FUJITSU

shaping tomorrow with you

Introduction And Status Update About COLO FT

Zhang Chen <zhangchen.fnst@cn.fujitsu.com>

Agenda

Background Introduction

Introduction COarse-grain LOck-stepping

COLO Design and Technology Details

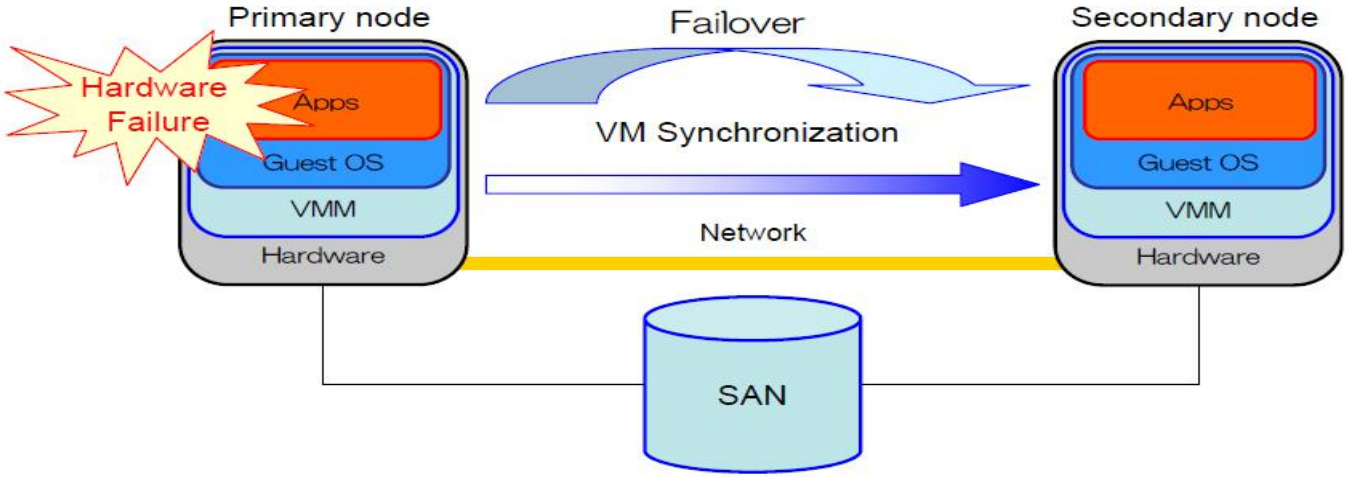
Current Status Of COLO

Future Work About COLO

FUJITSU

Virtual Machine (VM) replication

A software solution for business continuity and disaster recovery through application-agnostic hardware fault tolerance by replicating the state of primary VM (PVM) to secondary VM (SVM) on different physical nodes.



Existing VM Replication Approaches

FUJITSU

Replication Per Instruction: Lock-stepping

- Execute in parallel for deterministic instructions
- Lock and step for un-deterministic instructions

Replication Per Epoch: Continuous Checkpoint

- Secondary VM is synchronized with Primary VM per epoch
- Output is buffered within an epoch

Problems

FUJITSU

Lock-stepping

- Excessive replication overhead
  - memory access in an MP-guest is un-deterministic

Continuous Checkpoint

- Excessive VM checkpoint overhead
- Extra network latency

Agenda

Background Introduction

Introduction COarse-grain LOck-stepping

COLO Design and Technology Details

Current Status Of COLO

Future Work About COLO

What Is COLO ?

FUJITSU

VM and Clients model

VM and Clients are a system of networked request-response system

Clients only care about the response from the VM

COarse-grain LOck-stepping VMs for Non-stop Service (COLO)

PVM and SVM execute in parallel

Compare the output packets from PVM and SVM

Synchronize SVM state with PVM when their responses (network packets) are not identical

The idea is presented in Xen summit 2012, and 2013, and academia paper in SOCC 2013

Why COLO Better

FUJITSU

Comparing with Continuous VM checkpoint

No buffering-introduced latency

Less checkpoint frequency

On demand vs periodic

Comparing with lock-stepping

Eliminate excessive overhead of un-deterministic instruction execution due to MP-guest memory access

