Performance isolation across virtual machines in Xen

Diwaker Gupta,
Amin Vahdat
University of California,
San Diego

Lucy Cherkasova,
Robert Gardner
Hewlett-Packard Laboratories,
Palo Alto & Fort Collins

Middleware

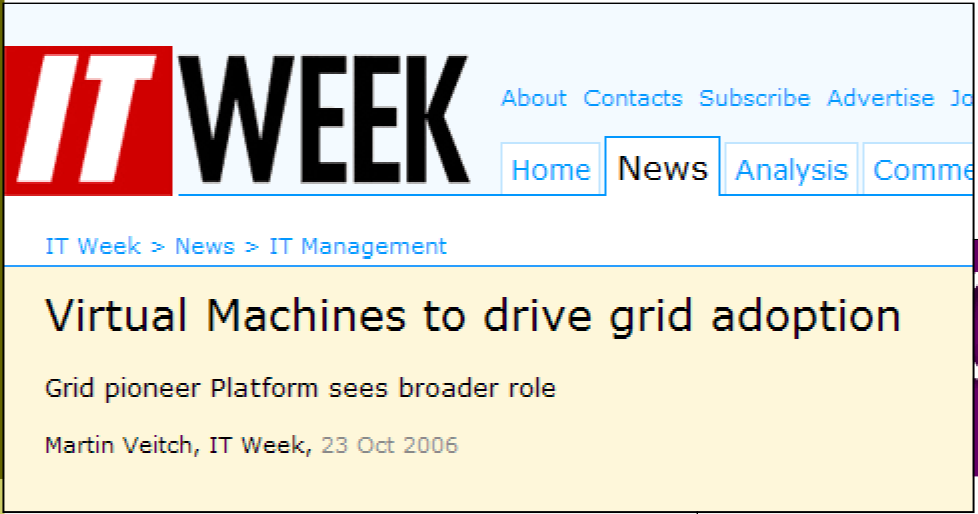
- Software that *connects* software components or applications, often to support complex, distributed systems (source: *Wikipedia*)
- All about virtualization of resources and abstracting out hardware heterogeneity
- Goal is to efficiently utilize a shared infrastructure
- It is *critical* to protect users from one another

Virtual Machines

- Software that creates a virtualized environment for the end-user (source: *Wikipedia*)
- Abstract out hardware heterogeneity
- Provides isolated execution environment for users

