Multilevel Differentiated Hypervisor Caching for Derivative Clouds

Sprint Thesis Talk, RISC ‘17

Prashanth, M.Tech. II

Advisor:
Prof. Purushottam Kulkarni

Cloud Provider Architecture

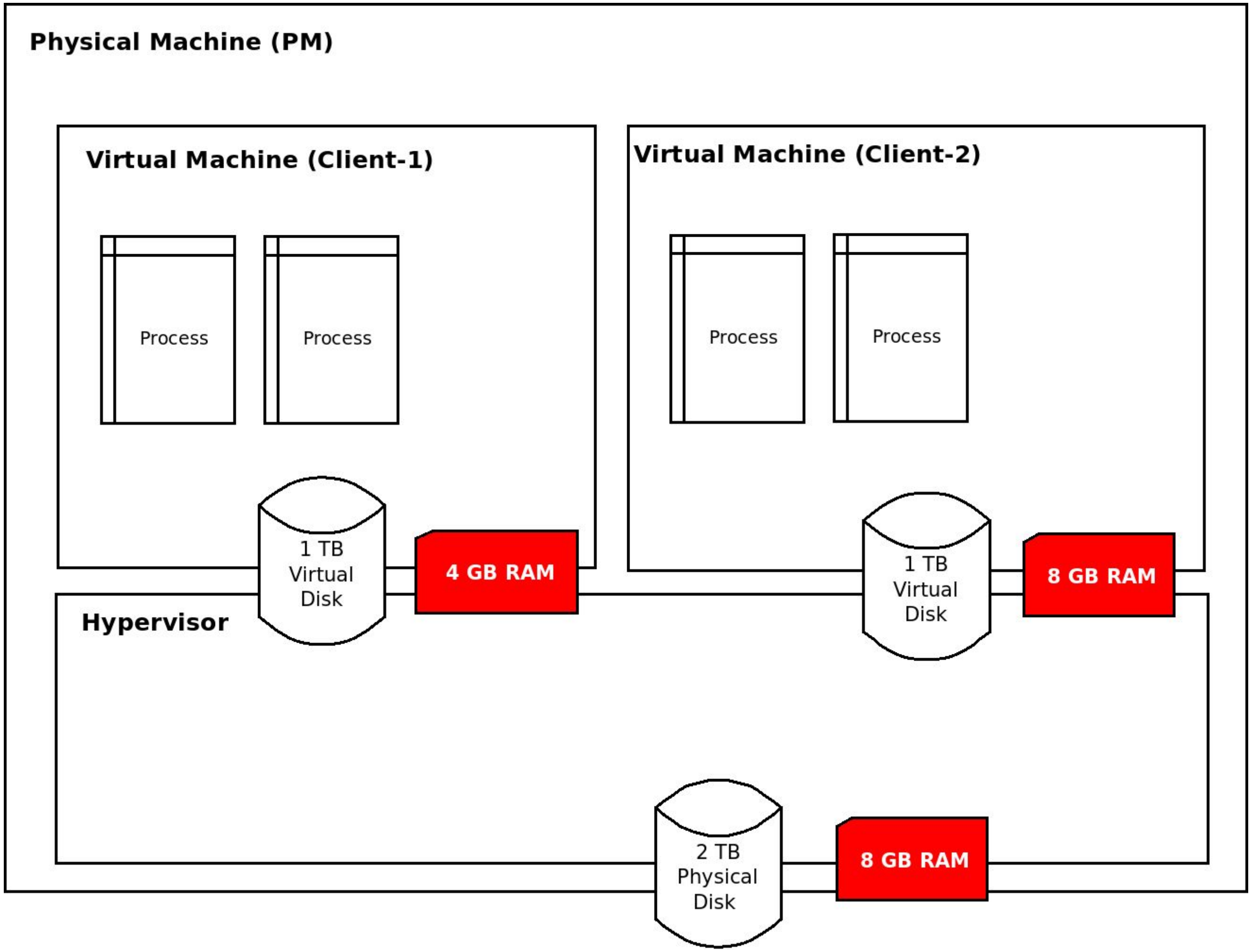


Fig-1: Cloud providers provisioning clients using VMs

Key Points

1. Provision clients on VMs
