OpenStack = Linux of the Cloud?

Ingo Gering, Fujitsu
Dirk Müller, SUSE

- Introduction to OpenStack
- Software Defined Anything (SDx)
- Containers and container orchestration with OpenStack Magnum
- Providing Bare Metal Compute Nodes with OpenStack Ironic
- Monitoring and logging based on OpenStack Monasca
- Summary / Discussion
Introduction to OpenStack

- 60 years in 60 minutes
DC Computing 65 years ago

WITCH
Year: 1951

Short for the **Wolverhampton Instrument for Teaching** Computing from **Harwell**, the WITCH was also known as The Harwell Dekatron Computer. It was slow (a multiplication took 5-10 seconds), but this was justified by its ability to run long periods of time unattended. It could therefore be left on its own with a large amount of input data. At one point it was left running over the Christmas and New Year holiday and was still working when the staff came back 10 days later.
DC Computing 6 months ago
„OpenStack is not (only) a Cloud, a Project, a Product, an API, a user interface, a set of standards. It is a framework for doing IT infrastructure as interoperable and interchangeable as possible.“

http://www.theregister.co.uk/2015/07/09/openstack_overview/
**OpenStack = „Linux of the Cloud?”**

- Replicating the success of Linux as operating system for **cloud management**
- **Universal, open cloud management** platform for public and private clouds
  - Free Open Source Software under the terms of the **Apache License**
  - IaaS cloud services from the beginning and more and more additional services (IaaS+)
- A compilation of various related technology projects / services
  - Currently 6 Core Services: Nova, Neutron, Swift, Cinder, Keystone and Glance
  - Currently 13 Optional Services: Horizon, Ceilometer, Heat, Sahara, Ironic, Manila,... → **Big Tent – Model**
- Development by a **global community of 850+ companies with 19000+ individuals**
  - Started as a joint project by Rackspace Hosting and NASA (2010)
  - Governed since 2012 by the non-profit **OpenStack Foundation**, promoting Software & Community
  - Agile and fast innovation with a **6-month release cycle** and appropriate Design Summits for developers
  - 13 releases since 2010 newest with the “**Mitaka**” community release in April 2016
  - OpenStack Summit in Austin End of April 2016 for the “**Newton**” design: about 7500 participants
  - End of October Design Summit in Barcelona for the “**Ocata**” release in Barcelona, Spain
  - **Has reached the level of maturity for productive use in enterprises**
OpenStack = Linux of the Cloud with a similar success!

- Speed of innovation
  - Community development
- Cost-effective
  - No license costs, commodity HW
- No vendor lock-in
  - Flexibility to customize and interoperate
- Massively scalable
  - Highly modular design
- Easy path to hybrid cloud
  - Standardized APIs enable easy integration of different cloud deployment models

Promise of OpenStack is to be the only cloud management platform that offers real possibilities for private cloud usage and standardization of hybrid clouds
OpenStack projects providing modular services

- **Base**: Set of interrelated well structured software modules to control large pools of compute, storage, and networking resources using common shared services
- **Additional**: More and more IaaS+ services for Database management and Data Processing of application clusters as 2 examples…
- Users can manage their resources through a self-service dashboard (Horizon), a more user-friendly portal as an alternative or general usage the the well and public documented OpenStack APIs
- No need to use all OpenStack services

---

*) Graphics are based on the OpenStack Kilo release; Up-to-date at https://www.openstack.org/software/project-navigator

---

Fastest growing open source cloud management platform with broadest industry support
OpenStack – project / service classification

Performing
- Heat
- Magnum
- Trove
- Sahara

Operational
- Monasca
- Barbican
- Ceilometer

Basic
- Cinder
- Horizon
- Ironic
- Swift

Essential
- Glance
- Nova
- Keystone
- Neutron
SUSE OpenStack Cloud distribution provided by the Fujitsu OpenStack Services (FOS) as an example

API Clients
- Lifecycle Man. (SUSE Manager)
- Image Tool (SUSE Studio)
- Dashboard (Horizon)
- Portal (OSCM)

OpenStack Cloud APIs + Eco System

SUSE OpenStack Cloud (V. 6 = Liberty)
- Admin Server
  - Chef
  - Crowbar
  - DHCP, PXE, ...
- Required Services
  - Authentication (Keystone)
  - Images (Glance)
  - Object (Swift)
  - Volume (Cinder)
  - Network (Neutron)
  - Orchestration (Heat)
  - Telemetry (Gatling)
- Compute (Nova)
- Hypervisor
  - KVM/Xen
  - Hyper-V
  - ESXi
- ETERNUS DX
  - Block (iSCSI, FC)
  - File (NFS)
  - vCenter
  - W2k12
- ETERNUS CD
  - Block (RBD)
  - S3 (RGW)
  - Object (RADOS)

