Distributed Health Checking for Compute Node High Availability

ZhengSheng Zhou@AWcloud
Alex Xu@Intel, JunWei Liu@ChinaMobile

Agenda

Compute Node High Availability

Distributed Health Checking

Nova Considerations

Ceilometer Considerations

Demo

Compute Node High Availability

Compute Node High Availability

Compute Nodes

- Some of our enterprise customers run pet workloads
- More than Hundreds of nodes per region
- Need scalable HA Solution

Compute Node Failure

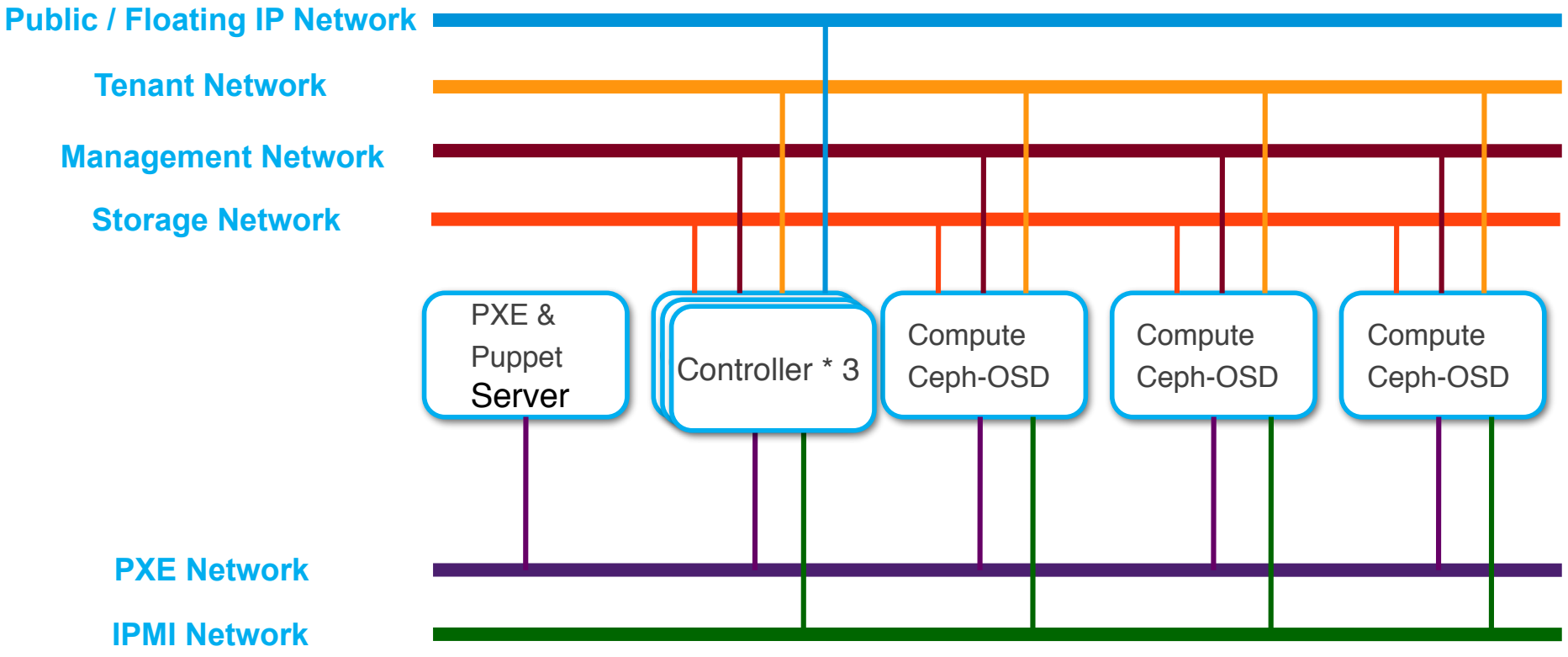
- Network Cable Broken / Loose
- Host / Switch Power Failure
  - Can not Connect to Host IPMI
- Kernel Stall / Panic
- Host Disk Failure
- Fan Broken, Overheat
- Migrate / Evacuate the Host

Compute Node HA@Home

```
#!/usr/bin/sh
while true; do
  nova service-list --binary nova-compute | \
  grep down | \
  awk '{print $4}' | \
  while read Host; do
   echo shutdown ${Host}
   echo evacuate ${Host}
  done
  sleep 10
done
```

~~Done, let's go and take a beer.~~

Deployment Topology



Compute Node HA v1

Collect Health Check Result

Consult Action Matrix

Send Email, IPMI Power off, Evacuate

Problems

- Who Supervises the Supervisor?
  - Monitoring Host or Service Failure
- What if It Loses IPMI Connection?
  - IPMI Network Failure
  - Compute Node Power Failure
- Scalability
  - 1 * Monitoring Service Instance
  - N * Hundreds of Compute Nodes