→ Virtual machines seem like good technology for building Middleware

HP SoftUDC, Amazon EC2

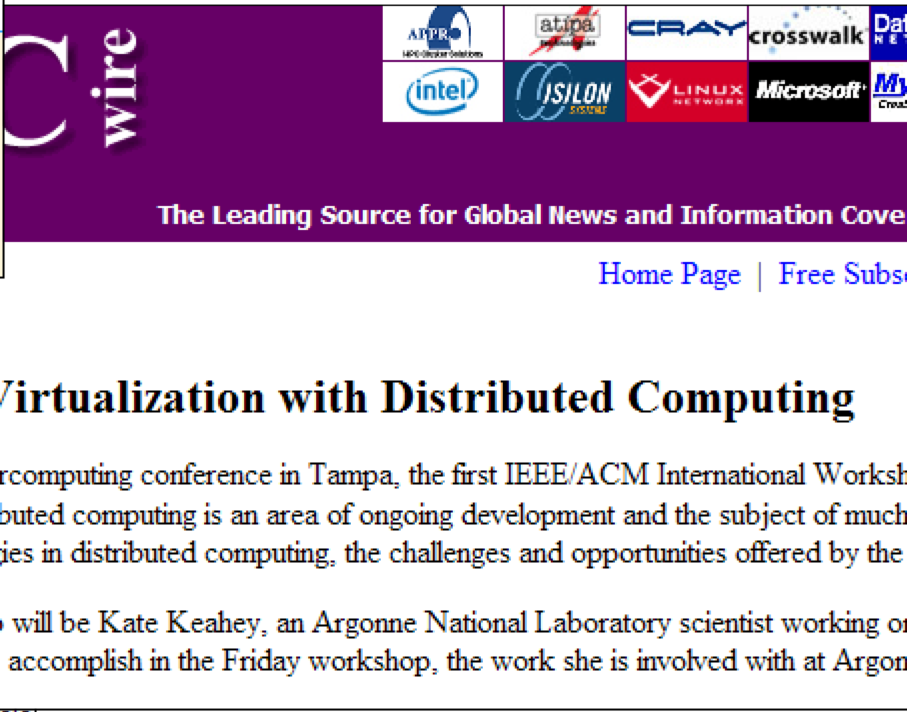


IT Week > News > IT Management

Virtual Machines to drive grid adoption

Grid pioneer Platform sees broader role

Martin Veitch, IT Week, 23 Oct 2006

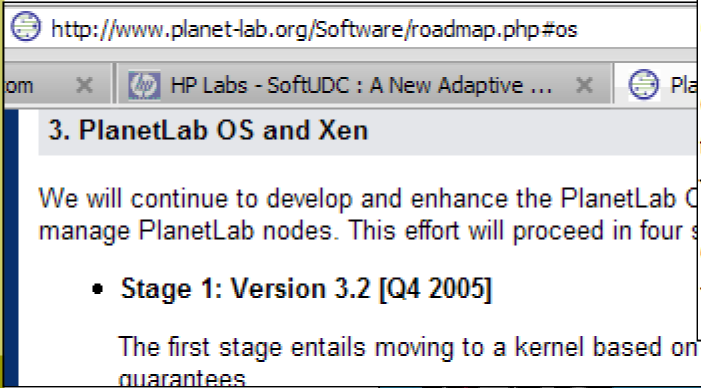


Grid@SC06

Converging Virtualization with Distributed Computing

On Friday at the Supercomputing conference in Tampa, the first IEEE/ACM International Worksh
technologies and distributed computing is an area of ongoing development and the subject of much
virtualization technologies in distributed computing, the challenges and opportunities offered by the

Chairing the workshop will be Kate Keahey, an Argonne National Laboratory scientist working o
what she would like to accomplish in the Friday workshop, the work she is involved with at Argon



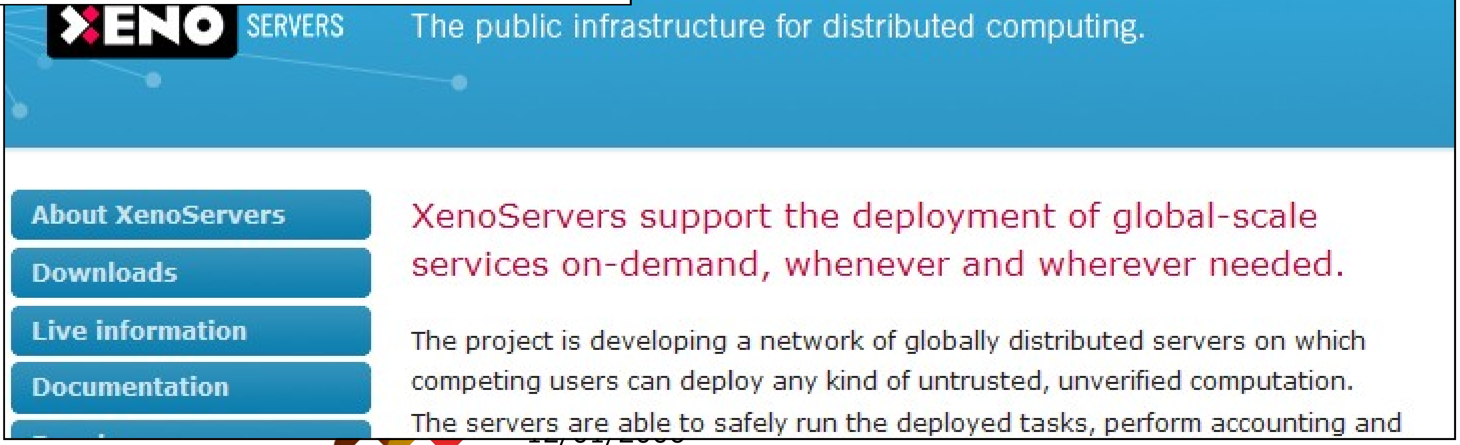
http://www.planet-lab.org/Software/roadmap.php#os
om
HP Labs - SoftUDC : A New Adaptive ...
Pla

3. PlanetLab OS and Xen

We will continue to develop and enhance the PlanetLab O
manage PlanetLab nodes. This effort will proceed in four s

- **Stage 1: Version 3.2 [Q4 2005]**

The first stage entails moving to a kernel based on Fedora Core's 1.1398 rele
guarantees



XenoServers support the deployment of global-scale services on-demand, whenever and wherever needed.

