# Asgard: Empowering Memory Scalability for Terabyte-Scale Hardware-Partitioned Servers

**Ángel Burgos Muñoz** **Konstantinos Chasialis** **Pedro Palacios Almendros** **Luis Romero Rodríguez**

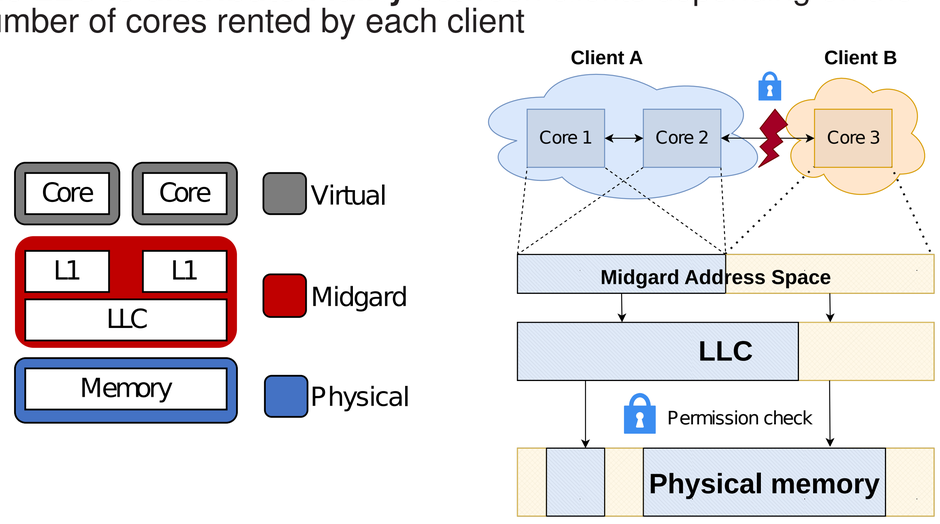
EPFL, Lausanne

## Introduction

- Virtual machines are vulnerable to hypervisor-level side channel attacks and incur performance overheads
  - Bare-metal cloud enables **hardware partition** at a **reduced performance overhead**, with solutions such as Core Slicing
  - Bare-metal cloud minimizes hypervisor intervention, **reducing security risks**
- Ever increasing memory and LLC sizes
  - Increased memory sizes also increase address translation overheads
  - Intermediate address spaces for cache addressing such as **Midgard increase performance on memory access** by reducing these overheads
- **Cloud rental services** such as AWS with multi-client colocation face these security issues and need memory scalability

## Asgard

- Hardware partitioning can be **synergistically** enforced using an intermediate address space
- **System-wide hardware-partitioned** Midgard address space used for cache indexing and isolation of different clients
  - The Midgard Address Space, which is big and sparse, is partitioned evenly between clients by prepending to each memory access address a client identifier
  - The **LLC is distributed fairly** between clients depending on the number of cores rented by each client




- Physical memory and core isolation management does not differ much from Core Slicing's implementation
  - Reduce physical address permission check frequency from TLB miss to LLC miss

## Addressing the LLC partitioning challenge

- Existing cache partitioning techniques are inadequate for our purposes:
  - *Page coloring* would **increase pressure** on Midgard's address translation by imposing a limit on Virtual Memory Areas size
  - *Way partitioning* would **decrease the associativity** of each client and have **limited granularity**
- Fixed distributions of accesses between LLC slices are **inflexible** and cannot adapt to the arrival and departure of clients

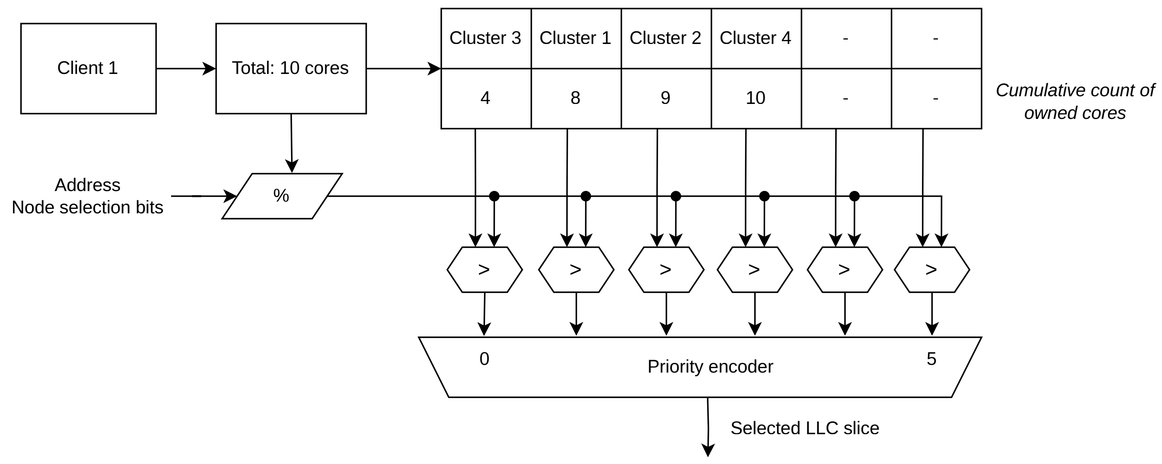
**How can dynamic LLC partitioning be ensured without sacrificing the advantages of bare-metal cloud and the intermediate address space translation?**