An Example Action Matrix

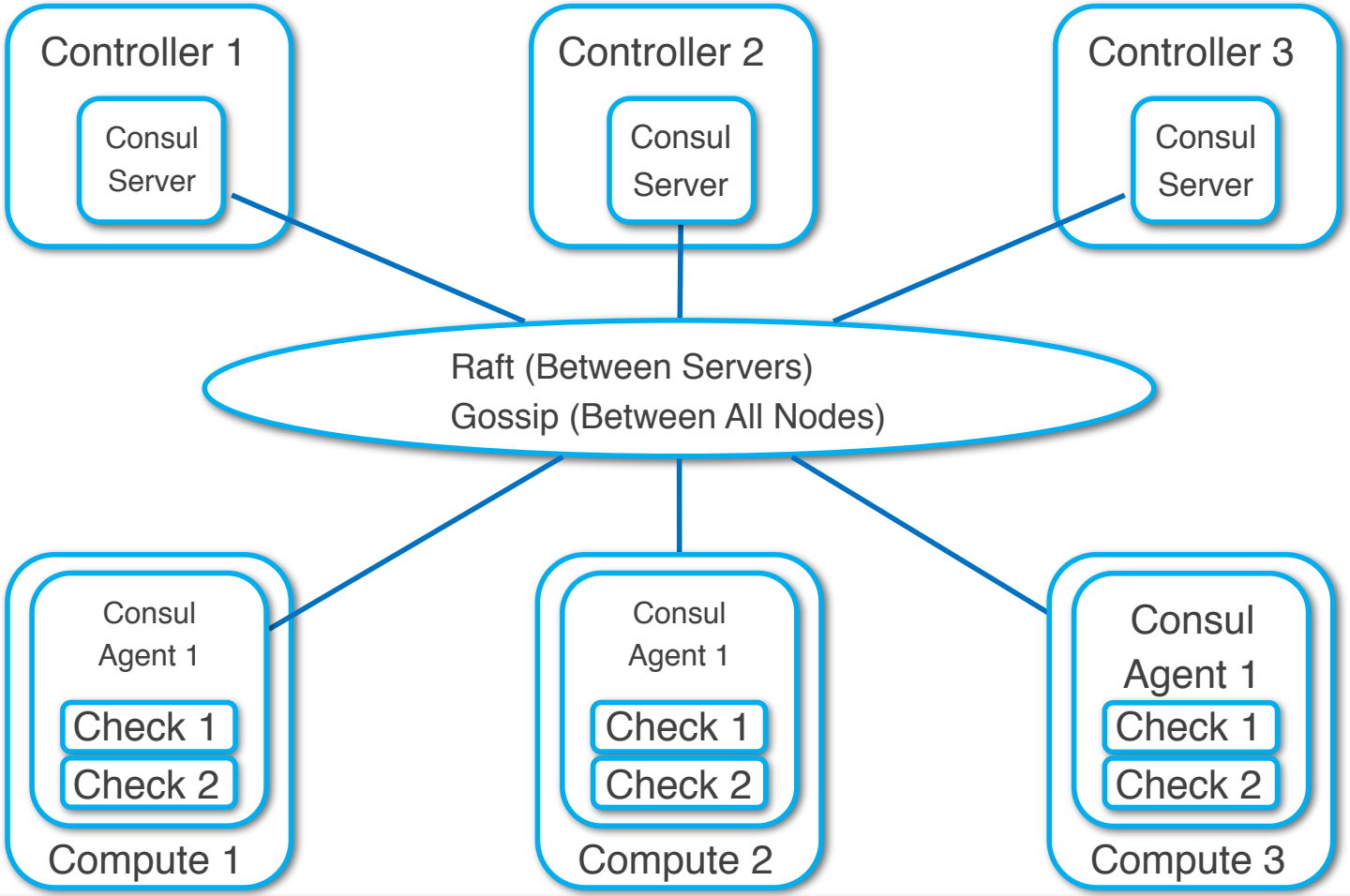
Mgmt Network	Storage Network	Tenant Network	Other Checks	Action
Down	Up	Up	...	Email
Up	Down	Up	...	Fence, Evacuate
Up	Up	Down	...	Migrate
Down	Up	Down	...	?
Down	Down	Down	...	IPMI? Fence, Evacuate