Agenda

Background Introduction

Introduction COarse-grain LOck-stepping

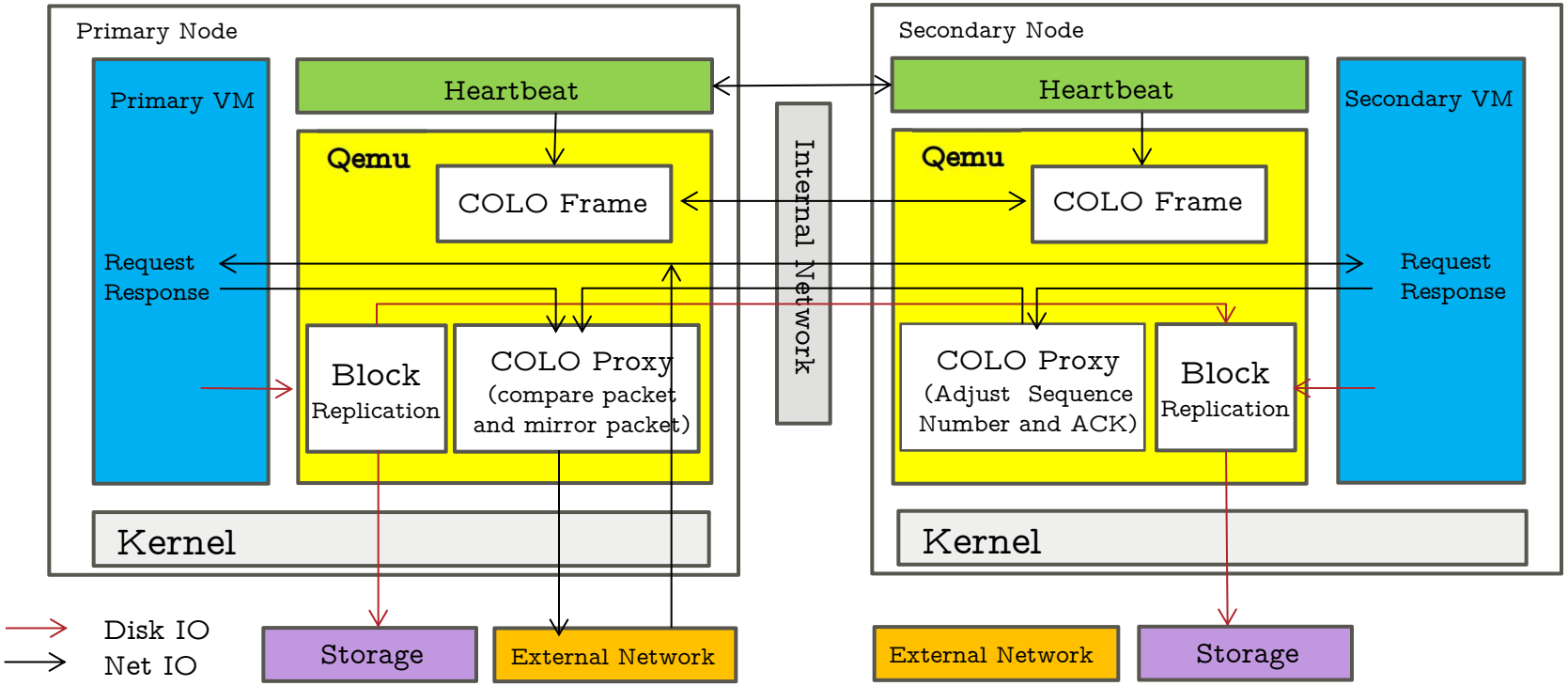
COLO Design and Technology Details

Current Status Of COLO

Future Work About COLO

Architecture Of COLO

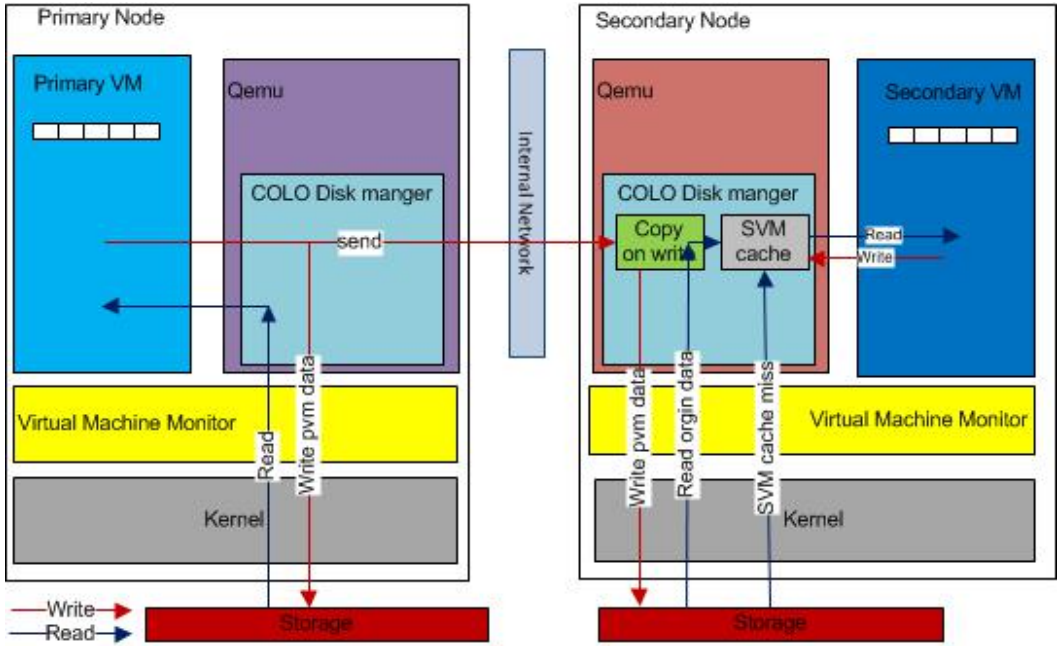
FUJITSU



COarse-grain LOck-stepping Virtual Machine for Non-stop Service

Block Replication(Storage Process)

FUJITSU



Checkpoint

Drop SVM cache

Failover

