

Open Infrastructure - Task #6869

Challenge OpenStack and OpenNebula with ucloud

06/22/2019 11:27 PM - Nico Schottelius

Status:	Rejected	Start date:	06/23/2019
Priority:	Normal	Due date:	09/18/2019
Assignee:	Nico Schottelius	% Done:	100%
Category:		Estimated time:	0.00 hour
Target version:			
PM Check date:			

Description

Introduction

For proper growth and stability, we need to challenge our setup. We will try to replace opennebula with ucloud and at the same time challenge OpenStack.

This is not only a technical, but also a public project, which Sanghee will write the public story of.

ucloud supports IPv6 first and might only support IPv4 via NAT64 or proxying.

General Requirements:

- a fully portable cloud management system that is API based, exposes all internals w/o secret keys
 - supporting console via guacamole
 - users in ldap
 - api authentication in ungleich-otp
 - firewalling in ufirewall (or similar)
- A great team
- To be built in less than 100 days

Technology stack

- python3
 - easy to read
- flask
 - easy to understand
- ldap
 - well known
- ungleich-otp
 - for API authentication
- etcd
 - storing VMs, networks, etc.
- nft (Linux), pf (BSD)
- JSON
 - describing the data, easy to handle
- Prometheus
 - For monitoring hosts and VMs
- VXLAN
 - For networks
- Ceph
 - as a datastore

Technical requirements

- ucloud should be portable
 - While the primary target is Linux, it should run on FreeBSD/OpenBSD as well
- There should be no single point of failure
 - APIs should be announced via BGP to the routers
 - Switches will then use ECMP to load balance
 - All APIs write to a distributed data store (v1: etcd) -> all data is distributed, too.

- Fast dead host detection
 - Dead hosts should be detected fast, VMs should be rescheduled fast

Components

User API (ucloud-api)

Entrance point / communication with the user by CLI. Flask based. Allows for the following actions in v1:

- Create VM
- Delete VM
- Create new network
- Attach network to VM
- Detach network from VM
- Delete network

User web interface (ucloud-web)

Might be based on original dynamicweb code (<https://code.ungleich.ch/ungleich-public/dynamicweb>).

Scheduler (ucloud-scheduler)

The scheduler knows about hosts, their capacities and their usage. The scheduler decides which VM gets scheduled where. The scheduler is also responsible for rescheduling VMs (f.i. due to another host becoming better for a specific VM).

How it works:

- Has a list of hosts for usage
- Knows about the capacity (installed cores, installed ram) of a host

Host manager (ucloud-host)

Manages hosts. If a host crashes, instructs scheduler to restart VMs (host will be selected by scheduler). If a host is added, the scheduler can use it.

VM manager (ucloud-vm)

Starts and stops VMs. Runs on every VM host.

Reacts on information from the scheduler. This component needs to support qemu on Linux and bhyve and co. on OpenBSD/FreeBSD.

How it works:

- watches a specific key in etcd, for instance: /v1/vm/
- if a key is added, check if it is a VM that should be started on THIS host.
 - if yes, start it
- if a key is modified, check if a VM that is on this host, should be stopped

Network manager (ucloud-net)

This service will run on every host and watch keys in the prefix /v1/network/

Creates and manages virtual layer 2 networks. Basically does the following:

- Create VXLAN with correct IPv6 multicast address

IPAM (ucloud-ipam)

Network address manager. Provides dhcpd/dhcp6d/router advertisements.

Legacy IP supporter (ucloud-legacy-ip)

For users who require legacy IP (IPv4), add a service that

- adds a 1:1 NAT64 entry

- adds a protocol based proxy entry
 - http(s)
 - smtp(s)
 - ssh jump host

Firewall (ucloud-firewall)

The VMs should not be able to interfere with other VMs or hosts in a malicious way. The following protective measures need to be implemented:

- prevent dhcpd answers in public networks
- prevent router advertisements in public networks
- prevent VM from using incorrect mac address
- prevent VM from using incorrect ipv6 addresses

Note to Nico: ping reyk for possible involvement

Image store (ucloud-image)

Ceph will be used for storing images.

Features:

- Allow uploading of images
- Allow cloning of images (required for starting a VM based on an existing image)
- Allow deleting of images
 - For cloned images after shutting down the VM

How it works for uploading:

- There is a server named image.datacenterlight.ch
- Every user in ldap can login via sftp and upload files
 - The server needs to authenticate against ldap for listing users
 - There needs to be a service to **pull** ssh keys from users into their home
- There will be a base path for users to store their stuff
 - like /var/www/\$USER/
 - reasoning for /var/www: to use nextcloud on top later
- The required image format is qcow2

How it works for using it as an image:

- User uses ucloud-cli image-create --name xyz
- The service picks up the image from /var/www/\$USER
- Checks if it is qcow2 -> if not reject
- If qcow2: use qemu-img | rbd import

Metadata (ucloud-meta)

Provides access to this to

- public ssh keys
- other data users provide

How it works

- VMs need well known entry point
- Should likely be DNS based
- Might be reachable by <http://metadata>
 - This excludes https!
 - Maybe network configuration can contain metadata server?
 - dhcp option
 - router advertisement?
 - Maybe by convention: metadata.\$domain
 - \$domain injected by dhcp/router advertisements

Payment service (ucloud-pay)

If a user requests a service and the service is not free, the user will be asked to pay for it. Should support at least

- credit card
- bank transfer

Might also manage existing money / coupon / etc.

CLI (ucloud-cli)

All services will be primarily available via API, web is a second class citizen. The CLI might be based on / related to <https://code.ungleich.ch/ungleich-public/ungleich-cli>.