Required Services
- Open Source Database (PostgreSQL)
- Messaging (RabbitMQ)
- Object (RADOS)
- Block (RBD)
- S3 (RGW)

Operating System: SUSE Linux Enterprise Server (SLES12 SP1)

Physical Infrastructure (PRIMERGY, Storage, Network)

SUSE Product
SUSE Cloud Enhancement
Partner Product
OpenStack Component
Fujitsu Product

Possible extensions
Software Defined Anything (SDx)
Software Defined Anything (SDx)

Moving control and management functionality from the hardware into an independent software layer

Orchestration of the infrastructure is no longer reliant on the many and varied proprietary embedded capabilities and delivers an high degree of automation

Everything from policy definition, though resource provisioning & configuration, to ongoing optimization, monitoring & administration can be done centrally in a more joined up and flexible manner

Using mainly Industry standard servers (x86 servers, …) as a hardware base for everything in the data center; also for Software Defined Storage (SDS) and Software defined Networking (SDN)
Software Defined Storage (SDS)
Block storage within OpenStack is SDS based on Open Source Ceph as a common standard!

- Ceph is used in more than 40% of productive OpenStack clouds for Cinder managed persistent Block storage
- Ceph is also very well integrated with OpenStack Keystone, Swift, Glance and Nova
- Traditional block storage usage in OpenStack is rather uncommon

Source: [http://superuser.openstack.org/articles/openstack-users-share-how-their-deployments-stack-up](http://superuser.openstack.org/articles/openstack-users-share-how-their-deployments-stack-up), 05/2015
Foundations of Ceph

Build for new challenges
- Every component must scale
- No single point of failure
- Run on readily-available, commodity hardware
- Everything must self-manage wherever possible

Driven by a new philosophy
- Open Source
- Complete software-based
- Community-focused equals strong, sustainable ecosystem
- Driven by big players e.g. Red Hat, Mirantis, SUSE, Fujitsu and others

Works with a new design
- Scalable in capacity and performance
- Self-managing / Self-healing
  - Automatically place & replicate
  - Automatically balance & migrate data
- Calculate instead of lookup
- Use objects instead of blocks or files
CRUSH algorithm – the crown jewel of Ceph

CRUSH - Controlled Replication Under Scalable Hashing

- Calculate placement instead of stored
  - No meta data necessary
  - Almost no central lookups
  - Infrastructure aware algorithm
  - Easy and flexible placement rules
  - Placement based on physical infrastructure

- Automatically place, replicate, balance, and migrate data
  - No hot spots
  - Quickly adjusts to failure
  - Pseudo-random, uniform distribution
  - Dynamic adaption to infrastructure changes
  - Automatic and fast recovery from lost disks
CRUSH algorithm – the crown jewel of Ceph

CRUSH – Infrastructure / topology aware algorithm

Every copy of an object should be placed in a different cabinet.

Cluster map hierarchy consisting of rows, cabinets, and shelves of disks.
Scalability and Reliability

Distributed Redundant Storage

- Intelligent data Distribution across all nodes and spindles = wide striping (64KB – 16MB)
- Redundancy with replica = 2, 3 ... 8
- Thin provisioning
- Fast distributed rebuild
- Availability, Fault tolerance
  - Disk, Node, Interconnect
  - Automatic rebuild
  - Distributed HotSpare Space
- Transparent Block, File access
- Reliability and Consistency
- Scalable Performance
- Pure PCIe-SSD for extreme Transaction processing
Ceph – Architectural components and interfaces

LIBRADOS
A library allowing apps to directly access RADOS (C, C++, Java, Python, Ruby, PHP)

Ceph Object Gateway (RGW)
A web services gateway for object storage, compatible with AWS S3 and OpenStack Swift

Ceph Block device (RBD)
A reliable, fully-distributed block device with cloud platform integration

Ceph FS
A distributed file system with POSIX semantics and scale-out metadata management

RADOS (Ceph Object Storage)
A software-based, Reliable, Autonomous, Distributed Object Store comprised of self-healing, self-managing, intelligent storage nodes and lightweight monitors
Software Defined Networking (SDN)
OpenStack Networking in general

Logical Network "overlay" (Network Virtualization Software) Switching, Routing, NAT, DHCP, FW, SLB, VPN

Physical network
OpenStack native SDN implementation

- Based on OVS – **Open vSwitch**
- Limitation in scalability and robustness
  - … due to centralized controller approach (central Control- and Network Nodes)
  - Scales up to ~30-150 compute nodes (depends on #VM, load, agility)
  - Single failure domain: Layer 2 network (switching) connectivity only (provider network)
- Limited functionality for example…
  - No possibility to offer secured shared services, no common DC security (3-tiered FW design)
    - one single Firewall per tenant, N/S traffic only
  - Limited redundancy
    - Not possible to scale out compute nodes (with DVR) in combination with redundancy (L3HA)
- High complexity
  - Multiple services, multiple bridges, IP tables, namespaces
MidoNet as one example for a stable and well scaling overlay network solution

