Chaos Heidi vs. Orchard:
Self-Disruption and Healing in a
Cloud Foundry-Based Service Environment

Diego Zamboni
Swisscom
diego.zamboni@swisscom.com

Bill Chapman
Stark & Wayne
billchapman@starkandwayne.com
Agenda

- Swisscom & Application Cloud overview
- Health Management (Orchard)
- Self-disruption (Chaos Heidi)
Swisscom

TV coverage

- around 93% SDTV
- over 89% HDTV

Ultra-fast broadband coverage (≥50 Mbps)

- Over 2.5 million homes and businesses

Mobile broadband coverage

- >99% 2G
- >99% 3G
- 98% 4G/LTE
- Winner of Connect test for the 6th time running

Universal service & ADSL

- 97% (>2Mbps)
Swisscom Application Cloud

Software

Functionality / APIs / Applications

Platform-as-a-Service

- Cloud Foundry OSS
- Container Orchestration
- F5 Web Firewall
- Integrated Load Balancer
- Command Line Interface
- Log Aggregator

Managed services

- Self-Service
- Lifecycle Management
- SLAs

Infrastructure-as-a-Service

- Red Hat OpenStack
- KVM Hypervisor
- ScaleIO Software-Defined-Storage
- PlumGrid Software-Defined-Network
- Quanta and Arista Hardware
- Swisscom Data Centres
The Application Cloud

- ELK
- MongoDB
- CF Services
- CF
- ScaleIO Nodes
- OpenStack Nodes
- OpenStack VMs
- Baremetal Servers (Quanta+Arista)
The Application Clouds

- **Virtual Private instance**
  - Swisscom internal

- **Virtual Private instance**
  - Customer A

- **Virtual Private instance**
  - Customer B

- **Virtual Private instance**
  - Customer C

- **Public instance**
  - http://developer.swisscom.com
Monitoring Cloud Foundry production environments
The OODA loop

Any infrastructure component
OODA loops are all over the place
(disjoint and incomplete,
AKA existing monitoring and management tools)
Design principle: don’t reinvent the wheel

<table>
<thead>
<tr>
<th>Bosh health monitor</th>
<th>Consul checks</th>
<th>CloudFoundry health management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Puppet policies</td>
<td>OpenStack health management</td>
<td>PlumGrid health management</td>
</tr>
<tr>
<td>Linux/Unix cron jobs</td>
<td>Logs</td>
<td>Flocker health management</td>
</tr>
</tbody>
</table>

Use existing mechanisms as much as possible
Design principle: aggregate and unify (loosely coupled systems)

Build on top to coordinate and add missing information

Use existing mechanisms as much as possible
Design principle: learn and improve

Delegate the hard decisions to humans, but use them to improve analysis.

Build on top to coordinate and add missing information.

Use existing mechanisms as much as possible.

Image credit: http://hyperboleandahalf.blogspot.ch/2010/06/this-is-why-ill-never-be-adult.html
Self testing: Chaos Heidi

Automated disruption to test the automated responses

Inspiration: Netflix’s Chaos Monkey (https://github.com/Netflix/SimianArmy/)
Why Orchard?

Improve visibility into our systems to make everyone’s lives easier
Service and BI view

CF services end-to-end checks

Ok

Business integration

Billing Spring Jobs
Failed Jobs: 66

Local login
Response Time: 362ms

Metis login
Response Time: 2981ms
<table>
<thead>
<tr>
<th>Service</th>
<th>Status</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>MongoDB</td>
<td>Ok</td>
<td></td>
</tr>
<tr>
<td>RabbitMQ</td>
<td>Ok</td>
<td></td>
</tr>
<tr>
<td>Redis</td>
<td>Warning</td>
<td></td>
</tr>
<tr>
<td>ELK</td>
<td>Ok</td>
<td></td>
</tr>
</tbody>
</table>

**Flocker nodes**

- Defined:
  - wflboxcmmg-elk-sc-pub-pub-50-100
  - wflboxcmmg-elk-sc-pub-pub-50-110
  - wflboxcmmg-elk-sc-pub-pub-50-120
  - wflboxcmmg-elk-sc-pub-pub-50-130
  - wflboxcmmg-elk-sc-pub-pub-50-140
  - wflboxcmmg-elk-sc-pub-pub-50-150

- Undefined:
  - wflboxcmmg-elk-sc-pub-pub-50-160
  - wflboxcmmg-elk-sc-pub-pub-50-170
  - wflboxcmmg-elk-sc-pub-pub-50-180
  - wflboxcmmg-elk-sc-pub-pub-50-190
  - wflboxcmmg-elk-sc-pub-pub-50-200
Architecture and implementation
Consul Agent
Exposes checks results

CF monitoring apps
(services, svc broker, APIs, etc.)

Each CF instance sends logs (syslog, Loggregator) and metrics (graphite) to Logstash

collectd daemon
Sends data in graphite format to Logstash

Consul Server Cluster

Logstash

Responder
Listens to Automated Resolution requests (Puppet, Mcollective, scripts, API calls, etc.)

RabbitMQ
Message bus

RIEMANN
Event Processing

Alerting

RunDeck
Automation

InfluxDB
Persistency

Dashboard

Baremetal Servers (Quanta+Arista)

OpenStack Nodes

ScaleIO Nodes

OpenStack VMs

CF Services

ELK

Checks services

CF

M

M

M

CF monitoring apps
(services, svc broker, APIs, etc.)

M

M

M

CF monitoring apps
(services, svc broker, APIs, etc.)

M

M

M

CF monitoring apps
(services, svc broker, APIs, etc.)

M

M

M

CF monitoring apps
(services, svc broker, APIs, etc.)

M

M

M
What is Chaos Heidi?

A generic resiliency testing framework
Similar to Netflix Chaos Monkey
Why do we want a Chaos framework?

Semper Vigilante!
How might can use Chaos Heidi

- Stack Validation testing along with Orchard
- Random daily resiliency validation
- Simple testing of Orchard components
- Regression testing and analysis
How does it work

Queue

Controller

Agent

Target Platform (Victim)

Agent

Target Platform (Victim)

Agent

Target Platform (Victim)
Example Attacks

- Kill an Openstack VM
- Kill a BOSH Job
- Kill a Docker Container running a CF Service
- Simulate a Network Outage
How to create useful Attacks

1. What is broken now?
2. What has broken in the past?
3. What will break in the future?
Something that misbehaved recently!

BOSH
Simulation vs Emulation

Implying a failure vs Causing a failure
Thank you

Diego Zamboni
Swisscom
diego.zamboni@swisscom.com

Bill Chapman
Stark & Wayne
billchapman@starkandwayne.com