Performance Isolation in Multi-Tenant Cloud Data Services

Vivek Narasayya, Sudipto Das, Manoj Syamala, Herodotos Herodotou, Badrish Chandramouli, Surajit Chaudhuri

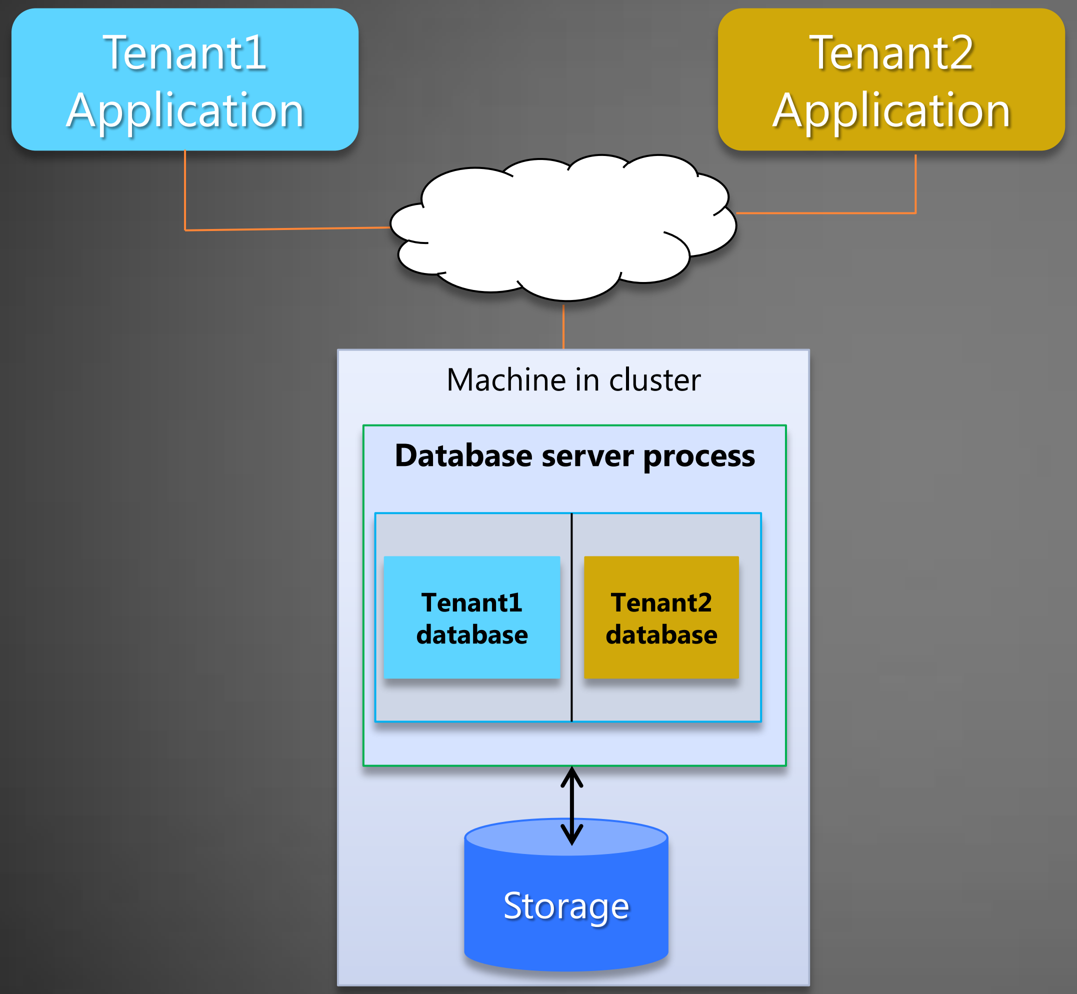
Microsoft Research

Interns: Feng Li (NUS), Hyunjung Park (Stanford)

