

Connecting Docker for Cloud Services IaaS Resource Consolidation with Network Virtualization and SDN

DaoliCloud Company
Beijing & Shanghai, China
nvi.daolicloud.com
nvi at daolicloud dot com
October, 2014

Presentation in 2014 Container Conference
China Software Developers Network (CSDN)



Abstract

The explosive scale of container CPUs needs highly efficient network virtualization

Challenge The number of container-based CPUs will be 10-100 multiples up over that of hypervisor-based virtual machines; cloud networking needs upgrade to fit the explosive scale of containers

Question Can SDN standards, e.g., Openflow, support highly efficient network virtualization for containers' networking?

DaoliCloud's Positive Answer Presented here is an innovative improvement to Openflow: Network Virtualization Infrastructure (NVI) + Tenant SDN Controller (TSC) for efficient network virtualization, and showing why/how proposed technologies are fit for networking container-based clouds of unbound scale

Application Overlay network for highly distributed containers, virtual machines, and hardware servers, to patch independent clouds of such nodes into a virtual cloud of any scale and elasticity



Outline

- New challenges and opportunities from container-based direct CPU virtualization
- Technology: Network Virtualization Infrastructure (NVI) + Tenant SDN Controller (TSC)
- Application: Crowdsourcing distributed IaaS resources to construct very large cloud services
- Long term value
- Conclusion

Technical material

- Inherent problems with existing networking technologies
- How the scale of container-based CPUs worsens the matter
- NVI and TSC technologies in detail



Container-based direct CPU virtualization

Lightweight and direct CPU virtualization

- All guest containers in a host share using the only operating system of the host, very lightweight, very resource efficient, and bare-metal direct I/O speed
- In contrast, hypervisor-based CPU virtualization is indirect; every VM consumes a guest operating system, very heavy, resource inefficient, and indirect I/O speed
- CPU utilization: On one same server, the number of Docker containers > 25 x that of hypervisor VMs with similar output from either virtual instances, thanks to the 80-20 rule
- Well-known player in container-based direct CPU virtualization: Docker, www.docker.com

Hello...



Pardon?



Not in service area !





Unfortunately, Docker networking is not scalable

- Every Docker host provides L3 services for hosted containers, NAT (Network Address Translation) is one such
- NAT is like a diode firewall, default blocking ingress traffics; Therefore, containers in different hosts have no IP connectivity
- NAT can be bypassed by L3 encapsulation protocols: e.g., STT, VXLAN, GRE, IPsec, LISP, VPN, ..., as shown in the figure below
- Unfortunately, encapsulation protocols **do not scale**; we shall discuss this in detail in technical material
- Probably a little unlike for PaaS use cases, networking non-scalability doesn't seem to feature an IaaS service very well

Payload encapsulation to nullify all network functions, e.g., NAT, and block all world visibility for containers



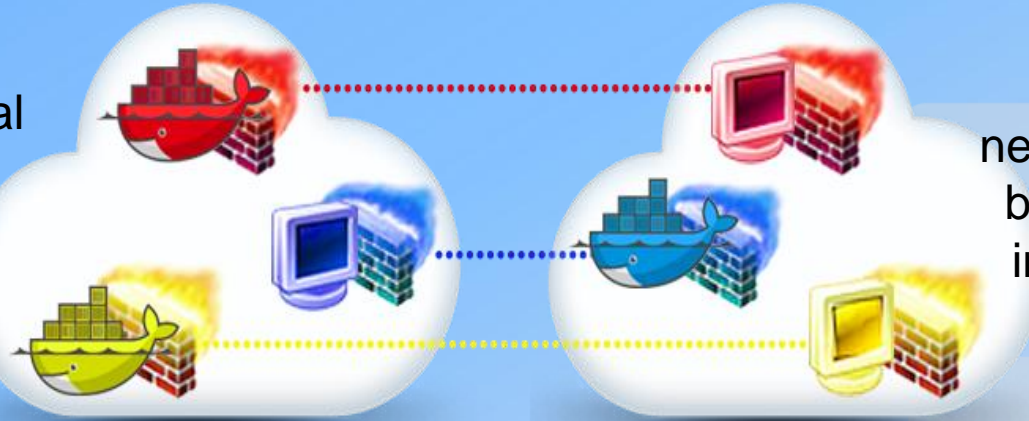
Encapsulation header label = control plane info placed in forward plane

Underlay packet headers L4/L3/L2



Positive side: Smallness in IaaS orchestration is NOT bad at all

Small scale of physical implementation for IaaS orchestration is like differentiation in math, to enable smooth integration




Provided network patch boundary is invisible by tenants

Without seeing any network boundary of small clouds, a tenant can “crowdsource” IaaS resources from independently orchestrated small clouds, just like integration-&-differentiation in math, to build a virtual private cloud of any scale and elasticity

