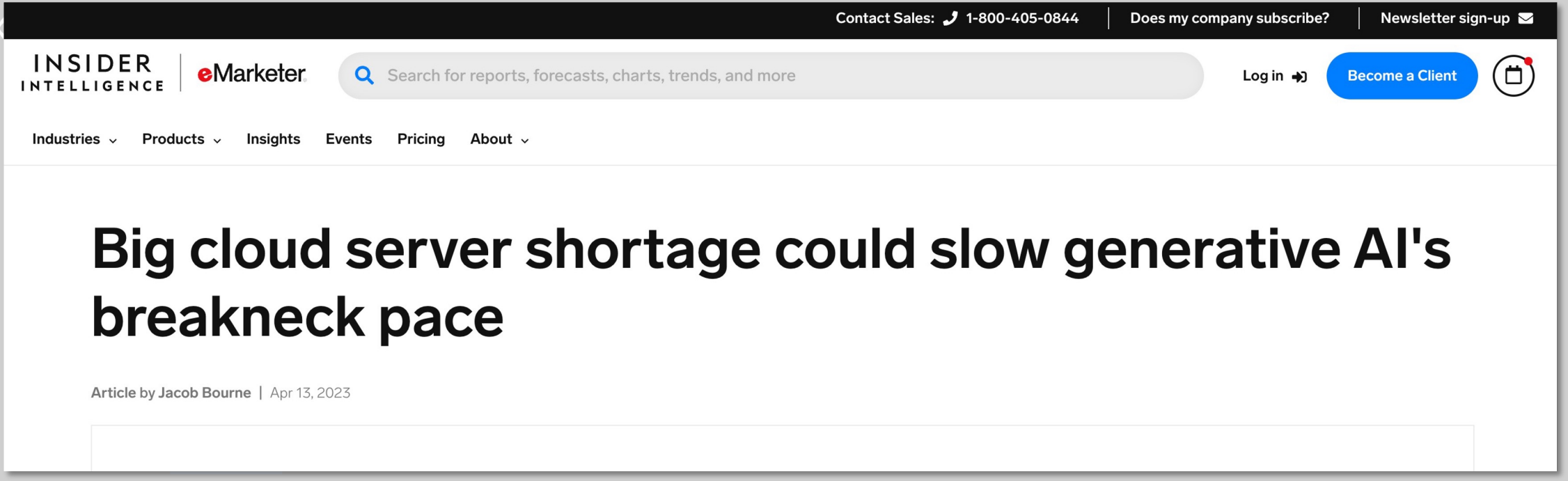
# *Kerveros*:
# Efficient and Scalable Cloud Admission Control

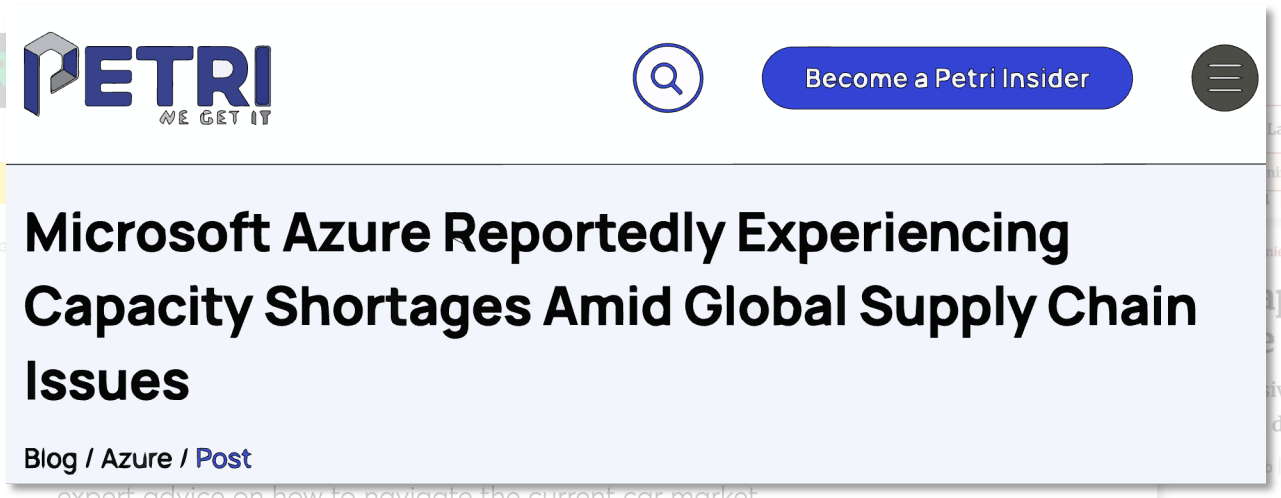
**Sultan Mahmud Sajal**,
Luke Marshall, Beibin Li, Shandan Zhou,
Abhisek Pan, Konstantina Mellou, Deepak Narayanan,
Timothy Zhu, David Dion, Thomas Moscibroda, Ishai Menache

Microsoft Research

Microsoft Azure

PennState


Contact Sales: 1-800-405-0844
Does my company subscribe?
Newsletter sign-up
INSIDER INTELLIGENCE
eMarketer
Search for reports, forecasts, charts, trends, and more
Log in
Become a Client
Industries
Products
Insights
Events
Pricing
About
Big cloud server shortage could slow generative AI's breakneck pace
Article by Jacob Bourne | Apr 13, 2023


PETRI
WE GET IT
Become a Petri Insider
Microsoft Azure Reportedly Experiencing Capacity Shortages Amid Global Supply Chain Issues
Blog / Azure / Post


The Register
Unofficial AWS SHD
@aws_shd · Follow
EC2 (London) - t2.micro Instance Capacity - 9:46 AM PDT We are temporarily running low on t2.micro instance capacity in the EU-WEST-2
12:48 PM · Mar 24, 2017
Reply
Copy link
Read more on Twitter