Multi-Tenant Cloud Data Services

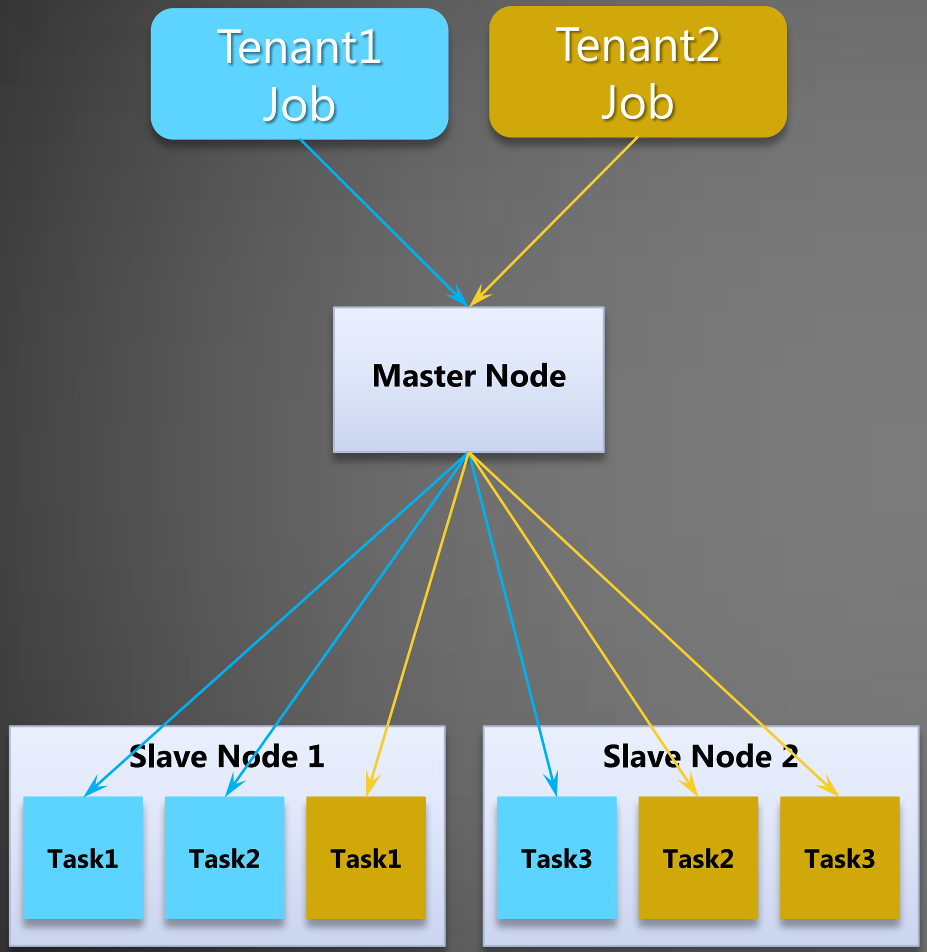
- Relational Database-as-a-Service (DaaS)
  - Examples: Microsoft SQL Azure, Amazon RDS
- MapReduce Cloud Platforms for "Big Data"
  - Examples: Windows Azure HDInsight, Amazon EMR, Cosmos (Microsoft internal)
- Cost vs. Performance
  - Low operational cost requires densely packing tenants
  - Tenants want good performance

Multi-Tenancy in Database-as-a-Service (DaaS)



- Queries from a tenant share DBMS resources with other tenants
- CPU, Memory, I/O, network shared across tenants
- Tenants seek isolation from SQL workloads issued by other tenants

Multi-Tenancy in MapReduce Platforms



- Each job is a collection of tasks
- Each task is an OS process
- Tasks of a tenant share machine resources with other tenants
- Tenants seek performance isolation at:
  - Task level
  - Job level

Focus of this talk

SQLVM: Performance Isolation in Multi-Tenant Relational Database-as-a-Service

Performance Isolation: Desiderata

- Tenants want performance unaffected by other tenant workloads
- Static resource allocation per tenant not cost effective
  - One VM per tenant each running a DBMS does not scale
- Service provider accountable for performance isolation
  - Increases confidence of customers to deploy in cloud