“Differentiation-&-Integration” of physically small implemented clouds have the following very desirable properties:

- IaaS orchestration of small server racks/farms can be stable and reliable; this is very important for service providers
- Arbitrary elasticity unbound by any provider, avoidance of cloud lock-in

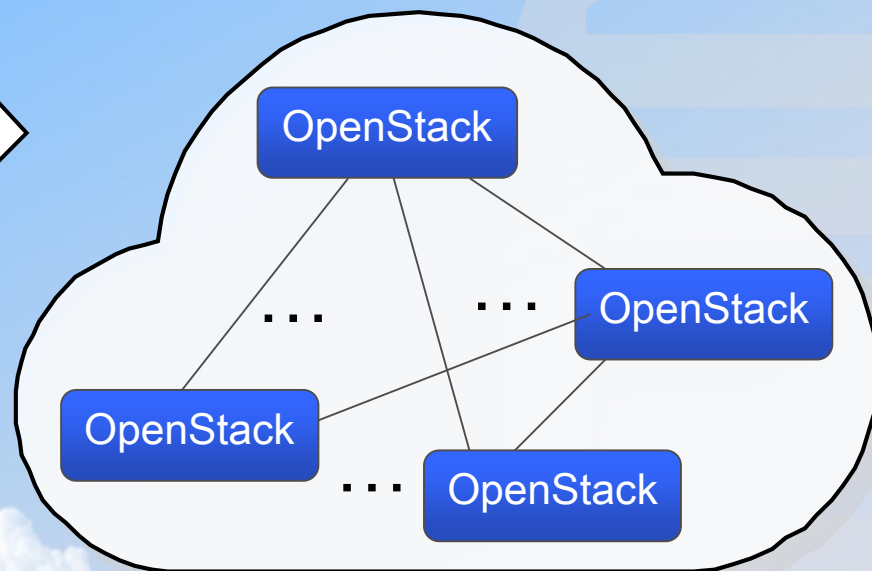
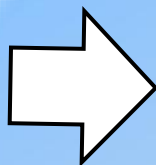


Does size matter for Openstack-like cloud orchestrator?

If does, it should be small

One cloud one Openstack?
Managing thousands of servers?
Tower of Bable Scale-up
Mission impossible

Better “One Cloud Two Openstack”
(“一云两治”) Only in distribution and
scale-out can cloud be operated with
scalability, elasticity, and most importantly,
STABILITY for operation and maintenance



Provided tenants cannot see smallness
It's the network virtualization that MATTERS!



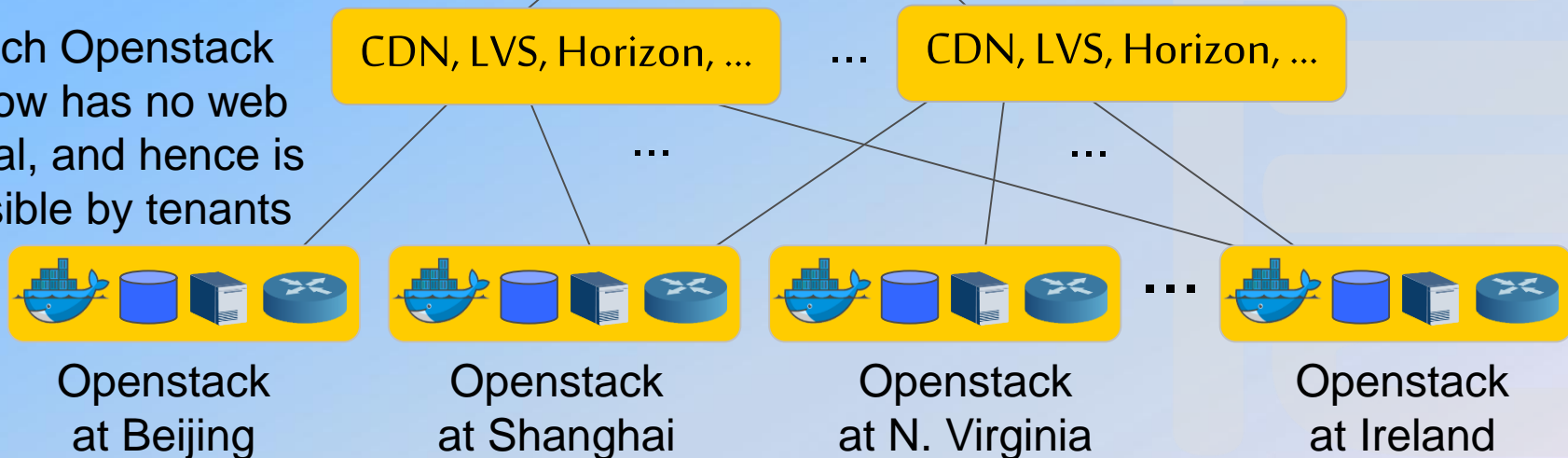
Application:
Crowdsourcing IaaS
resources from
multiple, different,
Independent, and
desirably small clouds

The screenshot shows a web portal for DAOLI. At the top is the DAOLI logo. Below it is a 'Log In' section with two input fields: 'User Name' containing 'jjalewang' and 'Password' containing seven dots. At the bottom of the login section are three buttons: 'Register', 'Forgot your password?', and 'Sign In'.

