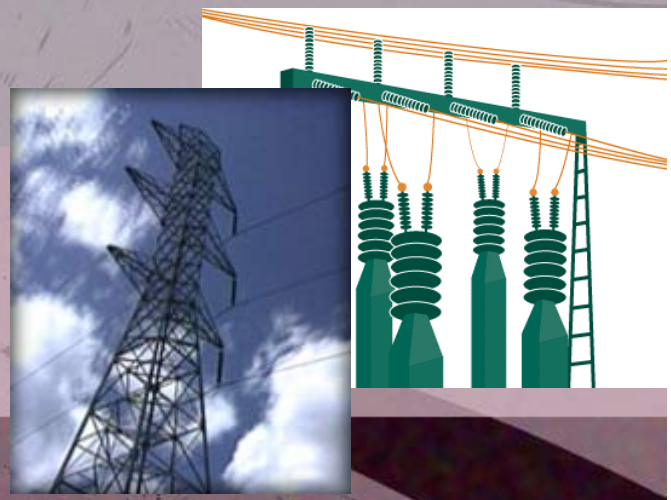
Cluster 2003
IEEE International Conference on Cluster Computing

Impact of Computational Resource Reservation to the Communication Performance in the Hypercluster Environment

Jim K.W.Tse and P.K Lun
Department of Electronic and Information Engineering
The Hong Kong Polytechnic University

Computational Grid

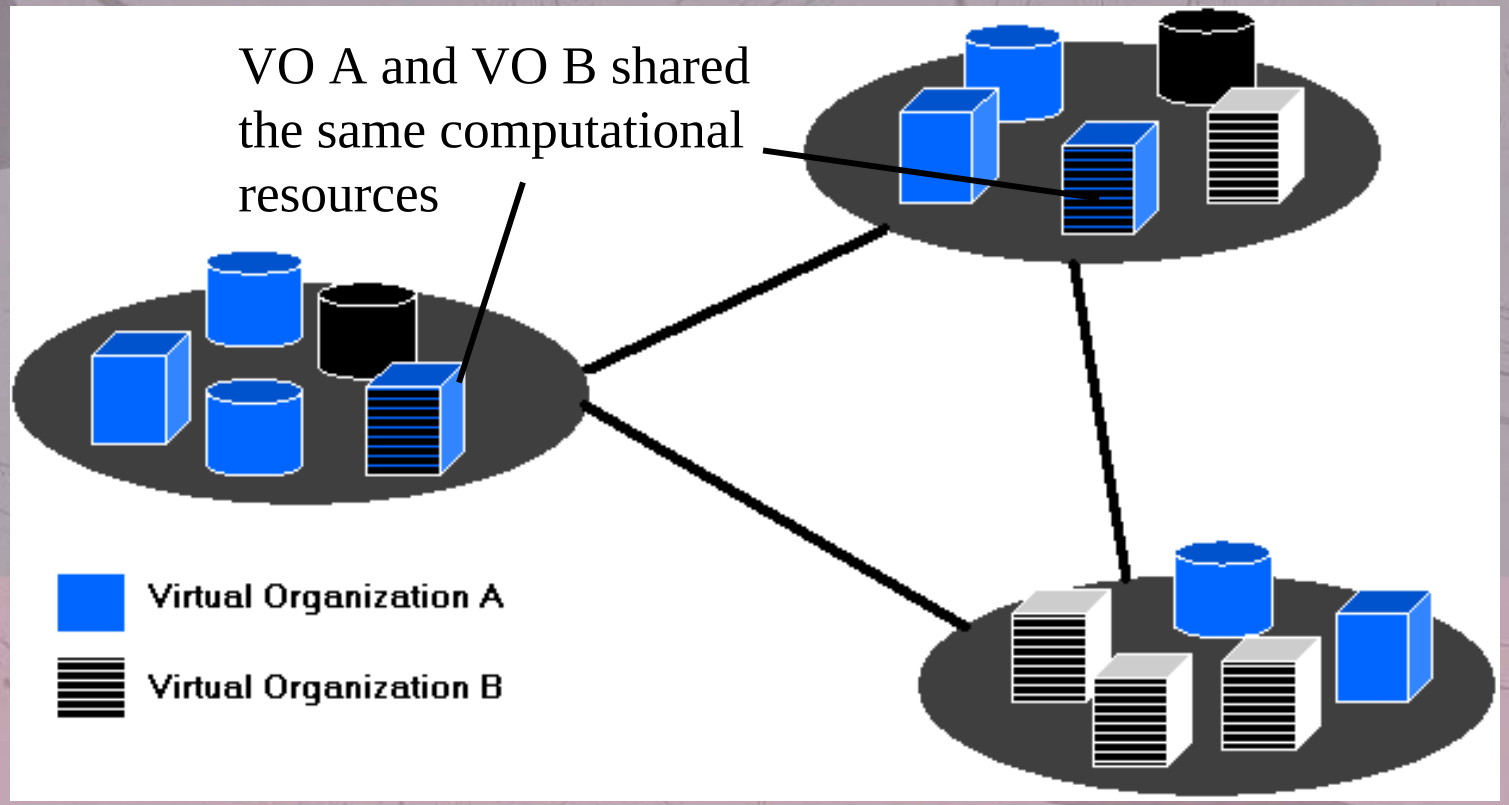
- Electric Grid
- Get computation power as easy as electricity (Plug and Play)
- Regardless of the details of the infrastructure



Grid Environment

- Aggregate computational resources from different administrative domain and disregards their geographic location
- Open environment, Multi-users, Multi-platform
- Dynamic, Unpredictable