2. Map SLA requirements to VM resources
3. Overprovision resources for cost benefits

Caching in a cloud setup

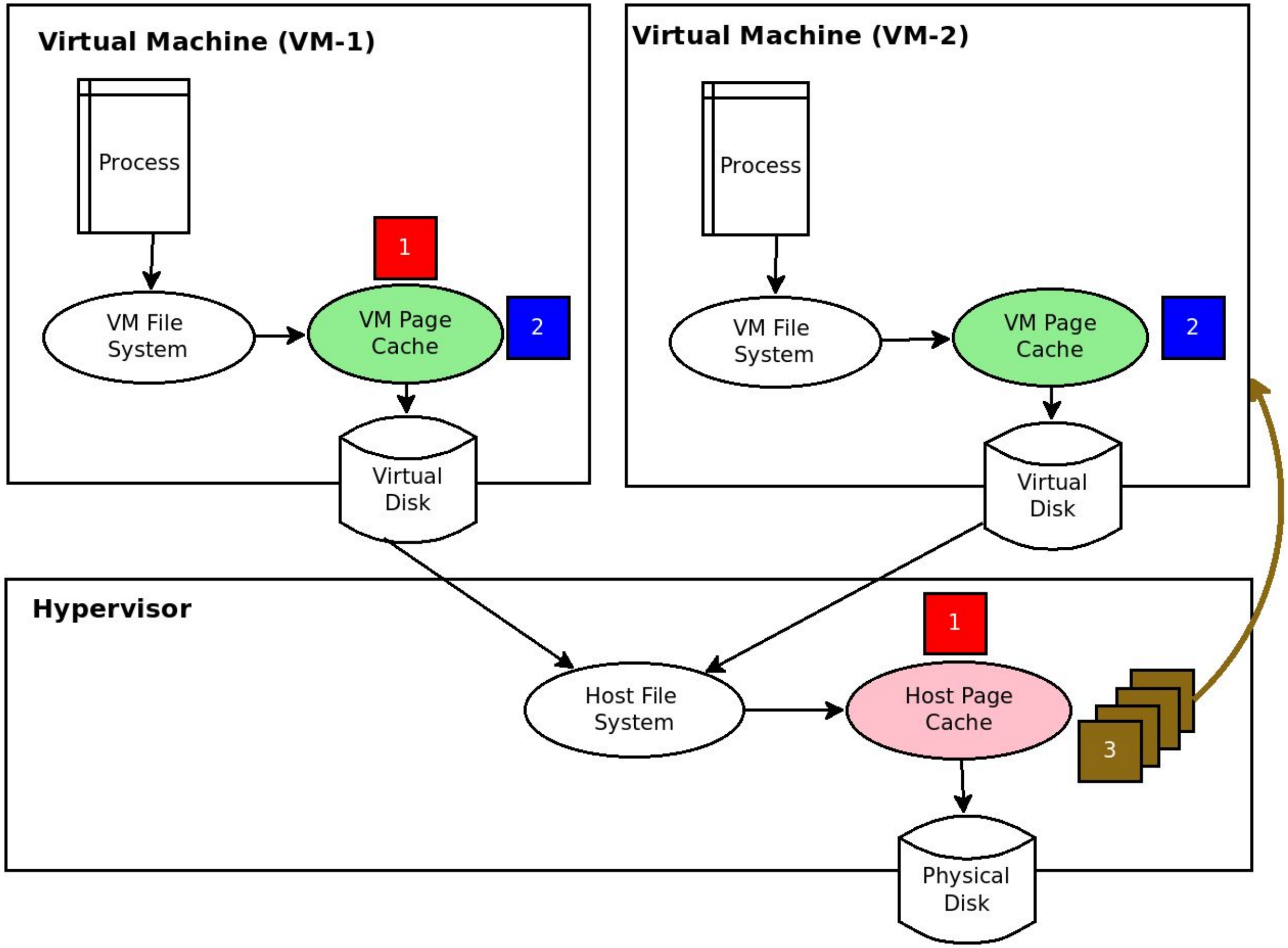


Fig-2: File I/O in traditional hypervisor based cloud setup

Drawbacks

1. Multiple copies of same page at host and VM
2. Multiple copies between VMs
3. Flooding of host page cache by a particular VM