Single-Sign-On
Web Portal
+
Tenant SDN
Controller

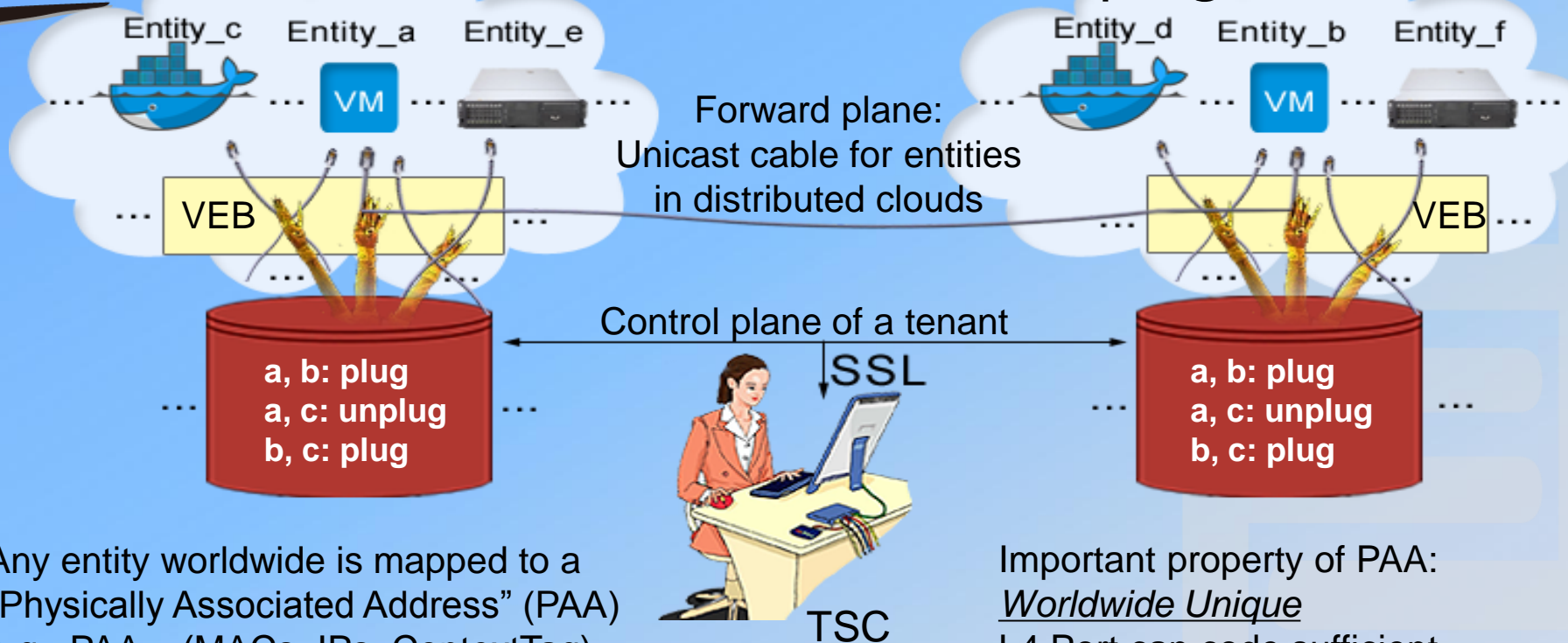
This “Openstack”
does no IaaS
orchestration

Each Openstack
below has no web
portal, and hence is
invisible by tenants



Worldwide distributed, completely independent and separate Openstack

SDN overlay know-how: Virtual Ethernet Bridge (VEB) distributed at vNICs is L2-L4 programmable



Any entity worldwide is mapped to a
“Physically Associated Address” (PAA)
e.g., PAA = (MACs, IPs, ContextTag)
L4 port: Very good usage for ContextTag

Important property of PAA:
Worldwide Unique
L4 Port can code sufficient
entropy for PAA uniqueness