Virtual Organizations



Building Hypercluster in Grid Environment

- Grid infrastructure enable Researchers, Scientists and Engineers build their own Hypercluster (Cluster of Cluster)
- Hypercluster inherits the dynamic behaviors, results in variable performance
- GRAB, DUROC
- “Some form of reservation will be ultimately required”

GARA

- Globus Architecture for Reservation and Allocation
- It provides a flexible architecture that can deliver quality of service for different types of resource

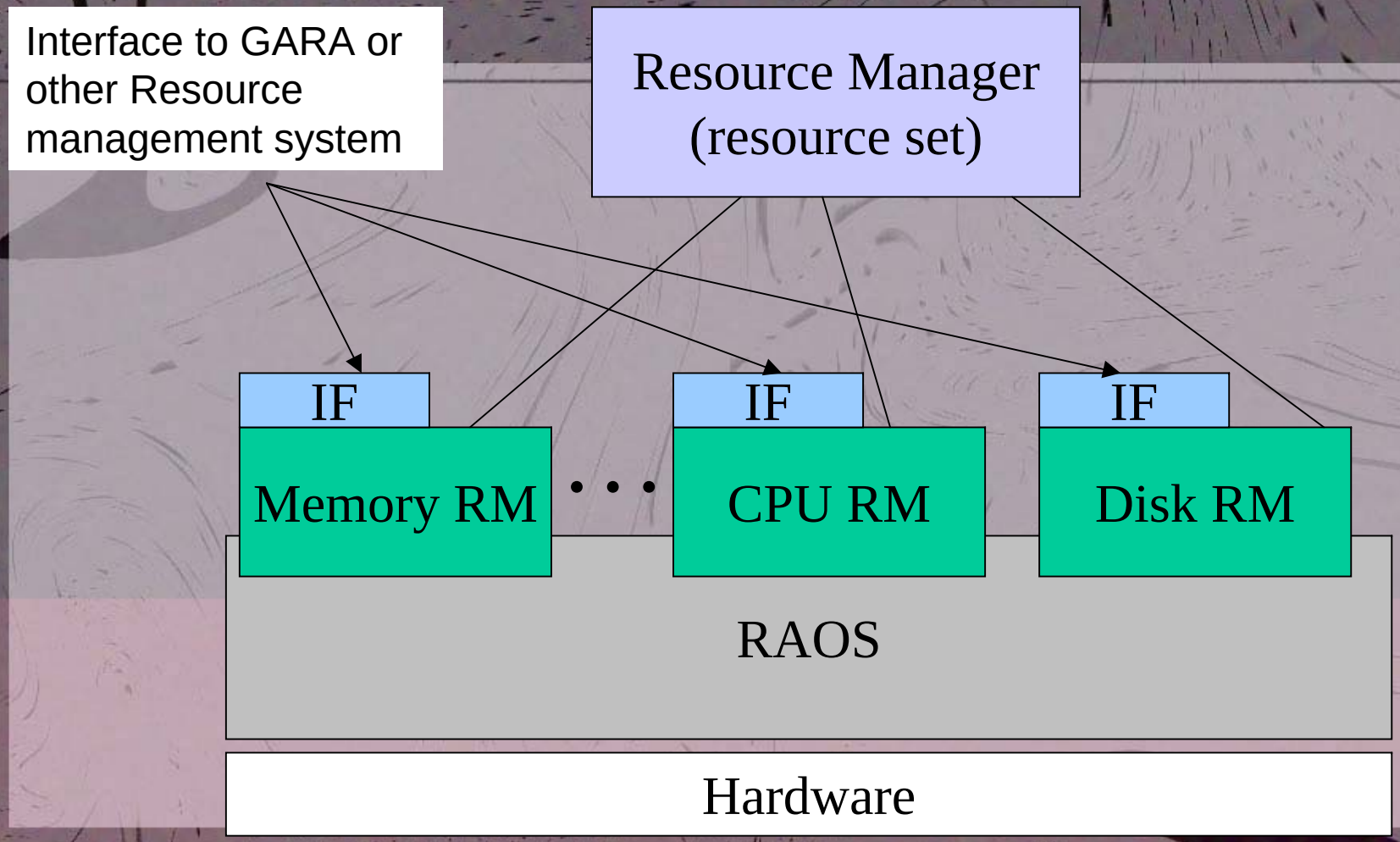
Limitation

- If the under resource provides advance reservation support GARA adopt it
- If not :
- (i) if GARA has total control over the resource, use “slot manager” to provide advance reservation
- (ii) Only provide probabilistic advance reservation

RAOS

- Reservation Aware Operating System
- Filling the gap between the Reservation Architecture and the Operating System
- Provides the foundation for the upper level scheduling abstract (e.g. GARA, Nimrod, etc.)

RAOS

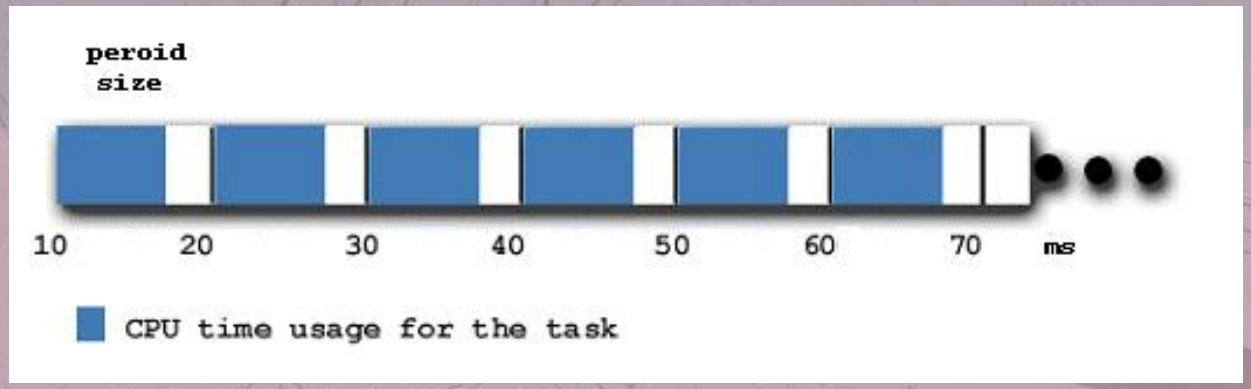

Interface to GARA or other Resource management system
Resource Manager (resource set)
IF
IF
IF
Memory RM
CPU RM
Disk RM
RAOS
Hardware

