Orchestration, Docker & NFV
A Real Case Study
Howdy

I am Uri Cohen

Head of Product @ GigaSpaces, makers of Cloudify

@uri1803
My (Other) Most Important Project
Agenda

- Orchestration 101
- Introducing Cloudify
- Cloudify + Clearwater
- Demo
The Vision
-*as-a-Service Automation
Awareness
Reaction to Self
The Reality
VNFs Are (Very) Complex
VNFs Are (Very) Complex

- Many tiers
- Load balanced
- Strict HW / Placement requirements
  - NUMA, DPDK, SR/IOV, Affinity / Anti-Affinity
- Firewalls, static networks, storage, etc. etc.
- Often hard wired
- What about day 1 and day 2?
- Scaling, Healing, elasticity?
Service Chains More So..

◇ Add Their Own Complexities:
  - (Often dynamic) forwarding graphs
  - Complex environments
  - Multiple geographic locations
  - Complex policies and SLA requirements
“It is not the strongest of the species that survives, It is the one that is most adaptable to change.”

-Charles Darwin
The only constant is change”

-Unknown
WHAT IF

You could orchestrate and manage any VNF the same way?
Key Aspects

Open Source
No more monolithic, vendor tied monsters

Open Standards
Standard and vendor neutral language to describe VNFs and forwarding graphs

Future Proof
Be ready for what’s coming and leverage new tools and technologies
The P-D-M-R Loop

- Model
- Provision
- Deploy & Configure
- Monitor & Detect
- Remediate
VNF Blueprint with TOSCA

Topology Orchestration Specification for Cloud Applications

Topology
Workflow
Policy
VNF Topology

Nodes:
- Bono (VM)
- Sprout (VM)
- Homer (VM)
- Network A (Subnet)
- Network B (Subnet)

Connections:
- Bono HostedOn Sprout
- Sprout ConnectedTo Homer
- Network A ConnectedTo Network B
VNF Topology

- Types, Nodes and Interfaces
- Inputs and Outputs
- Relationships
- Requirements and Capabilities

Node Type: Container

Hosted on

Connected-to

MongoDB

Old-School Java App
Application Blueprint (TOSCA)

Cloudify
- Provision
- Configure
- Monitor
- Manage

Monitoring & Alarming

IaaS Plugins

Container Plugins

Conf Mgmt Plugins

Monitoring & Alarming
Application Blueprint (TOSCA)

Cloudify
- Provision
- Configure
- Monitor
- Manage

Monitoring & Alarming
- InfluxDB
- Riemann
- elasticsearch

IaaS Plugins
- openstack
- VMware
- Google Cloud Platform
- Amazon Web Services

Container Plugins
- Docker
- Kubernetes

Conf Mgmt Plugins
- Saltstack
- Ansible
- Chef
- Fabric
- Puppet
Where It Fits In
Where It Fits In
HOW WE GOT STARTED

NFV Open Source First Steps

APPLICATION

APPLICATION MANAGEMENT

CLOUD INFRASTRUCTURE

INFRASTRUCTURE MANAGEMENT

Target

- Automatic deploy, scale and heal a VNF on different Openstack infrastructures
- Use only existing open source products
- Use generic management tools for orchestration and configuration – without an EMS and a specific VNFM
- Evaluate TOSCA maturity

 isEnabled: false

Successfully accomplished proof of concept
HOW WE GOT STARTED

NFV Open Source First Steps

- APPLICATION
- APPLICATION MANAGEMENT
- CLOUD INFRASTRUCTURE
- INFRASTRUCTURE MANAGEMENT

Target

- Automatic deploy, scale and heal a VNF on different Openstack infrastructures
- Use only existing open source products
- Use generic management tools for orchestration and configuration – without an EMS and a specific VNFM
- Evaluate TOSCA maturity

→ Successfully accomplished proof of concept
Solution Architecture

Service Portal
Self-Service Instantiated TOSCA Service Templates

OSS/BSS
Catalog, Inventory, Fulfillment

REST API

intelligent Policy Manager

Service Catalog

TOSCA/REST Service Model

TOSCA/REST Service Model

Cloudify NFVO

Network Controller

Service Orchestrator

Application Controller

3rd party VNFM

VNF Catalog, lifecycle, multi-VM, Native Integration with VMware NFVI

VNF Catalog: HEAT/ OVF

Lifecycle Management

Enhanced Monitoring

WAN Controller

VMware vCloud NFVI

vCloud Director

VMware Integrated OpenStack

Log Insight

vSphere

vSAN

NSX
OPNFV Functest vIMS
Cloudify + ClearWater

- We started with Chef
- We then moved to simple shell scripts using `apt-get`
- Now we’re doing Docker as well
  - More predictable
  - Simpler to setup
  - Self contained
  - Instantaneous
  - Lower footprint (no vm)
Why Cloudify?

- Will provision resources on any cloud
- Standards based (TOSCA)
- Will handle monitoring, log collection, healing and scaling of your deployment
- Will allow you to mix containerized and non containerized pieces
- It Works!
Demo Time!
References

◇ Cloudify community portal: http://getcloudify.org
◇ Project ClearWater: http://www.projectclearwater.org/
◇ Original ClearWater IMS Blueprint: https://github.com/cloudify-examples/clearwater-IMS-blueprint
◇ Cloudify ClearWater https://github.com/Orange-OpenSource/opnfv-cloudify-clearwater
◇ NFV related posts at the Cloudify blog: http://getcloudify.org/tags/NFV/
◇ ClearWater Docker images https://github.com/Metaswitch/clearwater-docker
◇ Demo Video: https://youtu.be/84gEy6Vvc0E
Thank You

Questions?

Find me at:
- Twitter @uri1803
- email uri@gigaspaces.com