- MidoNet „overlay“ network
  - Software defined - decoupled from physical network
  - „Cloud-admin“ responsibility
- High scalable and redundant services
  - Distributed agents on each hypervisor
  - Common database for network states (NSDB)
- Typical „underlay“ Data-Center physical Network
  - „Spine-Leaf“ architecture
- Static, Stable, performant
- „Network-admin“ responsibility
MidoNet – Decoupling logical topology from physical

Logical Topology

Physical Topology
Container Orchestration with Magnum
Why Container Orchestration?
The Road to Container Orchestration

Bare Metal Servers → Virtualized Servers → VM Sprawl

Virtualized Servers → System Containers → Container Sprawl
Container Sprawl
Different Container Types

System Containers

- „Lightweight OS“
- Usually either manually maintained or using config management
- OpenStack support via „Nova-Docker“

Application Containers

- Application centric
- Micro Services
- Layered deployment
- Container Orchestration Engines
- OpenStack support via „Magnum“
Introducing Container Orchestration

System Containers → Cloud

App Containers

Container Orchestration

Cloud

App Containers

Container Orchestration
Containers and Cloud

- Containers do not replace VMs
- VMs allow tenant isolation and hardware abstraction
  - Live Migration
  - Snapshotting
- Containers allow for better orchestration
  - Replication
  - Layering
  - Service Isolation
Magnum with Kubernetes

- OpenStack Magnum (COE)
- OpenStack Heat (VMO)
- OpenStack Nova (VM)

Kubernetes Components:

- Kubernetes Master VM
  - etcd
  - API
  - RC

- Kubernetes Minion VM
  - proxy
  - Containers

- Flannel
OpenStack Magnum Architecture

Nova Instance #n
- Heat Config Elements
- Kubernetes/Swarm
- Docker Containers
- Docker
- Micro OS (Fedora Atomic, CoreOS)

Cloud Init

Controller Node
- Magnum API
  - Bay
  - Baymodel
  - Node
  - Service
  - Pod
  - RC
- Magnum Conductor
  - Heat Templates
    - Handlers
    - Database
- OpenStack Heat
  - Nova
  - Neutron
  - Glance
  - Cinder
- Magnum Client
OpenStack Ironic

- Why use OpenStack with Bare Metal?
Bare Metal Use Cases

- High Performance Computing (HPC)
- Workload can not be effectively virtualized (access to specific peripherals, not certified)
- Big In Memory Databases
- Tenant isolation is unnecessary
- Workload by itself wants to virtualize
  - OpenStack on OpenStack (OoO)
  - Test Automation for Operating Systems :)
OpenStack Compute on Bare Metal

- Scheduling workload on a physical machine
  - Heterogeneous hardware can be selected via flavor properties
- Single tenant - maybe just use Flat + Standalone OpenStack Ironic?
- Multiple tenants
  - Network isolation is only partially possible
  - Tenant isolation via secure-erasing disks on redeploy
  - (Firmware-reflash between uses)
OpenStack Ironic – 2 views

Step 1: Operator enrolls hardware

$ ironic node-create ...

Ironic API

Ironic Conductor

hardware drivers

servers

Step 2: User creates instance

$ nova boot ...

Nova Compute

Ironic virt driver

Nova Scheduler

Nova API
PXE deployment process (I)
PXE deployment process (II)

- Nova
- API
- Conductor
- Neutron
- Node

- POST /vendor_passthru?method=pass_deploy_info
- runs deploy ramdisk
- exposes disks via iSCSI
- continues deploy
- mark node as ACTIVE
- iSCSI attach
- copies user image
- iSCSI detach
- sends "done" message
- terminates iSCSI endpoint
- reboots into user instance
Monitoring and Logging based on OpenStack Monasca

- Mission: Standardized OpenStack project instead of vendor specific solutions
- Logging-as-a-Service
- Monitoring-as-a-Service
- Multi-Tenancy, RBAC
Monasca - Functions and Technologies

- **Monasca:**
  - Metrics / Monitoring
  - Logging
  - Alarming

- **ElasticSearch:** Search Engine
- **Logstash:** Collection, Parsing and optionally Transformation
- **Kibana:** Dashboard Graphing/Trending
Monasca – some details

- **Metrics**
  - Supervise health status of your infrastructure
  - Alert on Service Unavailability
- **Logging**
  - Root cause analysis, Correlation analysis
- **Extensions over standard ELK**
  - Multi-Tenancy
  - Focus on Scalability / Performance
Monasca Architecture
Monasca Logging
Summary / Discussion
OpenStack and Linux – common features

- Success based on Open Source
- Efficient Resource Management
- Mechanism of Abstraction
- Sharing and Multitasking
- Manageable Complexity (Modularity, APIs, …)
- Scalability
- Similar definitions
shaping tomorrow with you