Design Philosophy

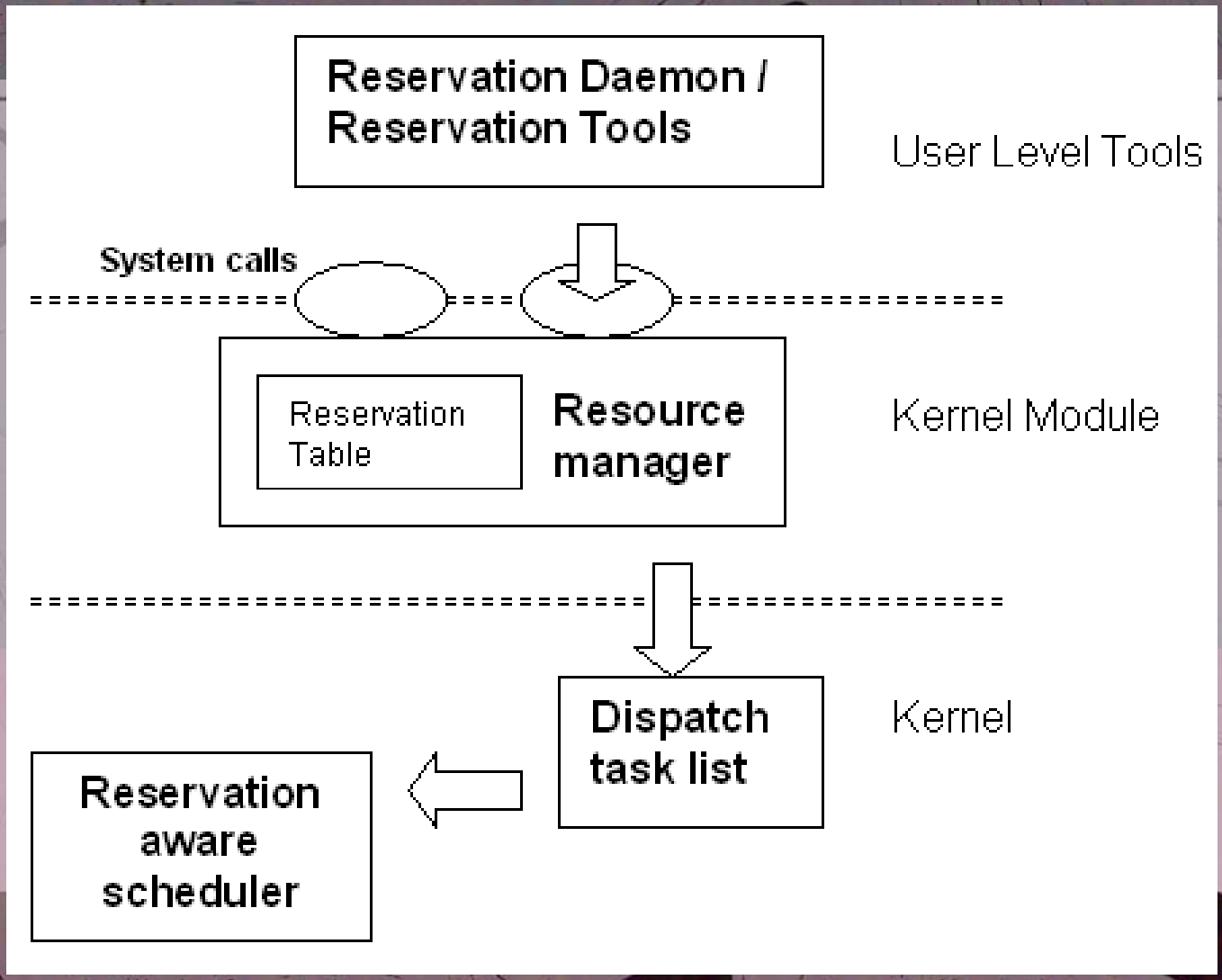
- Resource specific - Modified Kernel
  - Allocate and de-allocate resource
  - Quantify the resource
- General Reservation Module – Kernel module
  - Non resource specific
  - General reservation facilities
- Interface to upper level reservation system and scheduling system

Design Philosophy (cont')

- Quantify resource
  - Time (CPU bandwidth, Network bandwidth etc.)
  - Space (Disk space, Memory Space etc.)



Architecture of ROAS – CPU resource



How RAOS – CPU works ?