Distributed Health Checking

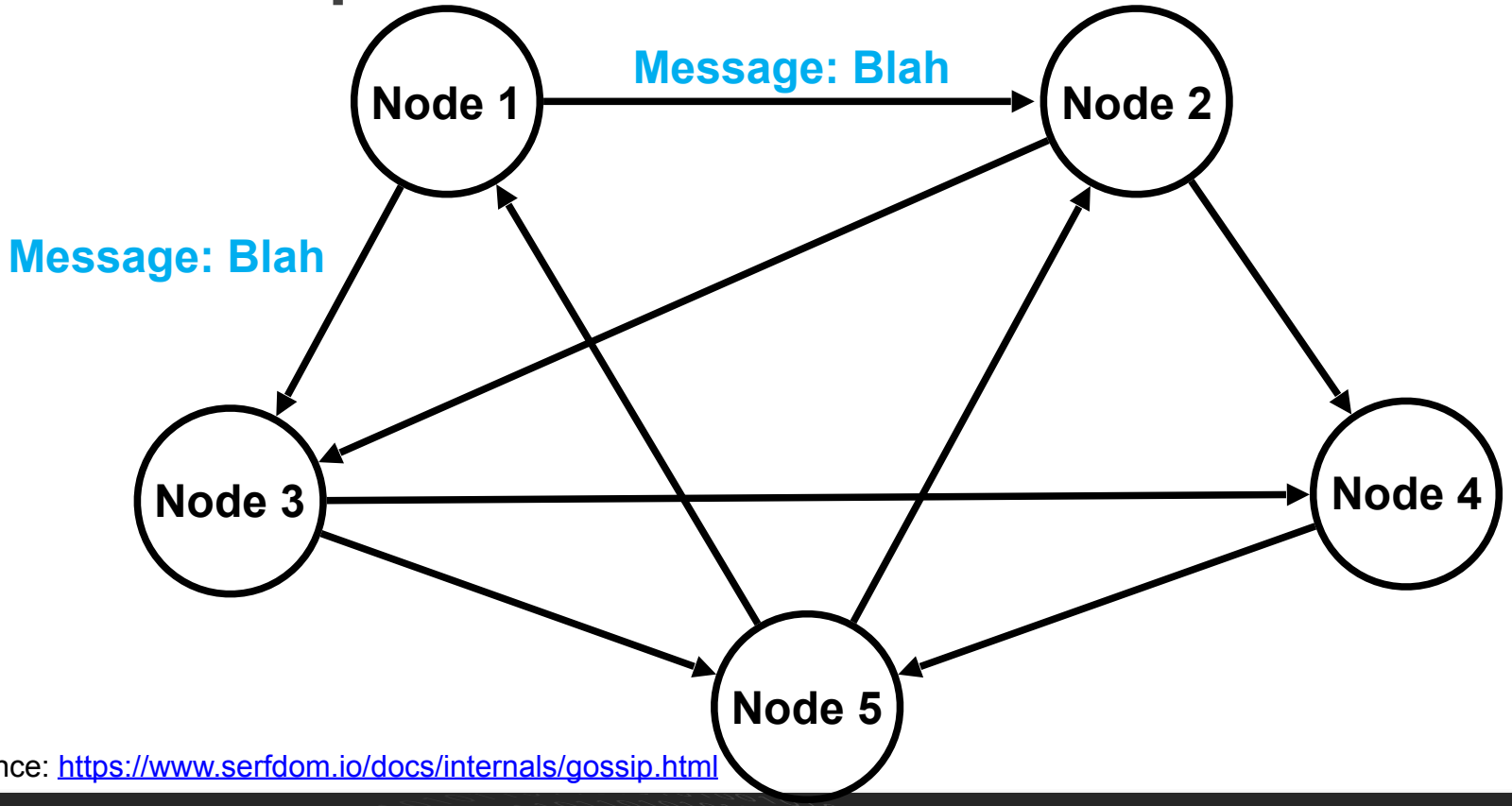
Distributed Health Checking

Consul

- Service Discovery Tool
  - Service Registry
  - DNS Interface
  - Configuration Template
- Distributed K/V Store
- Node Health Check
  - Script, HTTP, TTL, Edge Triggered
- Large Number of Nodes
  - Event Broadcast, Filtering and Watch
  - Gossip Implementation from Serf
- Session and Lock
  - Leader Election
- REST API

