Redundancy Doesn't Always Mean "HA" or "Cluster"

A cautionary tale against using hammers to solve all redundancy and resiliency problems ...

OpenStack Design Summit – Oct 2012

Randy Bias
@randybias
CTO, Cloudscaling

Dan Sneddon
@dxs
Sr. Engineer, Cloudscaling

cloudscaling

Our Journey Today

1. “HA” pairs are not the only type of redundancy

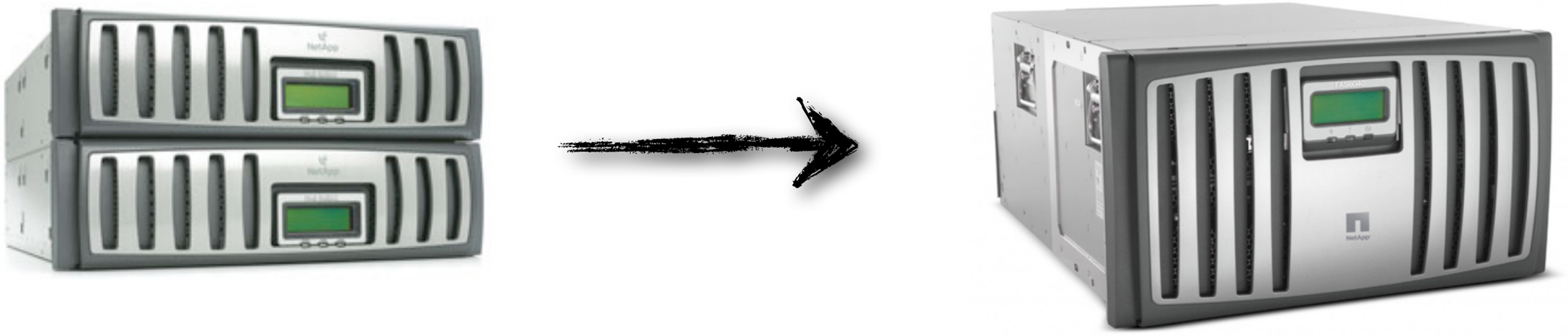
2. Alternative redundancy patterns for HA

3. Redundancy patterns in Open Cloud System*

** Cloudscaling’s OpenStack-powered cloud operating system (“distribution”)*

What Do We Mean By "HA"?

We mean what most people mean ...



Two servers or network devices that look like one

“HA HA”?

HA pairs come in a couple flavors

Active / Passive

“HA HA”?

People like this flavor best, but it’s not always possible...

Active / Active

“HA HA HA HA HA”??

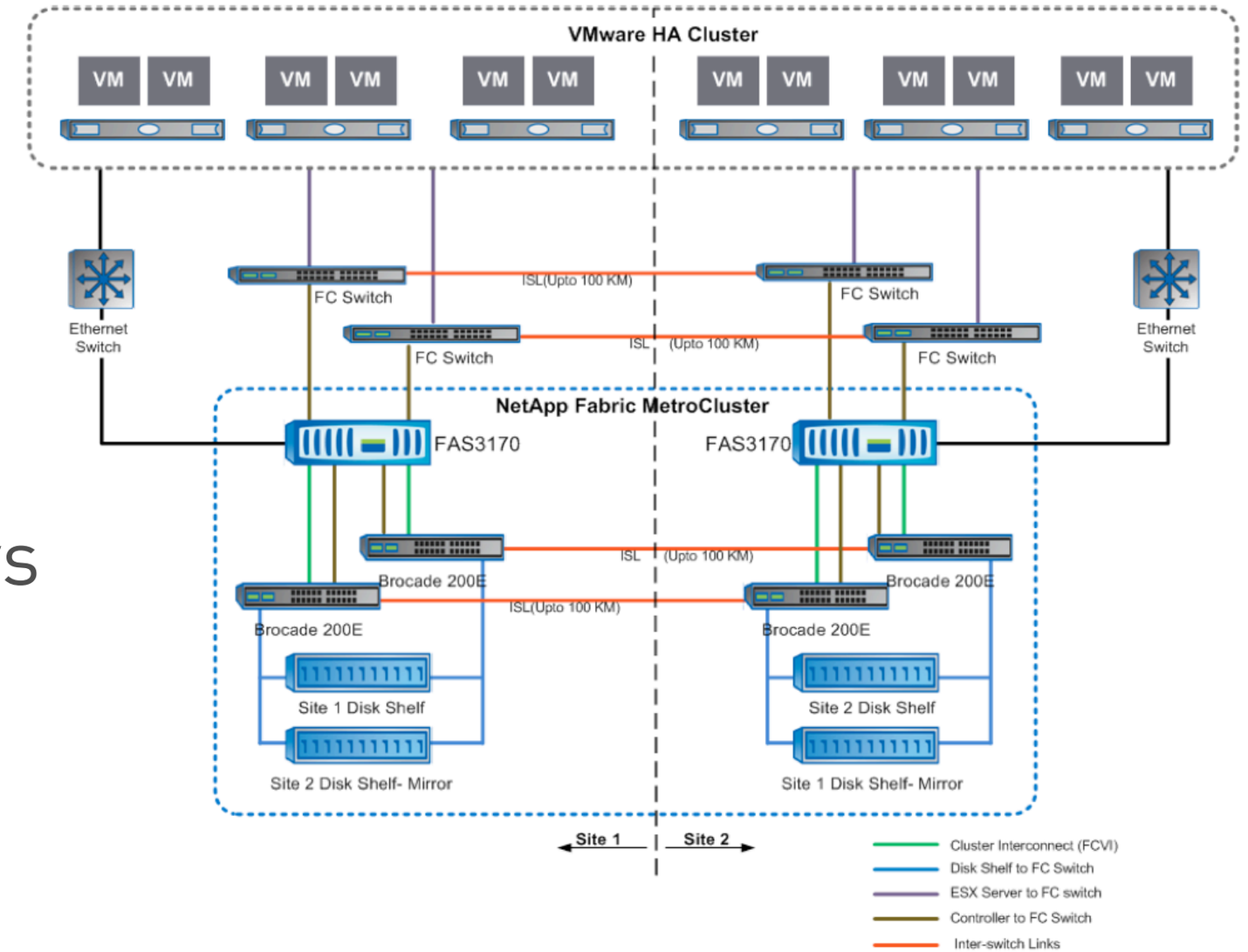
Many people wish they could get it more like this ...

HA cluster aka ‘massive operational nightmare’

Cluster<bleep>!

Imagine this was **4** or **6** nodes in the cluster

- 4 network tech.
- 7 NICs / node
- A million different ways to break



cloudscaling

"HA" Pairs Are One Type of Redundancy

Herein lies the problem ...