- Modified Linux Scheduler
- Resource management Kernel Module
- Reservation Tools and Interface

Capability

- Communication model:

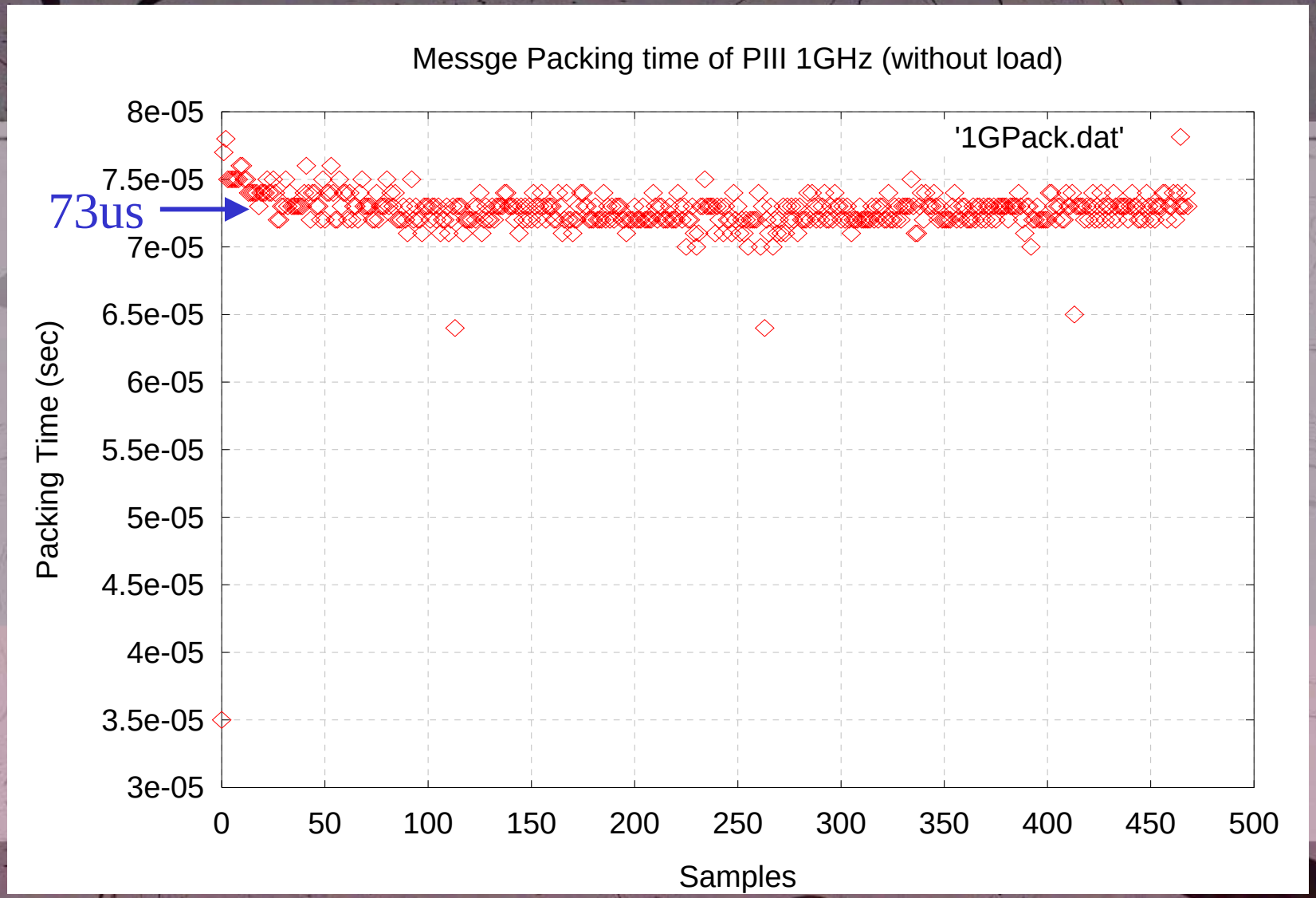
$$\xi + n\left[(\pi + \overline{\pi}) + \lambda + \frac{\delta}{\beta}\right]$$

ξ - Connection Setup Time
π - Message packing Time
$\overline{\pi}$ - Message unpacking Time
λ - Network delay
δ - Packet Size
β - Network Bandwidth