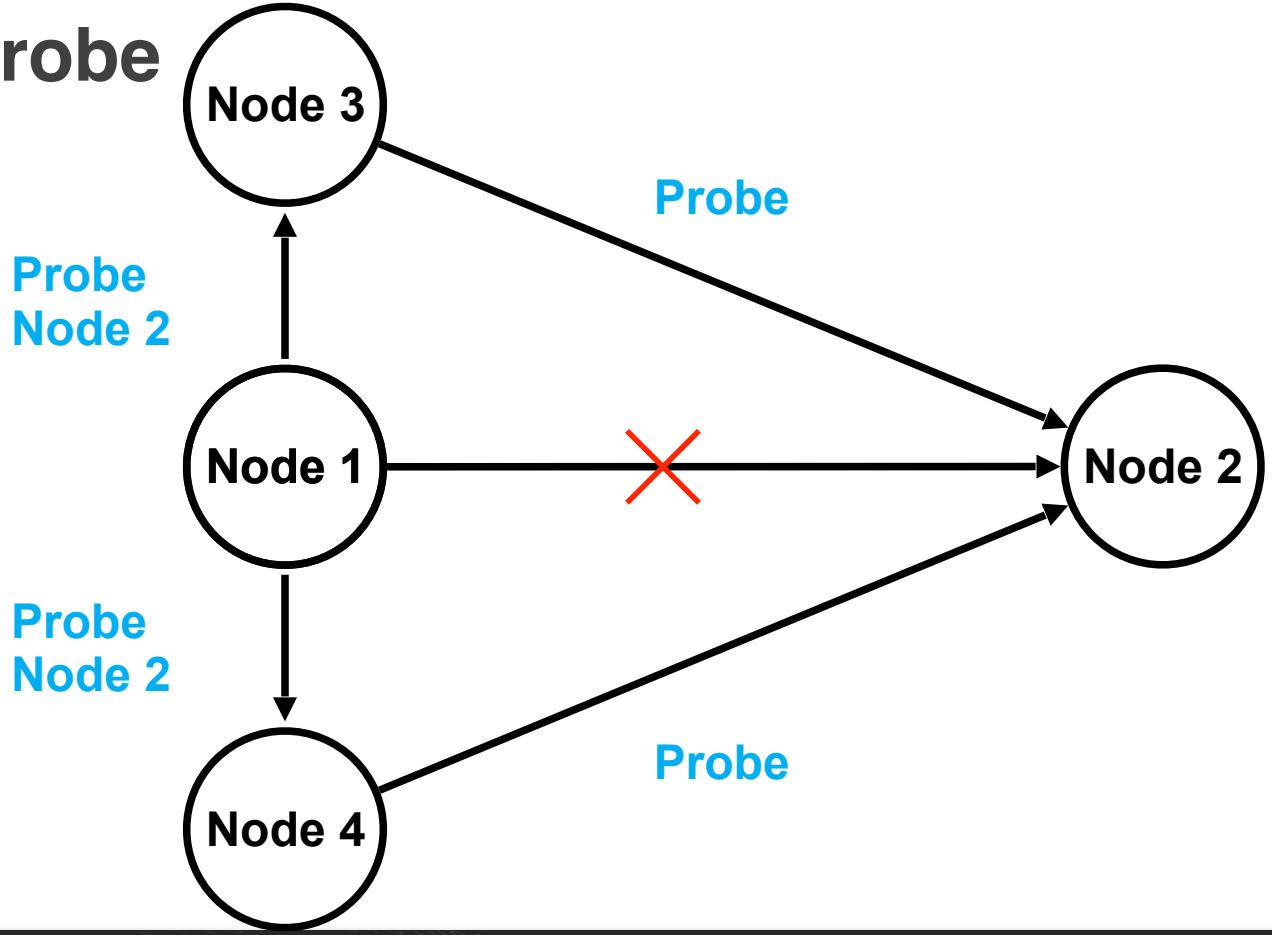

Controller 1
Consul Server
Controller 2
Consul Server
Controller 3
Consul Server
Raft (Between Servers)
Gossip (Between All Nodes)
Consul Agent 1
Check 1
Check 2
Compute 1
Consul Agent 1
Check 1
Check 2
Compute 2
Consul Agent 1
Check 1
Check 2
Compute 3

The Gossip Protocol



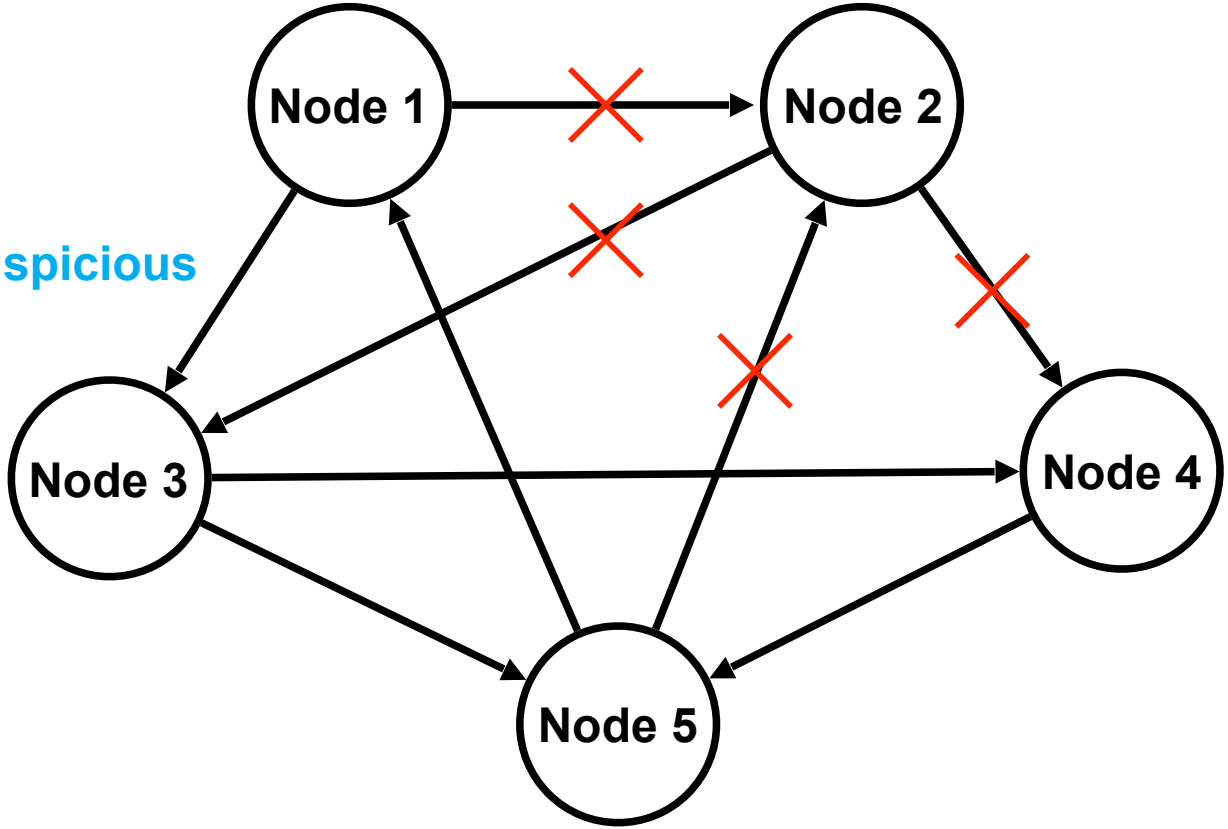
Reference: https://www.serfdom.io/docs/internals/gossip.html