The Problem With "HA"-mmers

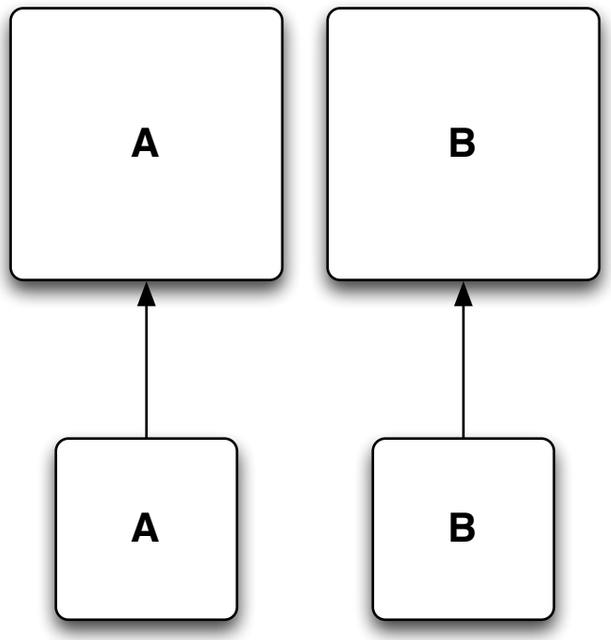
There are many, but these two matter most ...

- Catastrophic failures
- No scale out

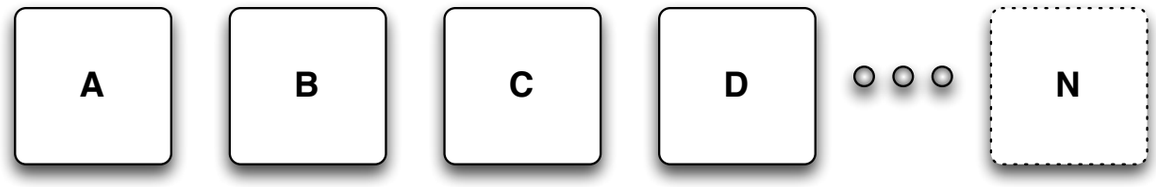
HA Pairs Have Binary Failures

Either working or dead, nothing in-between

What is Scale-out?



Scale-up - Make boxes bigger (usually an HA pair)



Scale-out - Make moar boxes

Scaling out is a mindset

Scaling up is like treating your servers as pets

bowzer.company.com

web001.company.com

Servers *are* cattle

HA Pair Failures* - 100% down

Hardware rarely fails, operators fail, software fails

Who	Type	Year	Why	Duration
Apple	Switch	2005	Bug	2 hrs
Flexiscale	SAN	2007	Ops Err	24 hrs
Vendio	NAS	2008	Ops Err	8 hrs
UOL Brazil	SAN	2011	Bug	72 hrs
Twitter	Datacenter	2012	Bug+Ops	2 hrs

* This is a handful of examples as a baseline; I'm sure you can find many more

"HA" Pairs Are an All-in Move

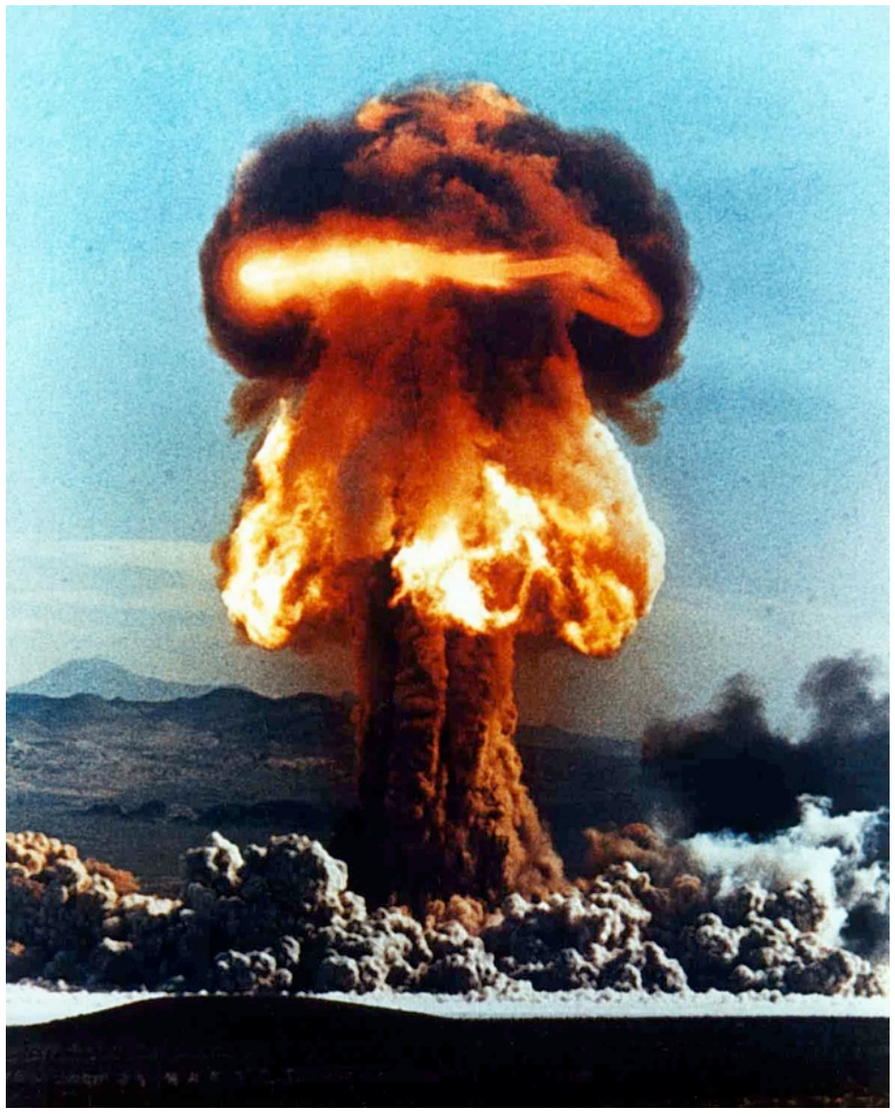
They better not fail ...

Risk Reduction

Many small failure domains is usually better

Big failure domains vs. small

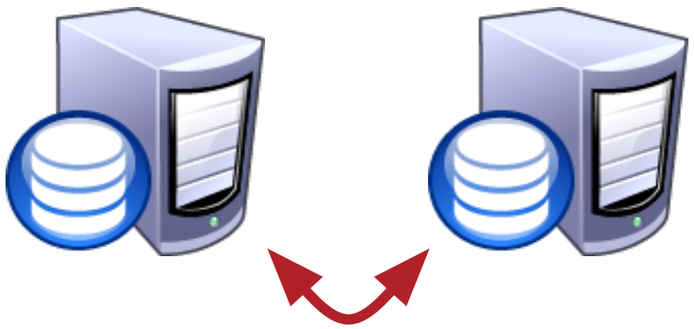
Would you rather have the whole cloud down or just a small bit for a short period of time?



Still a scale-up pattern ...
wouldn't you rather scale-out?

Pair vs. Scale-out Load Balancing

No scale-out



State Sync

(100% loss)

Shared-nothing Architecture

(20% loss)

Pair vs. Scale-out Load Balancing

No scale-out

State Sync

(100% loss)

Shared-nothing Architecture

(20% loss)

What's Usually an "HA" Pair in OpenStack?

Everything ...

Service Endpoints (APIs)

Messaging System (RPC)

Worker Threads (e.g. Scheduler, Networking)

Database (MySQL)

What needs to be an HA pair?