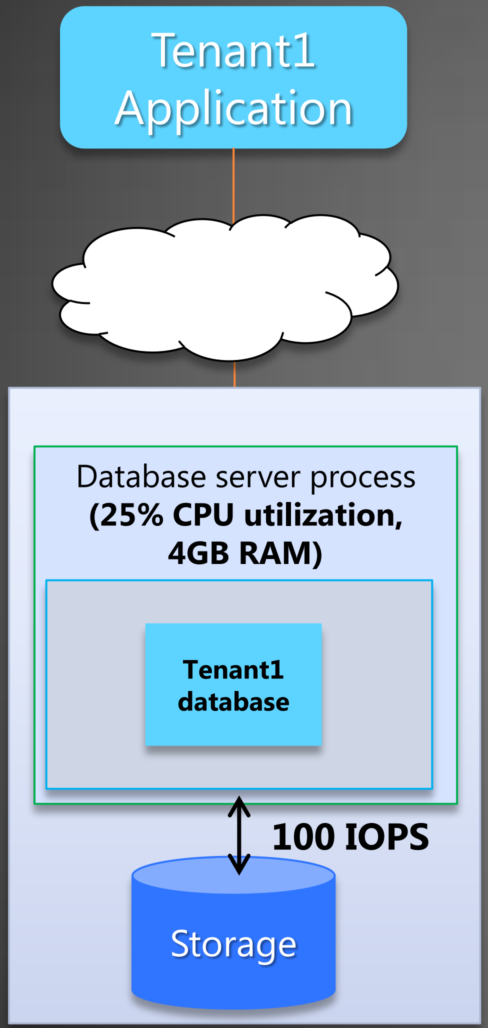
What Should Performance Isolation Mean?

- Can we promise queries/sec or query latency?
- Queries can access vastly different amounts of data

SELECT Product, SUM(Sales) as TotalSales
FROM FactSales F **JOIN** DimProduct P JOIN DimStates S
ON F.ProdID = P.ProdID and F.StateId = S.StateId
WHERE State = ~~'Vermont'~~ 'California'
GROUP BY Product

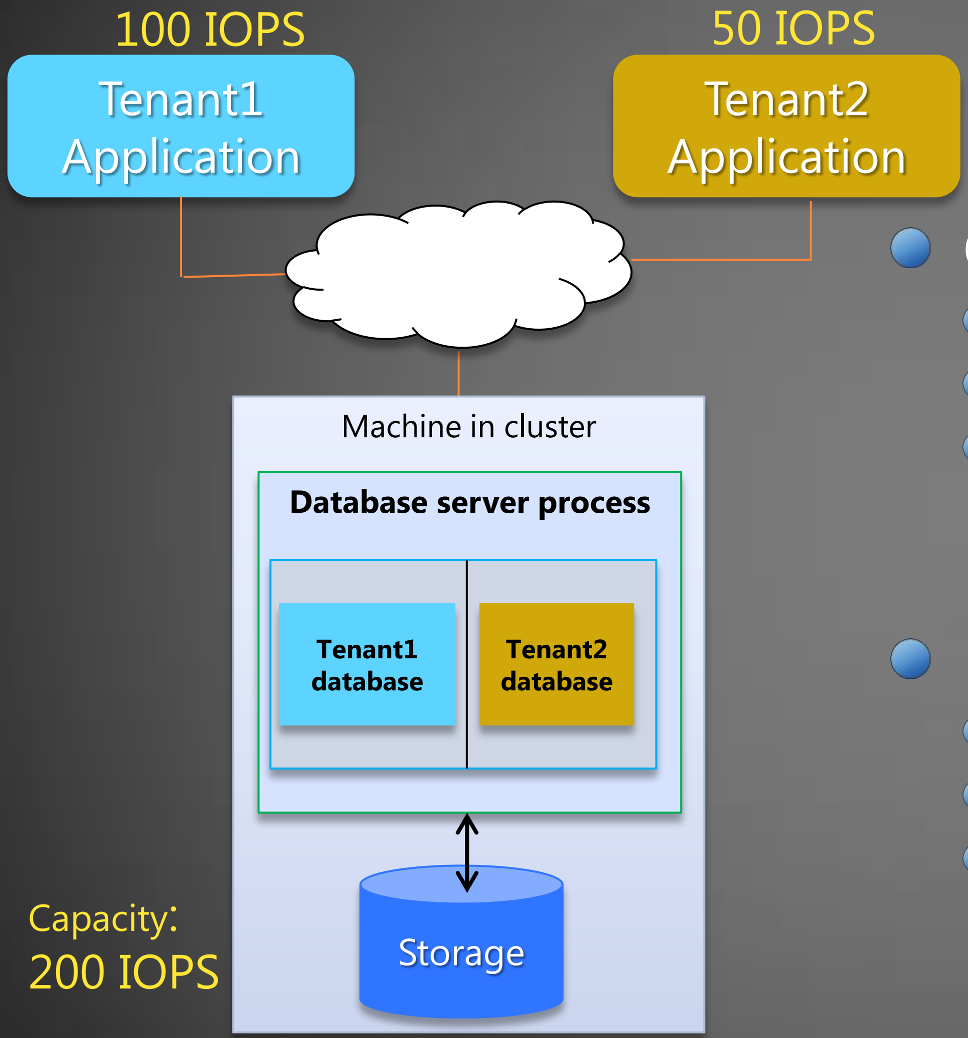
- DaaS providers aim to support most existing apps
  - Even ad-hoc queries

