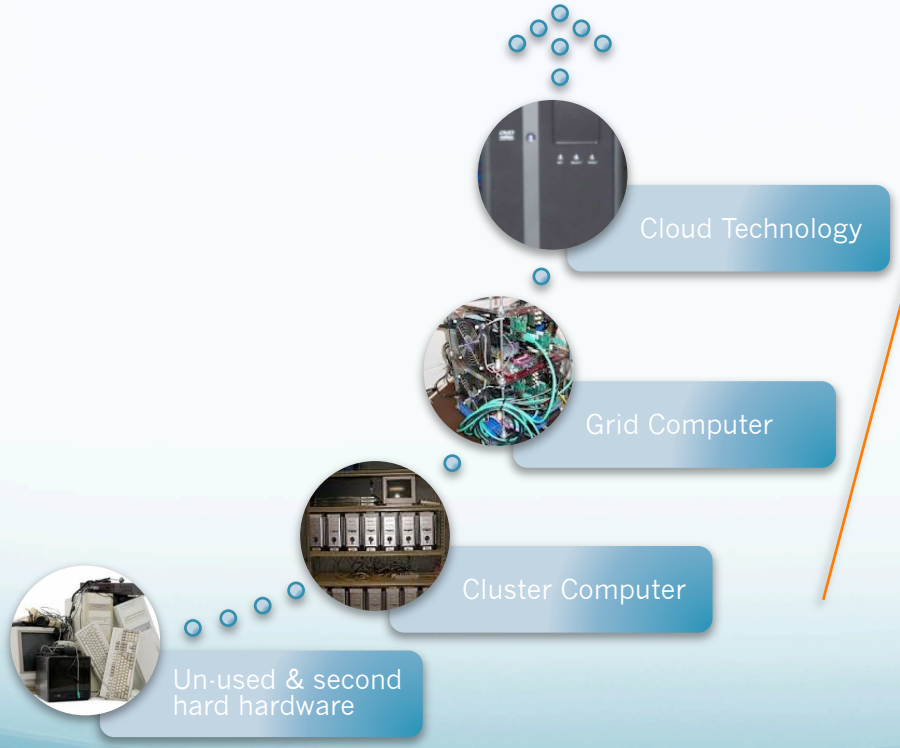
Department of Computer Science & Electronics
Faculty of Mathematic and Natural Sciences
Universitas Gadjah Mada
www.dcse.fmipa.ugm.ac.id

A Research and Working Group on
Grid & Cloud Technology
Universitas Gadjah Mada
www.cloud.wg.ugm.ac.id

Cloud.wg.ugm.ac.id

- Grup riset dan pengembangan, fokus pada teknologi Cloud and Grid
- Berlokasi di Lab SKJ, JIKE FMIPA UGM, Ged SIC lt 3 MIPA Utara
- Lead Researcher : Mardhani Riasetiawan, MT CompTIA Cloud Essential
  - Research students : 10+ person
- Diskusi
  - FB grup : Cloud.wg.ugm.ac.id

GamaBox (technology roadmap)



It's Produce GamaBox

GamaCloud

GamaExplorer

GamaBoxTV

Background

- Riasetiawan (2012), virtualisasi menyediakan teknologi untuk menyelenggarakan *cloud computing* dan layanan *cloud computing.*
  - *http://tekno.kompas.com/, dengan menggunakan virtual data center pengiriman data yang cukup besar hanya membutuhkan waktu sekitar 30 menit dari sebelumnya yang menggunakan server fisik membutuhkan waktu 3 jam.*

- Mehmood dkk. (2013), teknologi *cloud computing* memiliki resiko dari aspek keamanan sebab arsitekturnya yang bersifat terbuka dan terdistribusi sehingga rentan terhadap serangan.

Serangan

- *http://inet.detik.com/*, *cloud computing* di Indonesia rentan akan serangan DDoS, yang menyebabkan Indonesia berada di posisi nomor dua untuk negara yang sering diserang DDoS pada tahun 2012.

- *http://www.cert.or.id,* pada bulan September-Oktober 2013 angka pengaduan terhadap spam merupakan yang tertinggi, yang mencapai 62,51%.
  - Sedangkan malware dan *network incident* berada pada urutan kedua dan ketiga dengan masing-masing persentase 13,94% dan 9,88%.