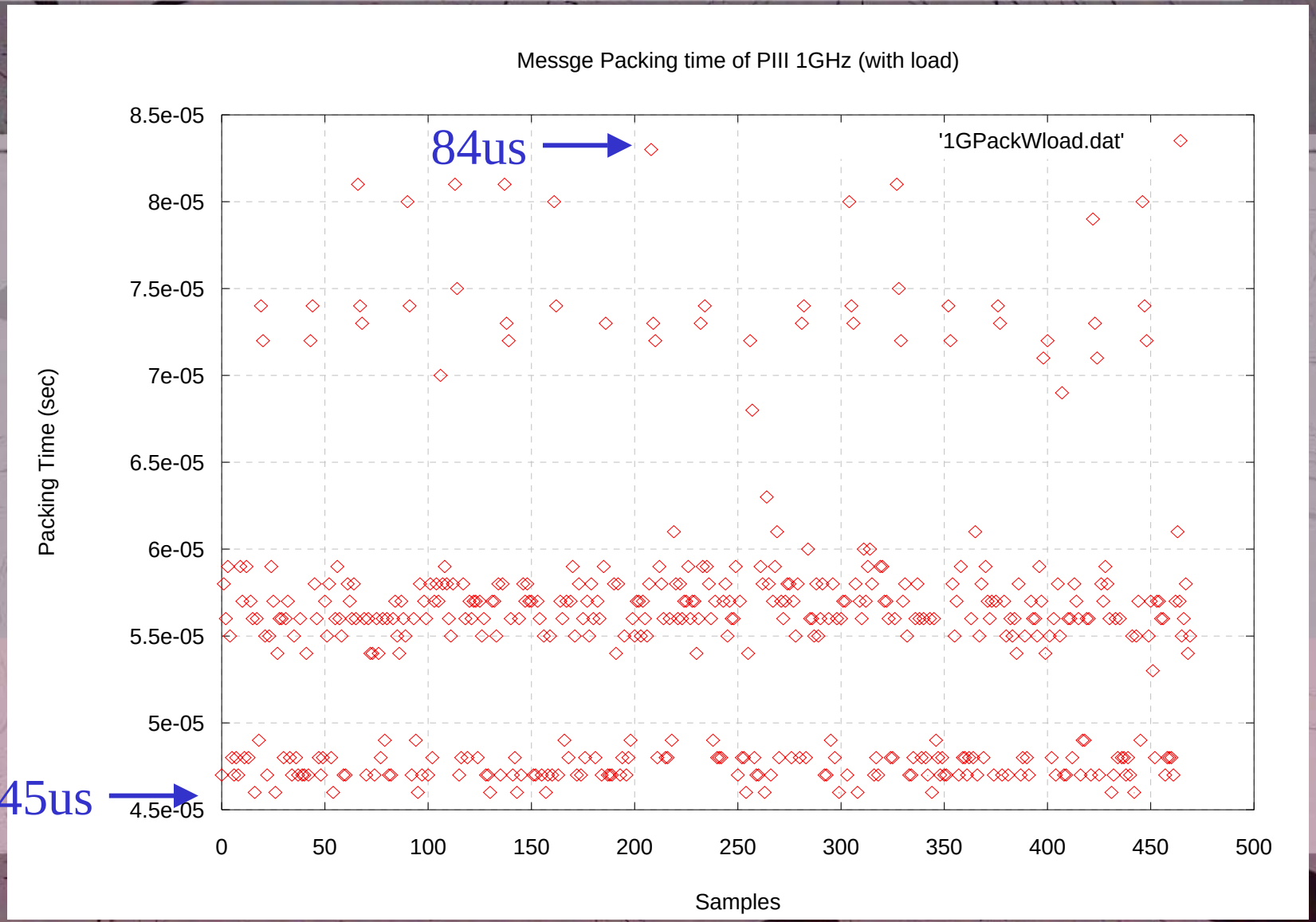
Experiment Platform

- Two computation nodes in a homogenous cluster
- PIII 1Ghz, 256MB RAM
- 100Base Ethernet
- RAOS-CPU installed

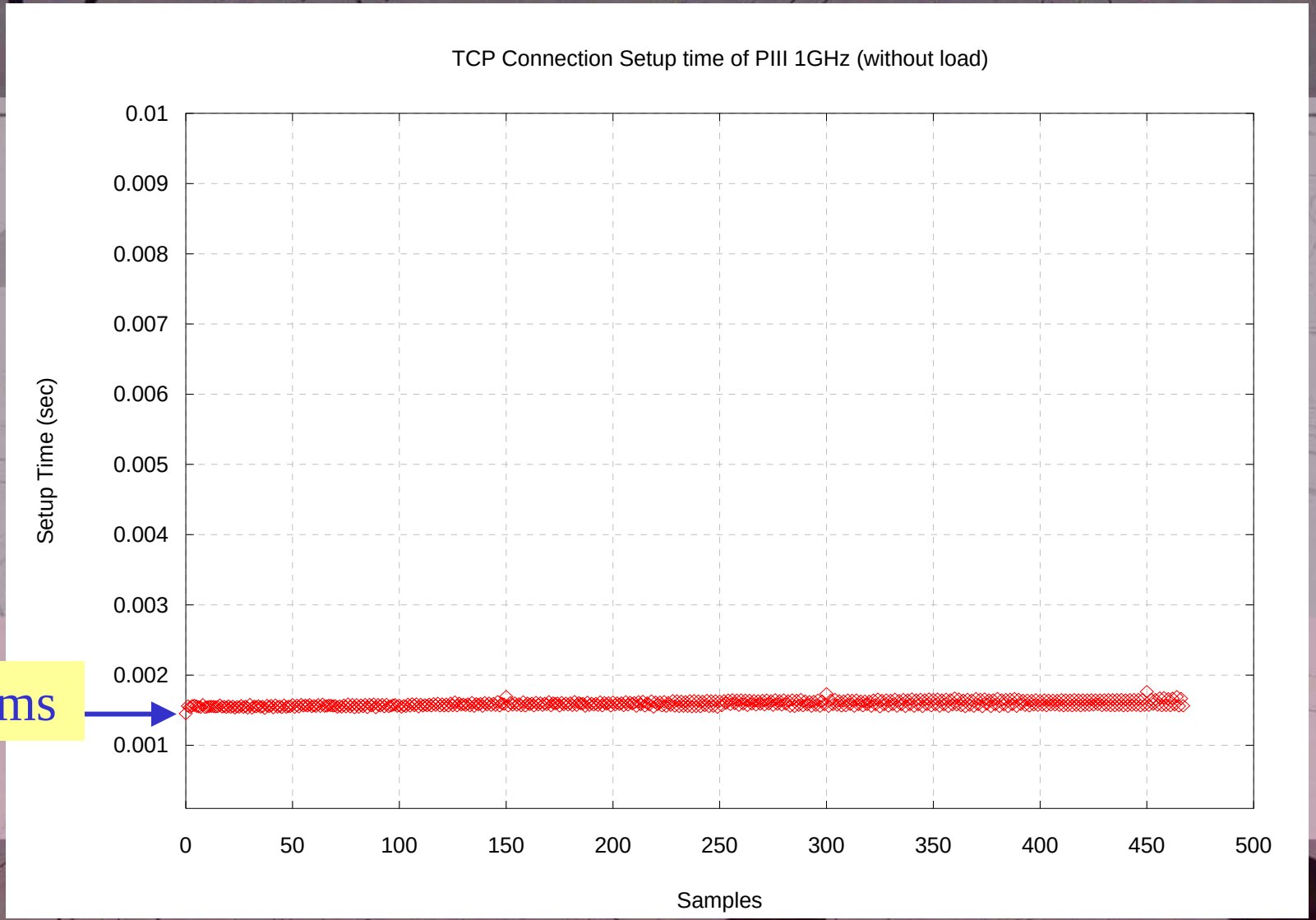
Message Packing Time without heavy system load