# Microsoft Azure

**65+** Azure regions

**200+** datacenters worldwide

**1K+** supported VM types

**3M+** machines

**14M+** VM Requests per hour

# Cloud is Finite

Availability Zone

Running VM

Capacity Reservation

Failure and Maintenance

# Cloud is Finite

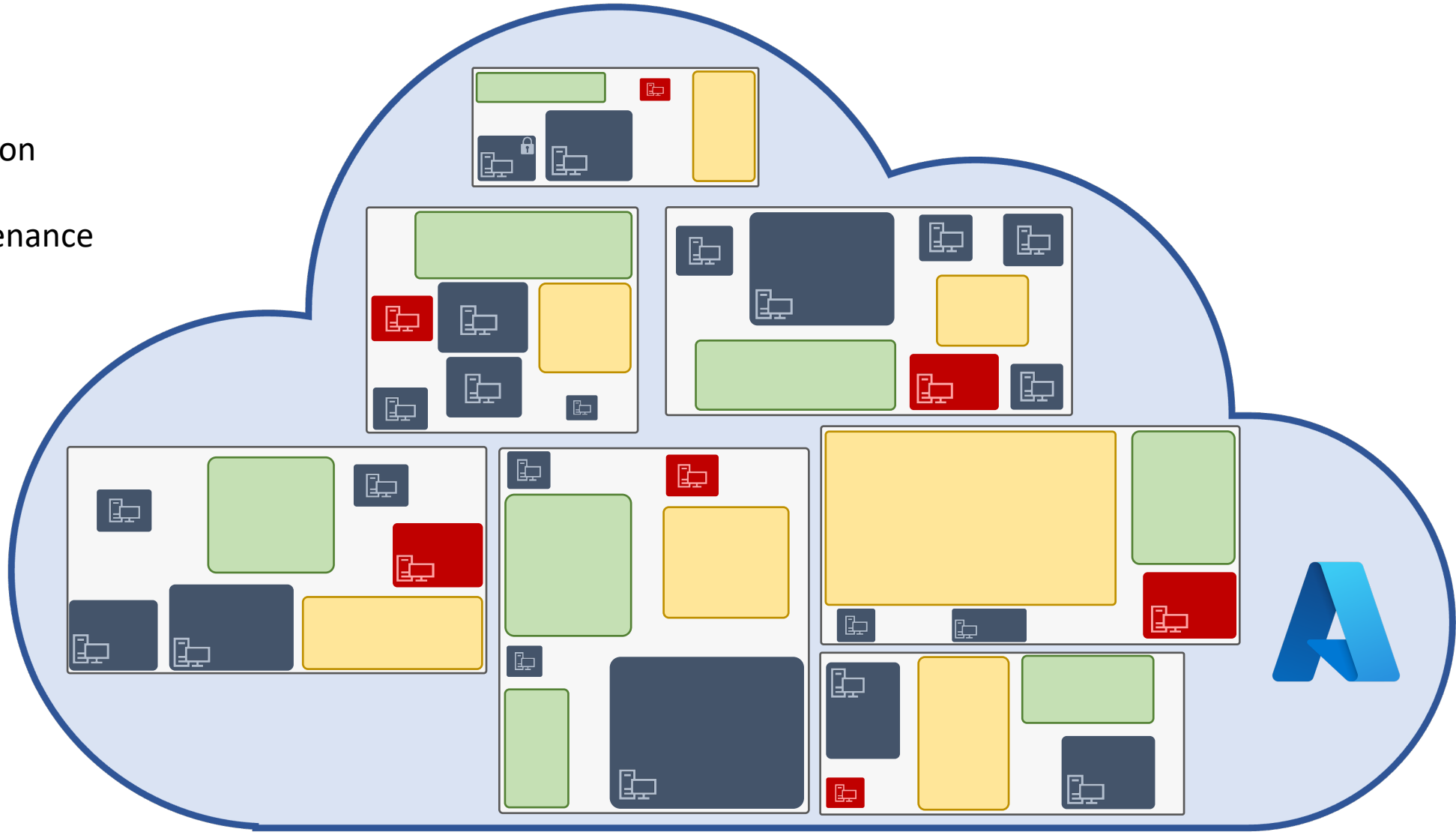
Availability Zone

Running VM

Capacity Reservation

Failure and Maintenance

Failed VM



# Cloud is Finite

Availability Zone

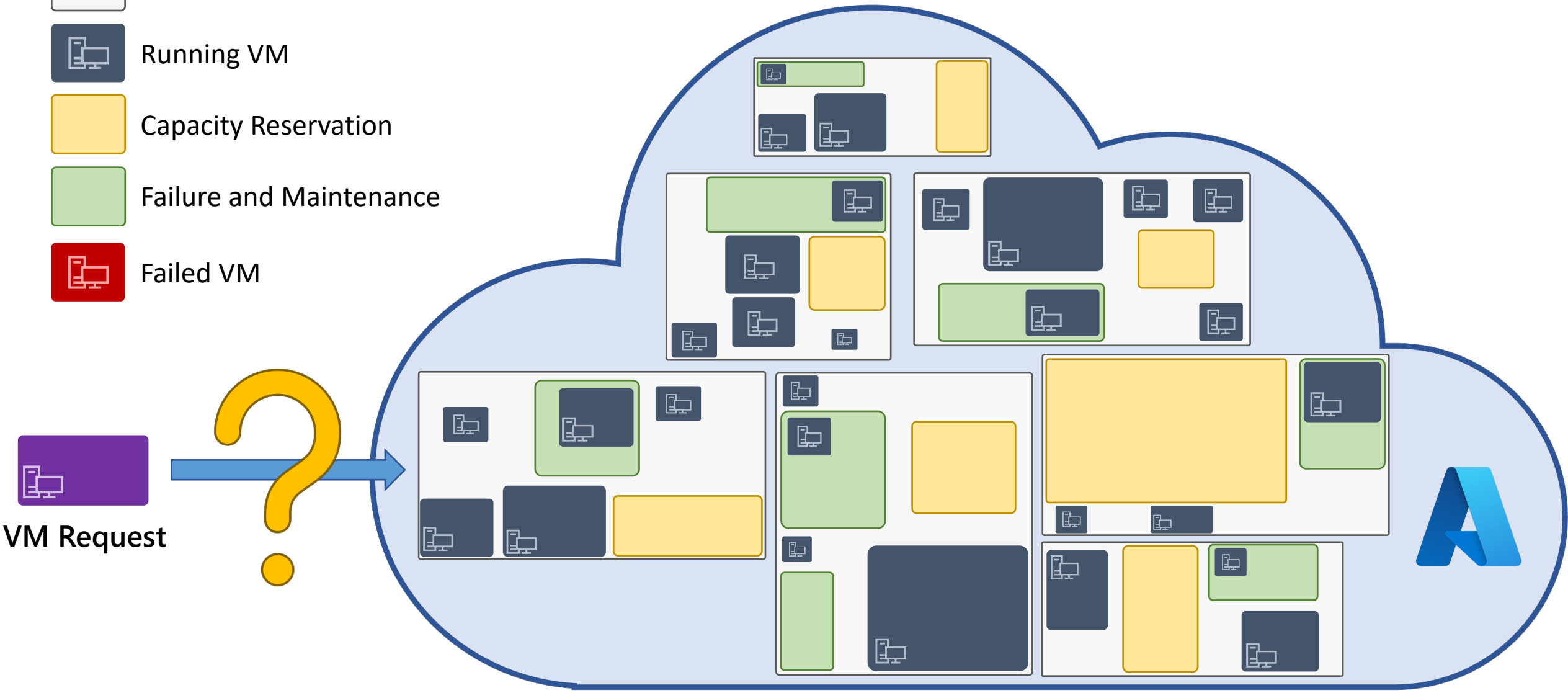
Running VM

Capacity Reservation

Failure and Maintenance

Failed VM

Admission Control: Should a new request be accepted?



VM Request

# Admission Control in Azure

Admission Control: Should a new request be accepted?

Available Resources = Total Resources − Allocated Resources

Supply

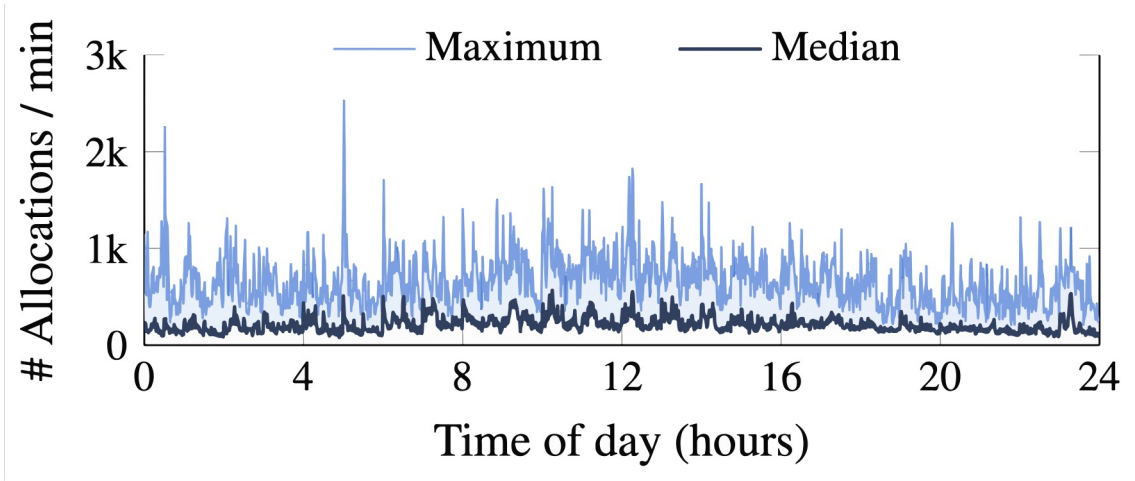
Demand

## Why is it hard?

- Network and Machine Failures
- Scheduled Maintenance
- Unscheduled Maintenance

- VM Requests
- Capacity Reservations
- Customer Scale-Outs

- **Variability affecting supply and demand**

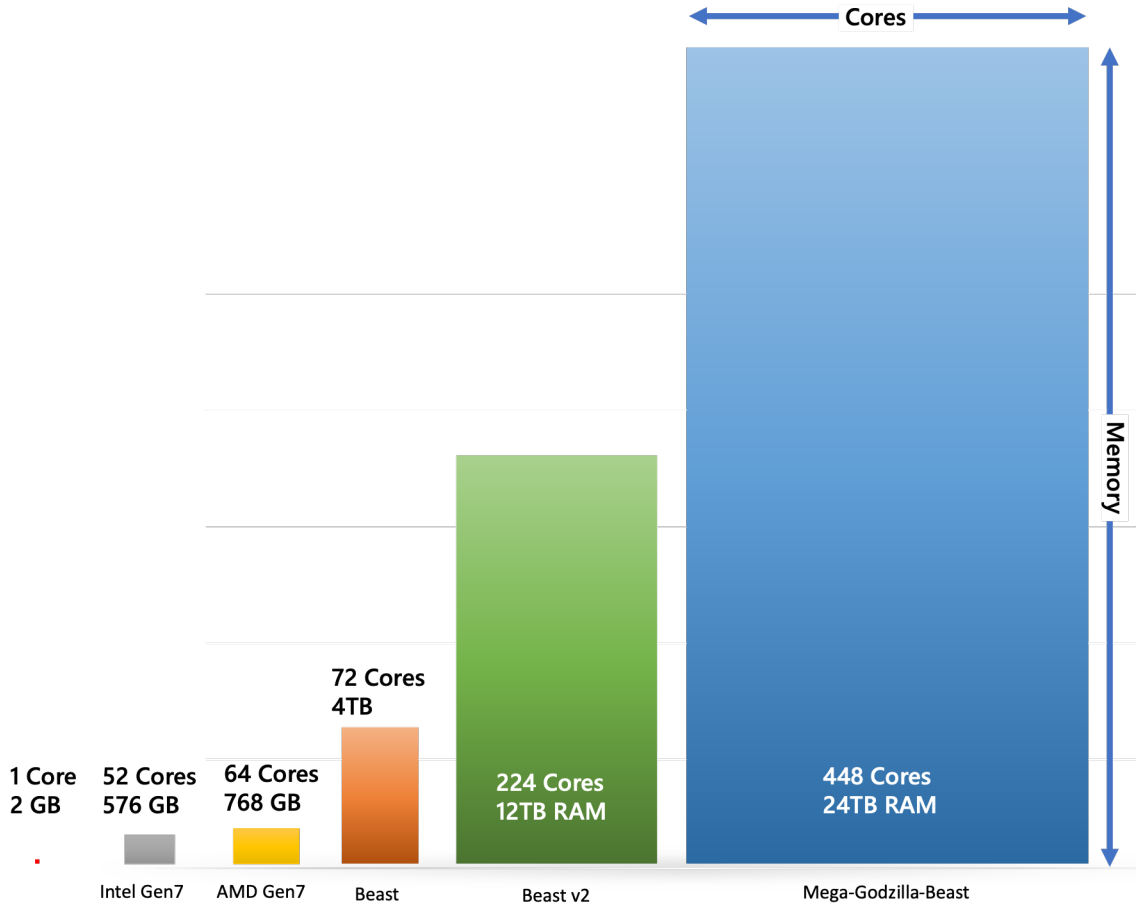


# Admission Control in Azure

Admission Control: Should a new request be accepted?

## Why is it hard?

- Variability affecting supply and demand
- **Hardware and VM type heterogeneity**



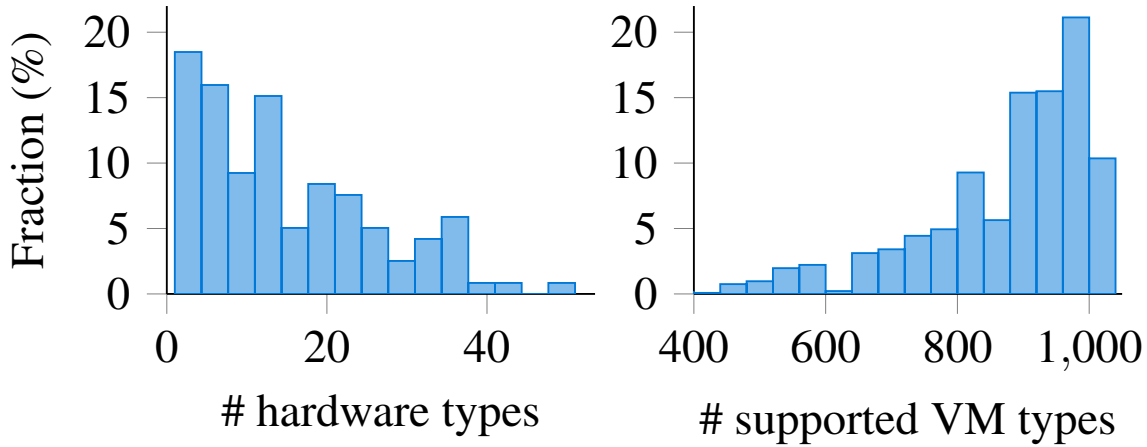
# Admission Control in Azure

Admission Control: Should a new request be accepted?

## Why is it hard?

- Variability affecting supply and demand
- **Hardware and VM type heterogeneity**

  → **fragmentation**



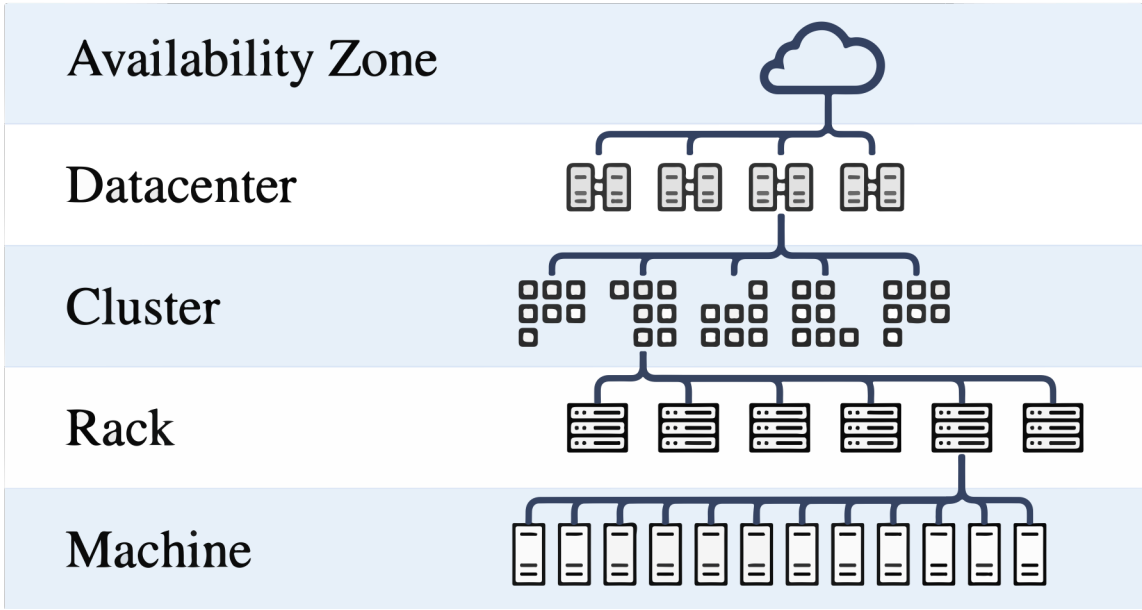
# Admission Control in Azure

Admission Control: Should a new request be accepted?