Not much needs state synchronization

Service Endpoints (APIs)

Messaging System (RPC)

Worker Threads (e.g. Scheduler, Networking)

Database (MySQL)

Fault Tolerance Methodologies

Fault Tolerance in OCS

Service Distribution

High Availability Without Compromise

Resilient

Stateless

Scale-out

Service Distribution

Combines Standard Networking Technologies

OSPF	/etc/quagga/ospfd.conf	**router ospf** **ospf router-id 10.1.1.1** **network 10.1.255.1 area 0.0.0.0**
Anycast	/etc/quagga/zebra.conf	**interface lo:2** **description Pound listening address** **ip address 10.1.255.1/32**
Load-Balancing Proxy	/etc/pound/pound.conf	**ListenHTTP** **Address 10.1.255.1** **Port 8774** **xHTTP 1** **Service** **BackEnd** **Address 10.1.1.1** **Port 8774** **End** **BackEnd** **Address 10.1.1.2** **Port 8774** **End** **End** **End**

Resilient OpenStack

Horizontally Scalable, No Single Point Of Failure

Service Distribution

Service Endpoints (APIs)

ZeroMQ

Messaging System (RPC)

Service Distribution

Worker Threads (e.g. Scheduler, Networking)

MMR + HA

Database (MySQL)

Service Distribution Advantages

What Makes This a Superior Solution?

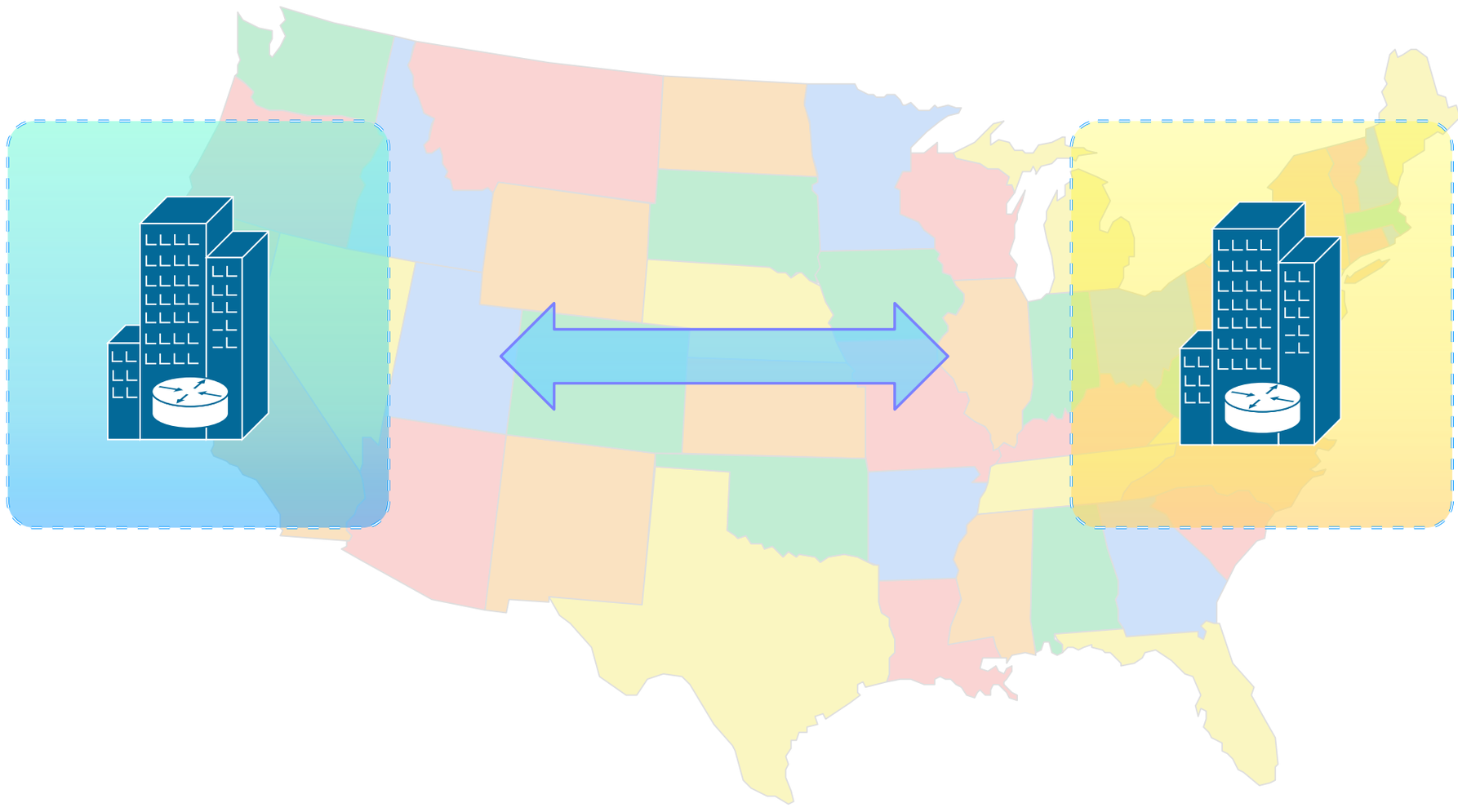
- True horizontal scalability with no centralized controller
- Services are always running, failover is nearly instant
- Reduced complexity, fewer idle resources
- No need for separate load balancers