Authentication

time based one time tokens as implemented in ungleich-otp will be used for service authentication:

<https://code.ungleich.ch/ungleich-public/ungleich-otp>

Milestones

v1 First MVP : 2019-07-02

This version should have the following features:

- create and delete VMs

v2 : 2019-07-16

Additionally supports

- network service: so VMs get an IPv6 address
- Cleanups / clarifications from v1

v3 : 2019-07-30

Additionally supports

- metadata service (for injecting ssh keys and more)
- Cleanups / clarifications from v2

From v3 on we should be able to setup test VMs for our own usage

v4 : 2019-08-12

Productive version. We are able to migrate our own production VMs to ucloud.

v5: 2019-08-26:

Customer usable version.

Additionally supports

- Console access via guacamole (ldap, totp, vnc)

Features not mapped to versions

- VNC should only be exposed as a unix socket from the VM
 - This is much more secure and requires local access to the socket
 - To access the console as a user, we will ssh into the host that runs the VM and start socat on the host
 - i.e. similar to ssh VMHOST "socat ..." and then access it via guacamole

Subtasks:

Task # 6857: Create ucloud-firewall

Task # 6871: Create ucloud-api

Task # 6872: Create ucloud-vm

Rejected

Closed

Closed

Task # 6875: Create ucloud-scheduler	Closed
Task # 6876: Define ucloud JSON formats	Closed
Task # 6877: Start the first VMs on place9 cluster using ucloud (ucloud v1)	Rejected
Task # 6897: Create ucloud-image service	Rejected
Task # 6899: Create ucloud-file-scan service	Rejected
Task # 6900: Allow creating an image from a file in ucloud-api and ucloud-cli	Rejected
Task # 6901: Describe on how to configure the files service	Rejected
Task # 6903: Create a cdist type for files.datacenterlight.ch	Rejected
Task # 6904: Implement ucloud-image-store management	Rejected
Task # 6914: ucloud-image-scanner	Closed
Task # 6902: Document the ucloud services and APIs in the open infrastructure	Closed
Task # 6908: ucloud v2 features	Rejected
Task # 6909: Accept units in ucloud-api, but store in non-unit format	Closed
Task # 6915: Introduce host status' and over/underbooking constraints	Rejected
Task # 6931: ucloud-host	Rejected
Task # 6995: ucloud-pay v1: Implement payment support into ucloud	Rejected
Task # 6996: ucloud-pay v2: add support for retrieving payments from ZKB	Rejected
Task # 7138: Nico's open / next points for ucloud	Rejected
Task # 7205: Try 1: Installing ucloud on Arch Linux	Rejected
Task # 7139: Approach Azure, AWS, Softwayer, OpenStack and Cloudstack users and ask the...	Closed
Task # 7206: Create ucloud page under ungleich.ch/ucloud	Rejected
Task # 7221: Pre 0.0.1 release fixes	Rejected
Task # 7278: Define how networking works in ucloud	Rejected
Task # 7283: Cleanups for ucloud 0.0.2	Closed
Task # 7285: Flows to implement and do document for go live (parent)	Rejected
Task # 7286: Flow 1: as a sysadmin, I want to setup ucloud w/o ceph and run a VM	Closed
Task # 7287: Flow 2: As a user I want to order an IPv6 only VM (via cli) (existing OS)	Rejected
Task # 7288: Flow 3: As a user I want to order a VM with my own OS (IPv6 only)	Rejected
Task # 7289: Flow 4: As a user I want to have a dualstack reachable VM	Rejected
Task # 7290: Flow 5: as a user I want to map an IPv4 address to another VM	Rejected
Task # 7291: Flow 6: as a sysadmin I want to know / install all requirements	Rejected
Task # 7292: Flow 7: as a sysadmin I want to be able to migrate VMs	Rejected
Task # 7293: Flow 8: As a user I want to be able to order a new network and use it	Rejected
Task # 7294: Flow 9: as a user I want to have an overview of my spendings and bills	Rejected
Task # 7295: Flow 10: as a sysadmin, I want to be able to see statistics	Rejected

History

#1 - 06/22/2019 11:33 PM - Nico Schottelius

- Project changed from 5 to Open Infrastructure

#2 - 06/22/2019 11:34 PM - Nico Schottelius

- Description updated

#4 - 06/22/2019 11:47 PM - Nico Schottelius

- Description updated

#5 - 06/22/2019 11:56 PM - Nico Schottelius

- Description updated

#6 - 06/22/2019 11:59 PM - Nico Schottelius

- Description updated

#7 - 06/23/2019 12:03 AM - Nico Schottelius

- Description updated

#8 - 06/23/2019 12:12 AM - Nico Schottelius

- Description updated

#9 - 06/23/2019 12:28 AM - Nico Schottelius

- Description updated

#10 - 06/23/2019 12:43 AM - Evil Ham

I think payment method --> pluggable thing that let's the system know sth can be processed.

That can be: codes, credit in account, credit card, bank transfer (aka manual validation because banks live in the 20th century (note: openbanking is a thing that **should** be working Europe wide, it can have gotten better/usable)).

Basically: you don't care about the specifics, design so payment is handled somehow.

The rest requires a non-available brain. It's too warm.

#11 - 06/23/2019 04:55 PM - Nico Schottelius

- Description updated

#12 - 06/23/2019 06:38 PM - Nico Schottelius

- Description updated

#13 - 06/28/2019 01:41 PM - Nico Schottelius

- Description updated

#14 - 07/01/2019 02:07 PM - Nico Schottelius

- Description updated

#15 - 07/08/2019 11:05 AM - Nico Schottelius

- Description updated

#16 - 01/02/2024 02:07 PM - Nico Schottelius

- Status changed from In Progress to Rejected