Hypervisor managed caching

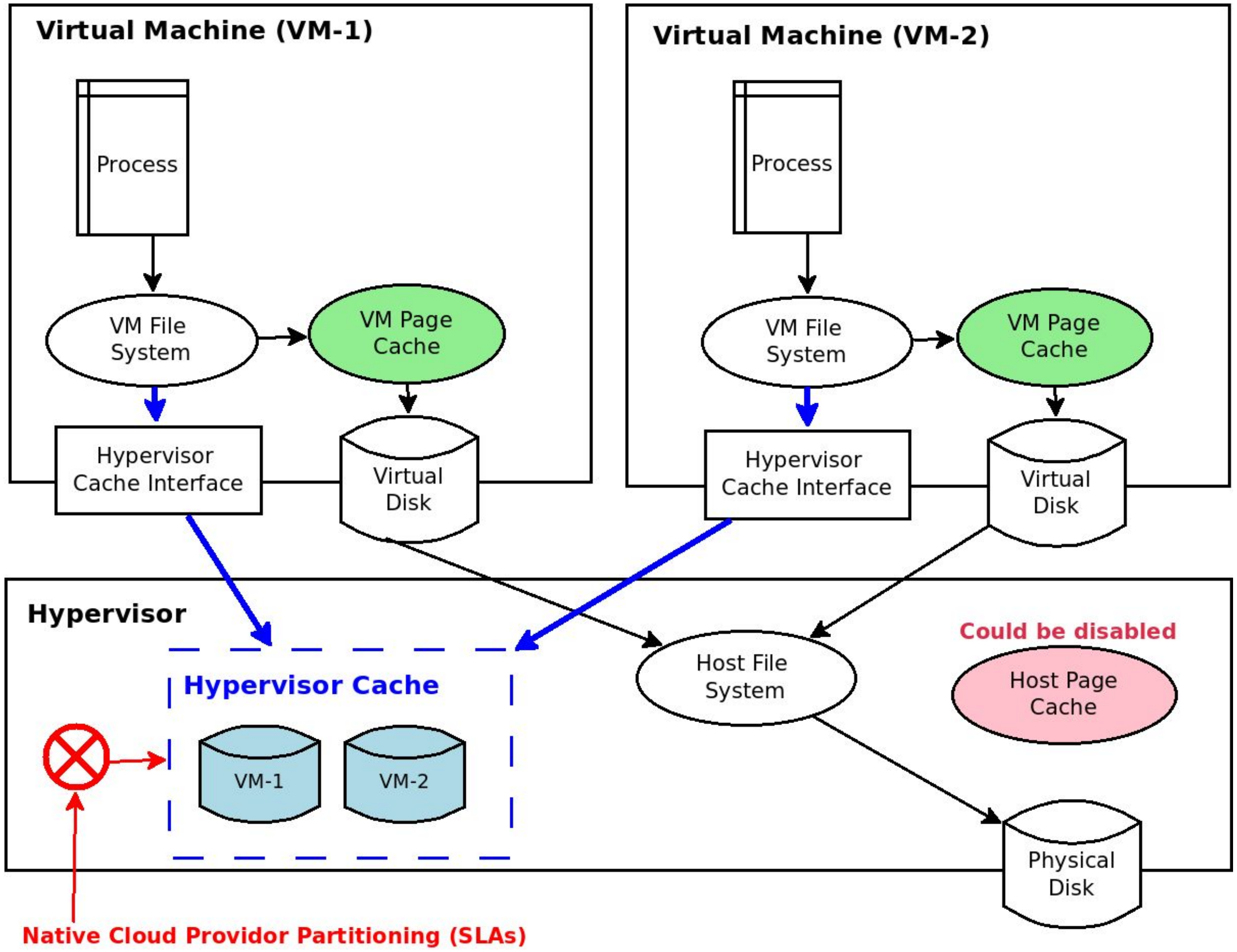


Fig-3: Hypervisor Caching

Key Points

1. Partitioning per VM based on high level SLAs
2. Exclusive cache
3. Dynamic readjustments
4. Existing works address this issue - [SDC SoCC '15] [Centaur ICAC '15]

⊗ Marks the controller

What are traditional caches backed by ?