Perfect For Site Resiliency

Service Distribution Works With Multiple Sites

- Traditional HA pairs do not support cross-site resiliency

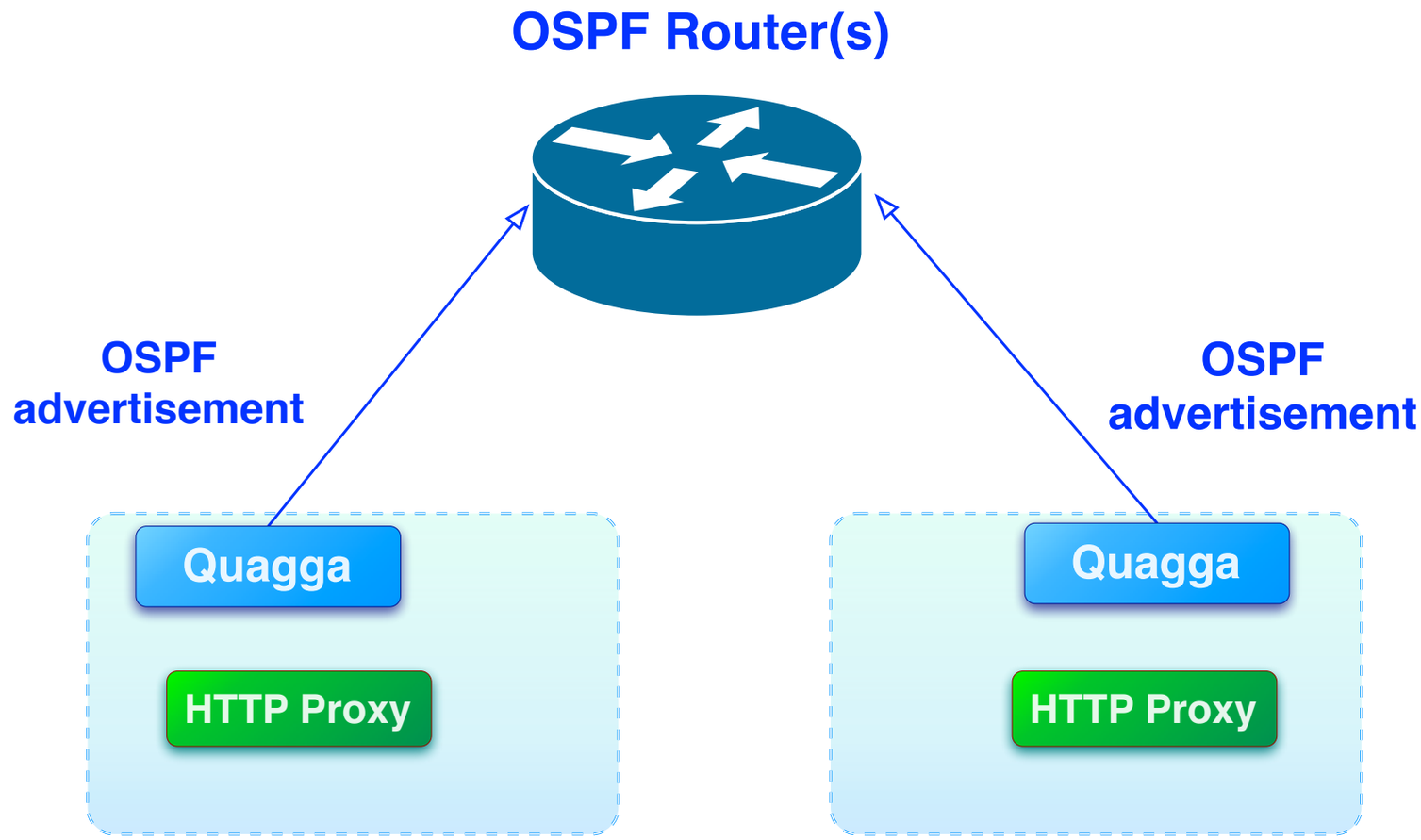
- Service Distribution fail across sites without DNS redirections