Write SVM cache to storage

Base on qemu's quorum,nbd,backup-driver,backingfile

Write

Pnode

Send the write request to Snode

Write the write request to storage

Snode

Receive PVM write request

Read original data to SVM cache & write PVM write request to storage(Copy On Write)

Write SVM write request to SVM cache

Read

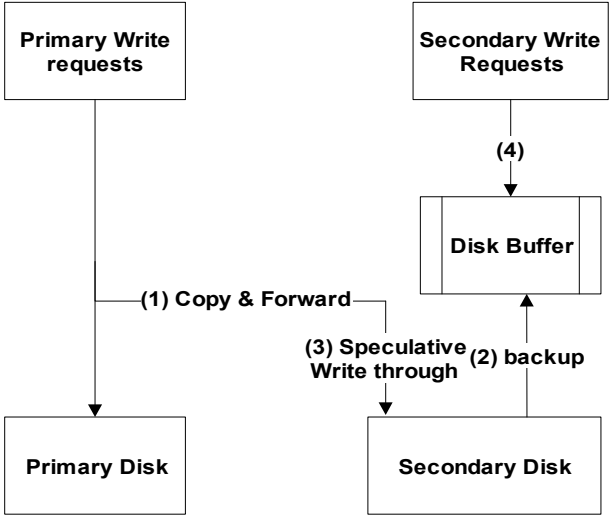
Snode

Read from SVM cache, or storage (SVM cache miss)