## Why is it hard?

- Variability affecting supply and demand
- Hardware and VM type heterogeneity

  → fragmentation
- **Placement constraints**



# Admission Control in Azure

Admission Control: Should a new request be accepted?

Solution→ **Kerveros**: Cloud admission control at scale

## Why is it hard?

- Variability affecting supply and demand
- Hardware and VM type heterogeneity

  → fragmentation
- Placement constraints

## Goals

- Fast and Scalable
  - Throughput = 120,000+ requests/minute[1]
  - Avg. Latency = 5 – 10 ms
- Resource Efficient
  - 1% efficiency gain → $100+ M/year savings [1]

[1] Protean, OSDI '20

# Kerveros

## Main Idea:

Late Binding of Reserved Capacity for Admission Control

### Why Late Binding?

- High packing efficiency
- Accurate accounting
  - Tracks across different VM types
- Flexible packing with low overhead
- Fast admission decision
- Unclaimed reserved resources reused as preemptable VMs (e.g., spot VMs)

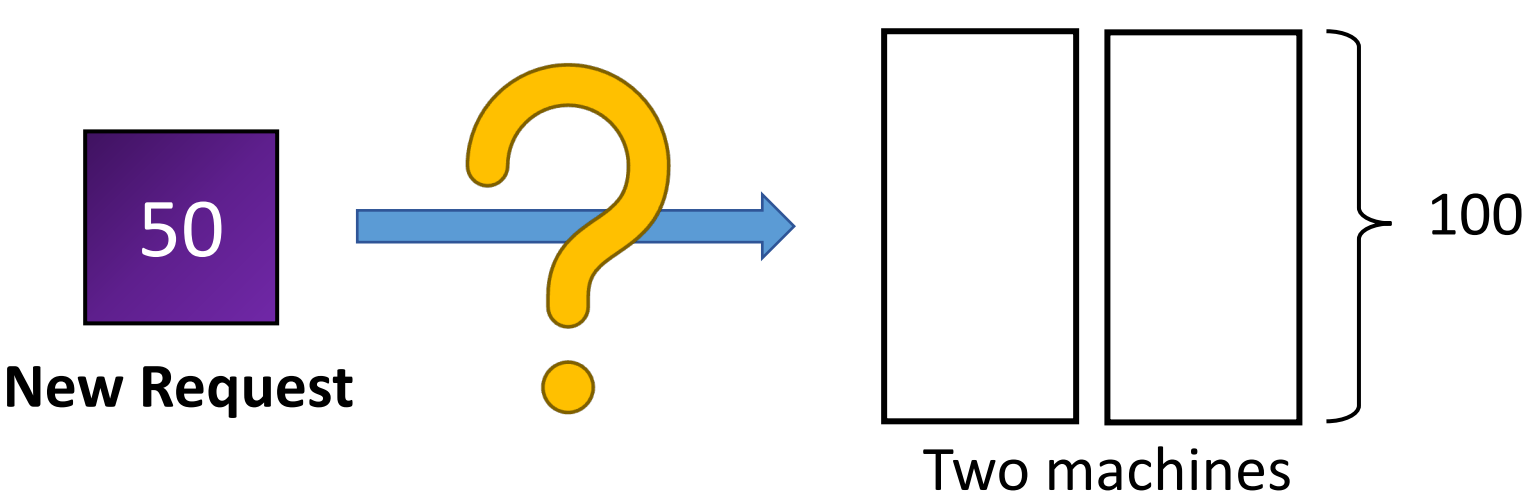
  → maximize ROI

# Challenges with Late Binding

Accept Request?

**“Available Capacity” ≥ New Request**

200-120 = 80 ≥ 50




**System Capacity:**

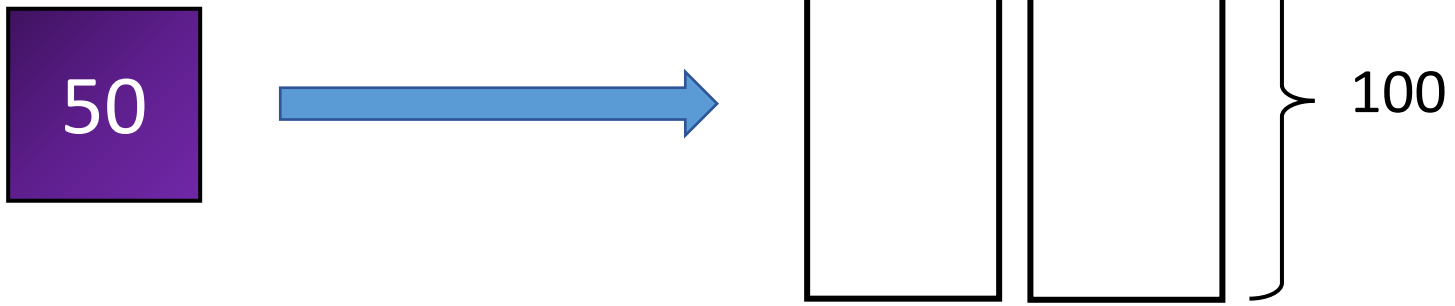
2 x 100 = 200

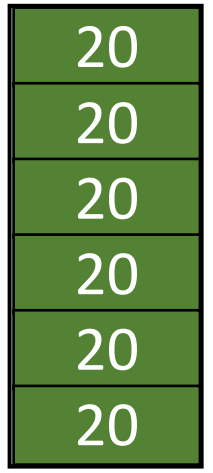
# Challenges with Late Binding

Accept Request?

**“Available Capacity” ≥ New Request**

200-120 = 80 ≥ 50




**Small VMs**

[120]

**System Capacity:**

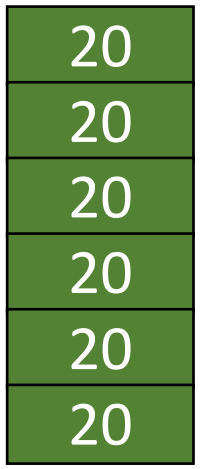
2 x 100 = 200

# Challenges with Late Binding

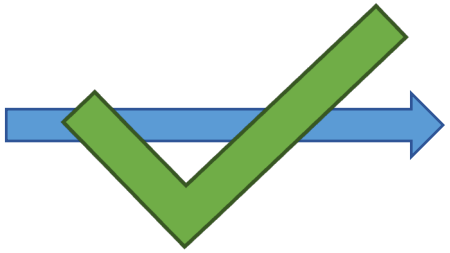
Accept Request?

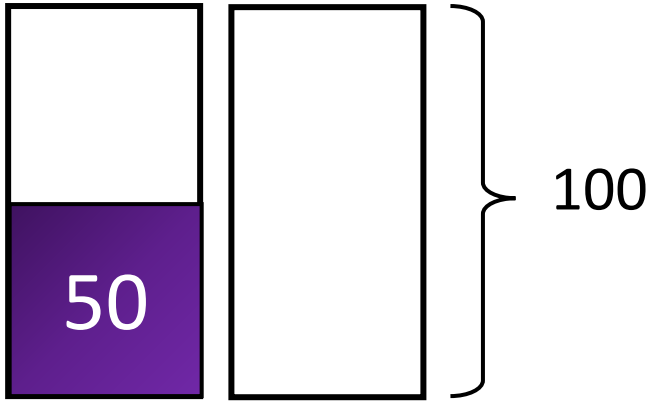
**“Available Capacity” ≥ New Request**

200-120 = 80 ≥ 50




**Claiming Small VM Reservations**






Two machines

**System Capacity:**

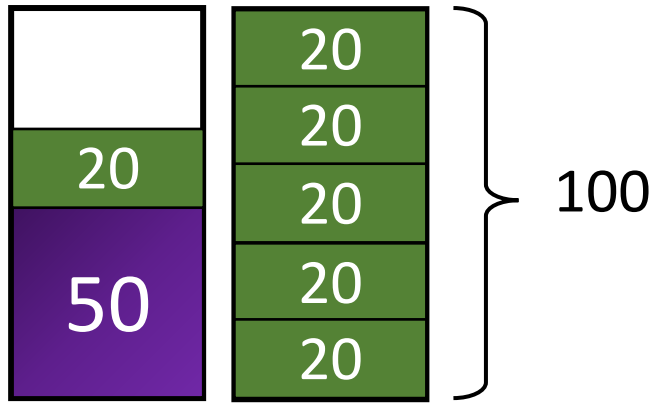
2 x 100 = 200

# Challenges with Late Binding

Accept Request?

**“Available Capacity” ≥ New Request**

200-120 = 80 ≥ 50




Two machines

**System Capacity:**

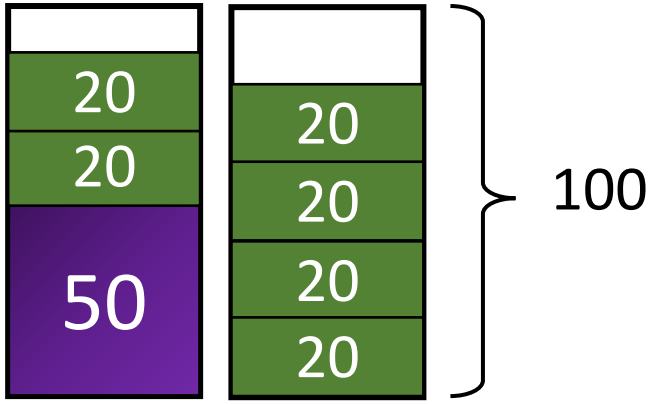
2 x 100 = 200

# Challenges with Late Binding

Accept Request?

**“Available Capacity” ≥ New Request**

200-120 = 80 ≥ 50 




Two machines

**System Capacity:**

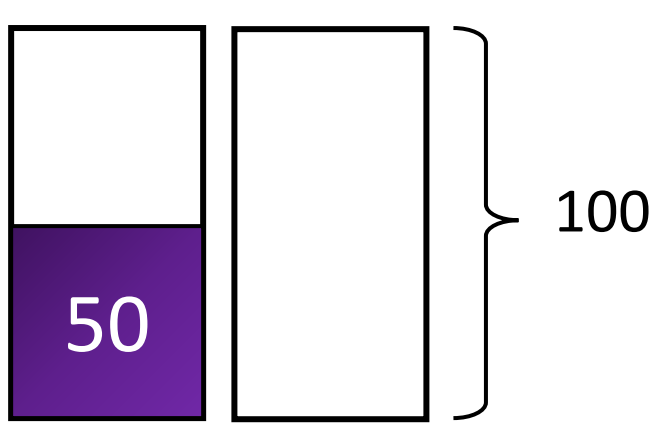
2 x 100 = 200

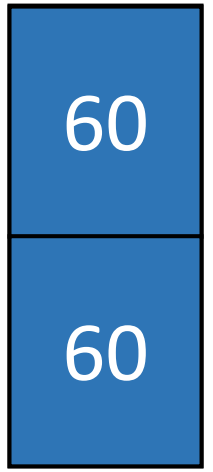
# Challenges with Late Binding

Accept Request?