Service Distribution in Action

Example: Distributed Load Balancing

1) **OSPF**



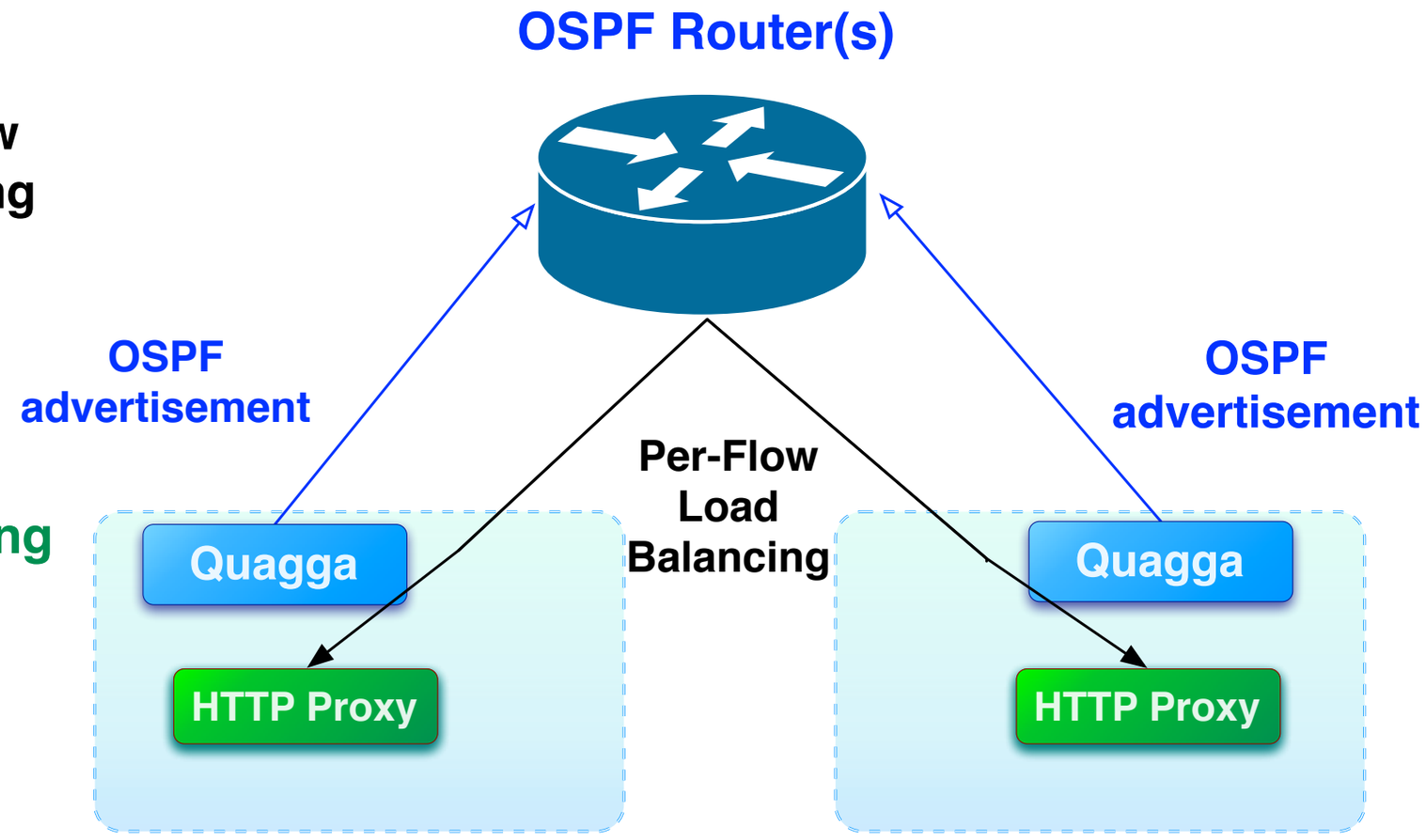
Service Distribution in Action

Example: Distributed Load Balancing

1) **OSPF**

2) **ECMP Per-flow Load Balancing**

3) **Load-balancing HTTP Proxy**



Service Distribution in Action

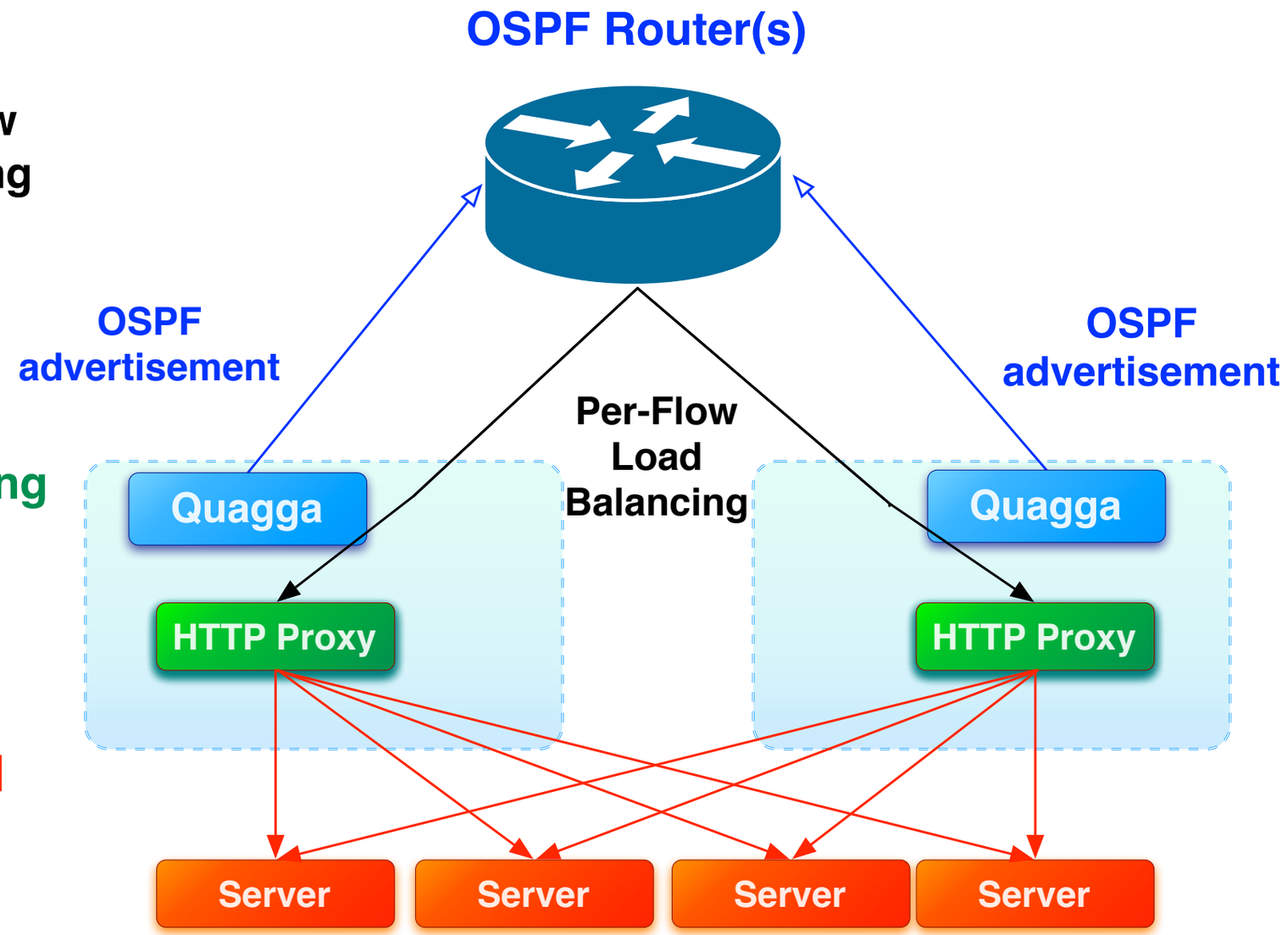
Example: Distributed Load Balancing

1) **OSPF**