SQLVM



- Tenant is promised reservation of DBMS resources
  - “VM inside SQL process”
  - CPU utilization, IOPS, Memory, …
- Resource governance
  - Fine-grained resource sharing
  - Novel mechanisms
- Metering (auditing)
  - Monitor actual and promised metrics for tenant
  - Determine violations

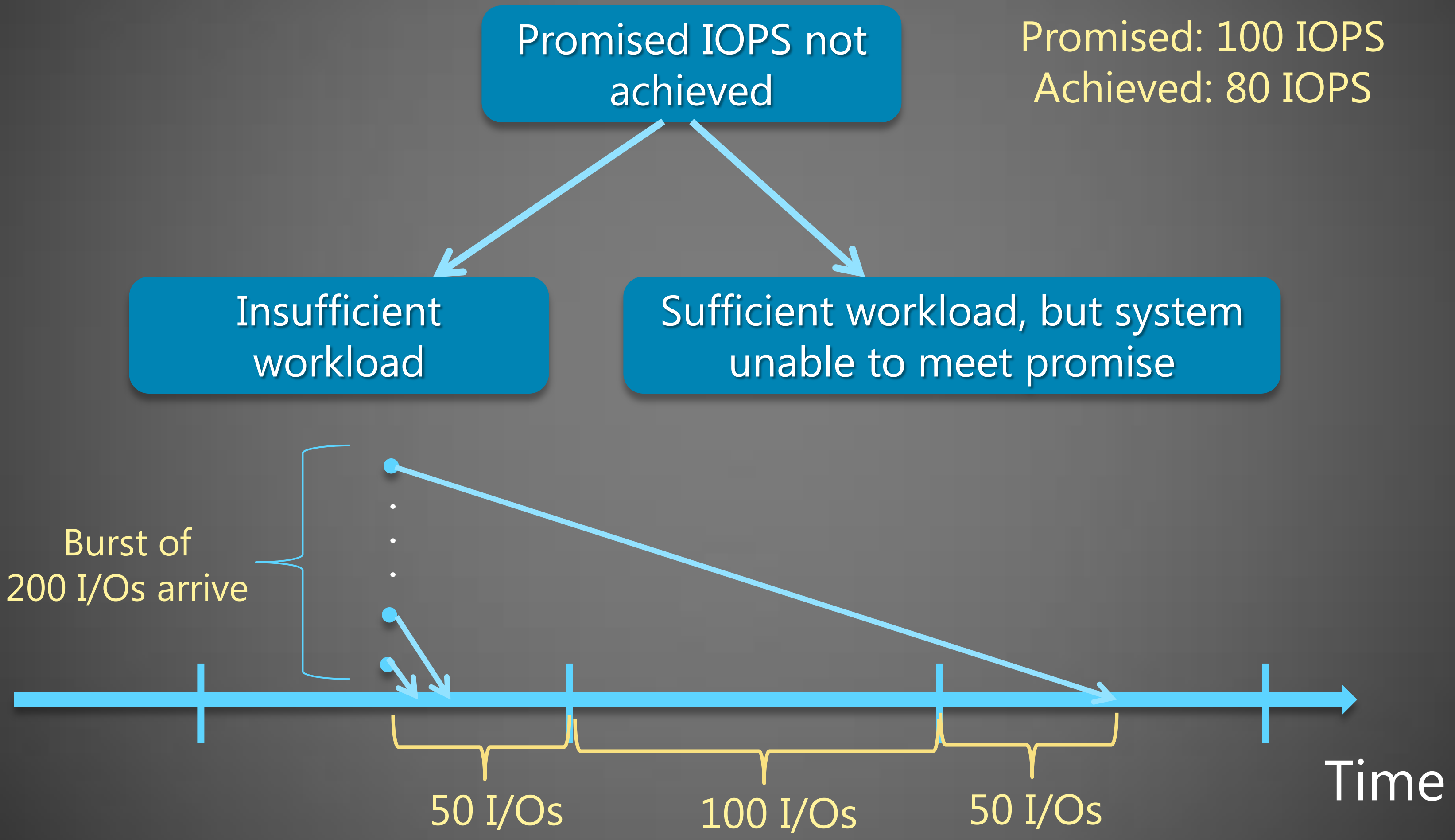
Resource Governance Mechanism



- Challenges
  - Bursty I/O patterns
  - Coordinating tenant I/Os across cores
  - Capturing I/Os issued indirectly on tenant's behalf
- Key idea: Shape I/O traffic
  - 50 IOPS $\Rightarrow$ one I/O every 20 msec
  - I/O request tagged with deadline
  - Issue I/Os whose deadline has arrived