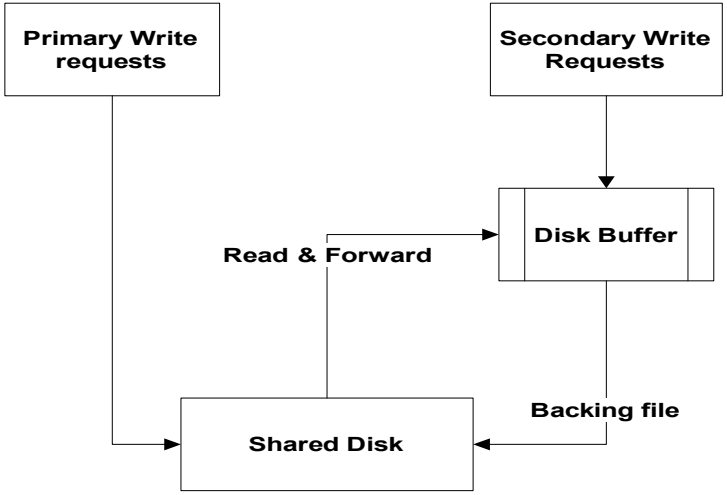
Pnode

Read form storage

How Block Replication Work



Non-Shared disk workflow

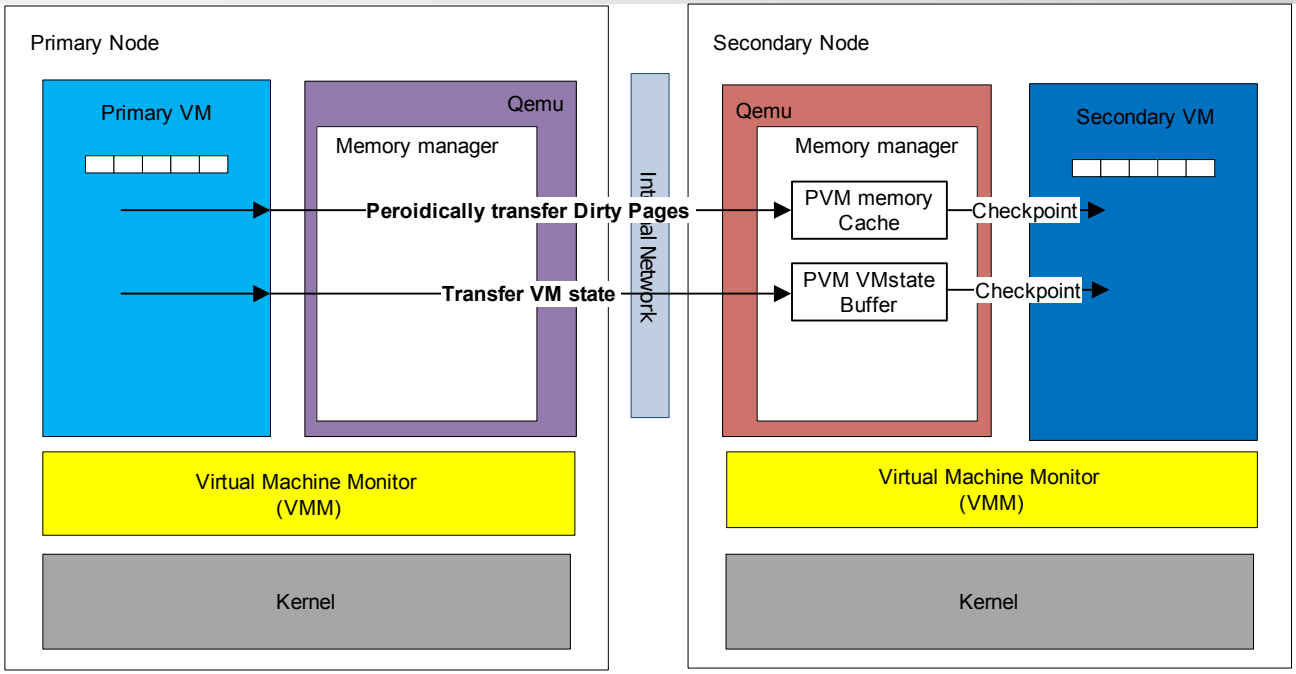


Shared disk workflow

Checkpoint: Disk buffer will be emptied to achieve block replication
Failover: Disk buffer will be written back to the ‘parent’ disk

COLO Frame (Memory Sync Process)

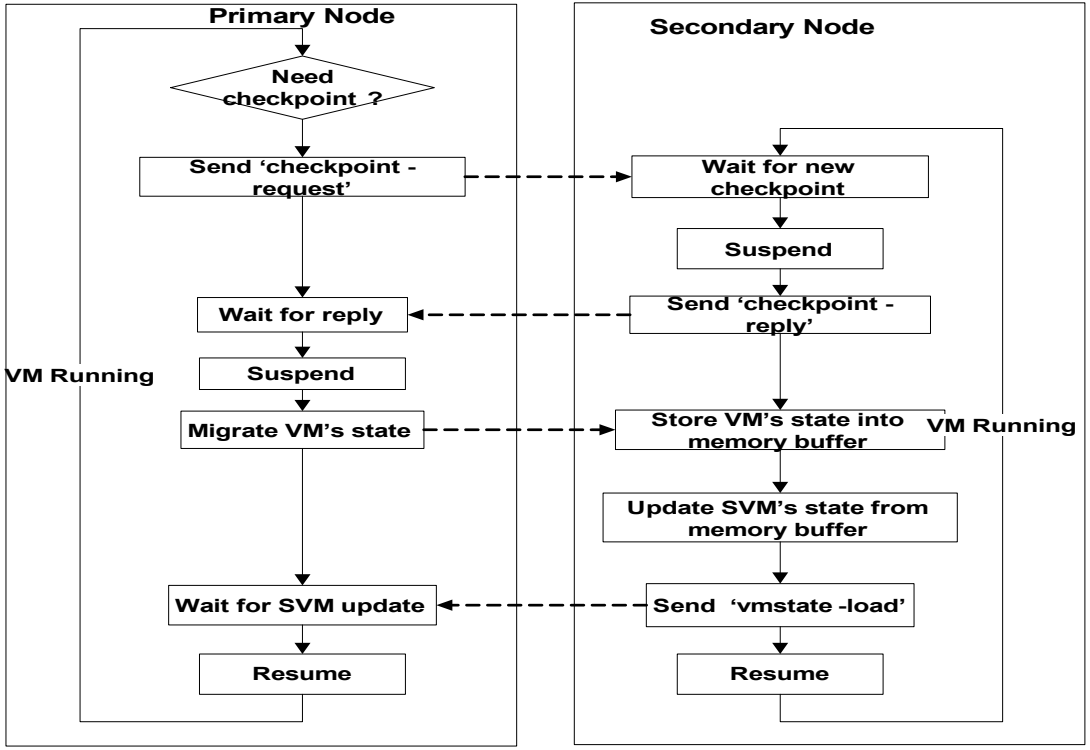
FUJITSU



- **PNode**
  - Track PVM dirty pages, send them to Snode periodically
- **Snode**
  - Receive the PVM dirty pages, save them to PVM Memory Cache
  - On checkpoint, update SVM memory with PVM Memory Cache