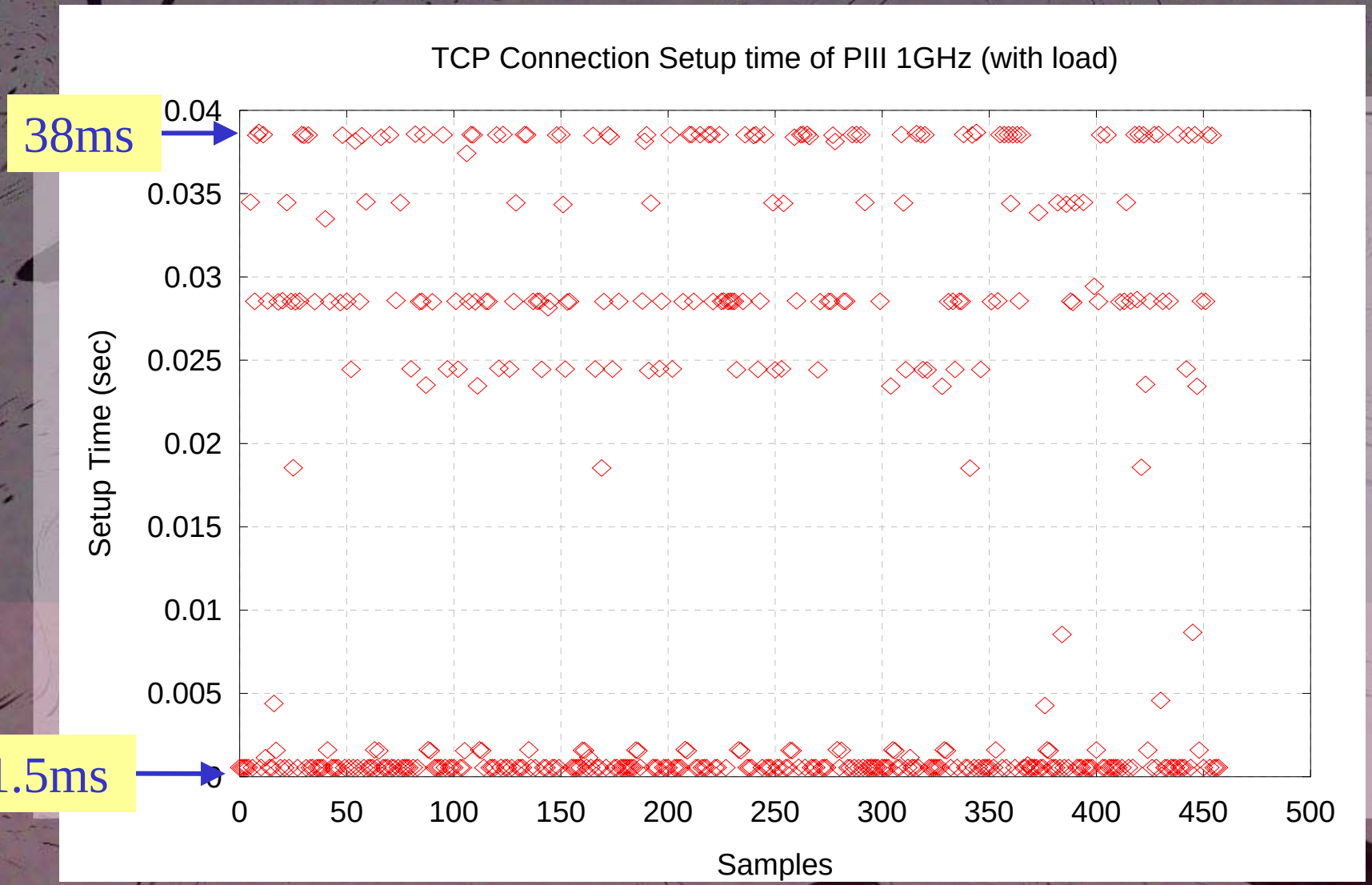
Message Packing Time with heavy system load