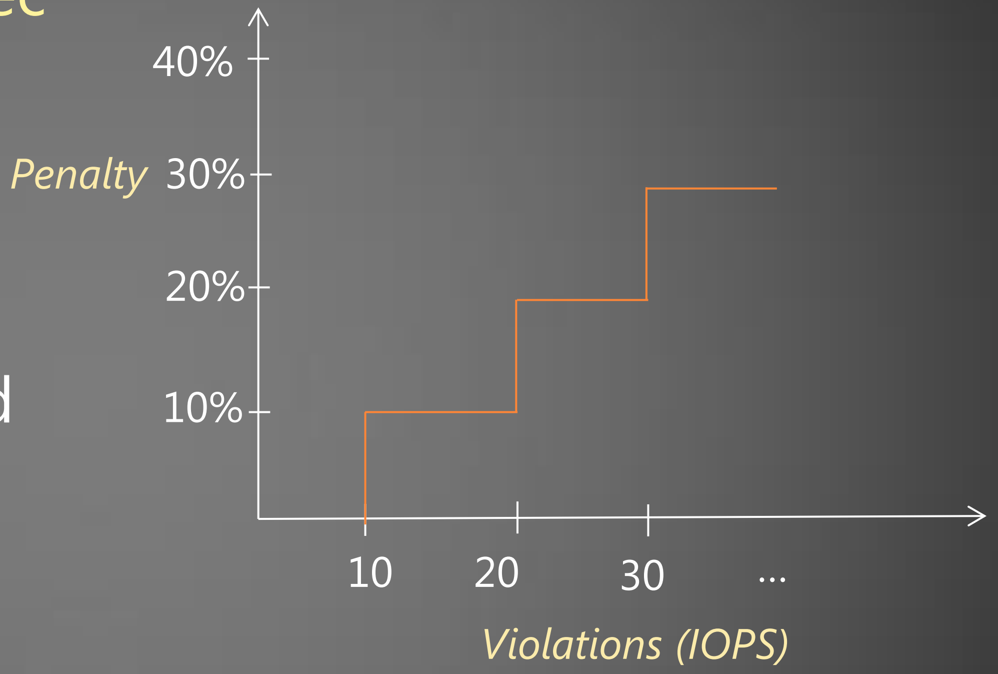
Metering

- Metering interval (e.g. 1 sec)