Indirect Probe



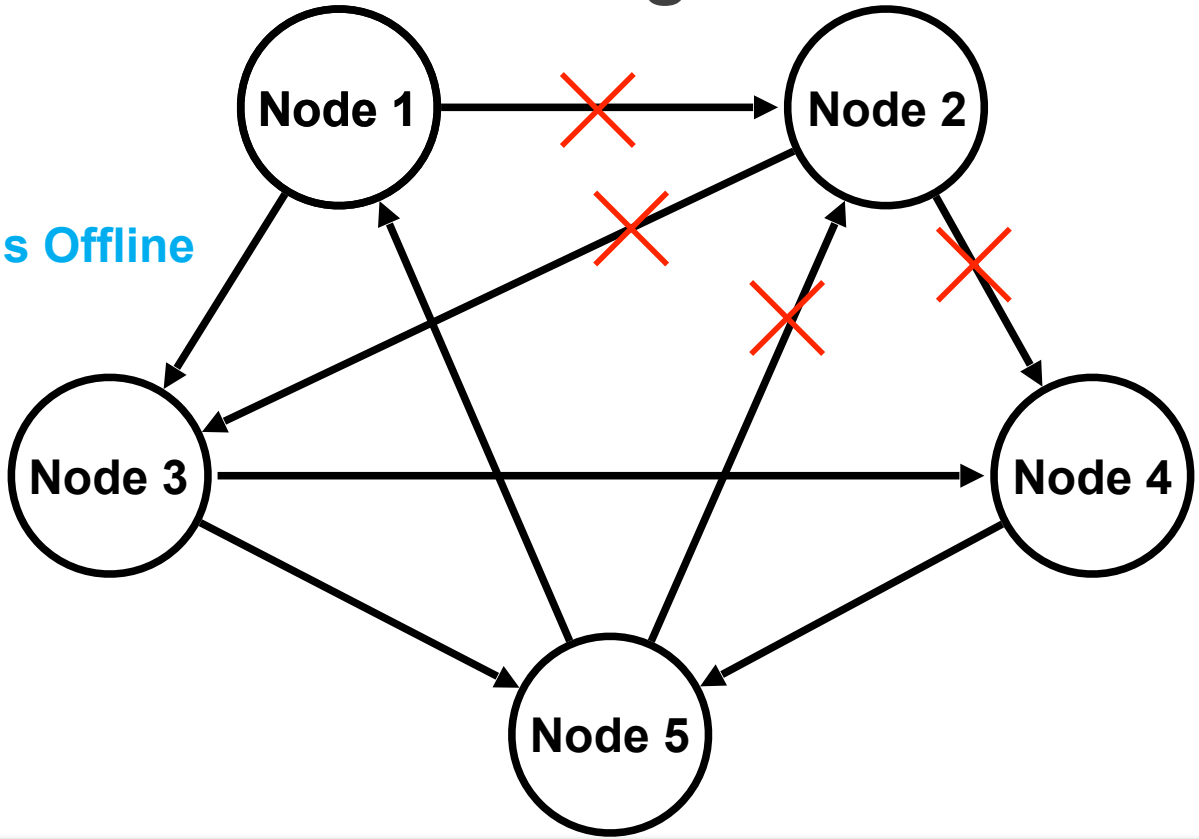
Suspicious Node

Node 2 is Suspicious



Gossip the Failure Message

Node 2 is Offline



Compute Node HA v2

Health Check

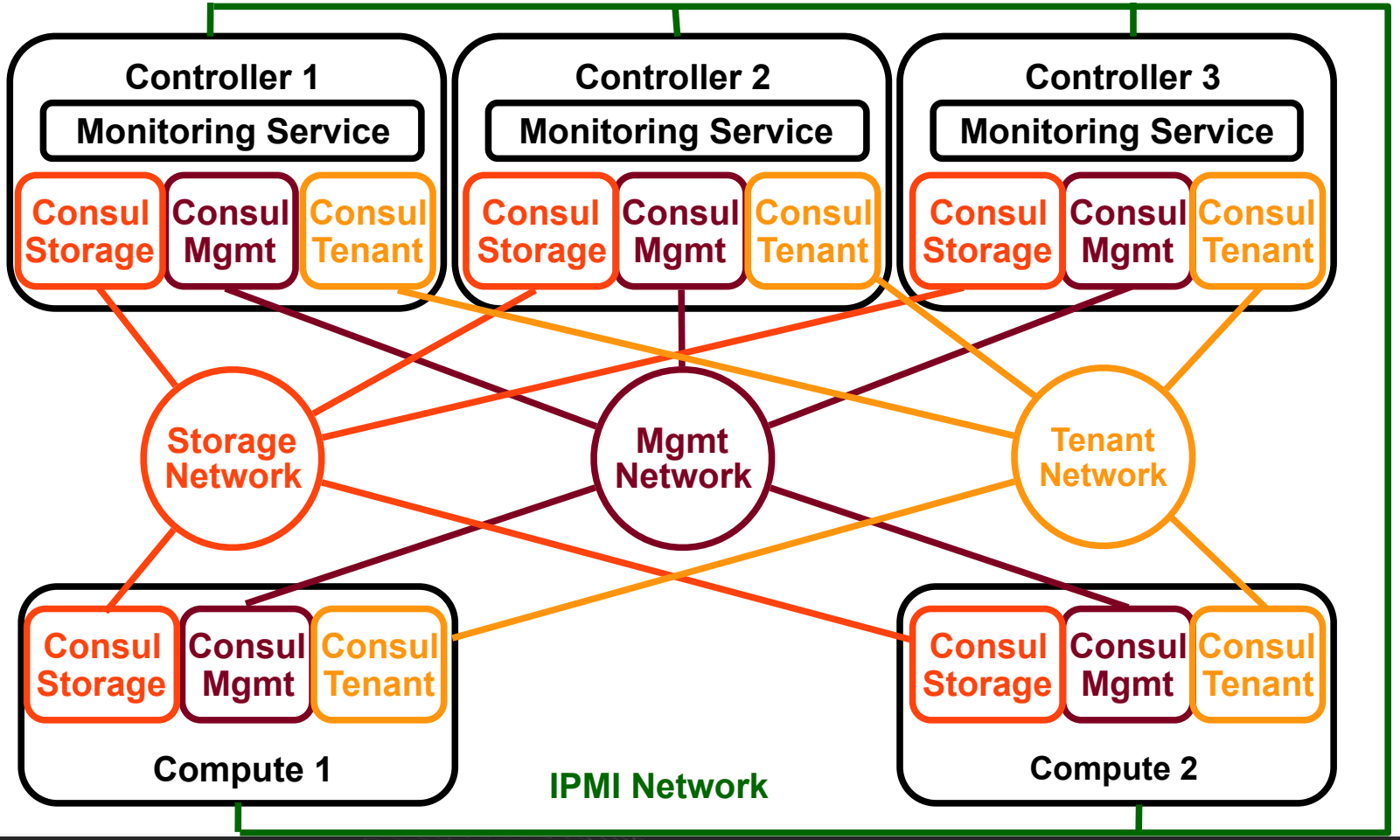
- Network Connectivity Status Provided by Consul Member Information
- Compute Nodes can Register Health Checks in Consul

Monitoring Service HA

- Leader Election and Release
- Consul Session and Lock

Fence

- IPMI Power off
- Consul Event Broadcast


Controller 1
Monitoring Service
Consul Storage
Consul Mgmt
Consul Tenant
Controller 2
Monitoring Service
Consul Storage
Consul Mgmt
Consul Tenant