Pencegahan

- Mathew dan Jose (2012) mengungkapkan bahwa *cloud computing* memerlukan sistem seperti IDS (Intrusion *Detection System)* untuk melindungi setiap mesin virtual dalam melawan serangan yang ada.
- Roschke dkk. (2009), IDS merupakan suatu ukuran keamanan yang efisien dan telah digunakan untuk mengamankan infrastruktur TI
- Bakshi dan Yogesh (2010) mengungkapkan jika terdapat serangan, maka *virtual server* akan merespon dengan cara memindahkan aplikasi target ke mesin virtual pada data center yang lain

Literature

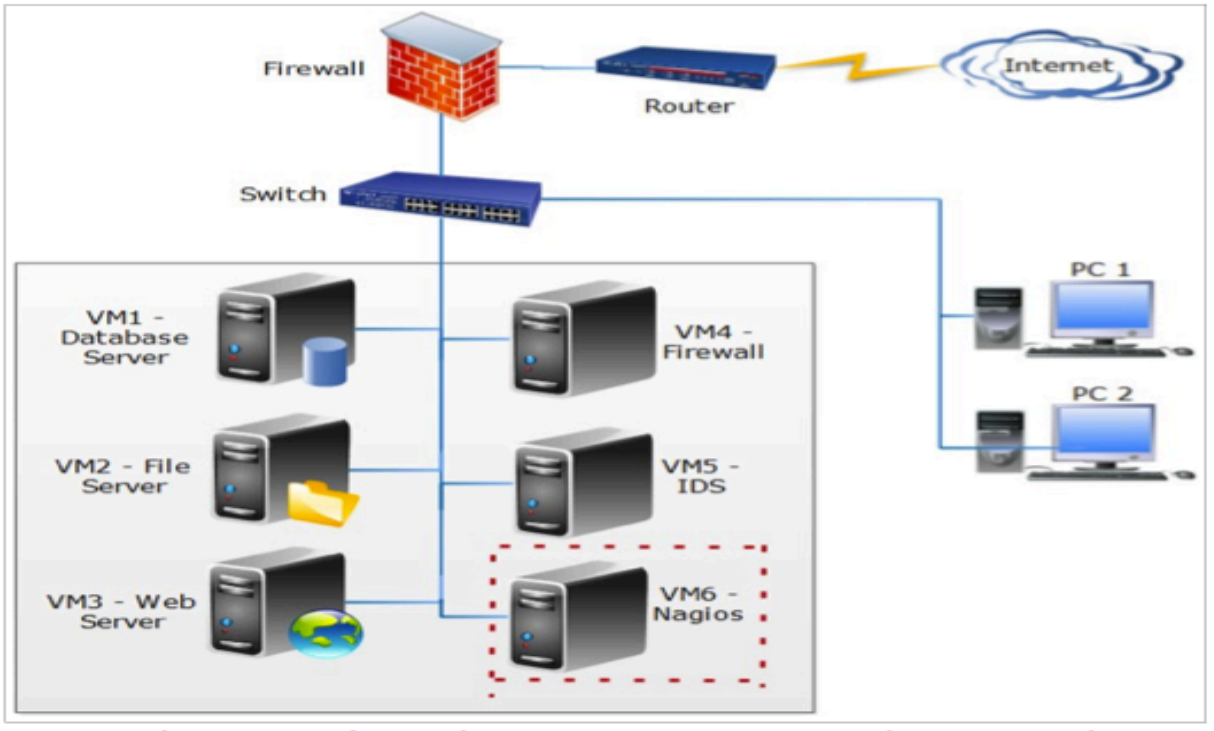
- Mathew dan Jose (2012) mengungkapkan bahwa pada infrastruktur *cloud* dibutuhkan suatu IDS untuk menutupi berbagai ancaman.
- Khumar dan Sharma (2013) menggunakan metode CBF *(Confidence Based Filtering)* untuk mencegah serangan pada *cloud.*
  - Metode CBF berupa *packet filtering* yang akan menyaring paket yang masuk ke *cloud* dengan cara membedakan apakah paket tersebut merupakan suatu serangan atau merupakan paket yang sah dengan cara mencocokkan pola yang ada.