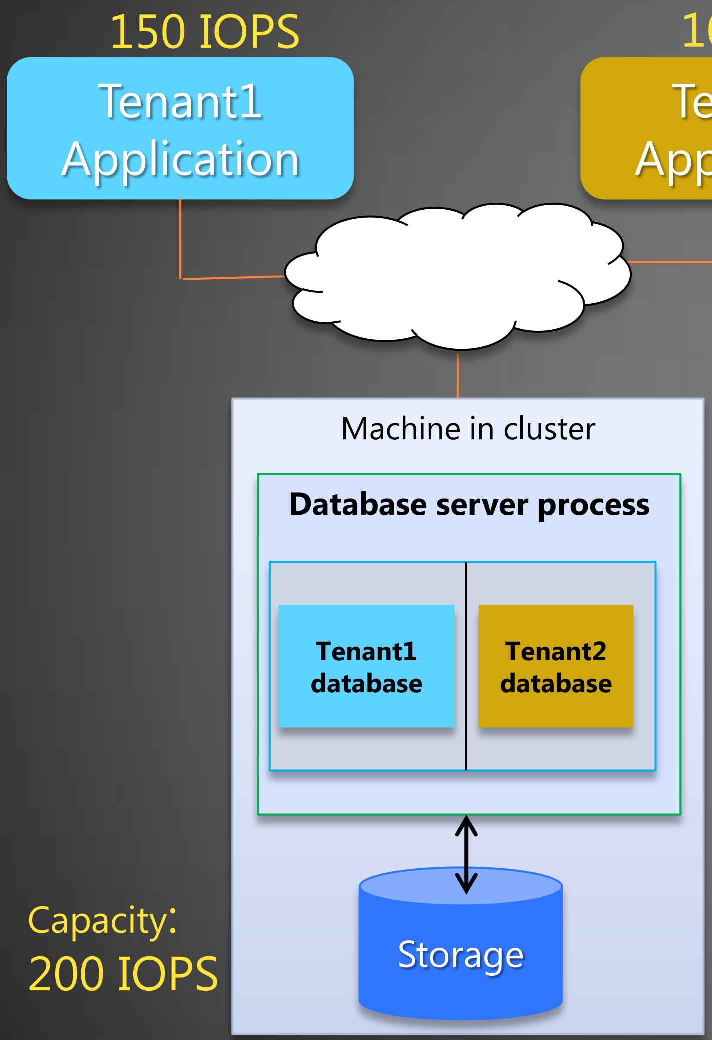
Violations and Penalties

- Metering interval, e.g. 1 sec
- Tenant is promised a Reservation of 100 IOPS
- Metering: Actual IOPS and Violations (IOPS)
- Penalty applied if Actual IOPS < promised IOPS



(SLA structure similar to Availability SLAs offered today)

Overbooking



- Enables denser packing, but…
- … may not be able to meet promises
- Resource governance objectives:
  - Minimize overall penalty
  - Fairness to tenants
- Online optimization
- Related problems:
  - How much to overbook?
  - Tenant migration

CPU and Memory

- CPU
  - Reservation: CPU utilization (e.g. 10%)
  - Resource governance challenges:
    - Variable quantums, number of connections, parallelism
  - Metering: Measure delay when tenant thread is ready to run but CPU is being used by another tenant
  - Upcoming VLDB 2014 paper

- Memory
  - Buffer pool memory is a cache of database pages
  - Reservation: Hit Ratio of workload for given memory size (e.g. 1GB)
  - Metering: “what-if” analysis to determine promised Hit Ratio

Demo

Related Work

- Resource/workload management for DBMS
  - Based on **maximum limits**, priorities etc.
  - Survey: [Krompass et al, IEEE Data Engg. Bulletin, 2008]
- SLA on Query Response Time
  - Cost-aware scheduling [Chi et al, VLDB 2011]
  - PIQL: Success-Tolerant Query Processing in the Cloud [Armbrust et al, VLDB 2011]
- Consolidating multiple database workloads
  - Database consolidation and resource modeling[Curino et al, SIGMOD 2011, Mozafari et al, SIGMOD 2013]
  - Towards multi-tenant Performance SLOs [Lang et al, ICDE 2012]

Status and Future Work

- Working in close collaboration with SQL team in Microsoft
  - Novel resource governance, metering mechanisms
  - SQL Azure, SQL Server 2014 CTP1
- Ongoing and Future Work
  - Resources: CPU, I/O, Memory
  - Exploiting SQLVM
    - Overbooking
    - Capacity planning
    - Higher-level performance SLAs