❑ Caches could be backed by

1. Memory (RAM)
2. SSD
3. NVMs etc.

❑ We could even combined them to form multi-level or hybrid cache designs - [ExTmem HPCC '14]

❑ Existing literatures on hypervisor caches are of single level cache

Derivative Cloud Environment

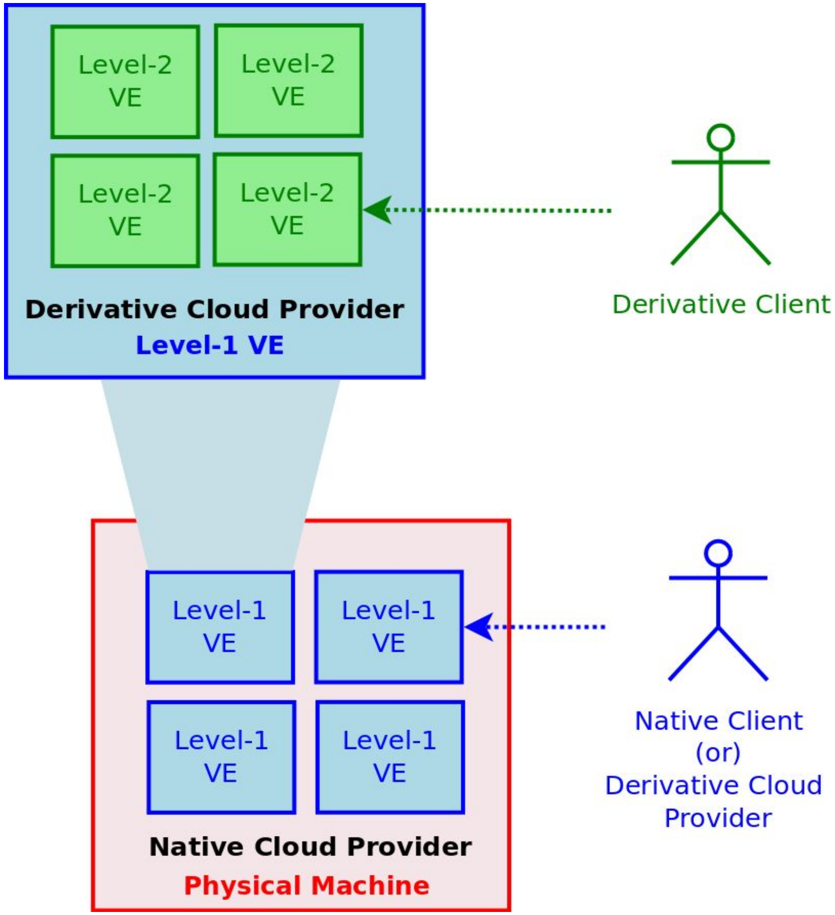


Fig-4: Comparison between native and derivative cloud environment

- ❑ **VE (Virtual Environment):** Virtual Machine (VM) or Container
- ❑ Level-1 VE - VM
  Level-2 VE - Container
- ❑ [Spotcheck EuroSys '15]
  [Heroku PAAS provider]