- Roschke dkk. (2009), bahwa virtualisasi sebagai salah satu teknologi kunci untuk *cloud computing* dengan menyediakan arsitektur yang *extensible* untuk mengintegrasikan manajemen VM dan manajemen IDS.
  - Dengan diletakkannya minimal satu IDS-VM tiap layer, maka ketika terjadi serangan yang besar pada pengguna akan lebih mudah terdeteksi dengan menghubungkan setiap *alert* yang masuk.

- Bakshi dan Yogesh (2010) menerapkan konsep virtualisasi untuk merespon serangan DDoS pada *cloud.*
- Alharkan dan Martin (2012), virtual mesin pada *public clouds* dapat dilindungi dengan menggunakan sebuah *framework* yaitu IDSaaS *(Intrusion Detection System as a Service).*

Metodologi

- Desain Implementasi



Secure Data Center

Concept & standard

Features & Capabilities

Consolidate all service on the single box

- Email
- Web hosting
- File management
- E-learning system and content
- Academics activities
- Multimedia services
- Based on Single sign on services

Mobility

- Mobile, Gadget and PC based access
- Unlimited transfer rate on local network

Simple On

- Single access user
- Collaboration platform
- Install execution

Management

- Secure content & access management
- “national” data
- Collaboration

Accountability

- Monitor for illegal content
- Backup management on data center
- Personal backup
- Live time