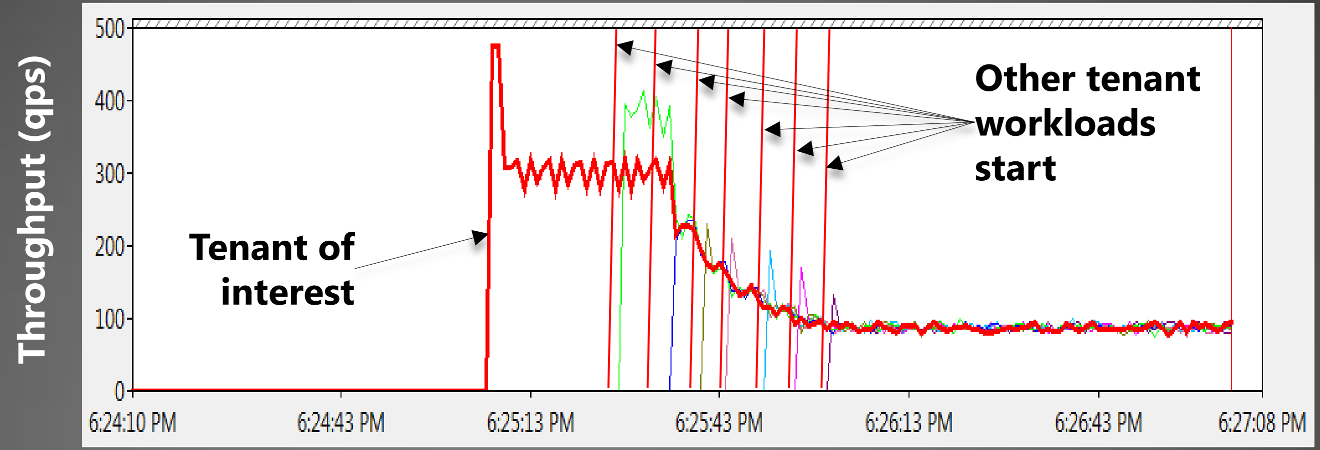
Backup Slides

Experimental Setup

- **Workloads:** TPC-C, Dell DVD-benchmark, TPC-H, CPUIO
- **Machine**: 12 core, 72 GB RAM, 3 HDD, SSD (log)
- **Number of tenants**: Up to 100

- **Example experiment**
  - **Eight** tenant databases sharing a *single* SQL Server instance
  - Each tenant executing a **CPU- and I/O-intensive** workload
  - **Tenant 1** (connecting to db1) is the tenant of interest
  - Tenant 1 shown in **Red**
  - Tenant 1 starts its workload, other tenants gradually added to the system
  - Execute without and with SQLVM

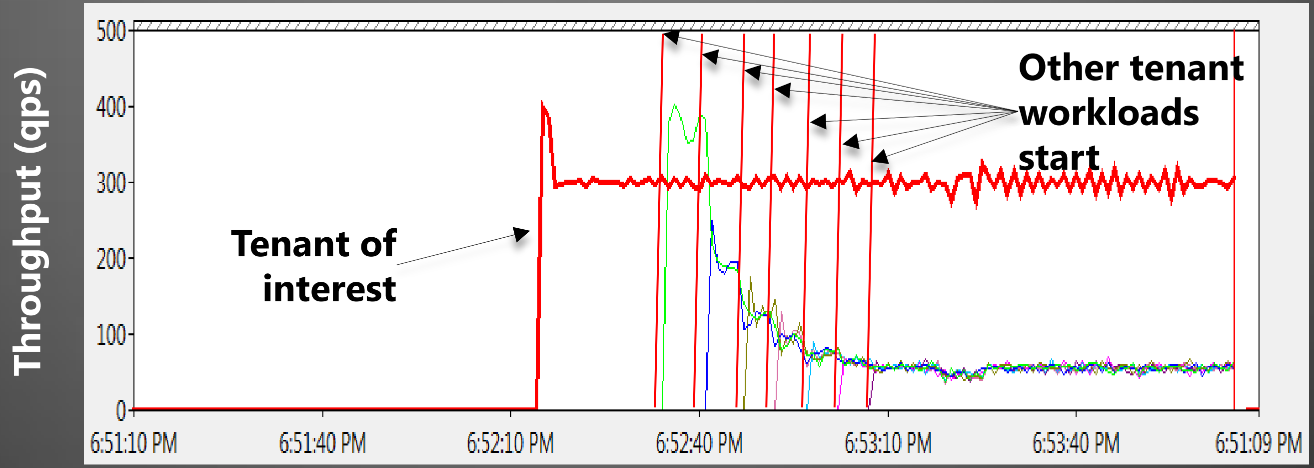
Without Performance Isolation



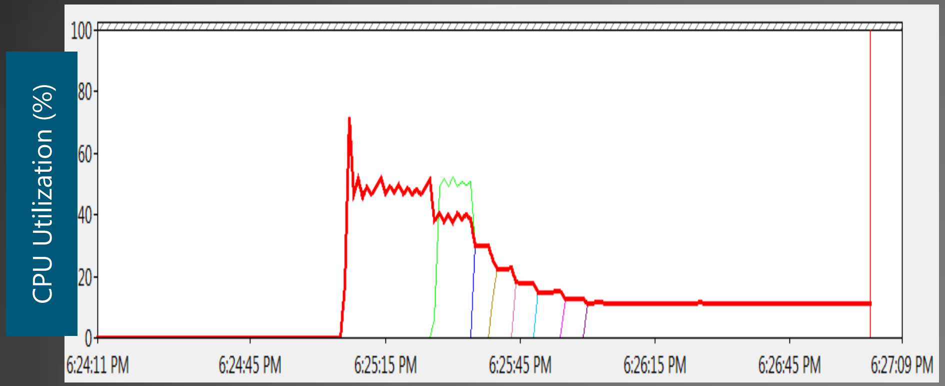
With Performance Isolation (SQLVM)