Controller 3
Monitoring Service
Consul Storage
Consul Mgmt
Consul Tenant
Storage Network
Mgmt Network
Tenant Network
Consul Storage
Consul Mgmt
Consul Tenant
Compute 1
IPMI Network
Consul Storage
Consul Mgmt
Consul Tenant
Compute 2

Monitoring Service HA

Run Multiple Monitoring Service

- Service Instance with Leader Role Does the Work

Leader Lock and Release

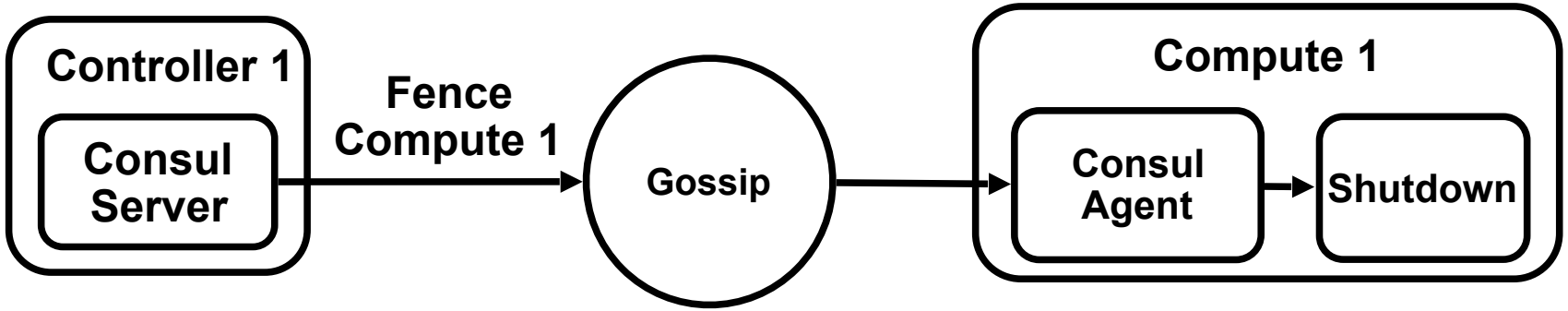
- Consul Session and Lock
  - Leader Checks Consul Member Info in Management, Storage and Tunnel Network
  - If it Loses Other Controllers in Mgmt/Storage/Tunnel, Release the Lock
  - Lock Timeout when Host/Service Failure