Implementasi

Open Nebula + Nagios + Secure VM with IDS

OPEN NEBULA

Cloud and Data Center Virtual Infrastructure Management

VIRTUAL DATACENTER MANAGEMENT

Integrate and federate existing IT assets in datacenters

VMWARE VCLOUD ALTERNATIVE

Open replacement for vCloud on VMware infrastructure

BIG DATA, HPC AND SCIENCE CLOUDS

Hosting virtualized compute & data processing environments

HYBRID CLOUD COMPUTING

Single management interface for private and public resource

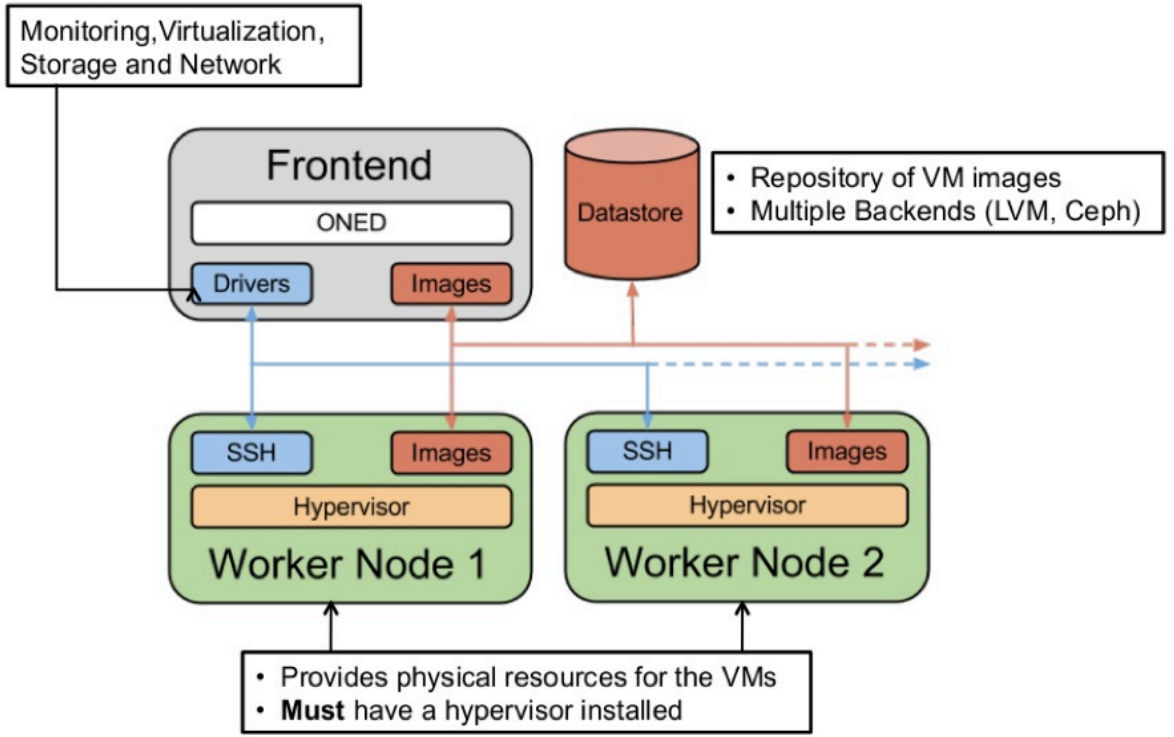
TELECOM CLOUD & NFVS

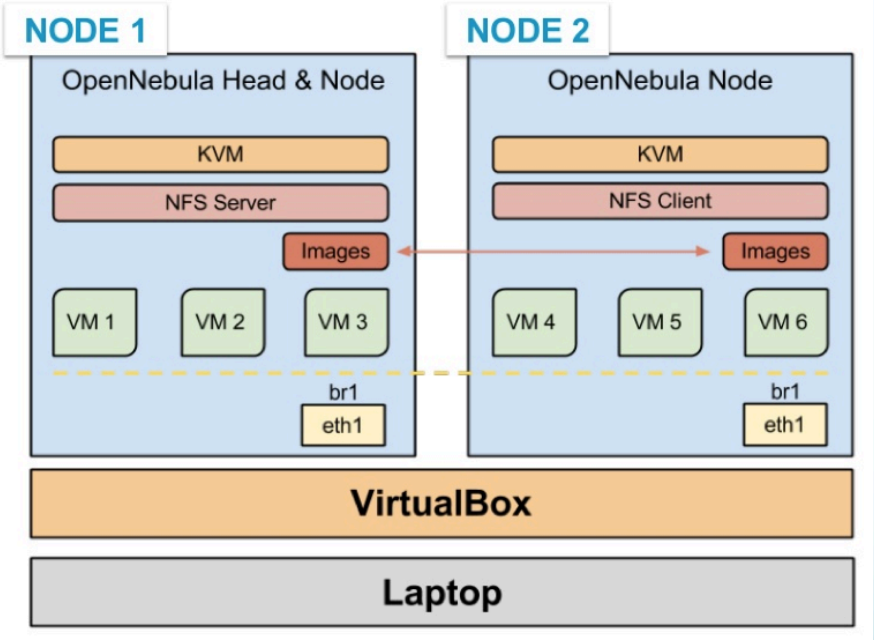
Deployment of private clouds to support VNFs

CLOUD HOSTING

Build a cloud provisioning service

- Planning the Installation
- Installation (node 1 - Frontend)
- Installation (node 2 - Worker Node)
- Configuration
- Basic Usage
  - Managing Hosts
  - Images, Networks, Templates and VMs
  - Managing Users, Quotas and ACLs
  - Logging & Debugging





- Head node
  - ssh, ruby
  - OpenNebula: oned, mm_sched, sunstone, …
- Worker nodes
  - Hypervisor (KVM, Xen or VMWare)
  - ssh, ruby (Xen & KVM)
- Optional
  - Storage Backends (LVM, iSCSI, Ceph, …)
  - Networking systems (VLAN, Open vSwitch, …)
  - Ganglia, LDAP, Apache, Nginx

Hands on (node1) !

- Activate repo and Install Packages

```
# cp /var/tmp/tutorial/opennebula.repo /etc/
yum.repos.d/
# yum install opennebula-server opennebula-sunstone
opennebula-node-kvm opennebula-flow
```

- Add QEMU drivers

```
# sed -i 's/"kvm" ]/"qemu" ]/' /etc/one/oned.conf
```

- Configure NFS Server

```
# cat /etc/exports

/var/lib/one
*(rw,sync,no_subtree_check,root_squash,anonuid=9869,anongid=9869)
```

- Configure Sunstone

```
# sed -i 's/127.0.0.1/0.0.0.0/' /etc/one/sunstone-server.conf
```

- Start Services

```
# service nfs start
# service libvirtd start
# service opennebula start
# service opennebula-sunstone start
```

- Quick overview of the CLI

```
# gpasswd -a oneadmin wheel
# su - oneadmin
$ oneuser show
$ oneuser -help
```

- OpenNebula CLI Commands

```
$ one[TAB]
```

oneuser	Manage Users	**oneimage**	Manage Images
onegroup	Manage Groups	**onetemplate**	Manage Templates
oneacl	Manage ACLs	**onevm**	Manage VMs
onehost	Manage Hosts	**oneacct**	Accounting Tool
onecluster	Manage Clusters	**onemarket**	Marketplace Tool
onevnet	Manage Networks	**onedb**	DB Tool
onedatastore	Manage Datastores		