COLO Frame (VM State Checkpointing)

FUJITSU

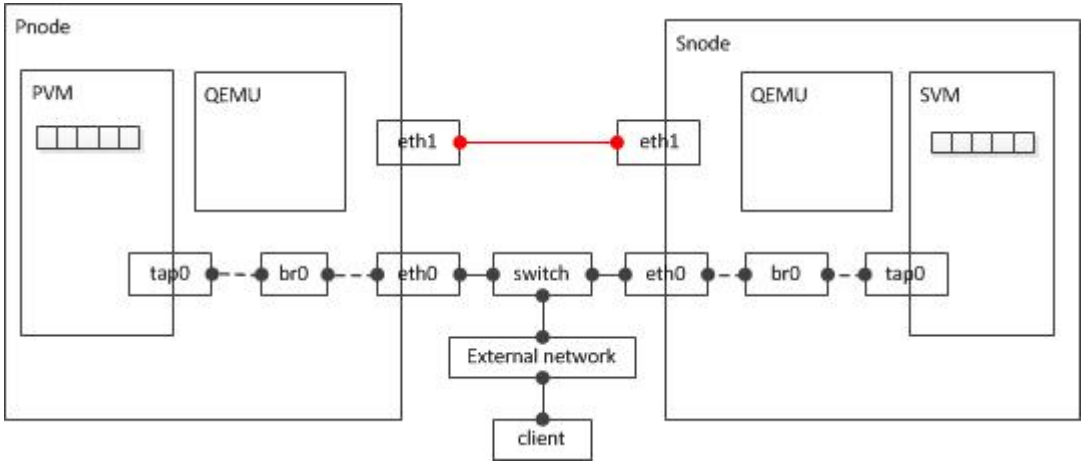


Execution and Checkpoint Flow in COLO

Based on live migration PVM's memory/device data be stored in extra memory-buffer of SVM before be synchronized to SVM