**“Available Capacity” ≥ New Request**

200-120 = 80 ≥ 50




**Large VMs** [120]

**System Capacity:**

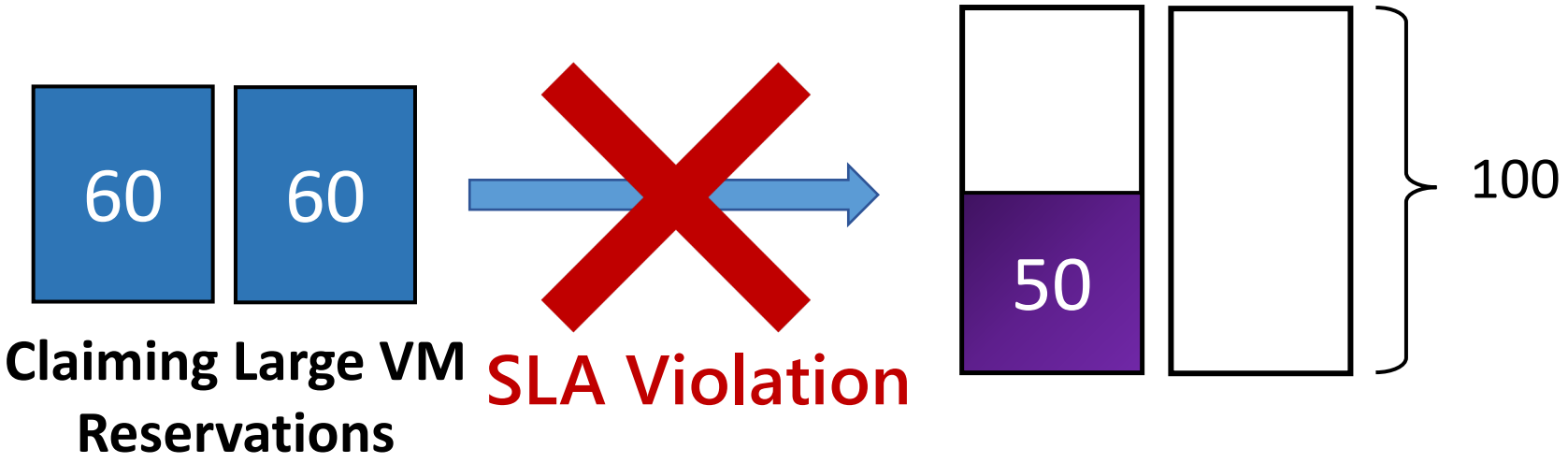
2 x 100 = 200

# Challenges with Late Binding

Accept Request?

**“Available Capacity” ≥ New Request**

200-120 = 80 ≥ 50




Admission Control depends on shape (i.e., VM type) of the reserved capacity

Solution: Allocable VM (**AV**)

# Allocable VM (AV)

- Novel bookkeeping of available capacity
  - For every VM type, count of additional VMs that can fit

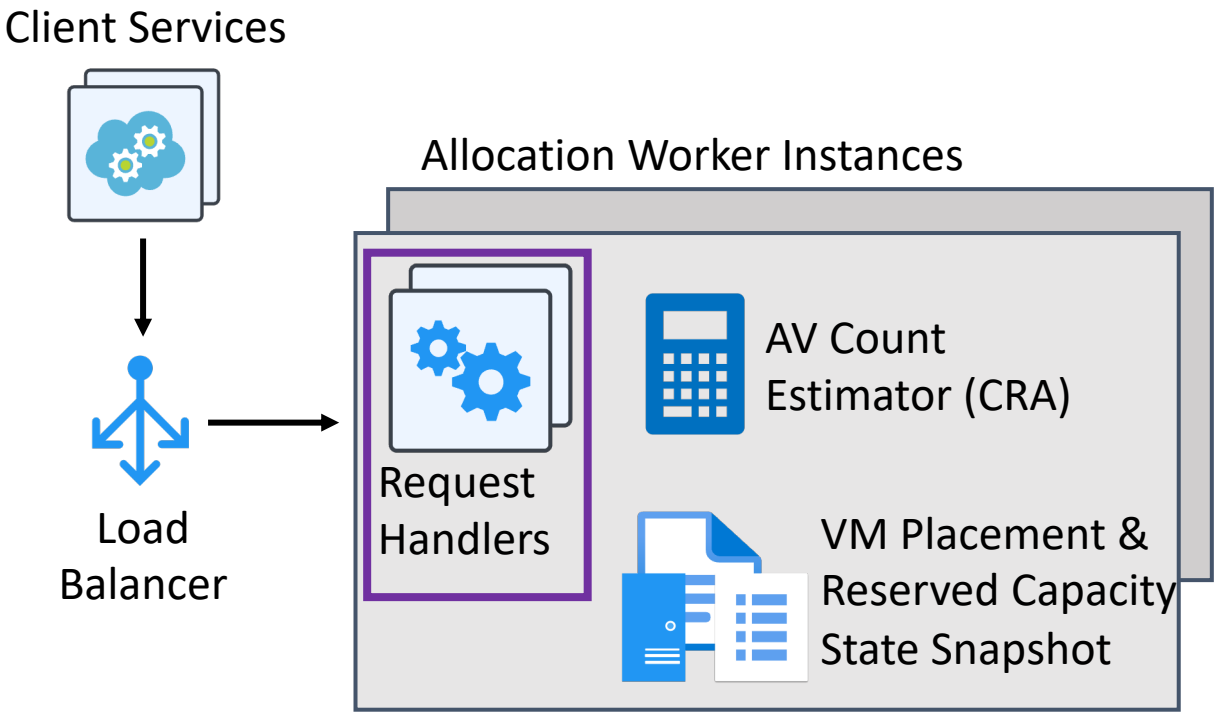
| VM Type | AV count |
|---|---|
| S | 27408 |
| M | 6724 |
| L | 1588 |

# Allocable VM (AV)

- Novel bookkeeping of available capacity
  - For every VM type, count of additional VMs that can fit
- Converts multi-dimensional demand to a single-dimension
- Develop two algorithms to adjust AV count for reserved capacity
  - Conversion Ratio Algorithm (CRA)
  - Linear Adjustment Algorithm (LAA)

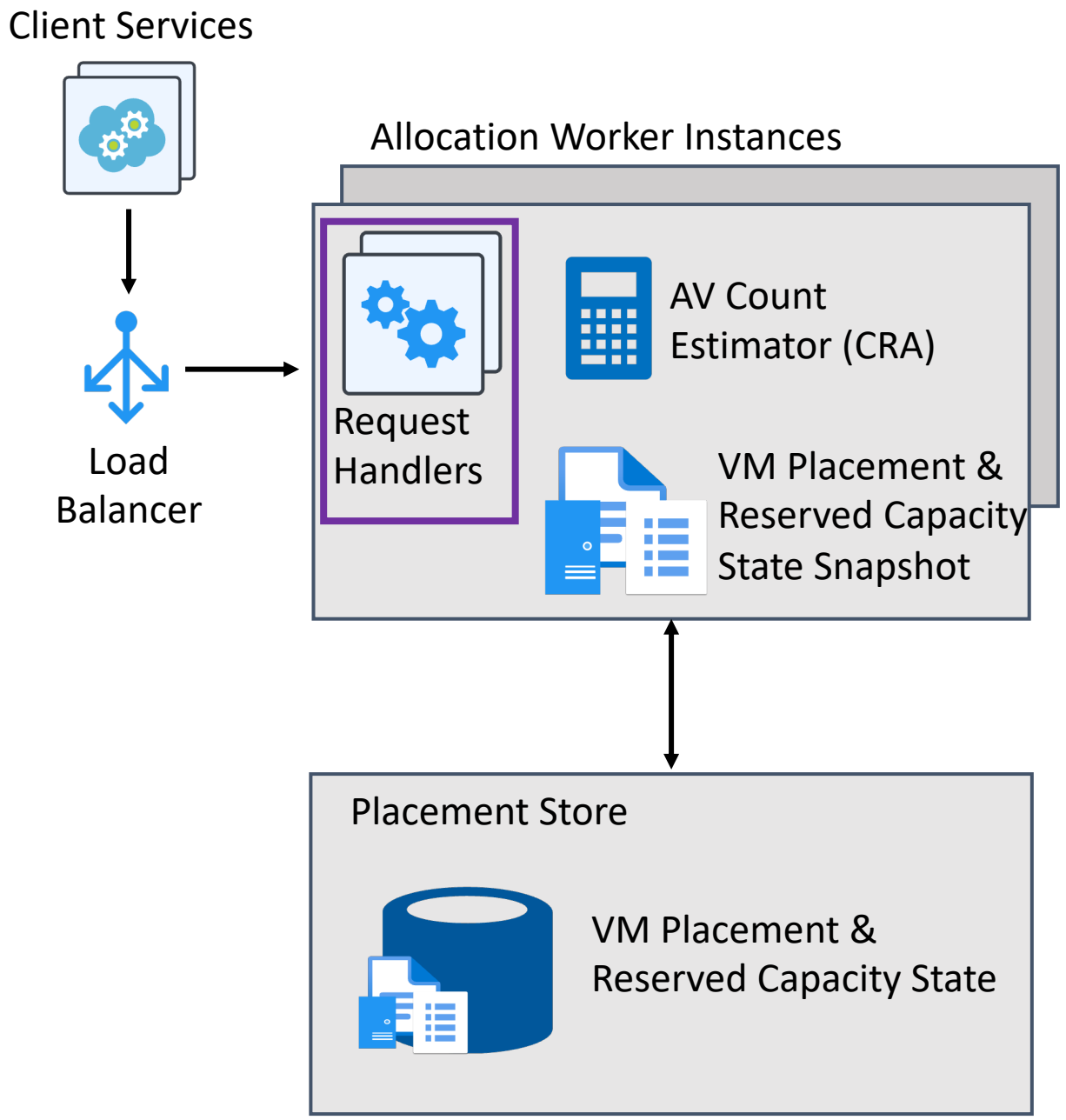
| VM Type | Multi-dimensional Resource demand | AV count |
|---|---|---|
| S | { CPU: 1, RAM: 2 GB, Disk: 64 GB, … } | 27408 |
| M | { CPU: 4, RAM: 8 GB, Disk: 256 GB, … } | 6724 |
| L | { CPU: 16, RAM: 32 GB, Disk: 1024 GB, … } | 1588 |

# Kerveros In Action




- Zonal admission control
- Considers all reserved capacity in zone
- Handles both VM and reservation requests

# Kerveros In Action




- Zonal admission control
- Considers all reserved capacity in zone
- Handles both VM and reservation requests