Problem Statement

To develop a caching framework that supports,

❑ Hypervisor caching

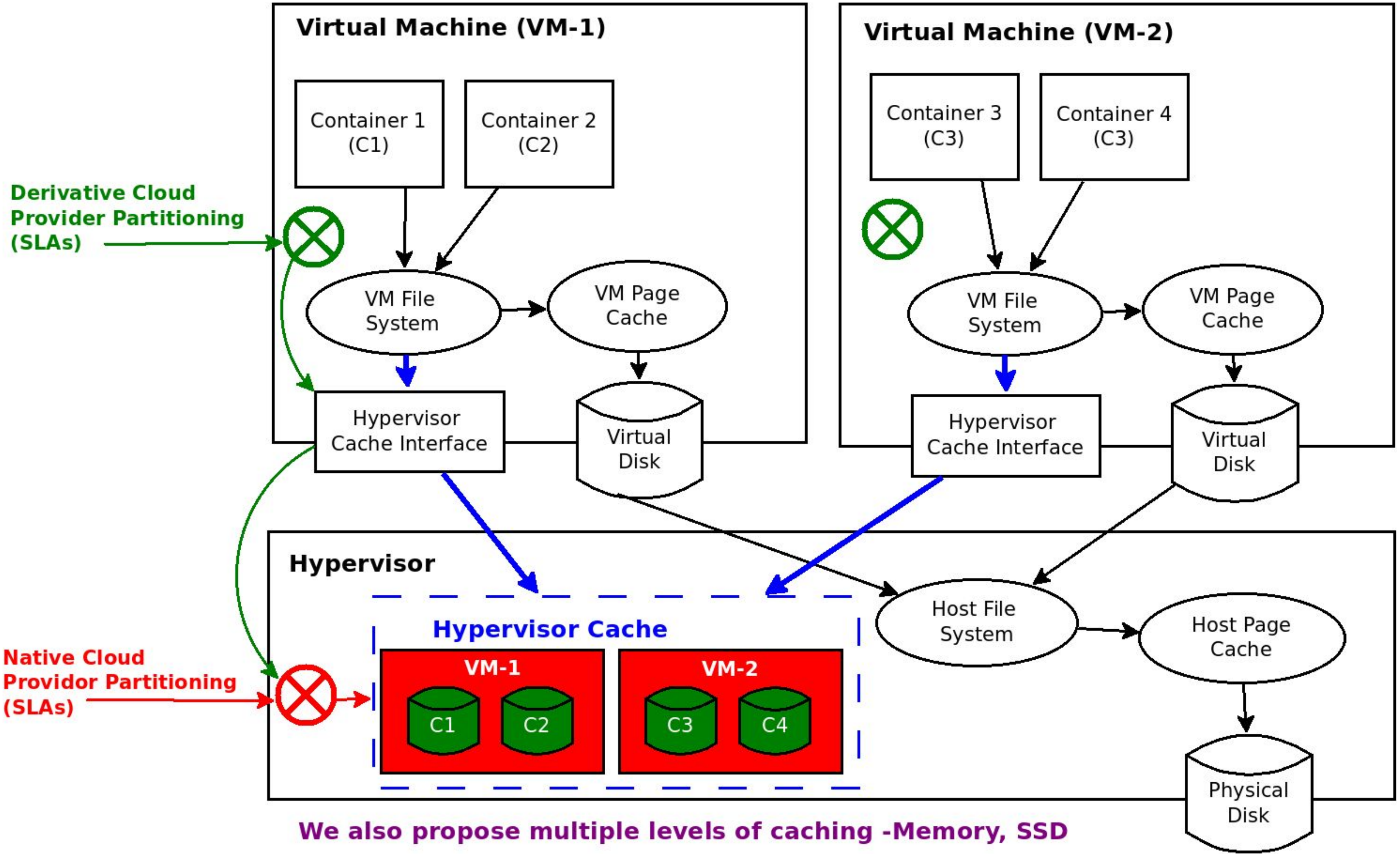
❑ Multiple levels of configurable cache

❑ A derivative cloud setup for enforce native and derivative provider SLA policies

Requirements of desired system

- Multiple levels of hypervisor managed caches
- Per VM and per container configurable caches at each level
- Resource conserving nature
- Spillover mechanism - Exceeds in L1 spilled over to the L2 cache
- Exclusive caching at all levels

Hypervisor caching in derivative clouds



We also propose multiple levels of caching -Memory, SSD

Fig-5: Proposed Architecture for cache partitioning in derivative clouds

Implementation Specifics

- T-MEM (Transcendent Memory) cache - A second cache caching framework for optimization of RAM
- Extended this to support hypervisor backed caches using memory and SSD
- KVM Hypervisor
- LXC containers (could be easily extended to other container managers)
- Control Knobs - Relative weights, Cache size