Network topology of COLO

FUJITSU



Pnode: primary node; PVM: primary VM; Snode: secondary node; SVM: secondary VM

[eth0] : client and vm communication

[eth1] : migration/checkpoint, storage replication and proxy

COLO Proxy Design

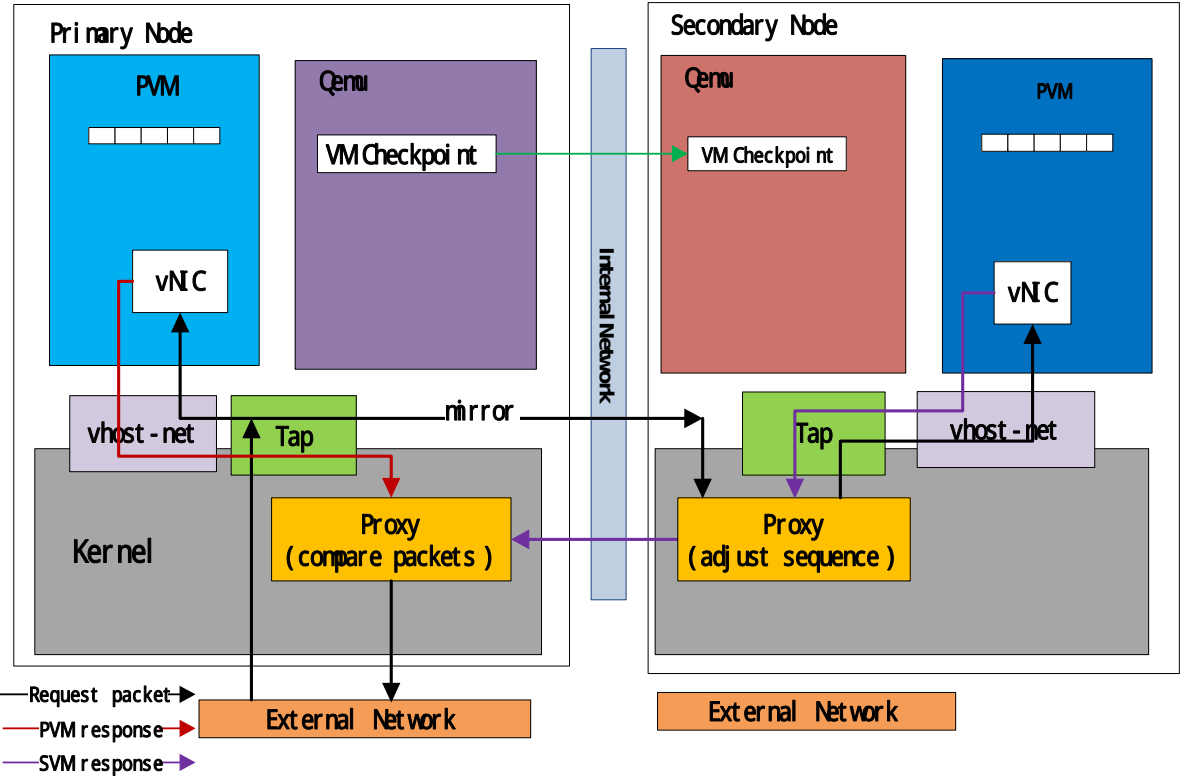
FUJITSU

Scheme:

- ~~Kernel scheme~~: (obsolete)
  - Based on kernel TCP/IP stack and netfilter component
  - Can support vhost-net, virtio, e1000, rtl8139, etc
  - Better performance but less flexible (Need modify netfilter/iptables and kernel)
- Userspace scheme:
  - Totally realized in QEMU
  - Based on QEMU's netfilter components and SLIRP component
  - Not support vhost-net, but e1000, rtl8139 ,`virtio-net`
  - More flexible

Proxy Design (Kernel scheme)

FUJITSU



Same: release the packet to client

Different: trigger checkpoint and release packet to client

Base on kernel TCP/IP and netfileter

Guest-RX

Pnode

- Receive a packet from client
- Copy the packet and send to Snode
- Send the packet to PVM

Snode

- Receive the packet from Pnode
- Adjust packet's ack_seq number
- Send the packet to SVM

Guest-TX

Snode

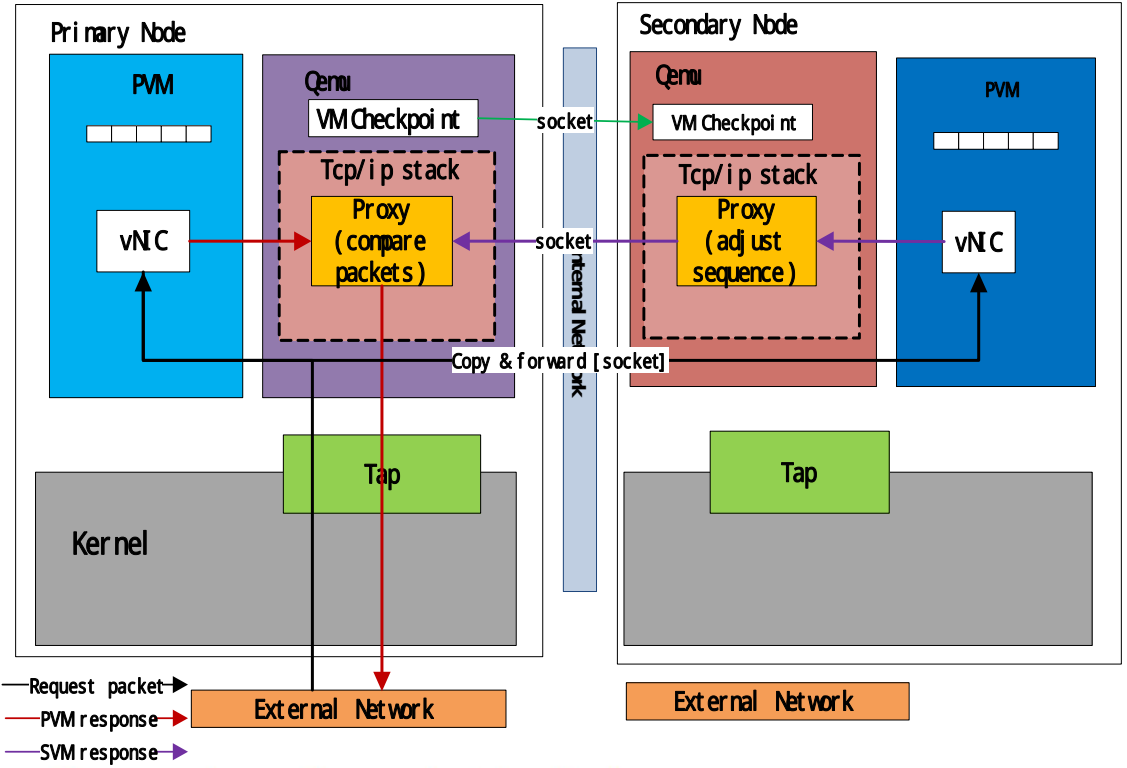
- Receive the packet from SVM
- Adjust packet's seq number
- Send the SVM packet to Pnode