- Activate repo and Install Packages

```
# cp /var/tmp/tutorial/opennebula.repo /etc/
yum.repos.d/
# yum install opennebula-node-kvm
```

- Configure Network, Hostname, NFS and sudo

```
# echo HOSTNAME=node2 > /etc/sysconfig/network
# hostname node2
# sed -i 's/1.1.1.1/1.1.1.2/' /etc/sysconfig/network-
scripts/ifcfg-br1
# ifconfig br1 1.1.1.2/24 up
# mount -t nfs 1.1.1.1:/var/lib/one /var/lib/one
# gpasswd -a oneadmin wheel
# service libvirtd start
```

- OpenNebula needs passwordless ssh access to all the nodes from all the nodes:

```
# (as oneadmin)
$ ssh-keyscan node1 node2 > ~/.ssh/known_hosts

# test it!

$ ssh node2
$ exit
$ ssh node1
$ exit
```

- Get the Sunstone Login information

```
# (as oneadmin)
$ cat ~/.one/one_auth
oneadmin:<password>
```

- Try out sunstone! http://localhost:9869

/etc/one/oned.conf (open this file and take a look!)

- OpenNebula Daemon:
  - LOG, PORT, DB
- Monitoring Intervals: MANAGER_TIMER, MONITORING_INTERVAL
- Configuration options for VMs:
  - VNC_BASE_PORT
  - MAC_PREFIX (MAC ⇔ IP)
  - DEFAULT_DEVICE_PREFIX = "hd“ (or vd, xvd, etc…)
- Drivers:
  - IM_MAD, VMM_MAD, TM_MAD, DATASTORE_MAD, HM_MAD, AUTH_MAD
- Resources:
  - DEFAULT_UMASK
  - VM_RESTRICTED_ATTR, IMAGE_RESTRICTED_ATTR

/etc/one/sched.conf (open this file and take a look!)

- Scheduler Daemon:
  - ONED_PORT, SCHED_INTERVAL, LOG
- Dispatch Options
  - MAX_VM, MAX_DISPATCH, MAX_HOST, LIVE_RESCHEDS
- Policy
  - DEFAULT_SCHED (packing, striping, load-aware, custom)

Hands on! (Sunstone)

- Create one host in Sunstone: **node1**
  - Virtualization: KVM
  - Information: KVM
  - Network: dummy
  - Cluster: dummy
- Watch transition **INIT → ON**
- Click on the row for more information
  - Automatic gathering of monitoring data
  - Take a look at the graphs

```
$ onehost -help
$ onehost create -help

$ ssh node2 ls -ld /var/tmp/one
$ onehost create node2 -i kvm -v kvm -n dummy
$ onehost list
$ onehost top

# Wait for ON ... and then CTRL-C

$ ssh node2 ls -ld /var/tmp/one
$ onehost show node2
$ onehost show 1
$ onehost show -x 1
```

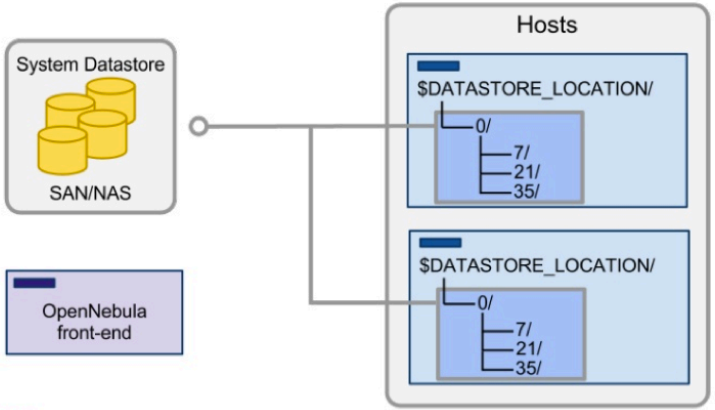
- Create an Image in Sunstone
  - Name: tty
  - Provide a Path: /var/tmp/tutorial/ttylinux.qcow2.img
  - Advanced → Driver: qcow2
  - Datastore: default
  - Create!
- Watch transition **LOCKED → READY**
- Ownership and Permissions (ala Unix!), Size, Driver...