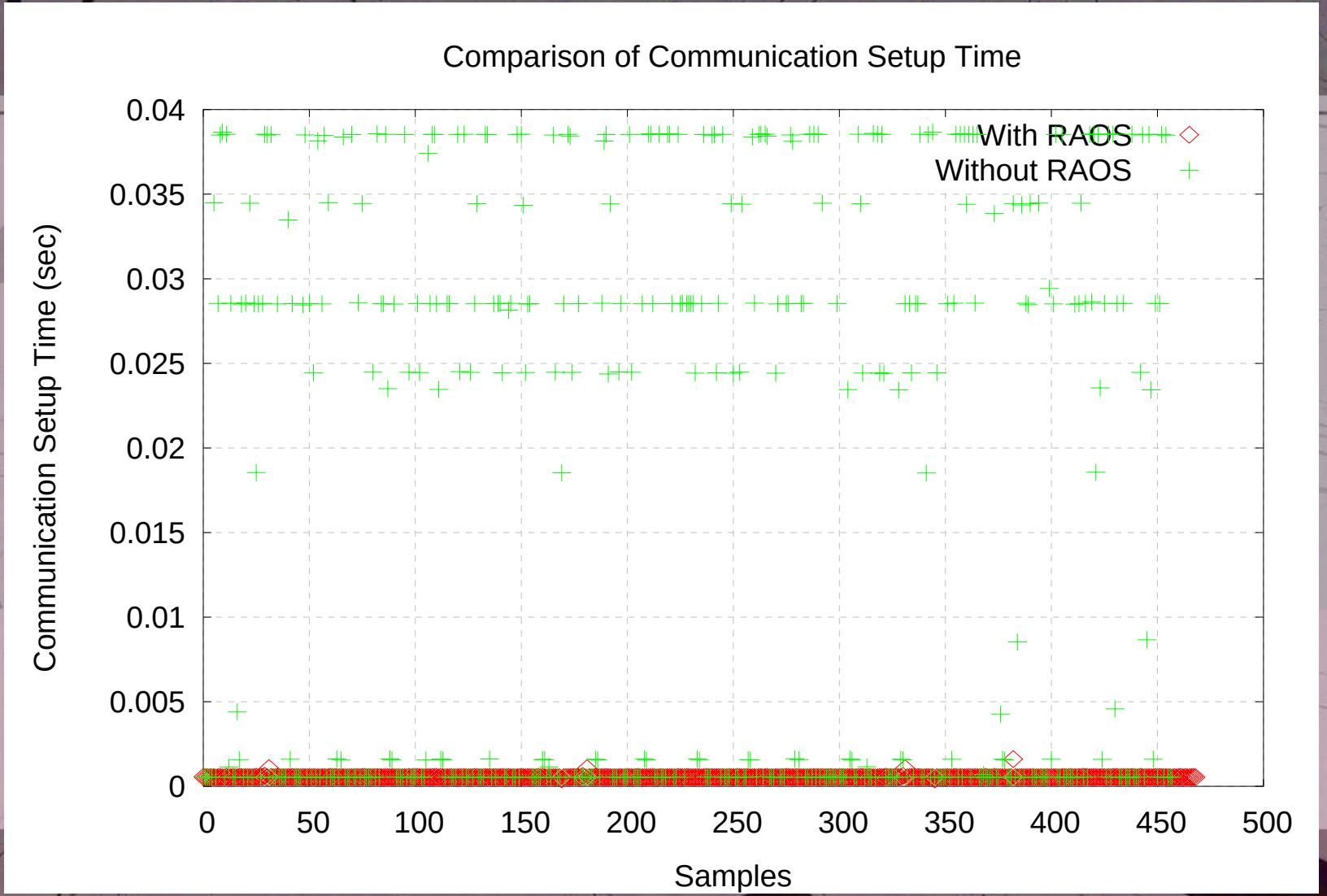
TCP connection setup time without heavy system load



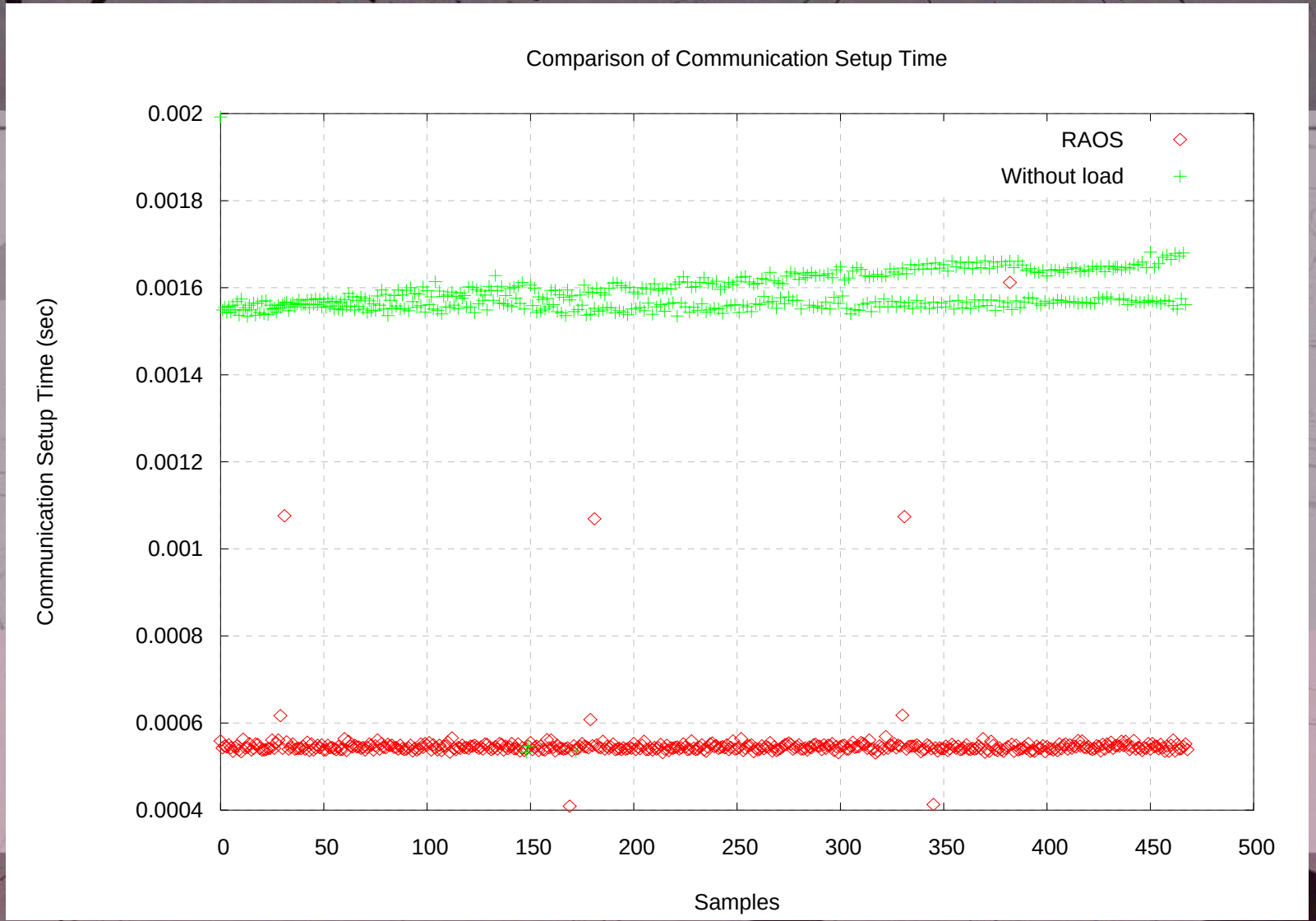
TCP connection setup time with heavy system load