The project is developing a network of globally distributed servers on which competing users can deploy any kind of untrusted, unverified computation. The servers are able to safely run the deployed tasks, perform accounting and

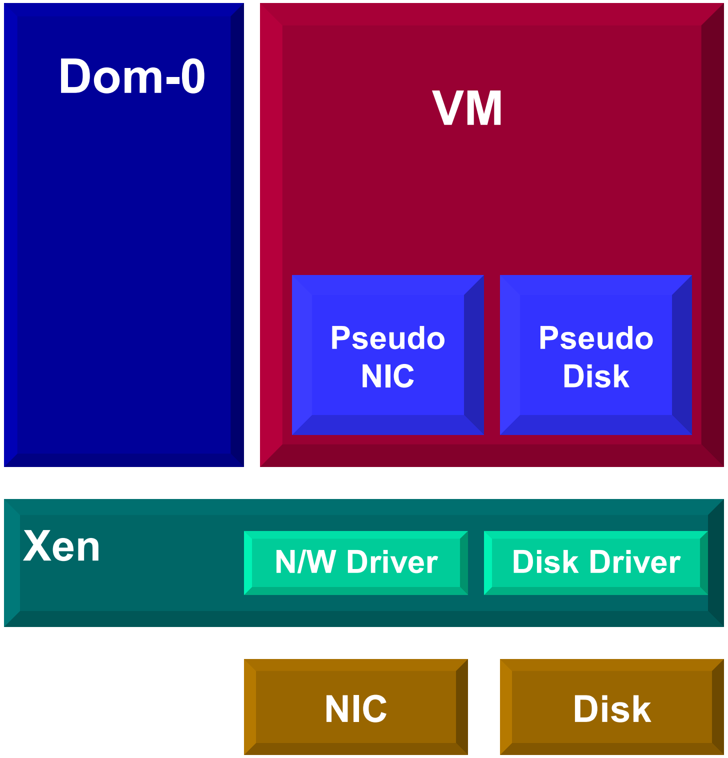
Requirements from VM platform

- Fault isolation
- Performance isolation
  - Performance of one VM should not impact performance of another VM
  - Related concept: *resource isolation*
  - Resource isolation is *necessary* for performance isolation, but is it *sufficient?*

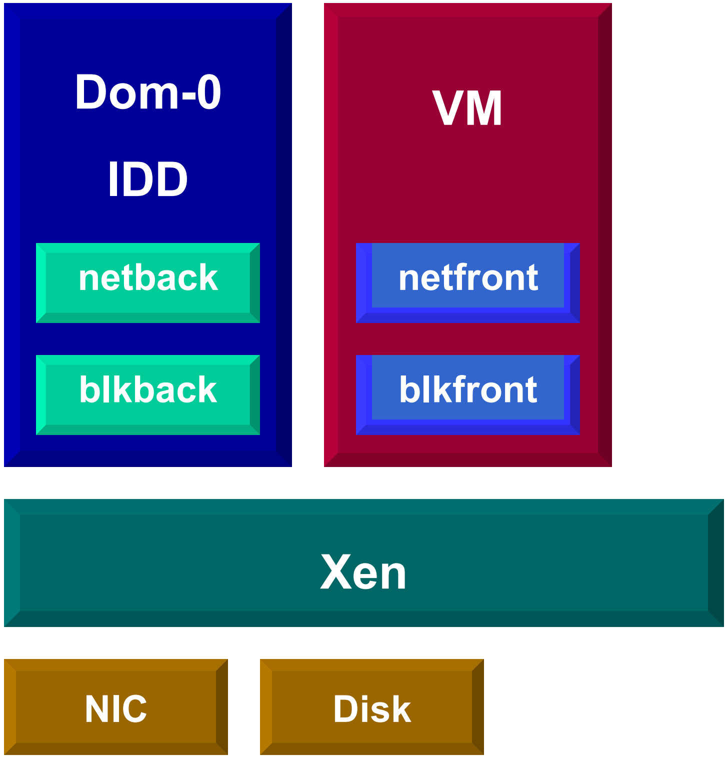
This work focuses on the *performance isolation* in Xen [SOSP 2003]

