong Kubernetes Pod to Node affinity
- Not built (yet)

We're Happy To Be Part of the Community

We're **contributing and want to contribute more**

We're **using Kubernetes in other areas** as well

We have **Kubernetes deployed and hosting live services**

Our **service owners are excited**

We are hiring!

Thank You