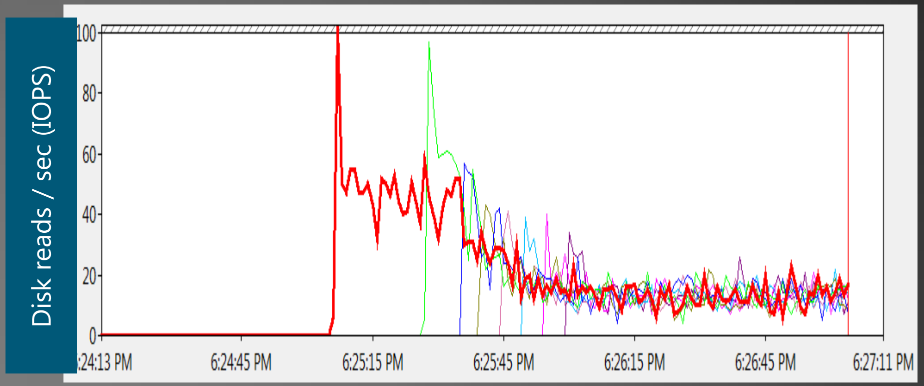
Tenant1:

- **50% CPU utilization**
- **50 IOPS**
- **2 GB RAM**

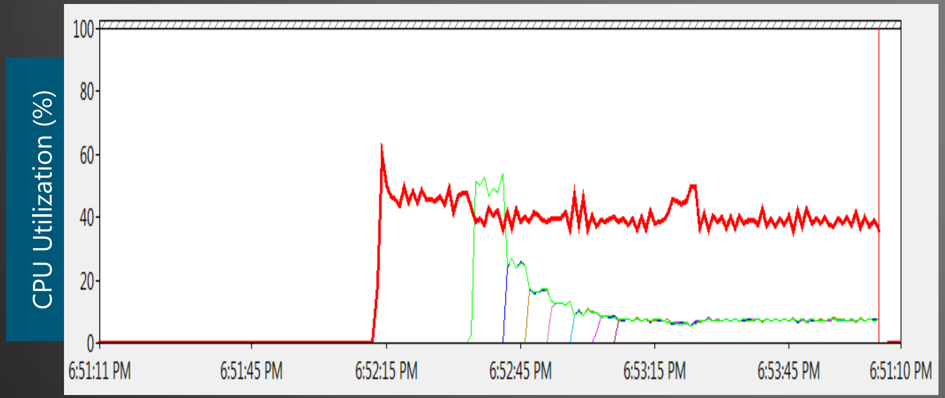


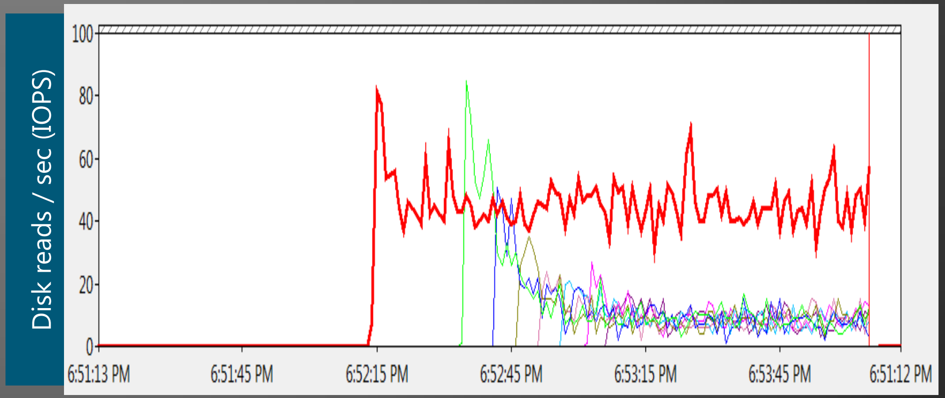
Without SQLVM





With Performance Isolation (SQLVM)





Relational Database-as-a-Service Providers

- Microsoft SQL Azure
  - Single SQL Server process per node
  - Each tenant gets a database
- Amazon RDS
  - MySQL hosted in VM
  - SQL Server, Oracle
- Oracle 12c
  - Multi-tenant Oracle DBMS as a service
- Google Cloud SQL
  - MySQL database
  - Allows DBMS access from AppEngine