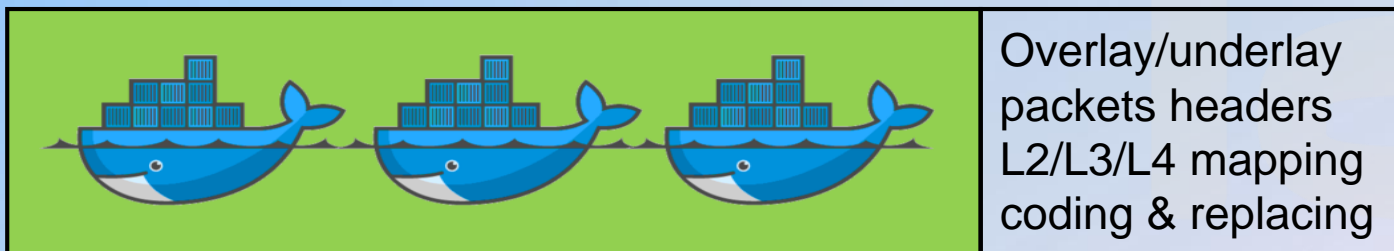
Overlay network of arbitrary topology on distributed VEB no longer need encapsulation
A unicast cable plugging entities x and $y = (PAA_x, PAA_y)$, in which:
L2 MAC addresses link x, y to their respective default gateways
L3 IP addresses = underlay IP addresses of the respective default gateways
L4 Port numbers encode unique mappings between overlay entities and PAAs



Technical material

Non-encapsulation technology to patch clouds

- Novel and useful improvement to Openflow standard
- L2/L3/L4 header metadata mapping, coding and replacing technology (Compare packet figure below with that in Slide 5)
- Random mappings are non-secret; SDN controller can help agree mappings to connect separate L2s while seeing neither intranets info
- That's how notion of Tenant SDN Controller (TSC) rises: TSC working with independent clouds connect distributed nodes within for tenant!
- Minus encap, all other virtues of Openflow are kept, e.g., efficient per flow checking routing table in VEB fastpath, instead of inefficient per frame checking underlay label (yellow header in Slide 5)
- Extremely efficient: Header metadata replacement operated in nest eliminates MAC populating in exponential reduction speed! Also no packet enlargement, no fragmentation, no broadcast via TSC, ...





Technical material

Inherent problems for cloud networking

The following cloud networking problems are already bad enough for the scale of hypervisor-based CPU virtualization; the explosive scale of container-based CPUs will only worsen the matter

MAC address explosion One rack of servers in current CPU condensity can host 10s of thousands containers. In conventional flood-&-learn MAC populating, a ToR switch must hold multiple such numbers of MACs since a cloud should be larger than one rack. Moreover, can so MAC populated ToR work efficiently, and in an affordable cost?

L2 broadcast control ARP broadcast is the only practical way to plug-&-play construct a physical L2. However broadcast has prohibitively high cost; to build a very large physical L2 is certainly to look for trouble. In the next slide we shall discuss how current technologies for L2 broadcast control, and their irrelevance to large scale cloud networking.



Technical material

Cloud networking current technologies analysis

Encapsulation protocols in Slide 5 can L3 tunnel connect separate L2s

Key issue They are peer connection protocols: SDN controller must see both L2s intranets to orchestrate connection. That's why they're aka "large L2" protocols. Enlarging intranets hopelessly kills scalability for cloud services. Also killed enroute is cloud service interoperability.

Technical assessment

1. To avoid MAC explosion and control L2 broadcast, encap for servers/hosts; to isolate tenants, encap for each tenant; to patch cloud for truly large scalability, encap further for IDCs; in general, to connect n instances, $O(n^2)$ encapsulations are needed.
2. IP connectivity is carefully architected to be connectionless flows so that forward plane only conducts per flow checking for routing, this very important architecting is nullified by encap into per packet checking labelling (yellow header in Slide 5), that's why encap is inefficient.
3. Encap enlarges packet over MTU (Maximum Transmission Unit), and hence fragmentation/reassemble, additional cost.



Long term value for inter-cloud patching

The cloud is unbound large, meaning its service logic scalability. However the physical implementation of the cloud should follow distributed computing principle, and open standards for interoperability.

DaoliCloud believes: If cloud services remain in the current practice of each provider encapsulating its own connectivity without interoperability, then provider lock-in would be the inevitable norm, which is obviously not good for users. Non-scalable cloud is not good for provider either.

Non-encapsulation based cloud networking enabled inter-cloud connectivity and interoperability hence provide very important value to all.





Conclusion

Non-encapsulation based cloud networking technologies (NVI & TSC) provide a practical solution to cloud network virtualization, eliminate the physical network boundaries between physically implemented small and hence stably running clouds, and make cloud network with unlimited scalability and elasticity.

Non-encapsulation based cloud networking technologies enable lightweight connectivity for containers, VMs, and physical servers.

Non-encapsulation based cloud networking technologies form a hopeful solution to inter-cloud interoperability.





Sign-up for free trial account now at
nvi.daolicloud.com