Pnode

- Receive the packet from PVM
- Receive the packet from Snode
- Compare PVM/SVM packet

Proxy Design (Userspace scheme)

FUJITSU



Same: release the packet to client

Different: trigger checkpoint and release packet to client

Base on Qemu's **netfilter** and **SLIRP (Userspace TCP/IP stack)**

Guest-RX

Pnode

- Receive a packet from client
- Copy the packet and send to Snode
- Send the packet to PVM

Snode

- Receive the packet from Pnode
- Adjust packet's ack_seq number
- Send the packet to SVM

Guest-TX

Snode

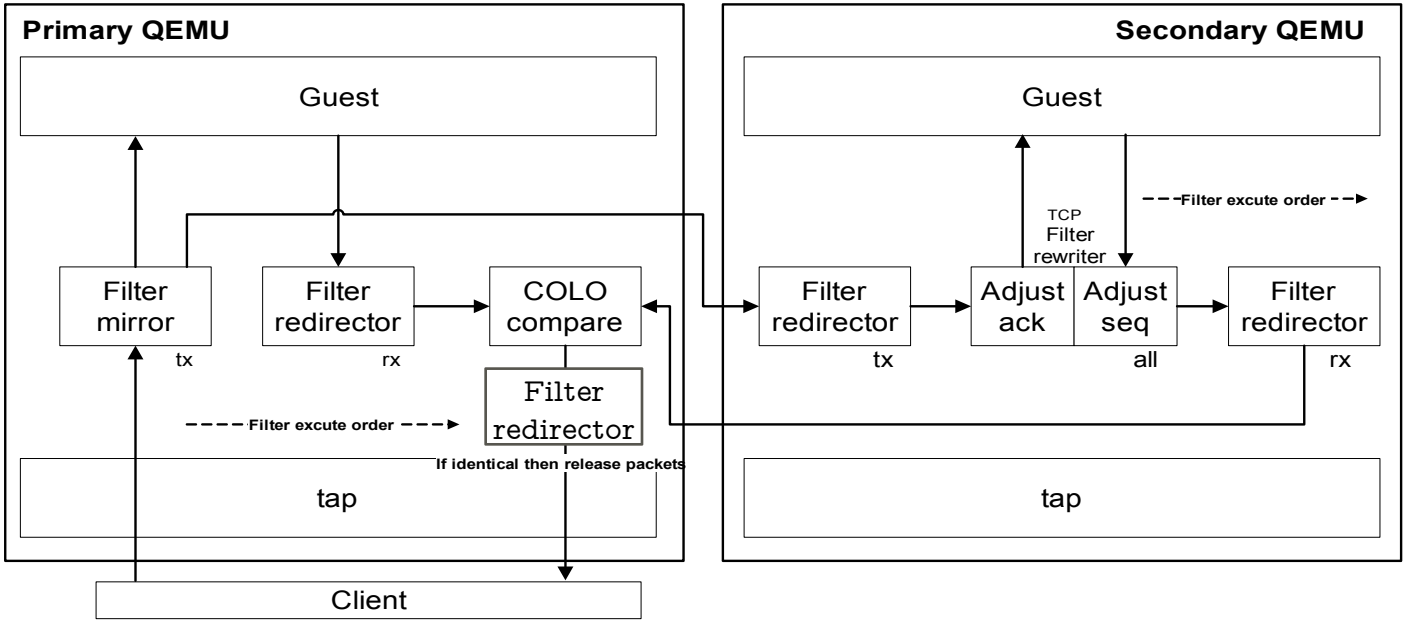
- Receive the packet from SVM
- Adjust packet's seq number
- Send the SVM packet to Pnode

Pnode

- Receive the packet from PVM
- Receive the packet from Snode
- Compare PVM/SVM packet

Proxy Design (Userspace scheme)