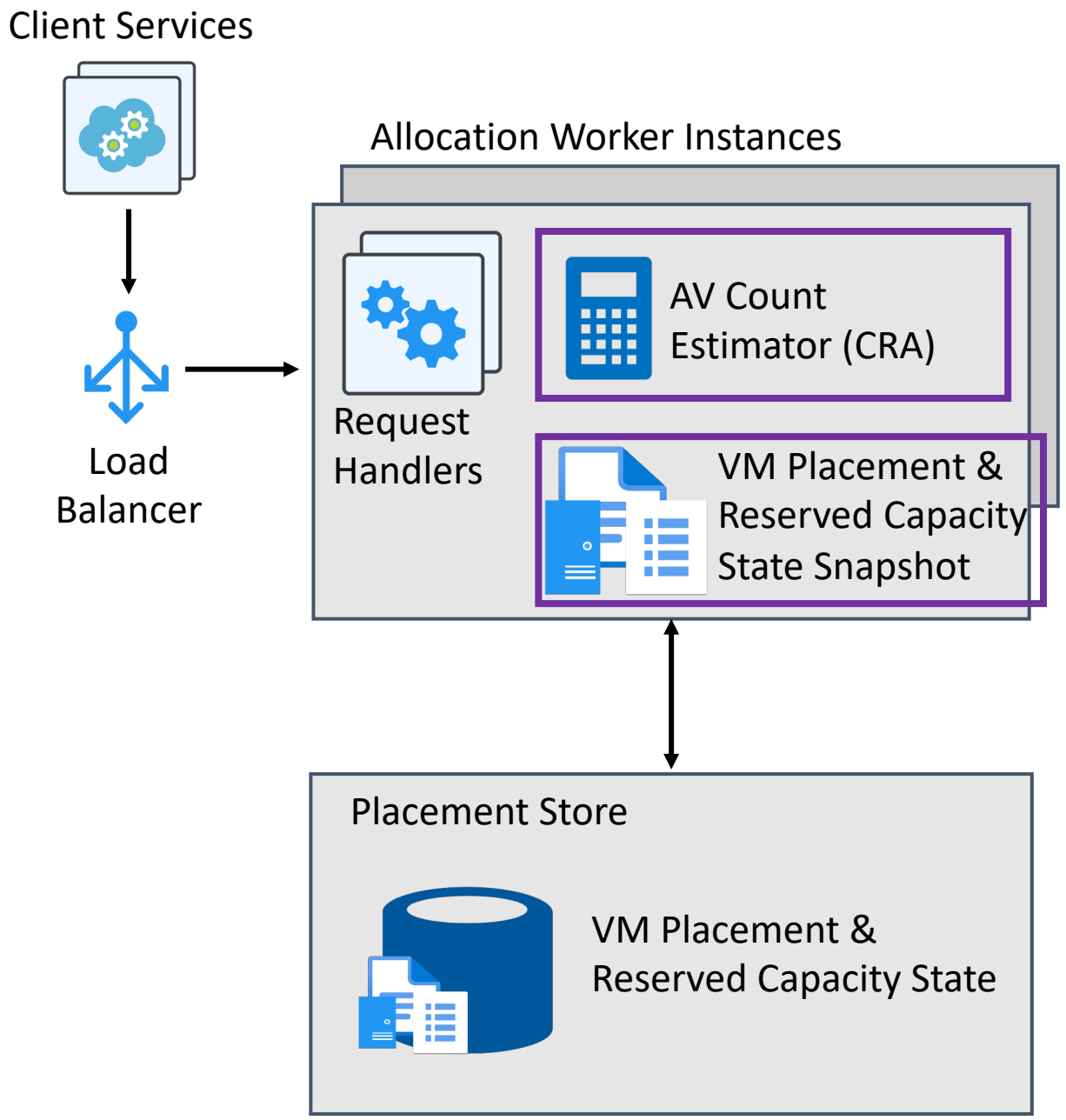
**Request Handler Process**

- Request arrives → check AV count
- If enough AV in system, Accept
  - Update VM placement & reserved capacity state
- Else Reject

**How do we get it?**

| VM Type | AV Count |
|---|---|
| S | $AV_S$ |
| M | $AV_M$ |
| L | $AV_L$ |

# Kerveros In Action




## AV Count Estimation

- Initialize AV count in zone
  - Uses in-memory state snapshot
  - Counted independently for each VM type
- Subtracts AV count for reserved capacity
  - Convert between VM types

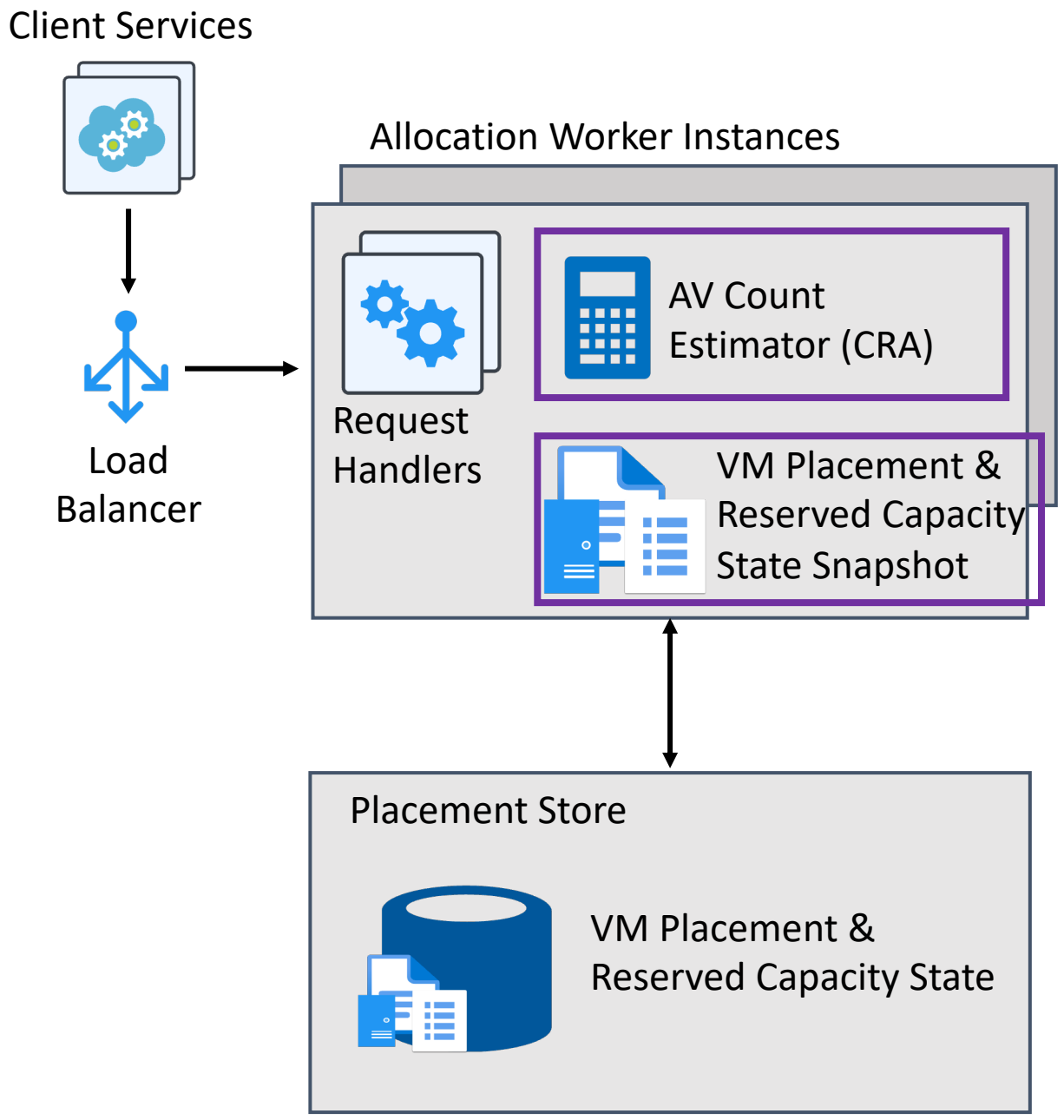
## Conversion Ratio Algorithm (**CRA**)

- Converts AV count between VM types
- Handles multi-dimensional conversion
- Frequent AV count estimation: 1 minute

**How do we get it?**

| VM Type | AV Count |
|---|---|
| S | $AV_S$ |
| M | $AV_M$ |
| L | $AV_L$ |

# Kerveros In Action




**AV Count Estimation**

- Initialize AV count in zone
  - Uses in-memory state snapshot
  - Counted independently for each VM type
- Subtracts AV count for reserved capacity
  - Convert between VM types

Conversion Ratio Algorithm (**CRA**)

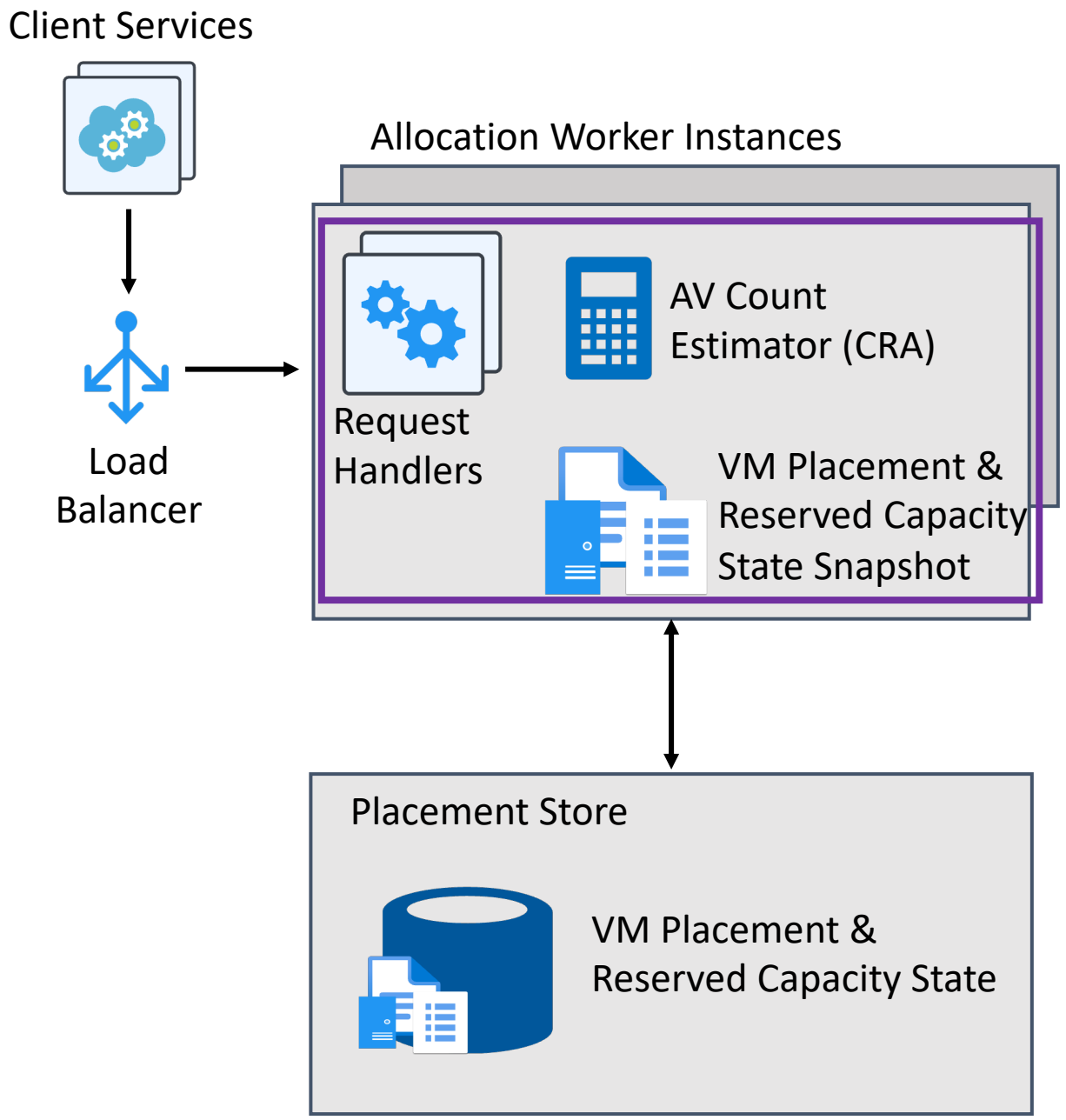
- Converts AV count between VM types
- Handles multi-dimensional conversion
- Frequent AV count estimation: 1 minute