## Asgard's cache partitioning

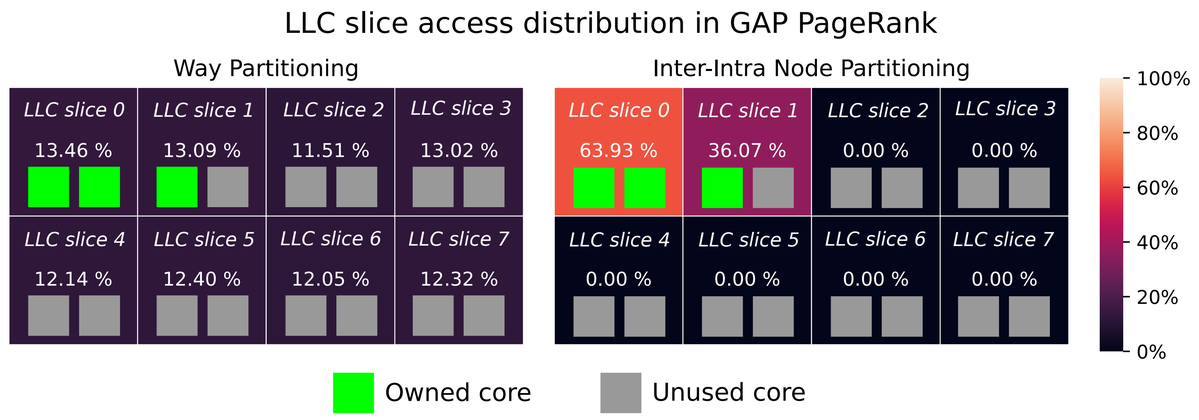
- Partitioning takes effect at two different levels at shared last level caches: **Intra-node** and **Inter-node** partitioning
- Partitioning between different memory nodes (*inter-node*) enables keeping data close to each client's core, reducing LLC average hit latency
- Partitioning inside each memory node (*intra-node)* guarantees that clients have a **proportional amount of cache** to the number of cores rented in the cluster of each LLC slice
- Both partitioning schemes support **dynamic reconfiguration** for incoming and departing clients

## Inter-node Partitioning

- Distribute accesses between LLC slices where the client owns cores by extracting bits from Midgard addresses and indexing an additional structure

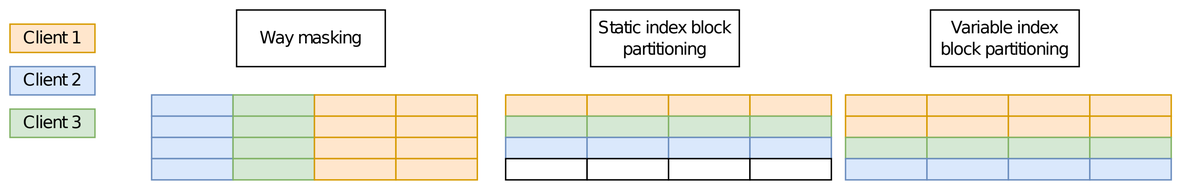



- Allows control over the distribution of LLC accesses
- Move each client's **data closer to its cores**

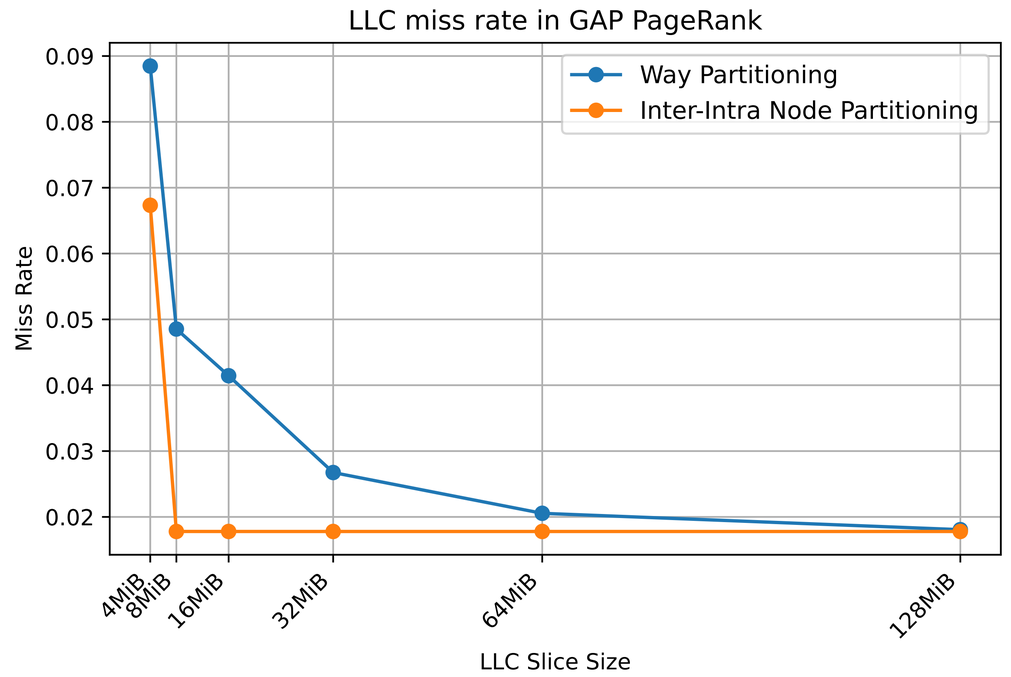



## Intra-node Partitioning

- Partitioning over sets rather than ways preserves the LLC associativity for each client's partition




- **Higher associativity** translates into **lower LLC miss rates**: up to 63% lower miss rates when compared to way partitioning




## Conclusion

- Asgard combines Midgard and Core Slicing to allow **hardware-partitioning** of servers that benefit from Midgard's **memory scalability**
- Our Midgard-adapted LLC partitioning provides **higher associativity**, **better granularity**, **lower miss rates** and distributes **data close to the cores**

EPFL

PARSA PARALLEL SYSTEMS ARCHITECTURE LAB