Hands on! (CLI)

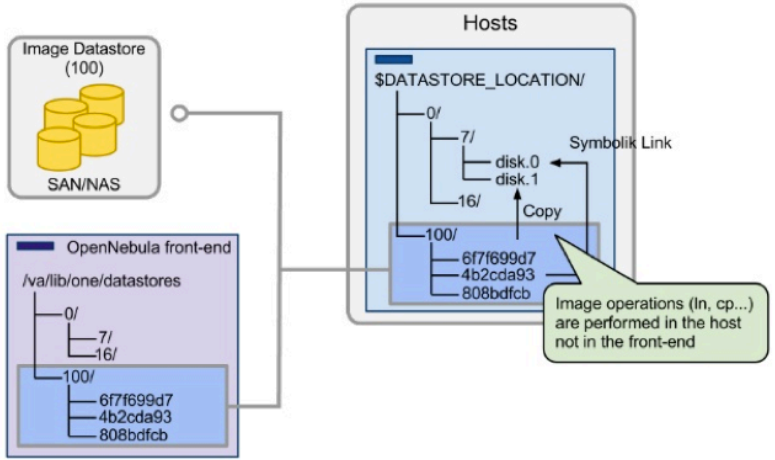
```
$ oneimage list
$ oneimage show tty

# DO NOT EXECUTE THE FOLLOWING COMMAND
$ oneimage create --name tty --driver qcow2
  --path /var/tmp/tutorial/ttylinux.qcow2.img -d
default
```

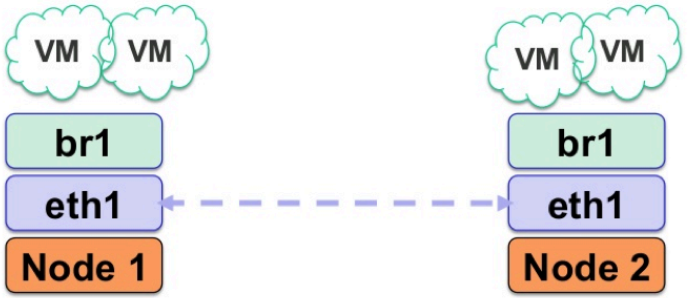
- Inspect each Datastore:
- The **system** datastore:
  - Holds images for **running** VMs
  - The TM_MAD (transfer manager driver) is **shared** which means:



- The **default** datastore:
  - Holds images ready to be cloned or linked for VMs
  - The DS_MAD is **fs** because our image is a regular file
  - The TM_MAD (transfer manager driver) is **shared** which means:



- Create a new Network
  - Name: private
  - Type: Fixed Network
  - IP: 192.168.0.1 -> [ENTER] -> repeat ... -> 192.168.0.4
  - Network Model: default
  - Bridge: br1




```
$ cat private2.net
NAME = private2
TYPE = fixed
BRIDGE = br1
LEASES = [ IP = 10.0.0.1 ]
LEASES = [ IP = 10.0.0.2 ]

$ onevnet create private2
$ onevnet list
$ onevnet show private
$ onevnet addleases private 192.168.0.105
$ onevnet hold private 192.168.0.105
```

- A template is a Virtual Machine definition ready to be **instantiated**
- It has CPU, Memory, Disks, NIC, Graphical Ports, etc...
- Create a new Template:
  - Name: ttylinux
  - CPU: 0.1
  - Memory: 64M
  - Storage: tty
  - Network: private
  - Input/Output: VNC + Keymap
  - Random values in Context → Custom Variables
  - Create!

```
$ onetemplate create --help
$ onetemplate create --name ttylinux --cpu 0.1 \
    --memory 64 --disk tty --nic private --vnc --dry

NAME="ttylinux"
CPU=0.1
MEMORY=64
DISK=[
  IMAGE="tty"
]
NIC=[
  NETWORK="private"
]
GRAPHICS=[ TYPE="vnc", LISTEN="0.0.0.0" ]
```

- Instantiate the template
  - Deploy 2 VMs
  - Leave the name blank
- Watch the transition **PENDING → RUNNING**
- In which host is running each VM?
- vnc (root / password)
- ifconfig → configured using context
- migrate
- live-migrate
- ping the other machine

- Instantiate the template
  - Deploy 2 VMs
  - Leave the name blank
- Watch the transition **PENDING → RUNNING**
- In which host is running each VM?
- vnc (root / password)
- ifconfig → configured using context
- migrate
- live-migrate
- ping the other machine

Nagios & SecureVM

DEMO