Fast and Scalable

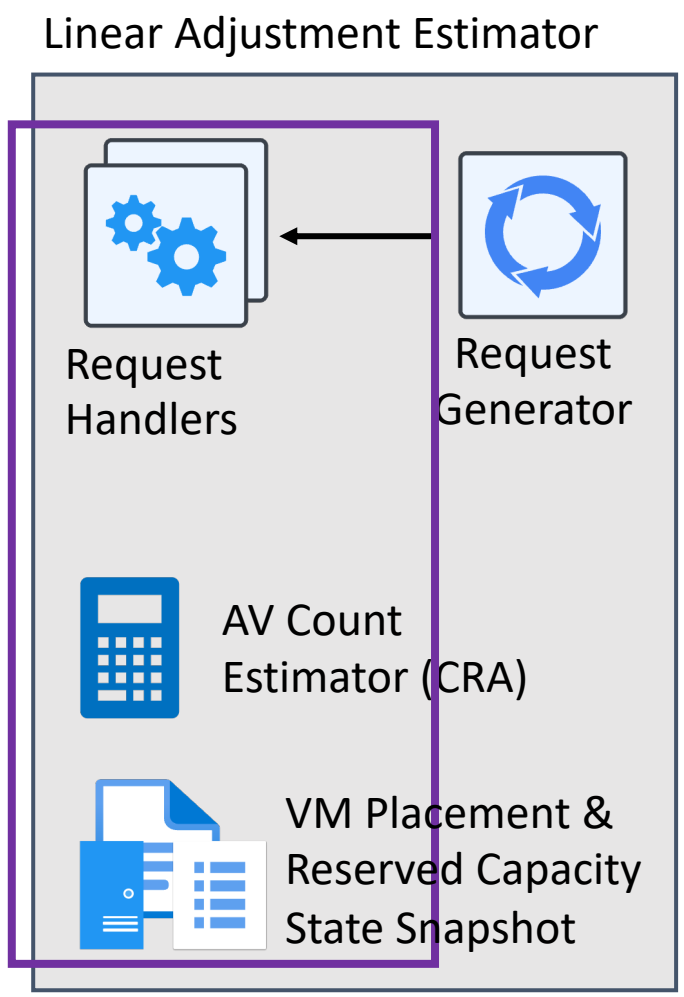
Rounding Errors → Fragmentation

Conservative Estimation

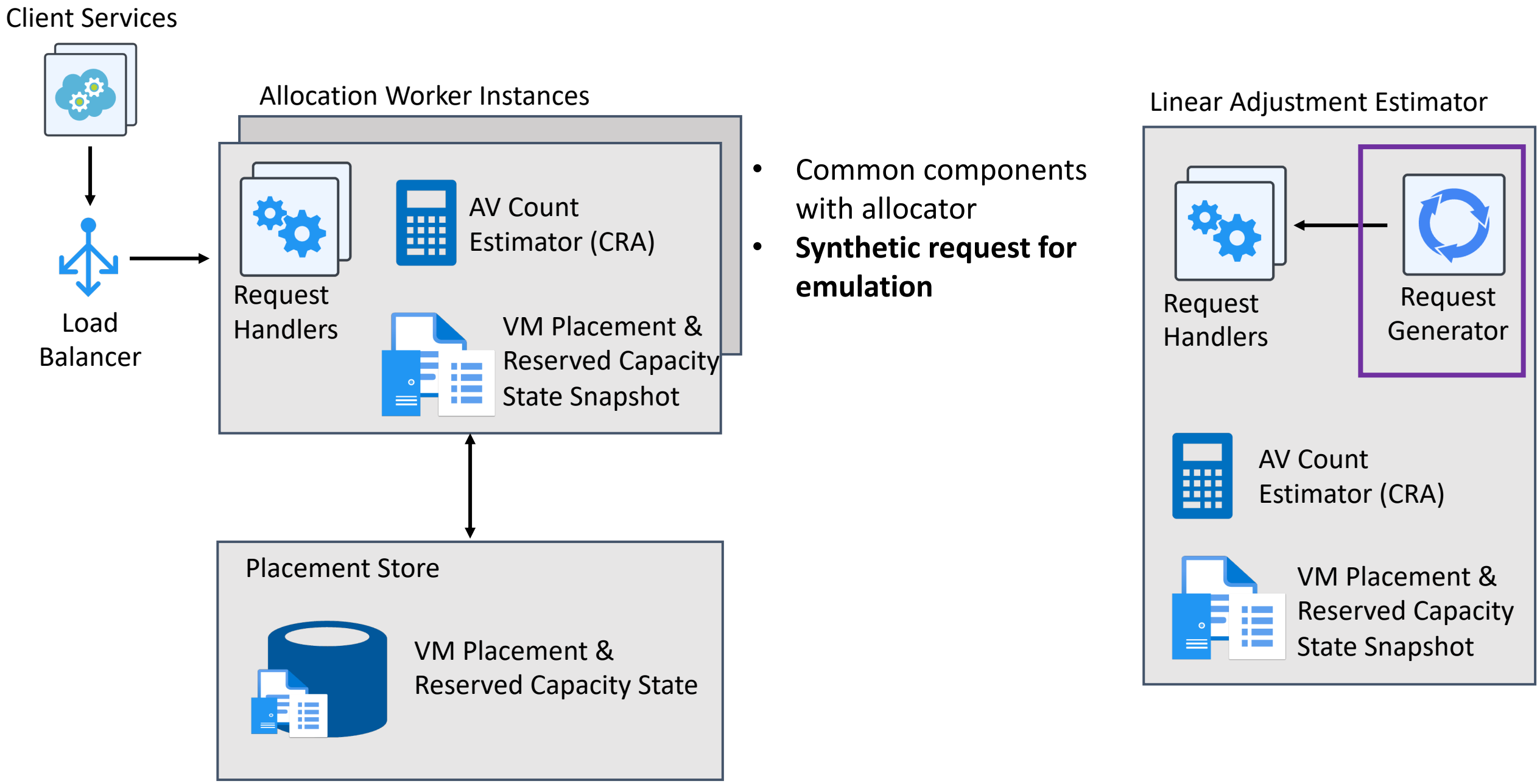
# Kerveros In Action



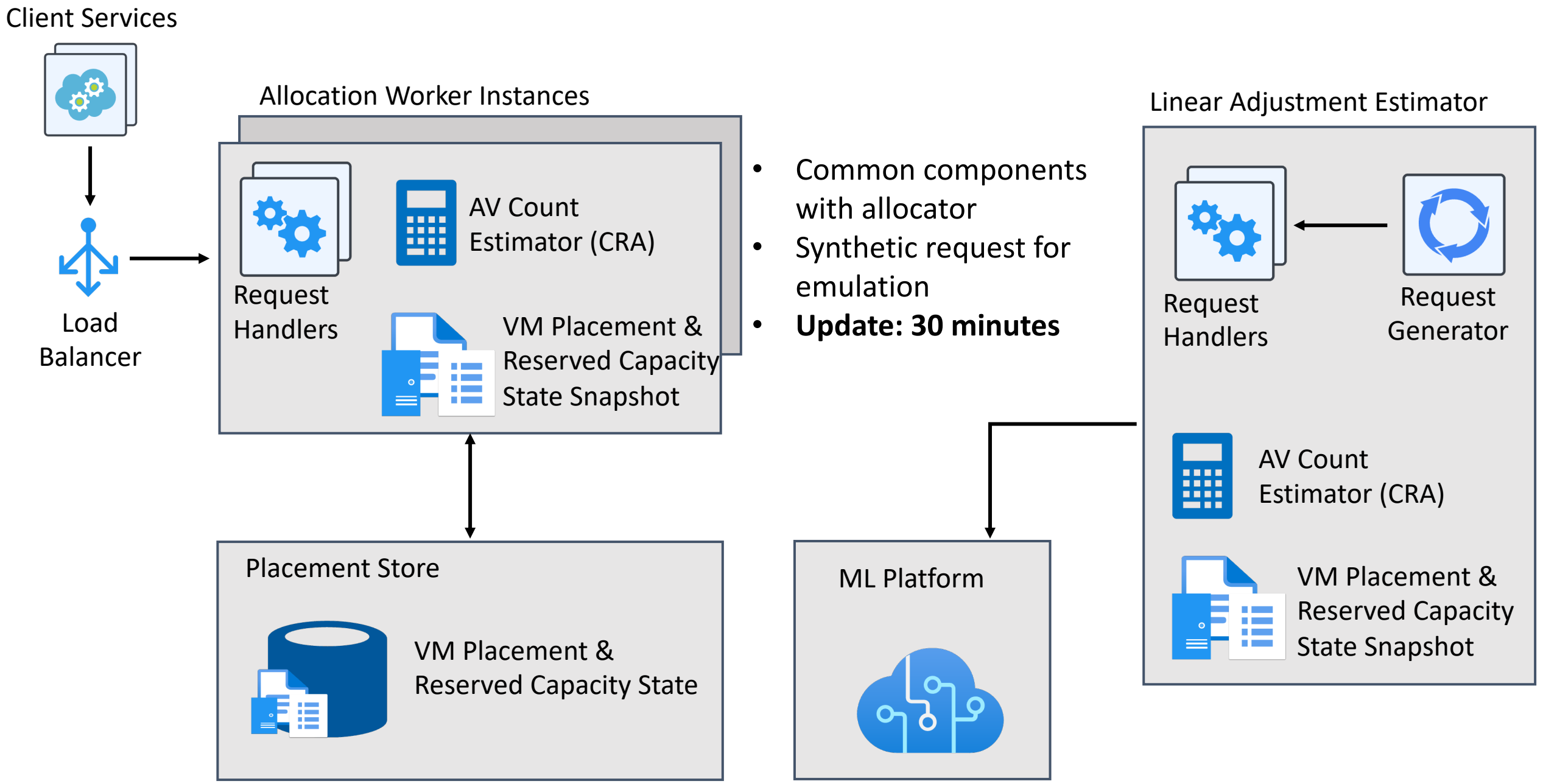

- **Common components with allocator**



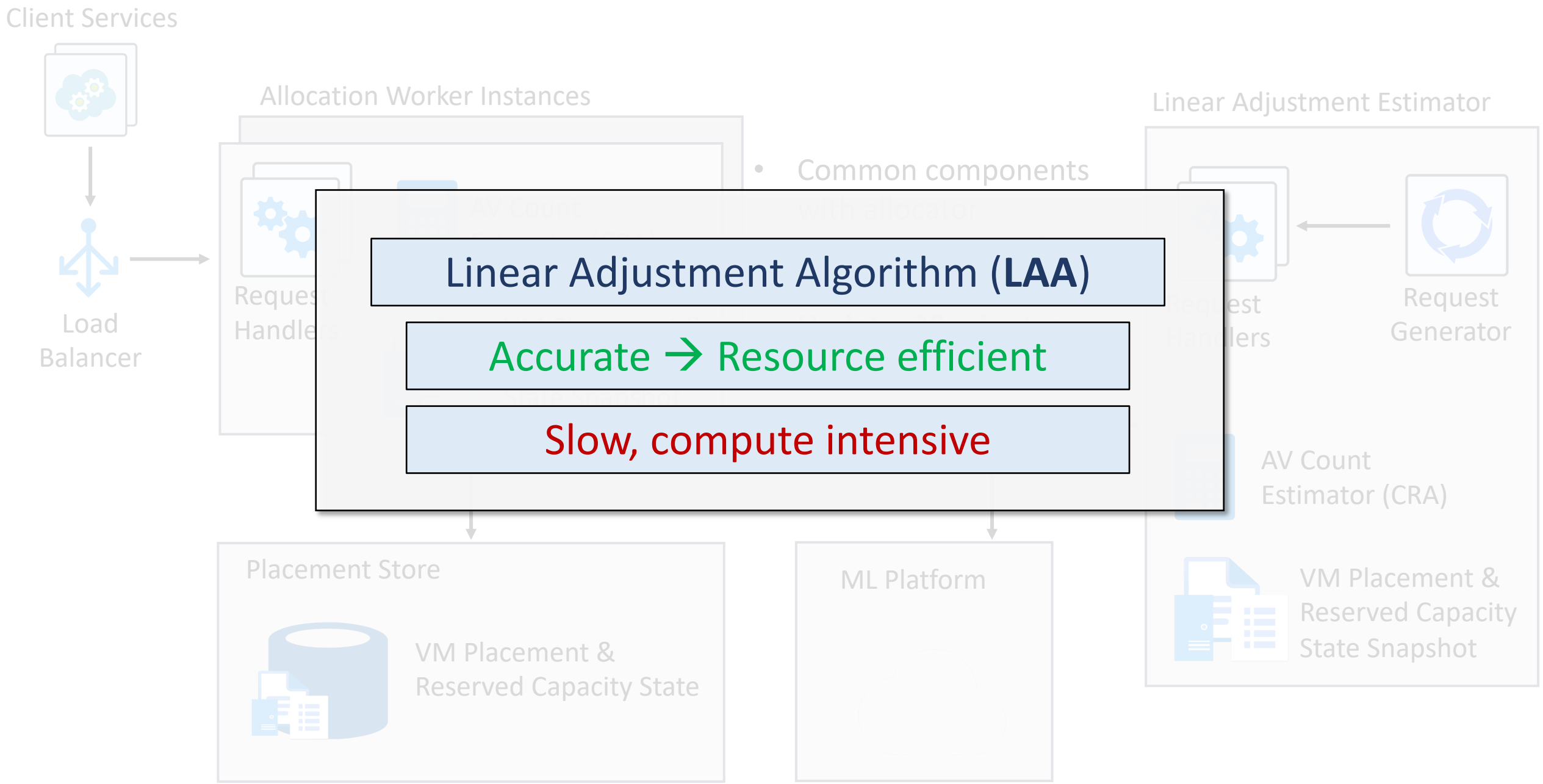
# Kerveros In Action



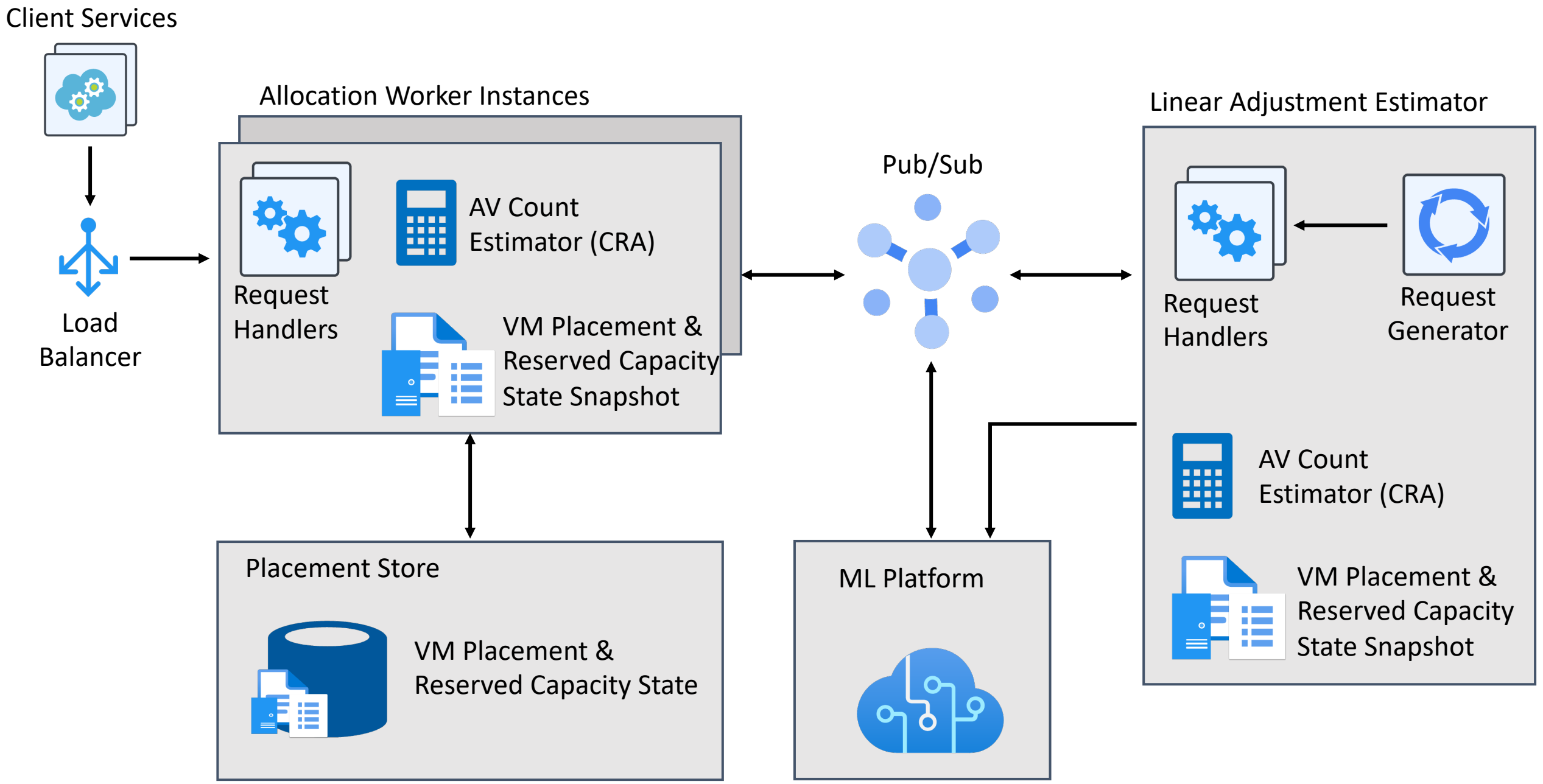
# Kerveros In Action



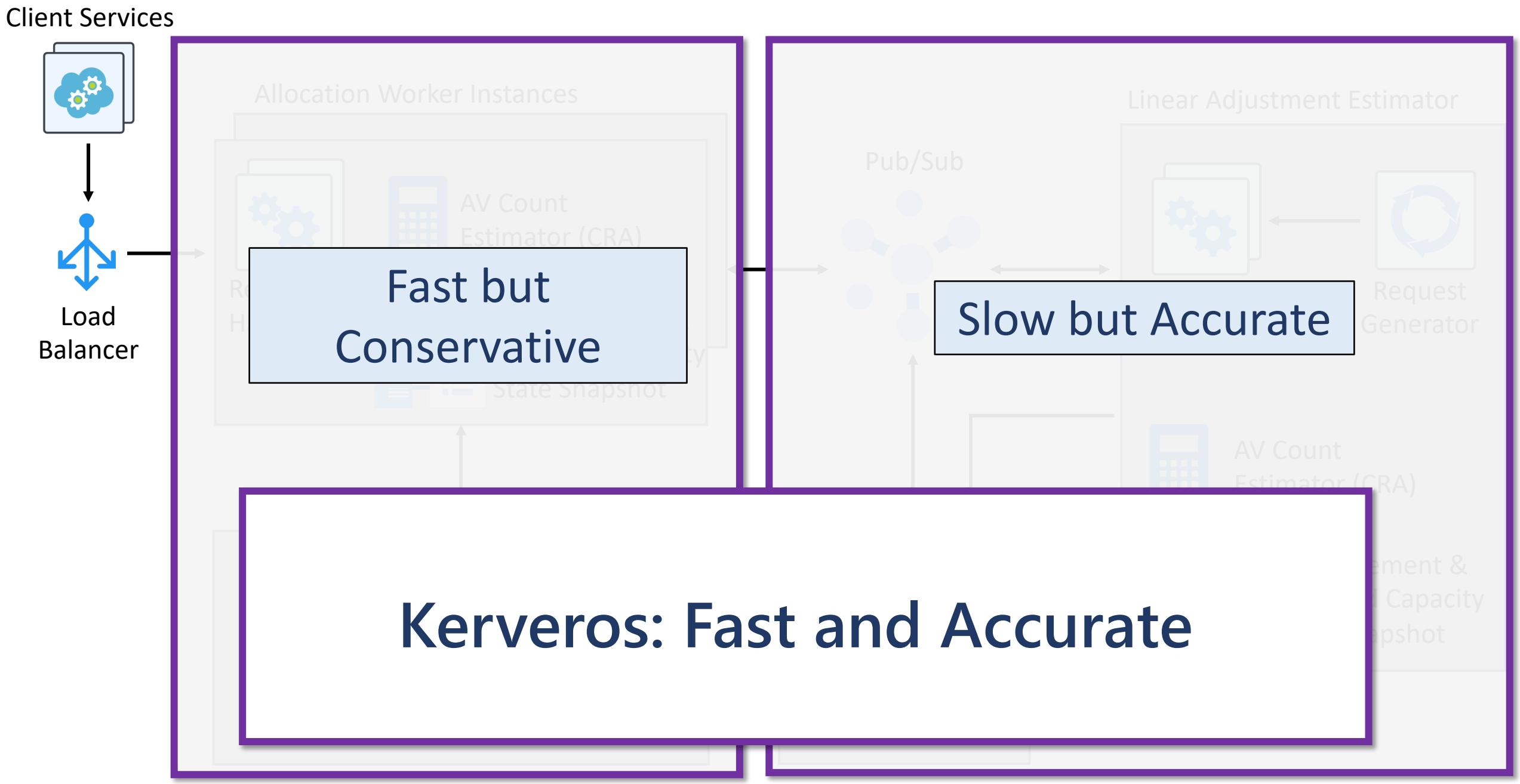
# Kerveros In Action



# Kerveros In Action



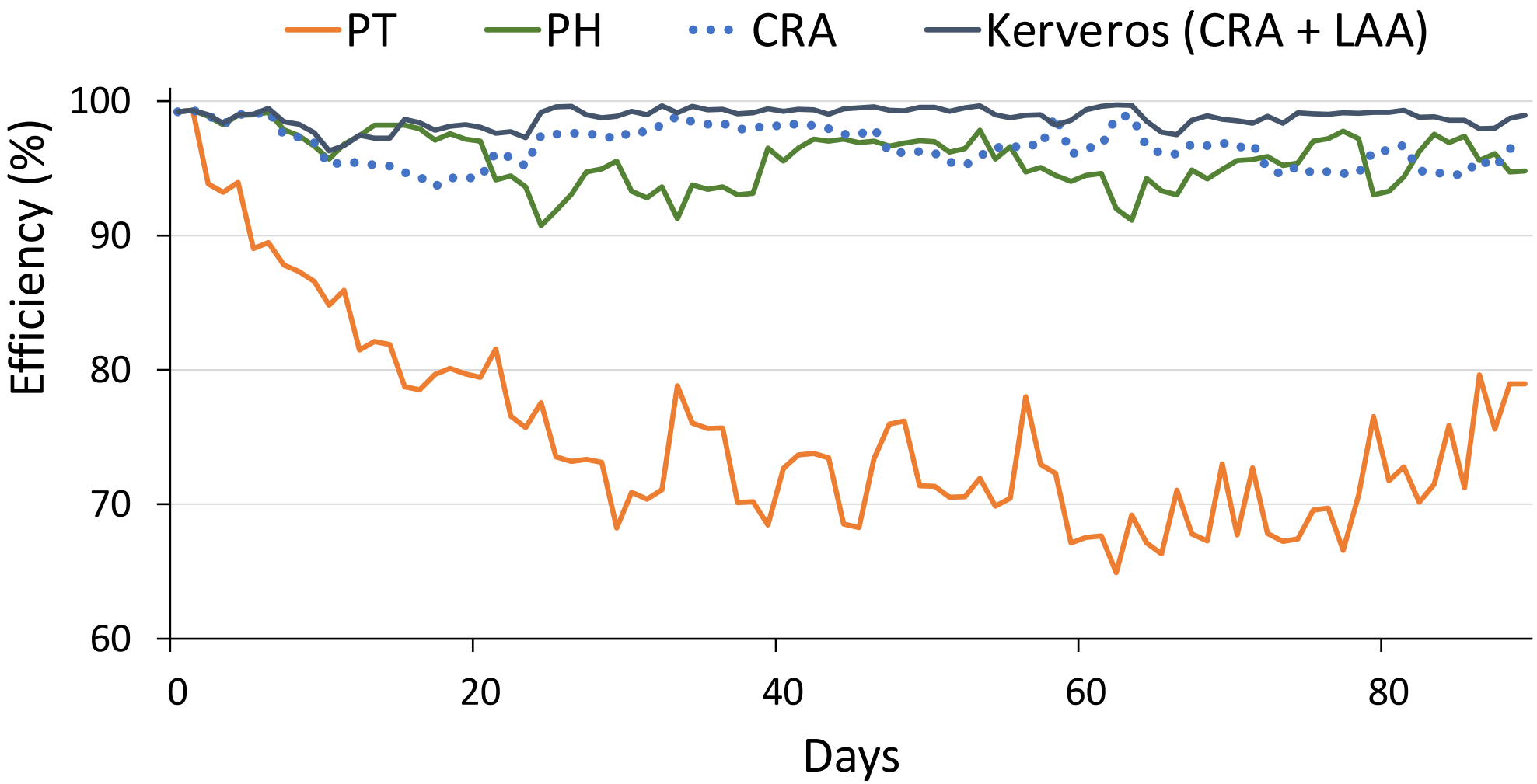
# Kerveros In Action



# Alternate Solutions

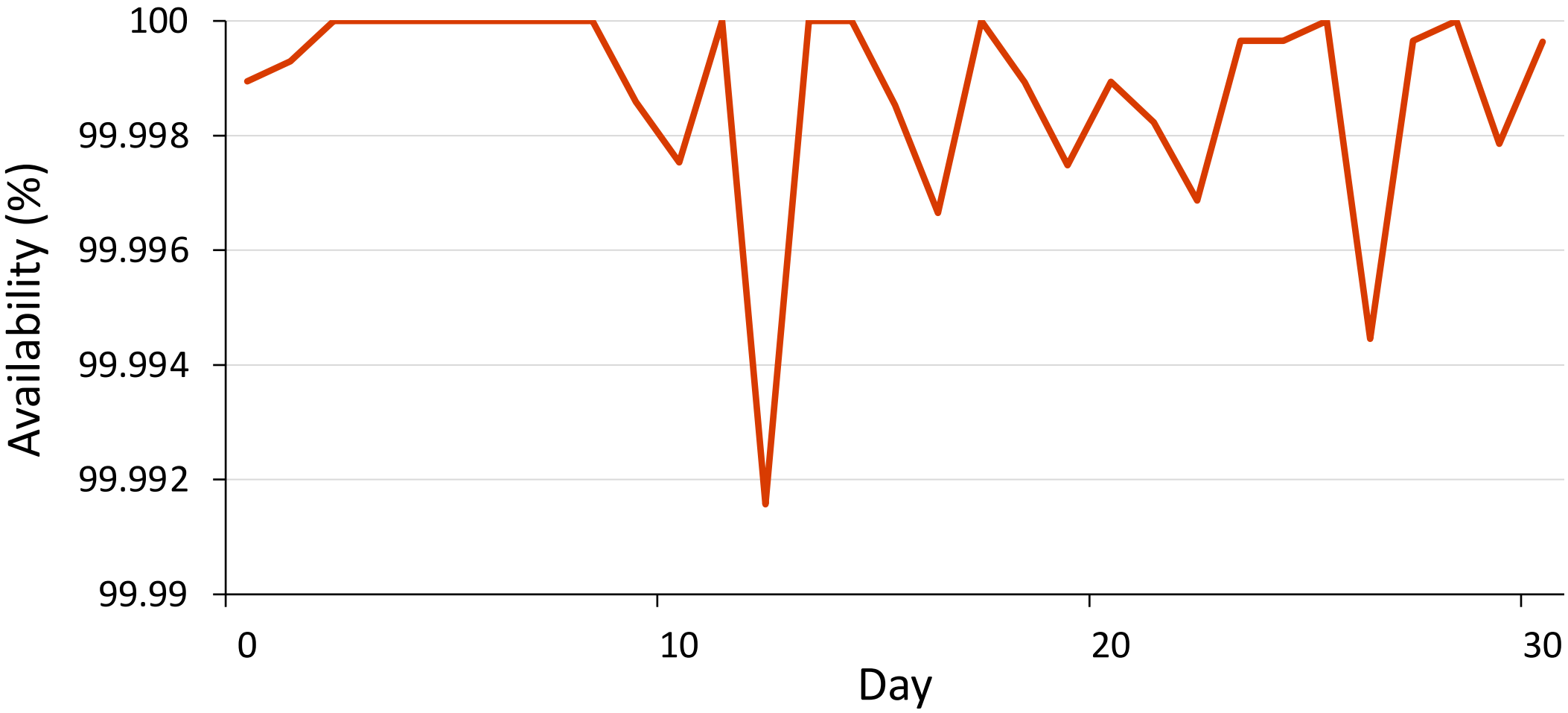