Fenced Node List

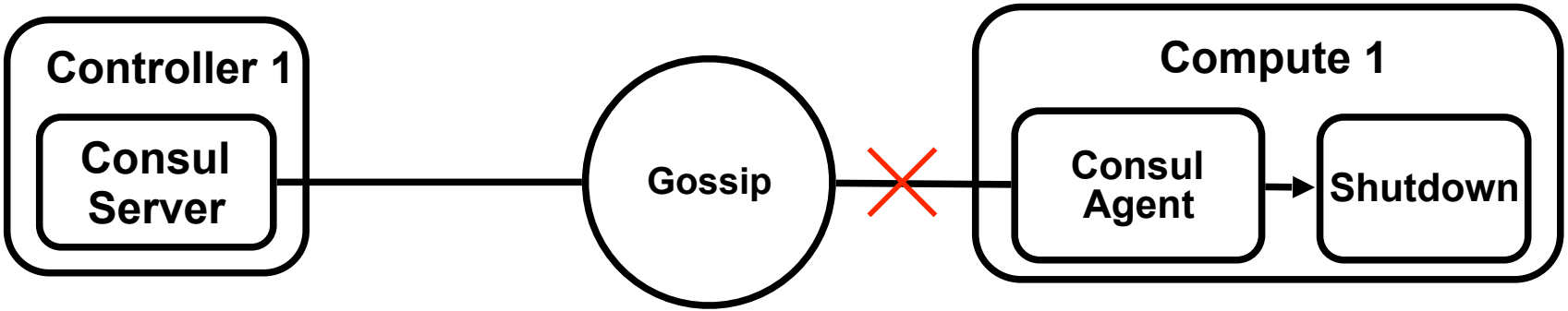
- Avoid Shutting Down the Machine in Loop
- Consul K/V Store

A Complement Fence Method

Send & Receive Fence Message via Consul Event Mechanism



Compute Node Suicide when Losing Fence Channel



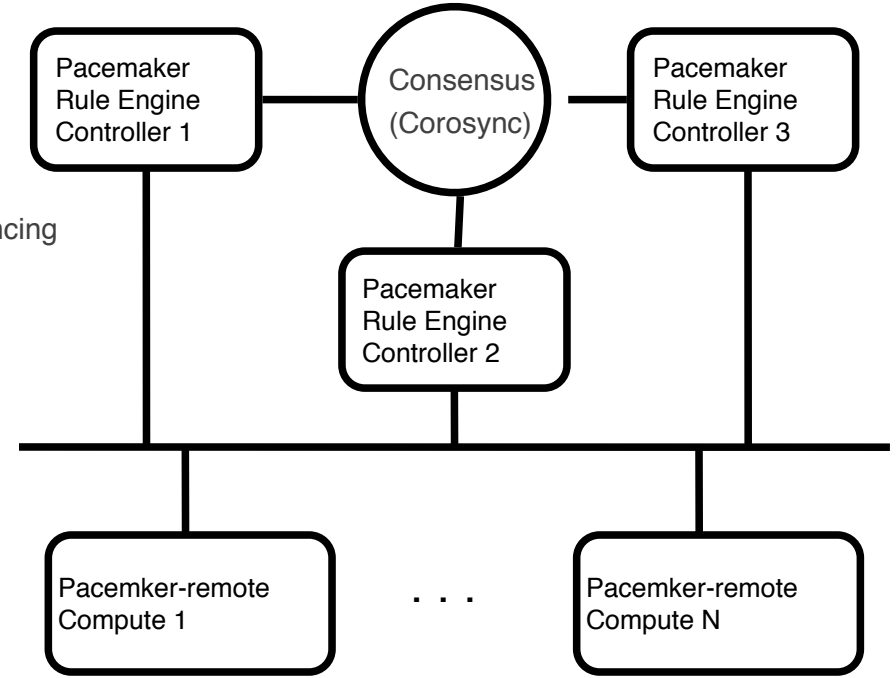
vs. Pacemaker-remote

Pacemaker

- Cluster Orchestration Tool, Based on Corosync, Support Fencing
- Resource, Clone, Master-slave
- Location, Colocation Rules
- Limited Cluster Size
- Corosync RRP Mode: Passive or Active

Pacemaker-remote

- No Limit on Cluster Size
- One Heartbeat Network
  - Usually Management Network
  - Fence the Node when Lost Heartbeat!!!
- ocf:pacemaker:pingd
  - Ping who?



vs. Zookeeper

Zookeeper

- Coordinate Distributed Applications
- Service Discovery
- Hierarchical K/V Store
- Distributed Lock
- Ephemeral ZNodes
- Nova Service Group API Integration