Current Implementation

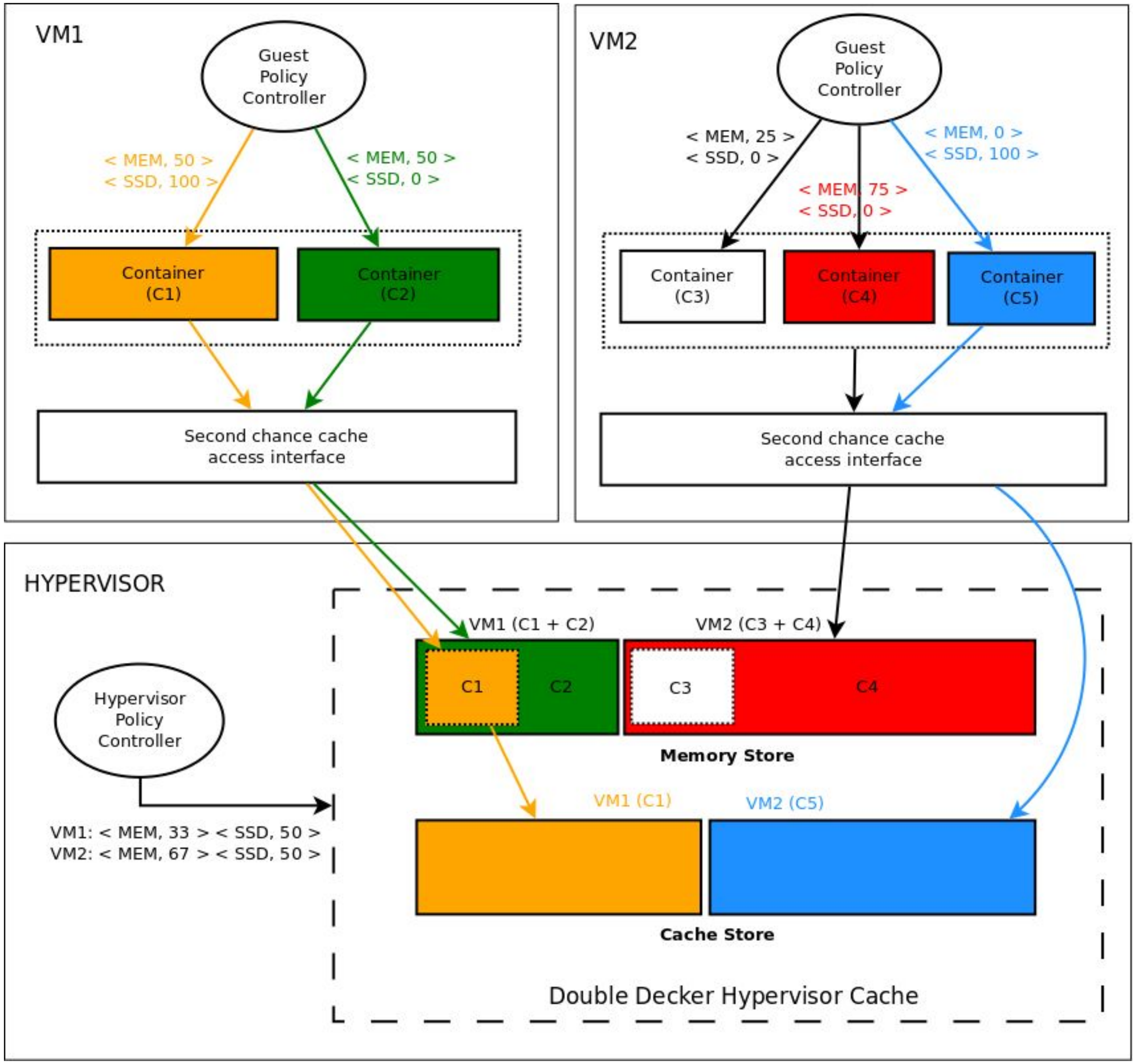


Fig-6: Implementation Details

Set of APIs to control caches

Set of APIs,

1. CREATE_CACHE
2. PUT_OBJECT
3. GET_OBJECT
4. DESTROY_CACHE
5. SET_WEIGHTS
6. EVICT_SPECIFIC_OBJECTS
7. EVICT_SET_OF_OBJECTS
8. MIGRATE_FROM_L1_TO_L2
9. MIGRATE_FROM_L2_TO_L1

Future Work

- Making use of developed APIs to map SLA policies into cache partitions
- Would hint passing from VM to Host help in cache partitioning ?

Thank You !

Any Questions ?