- Partition (PT)[SOSP ‘21]
  - Approach: Reserve capacity by partitioning machines
  - Pro: Greater control over resources and isolation → Works on private cloud
  - Con: Fragmentation with high heterogeneity → Wastes resources in public cloud
- Placeholder (PH)
  - Approach: Allocate and reserve resources for reservations
  - Pro: Simple and Guarantees SLA
  - Con: Early binding to allocated resources → Low packing efficiency

# How Resource Efficient is Kerveros?



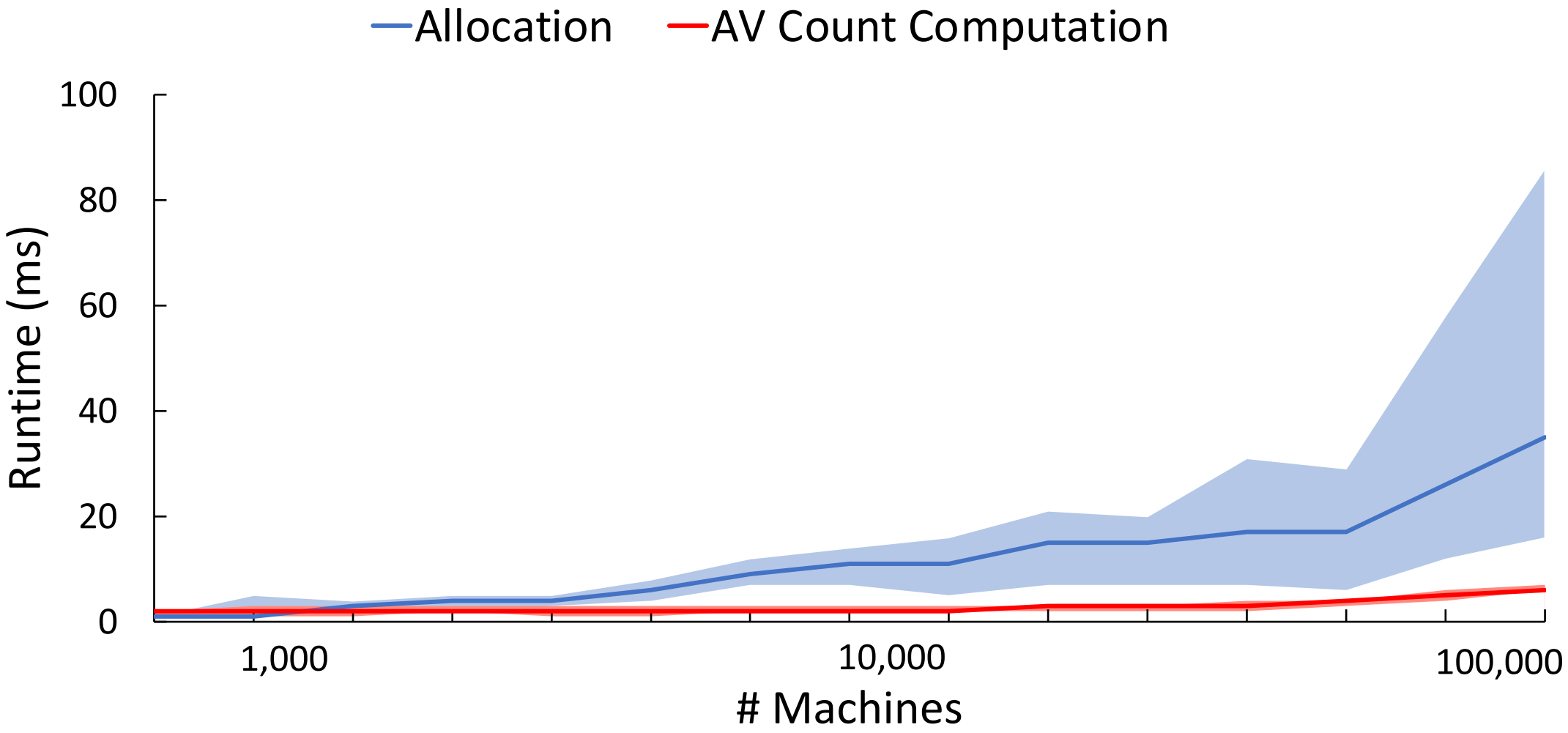

Kerveros ensures high resource utilization

# How does Kerveros Deal with Failures?




Kerveros achieves consistent fours 9s of availability

# How Scalable is Kerveros?




Kerveros scales well with inventory size

# Conclusion

- Kerveros : Admission control system in Microsoft Azure
  - Variable supply and demand
  - Hardware and VM type heterogeneity
- Scalable and resource efficient in cloud scale
- Achieves high resource utilization while maintaining SLA
  - Late binding of reserved resources for admission control
  - Allocable VM (AV)

Microsoft Research

Microsoft Azure

PennState