Server Heartbeat Load is Linear to Node Number

No Other Health Checks

Message Queue is Handled by Servers with Quorum

- Not for High Message Rate and Large Cluster

Future Improvements

Limit on Evacuate Rate and Count

Add More Health Check Items

Configurable Action Matrix

Reserve Bandwidth for Gossip Message

- Gossip Traffic Calculation
- 5 Message / s to 3 Nodes = 41 KB/s
- Takes 1.25s for 99 % Convergence in a 30 Node Cluster

Watchdog Device

Nova Considerations

Nova Considerations: Force Service Down

Service API

```
PUT /v2/{tenant_id}/os-services/force-down

{
        “host”: “HostA”
        “binary": “nova-compute”,
        “forced_down”: true
}
```

Enabled In Liberty with Microversions 2.11

Nova Considerations: Evacuate Instance

Evacuate API

```
POST /v2/{tenant_id}/servers/action

{
    "evacuate": {
        "host": "hostA",
        "onSharedStorage": "False"
    }
}
```

Nova Considerations: Evacuate Instance

Evacuate API

```
POST /v2/{tenant_id}/servers/action

{
     "evacuate": {
        "onSharedStorage": "False"
     }
}
```

Host Become optional in Juno

Nova Considerations: Evacuate Instance

Evacuate API

```
POST /v2/{tenant_id}/servers/action

{
     "evacuate": {
        "onSharedStorage": "False"
     }
}
```

Host Become optional in Juno

Persist scheduling policy in Progress

Nova Considerations: Evacuate Instance

Evacuate API

```
POST /v2/{tenant_id}/servers/action

{
     "evacuate": {
     }
}
```

Host Become optional in Juno

Persist scheduling policy in Progress

onSharedStorage optional in Progress

Ceilometer Considerations

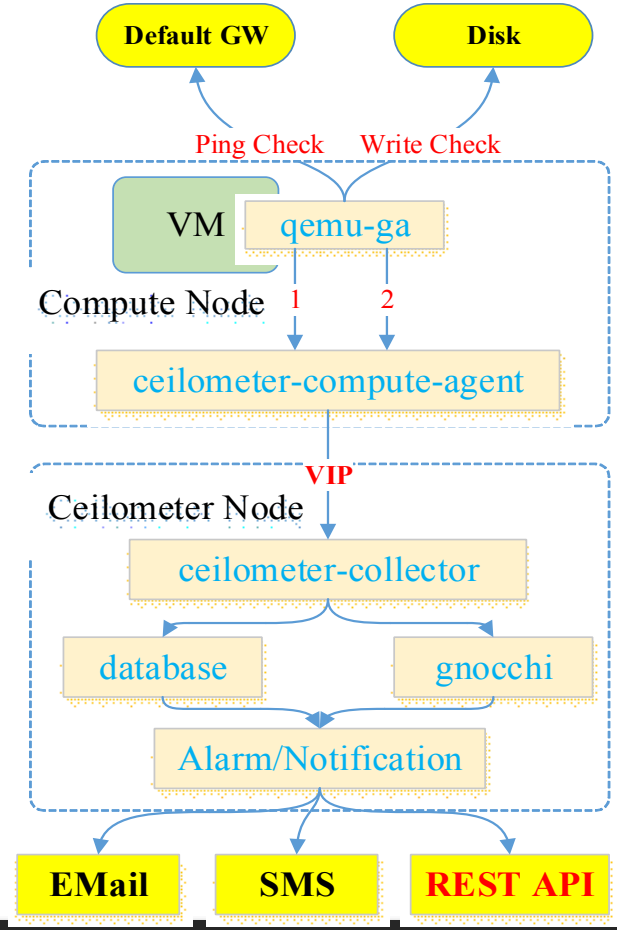
High Availability Based Ceilometer

Two Ceilometer Metrics

- Ping Check Metric - instance.ping.delay
  - delay <= timeout, return delay
  - delay > timeout, return 999
- Storage Health - instance.disk.health
  - writable, return health (health = 0)
  - unwritable, return -1

Assumed Conditions of HA Alarm

- every num period:
  - avg(delay) > 500, trigger alarm
  - sum(health) < 0, trigger alarm



High Availability Based Ceilometer

- **Two Ceilometer Alarm: ping-delay-alarm and disk-health-alarm**

ping-delay-alarm： --meter-name instance.ping.delay --threshold 500 --comparison-operator gt --statistic avg --period 10 --evaluation-periods 3

disk-health-alarm： --meter-name disk.health --threshold 0 --comparison-operator lt --statistic avg --period 10 --evaluation-periods 3

- **Two Ceilometer Alarm Actions: Email, SMS and REST API**
- **REST API as the HA handler**
  - nova live migrate
  - nova reboot api
  - nova rebuild api

High Availability Based Ceilometer

Pros

- high vailability of virtual machine level
- tenant network failures, storage network failures

Cons

- Don’t deal with management network and IPMI network failures
- Too many duplicatable checks if the host is failures
- Must depend on qemu guest agents

Optimizing - Consul mechanism can overcome cons above

Demo

Q & A

海云捷迅 AWCLOUD

intel Software

Intel OpenSource TECHNOLOGY CENTER

中国移动 China Mobile

Thanks


海云捷迅
AWCLOUD


intel
Software


Intel
OpenSource
TECHNOLOGY CENTER


中国移动
China Mobile