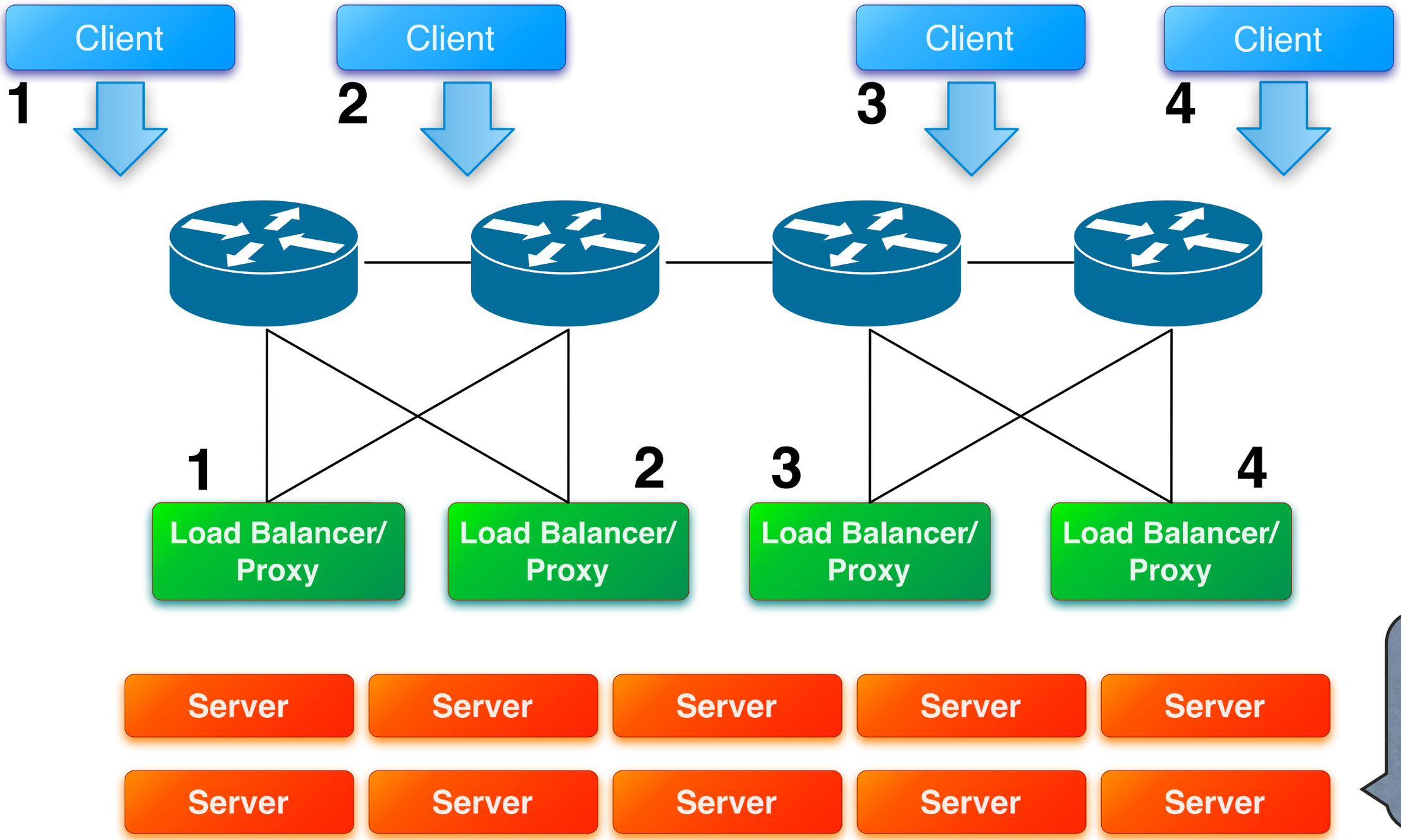
2) **ECMP Per-flow Load Balancing**

3) **Load-balancing HTTP Proxy**

4) **Unlimited # of Back-End Servers**

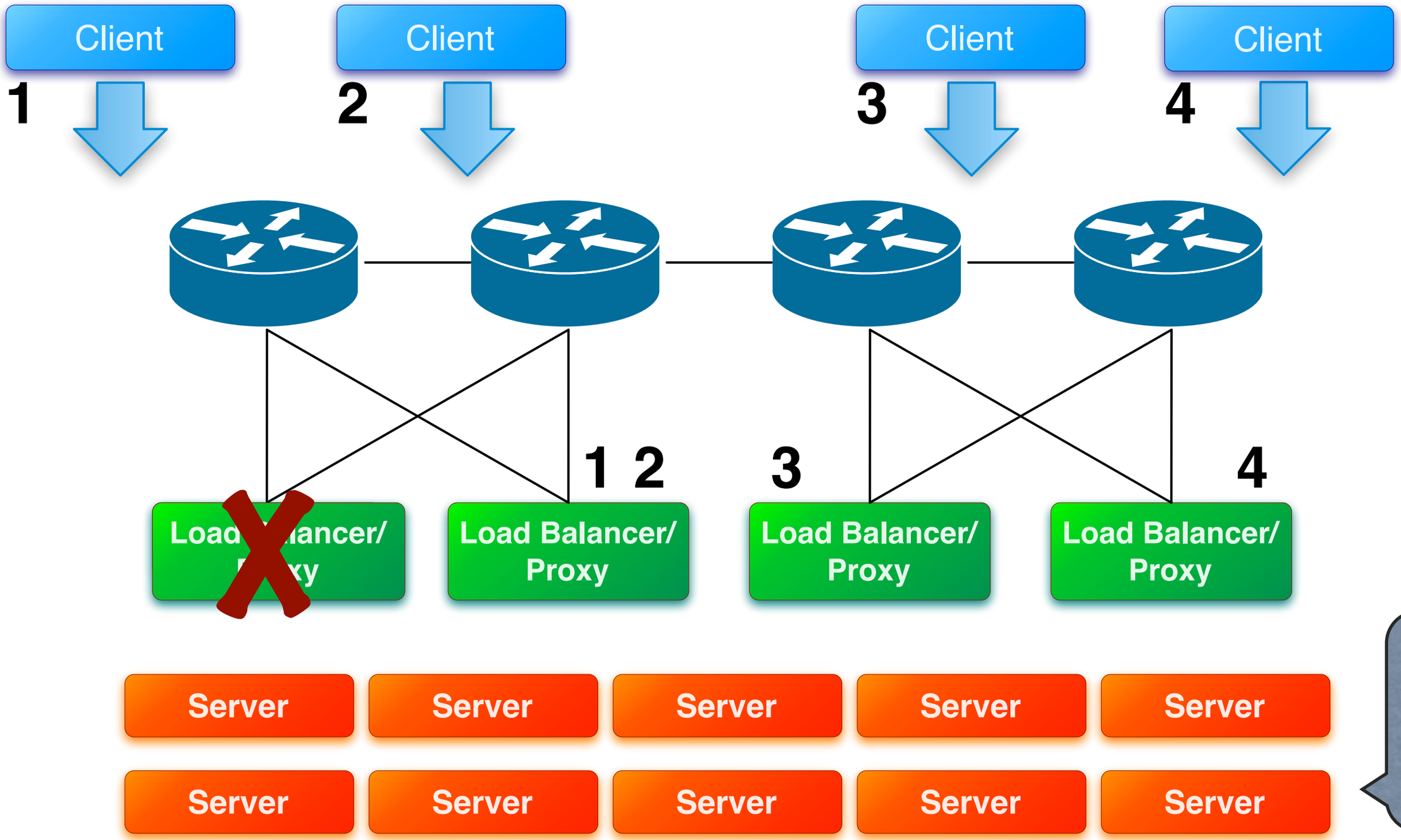


Failure Resiliency



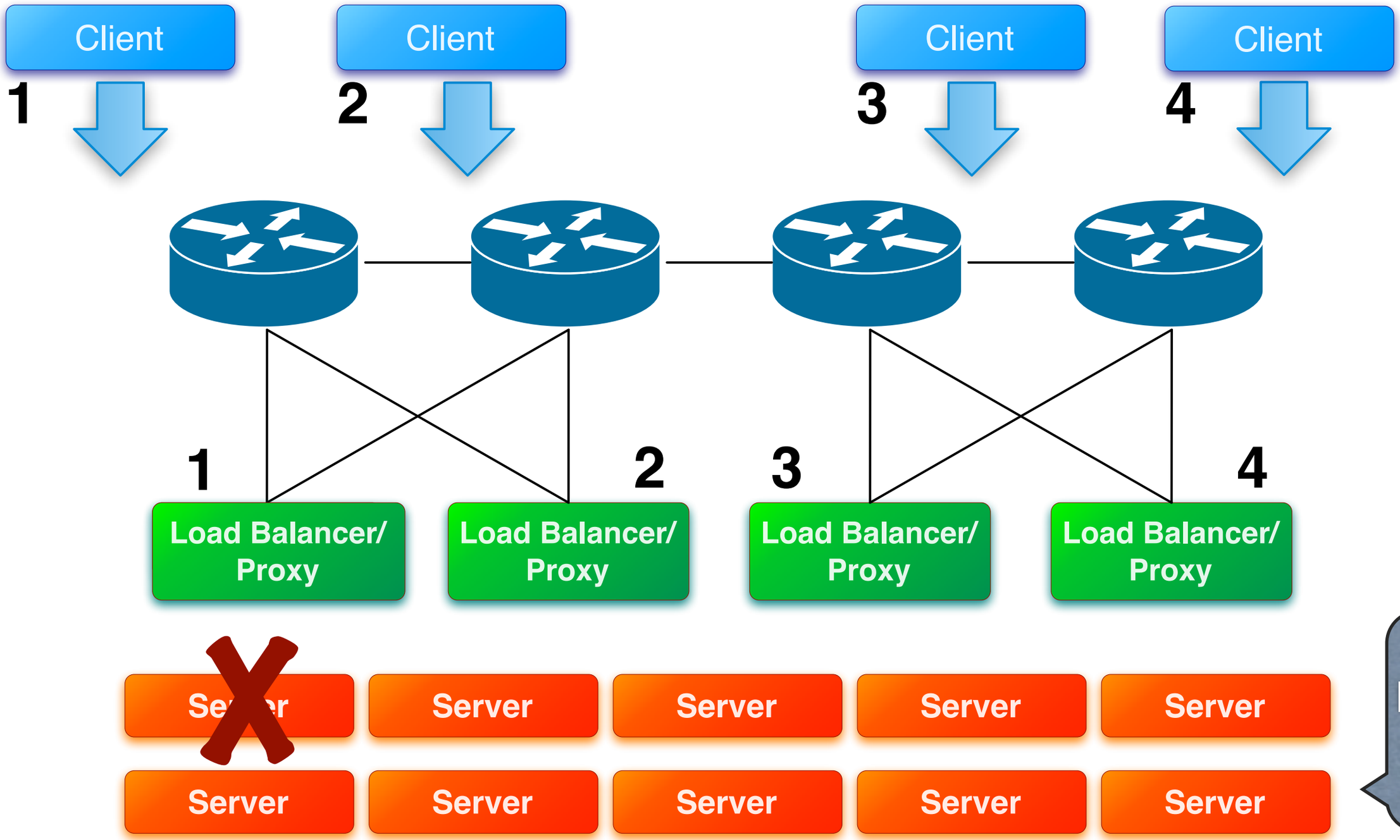
cloudscaling

Failure Resiliency



cloudscaling

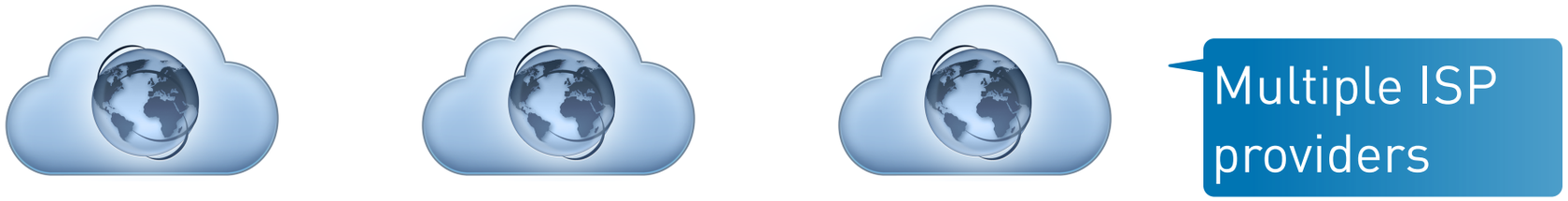
Failure Resiliency



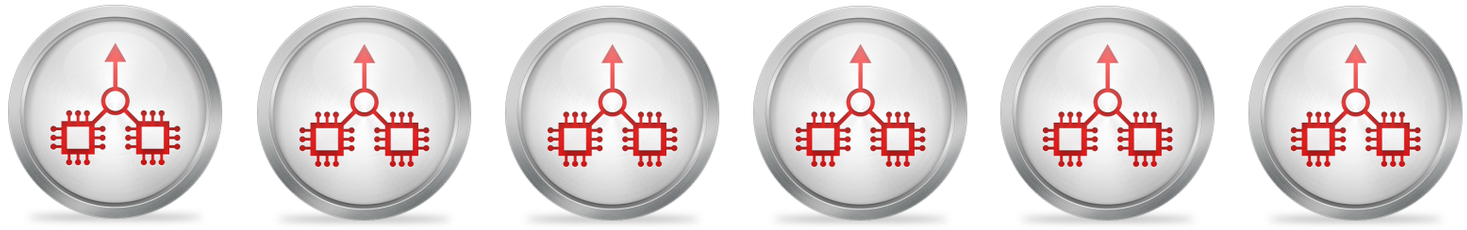
OCS NAT Service