FUJITSU



Filter mirror: copy and forward client's packets to SVM

Filter redirector: redirect net packets

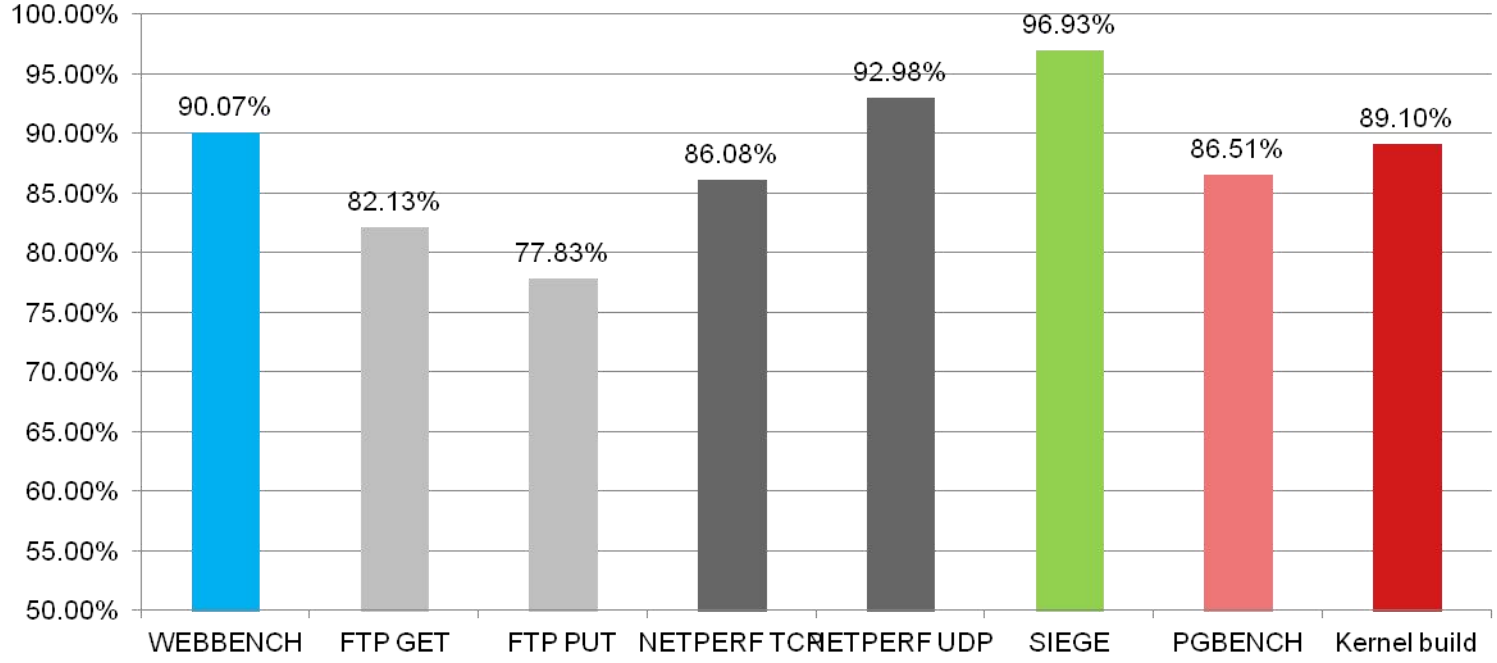
COLO compare: compare PVM's and SVM's net packets;

Filter rewriter: adjust tcp packets' ack and tcp packets' seq

COLO Performance In KVM

FUJITSU

Performance (Based on kernel proxy)



The experimental data is normalized to the native system

Agenda

Background Introduction

Introduction COarse-grain LOck-stepping

COLO Design and Technology Details

Current Status Of COLO

Future Work About COLO

Status of COLO In KVM

FUJITSU

COLO Framework:

Include VM state checkpoint process, failover process

Already been merged to master branch

Notify block replication and colo-proxy related patchset V2 has been post.

COLO block replication:

Only include non-shared storage replication scheme

Already been merged to master branch

COLO proxy:

Include netfilter /mirror/redirector/rewriter/ compare plugins

Already been merged to master branch

Status of COLO In Xen

FUJITSU

COLO Framework:

Already been merged to master branch

COLO block replication:

Only include the non-shared storage replication scheme

Have been synced with the last qemu branch

COLO proxy:

Abandoned implementation scheme based on kernel proxy

Have been synced with the last qemu branch

Notify COLO Framework qemu side patchset have been post V1

Xen side patchset have been merged

Agenda

Background Introduction

Introduction COarse-grain LOck-stepping

COLO Design and Technology Details

Current Status Of COLO

Future Work About COLO

Future Work

FUJITSU

- Revise patches according review feedbacks, get patches accepted into upstream
- Continuous VM replication development
- Support shared storage
- Network performance optimizations
- Libvirt support

FUJITSU

shaping tomorrow with you