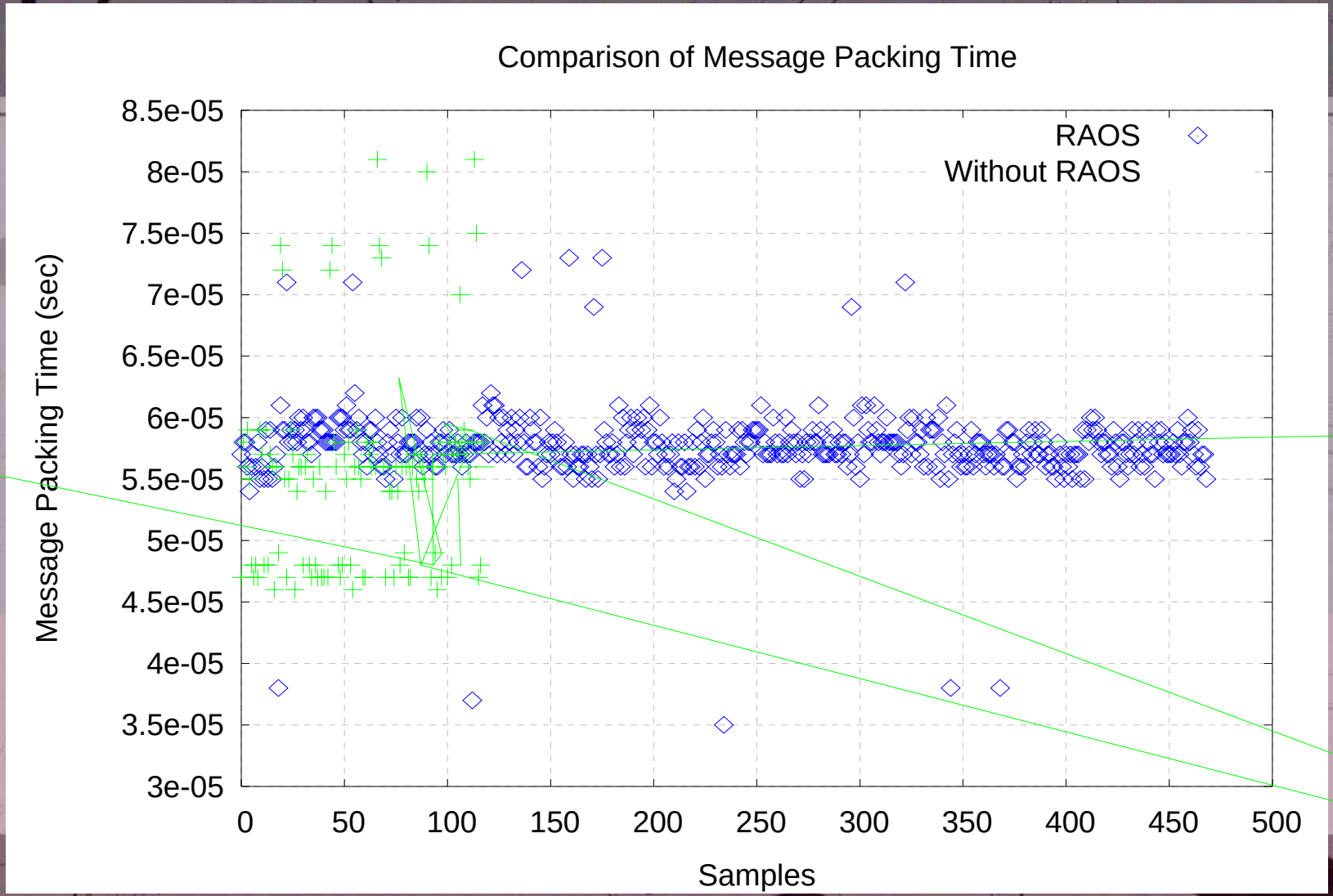
Comparison of Communication Setup time – with and without RAOS



Comparison of Communication Setup Time – Without load and RAOS



Comparison of Message Packing Time – with and without RAOS



Comparison of Message Packing Time – without load and RAOS

RAOS vs DSRT

- DSRT
  - User level scheduler
  - CPU resource only
- RAOS
  - Kernel level
  - Target for all the computation resources that the Operating System controls

Potential Benefit of RAOS

- Provide quality computation resource for Grid Economy
- Power saving in Cluster environment
- Process migration (Reserve before migrate)
- And more…..

Points to Consider

- Is the computer architecture today suitable for the future Grid paradigm?
- Would the reservation based resource management better then competition based ?

Question ?

?