Example: Scale-out Network Address Translation

BGP

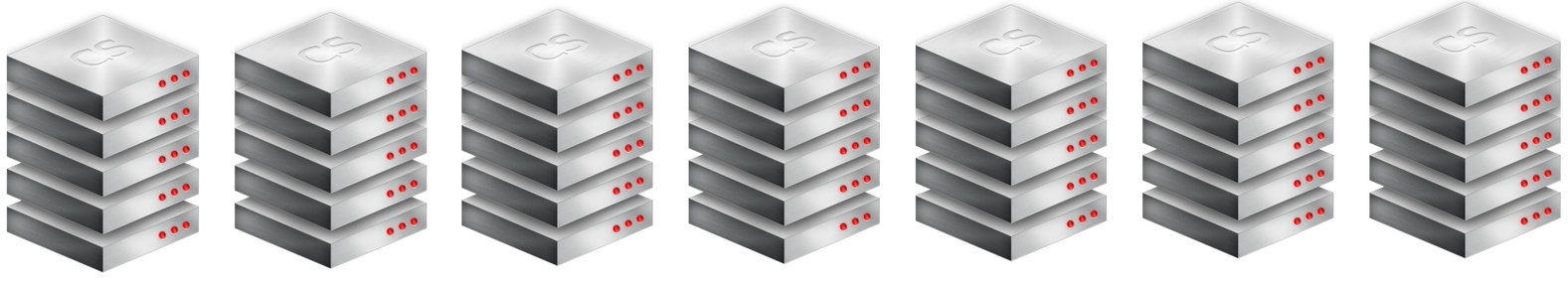


NAT



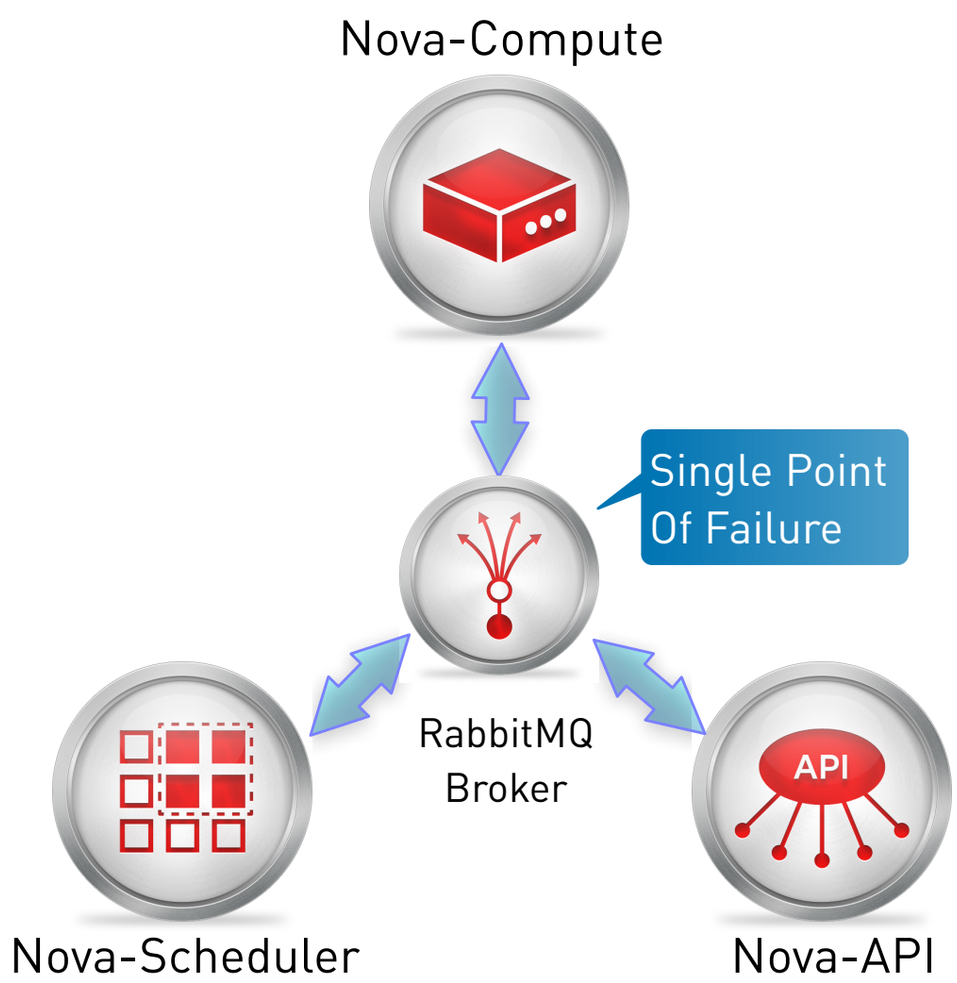
Service Distribution

VMs



Brokerless Messaging With ZeroMQ

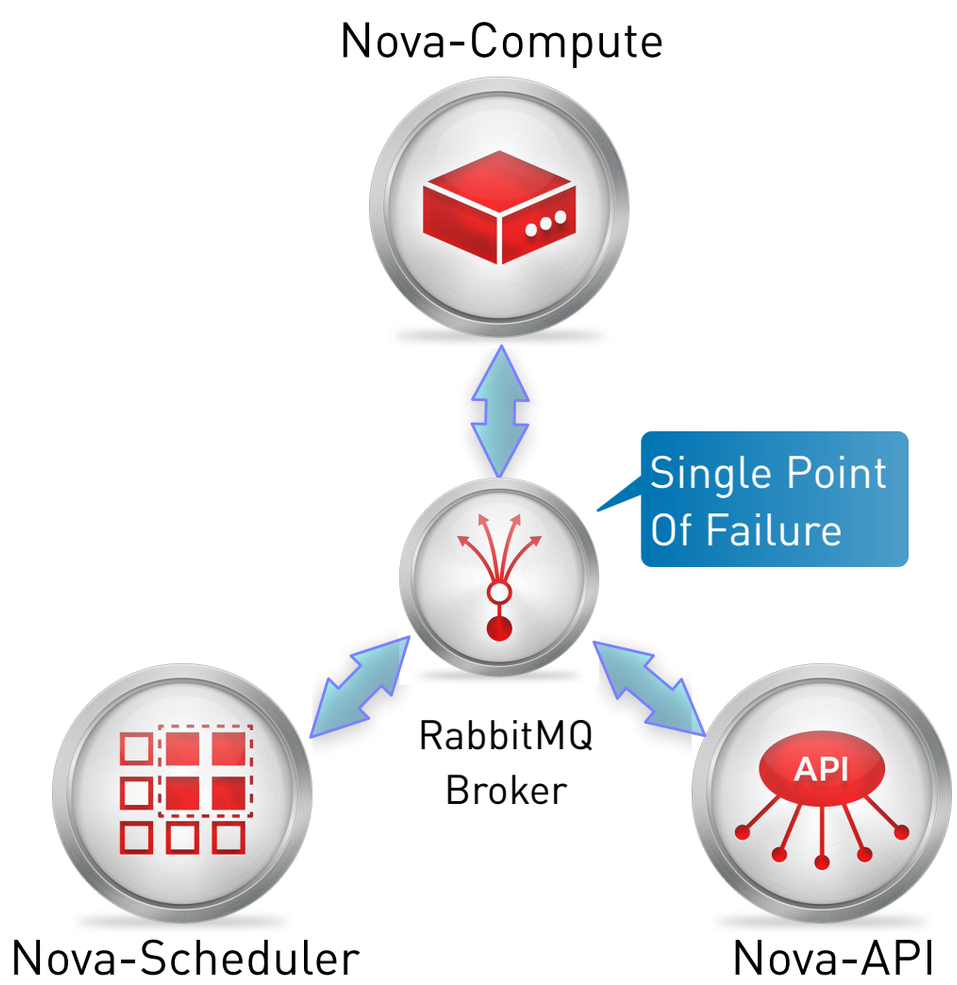
Avoiding RabbitMQ's Single Point Of Failure



RabbitMQ
(Brokered)

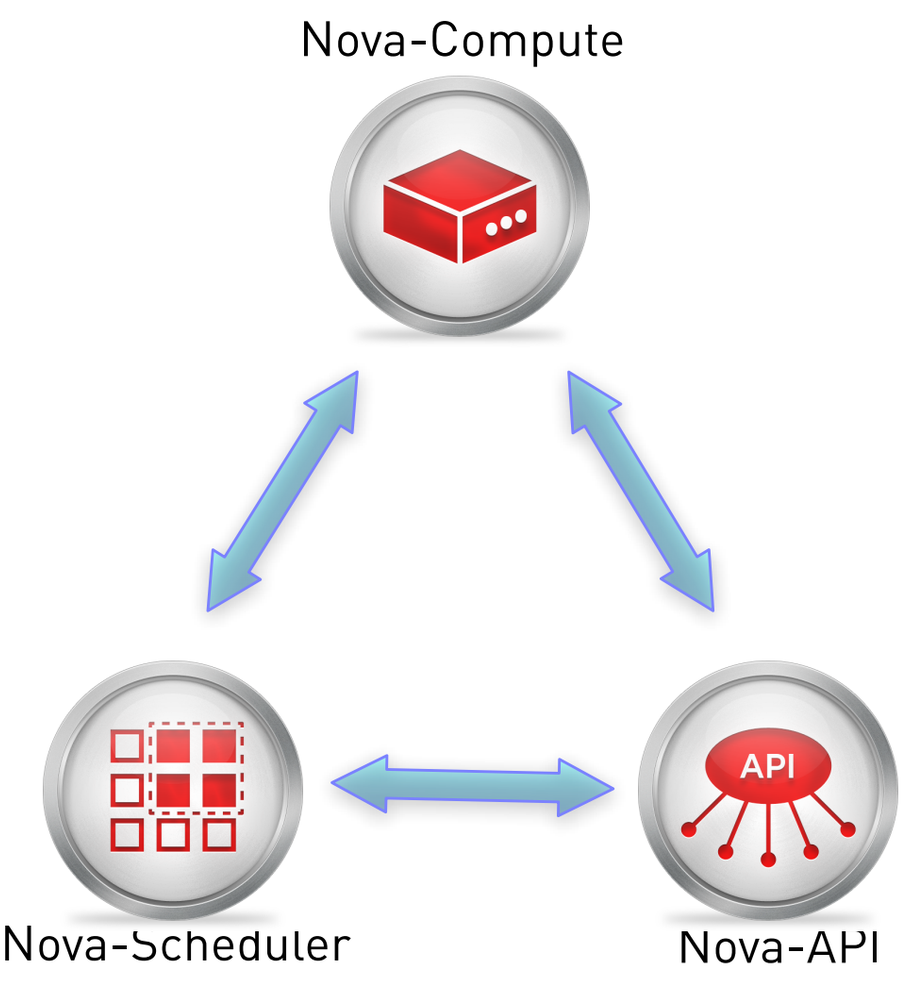
Brokerless Messaging With ZeroMQ

Avoiding RabbitMQ's Single Point Of Failure



RabbitMQ
(Brokered)

vs.



ZeroMQ
(Peer To Peer)

What did we learn today?

1. HA-mmers are for nails

2. Scale-out rules for redundancy

3. Design-for-failure is a mentality, not a pair

4. Resiliency over redundancy

Q&A

Randy Bias
@randybias
CTO, Cloudscaling

Dan Sneddon
@dxs
Sr. Engineer, Cloudscaling

OCS 2.0

Public Cloud Benefits | Private Cloud Control | Open Cloud Economics