Evolution of I/O Model in Xen

Xen 1.x: Device drivers in hypervisor

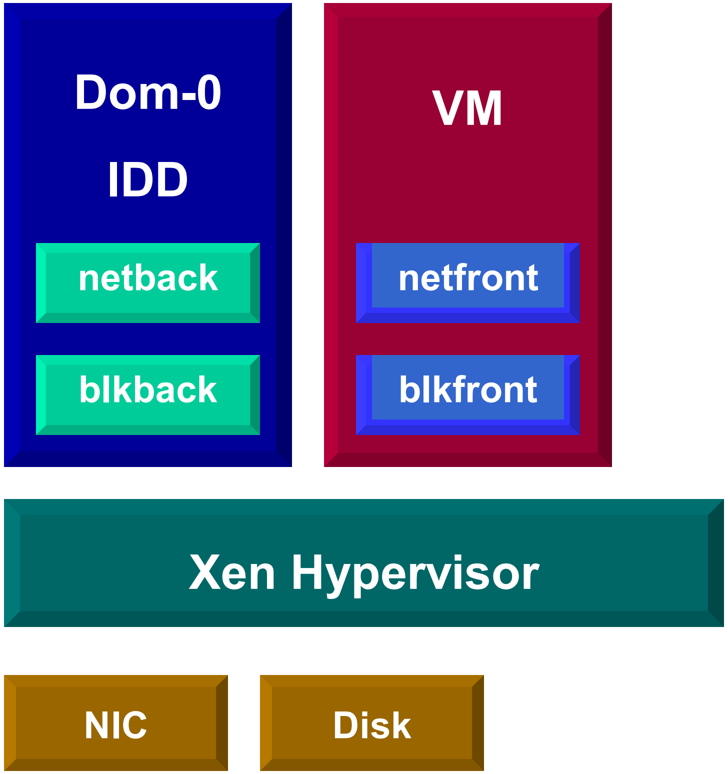


Xen 3.x: Device drivers in driver domains



Driver Domains

- Execution container vs. resource principle
  - Resource consumption of a VM may span several driver domains
- Accurate accounting and resource allocation
  - Resource consumption by an IDD on behalf of a VM



Two concrete problems

- How does one control the *aggregate* resource consumption of a VM (including resources consumed in a driver domain on its behalf)?
- How does one control the resource consumed by a VM within a driver domain?

General Strategy

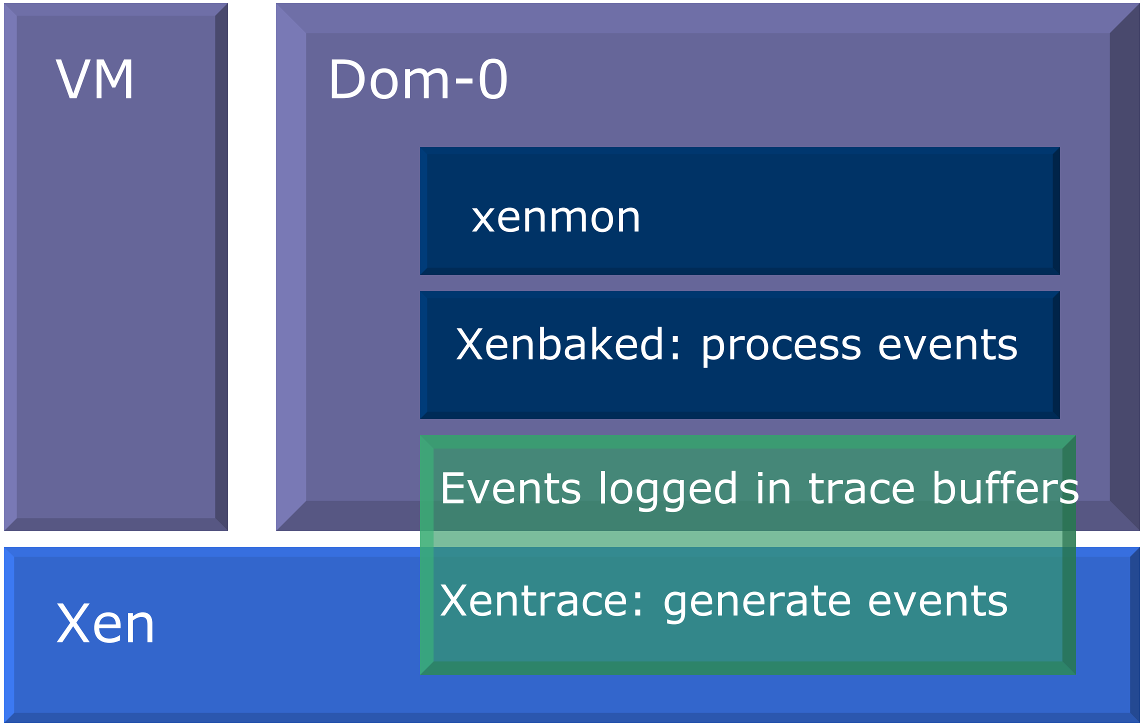
- Measure
  - Profiling tools
- Allocate
  - Modifications to the CPU scheduler
- Control
  - Mechanisms to control resource usage

Our work focuses on CPU and network I/O.

XenMon

- Events: anything "interesting" (domain started running, a packet was sent, domain woke up etc)
- Events analyzed in user space to generate meaningful metrics (e.g. blocking time, waiting time etc)
- Flexible measurement granularity: over 10s, over 1s, avg per *execution period*
- Included in the official Xen code tree

XenMon Architecture



More details on XenMon available in HP Labs tech report HPL-2005-187

Two concrete problems

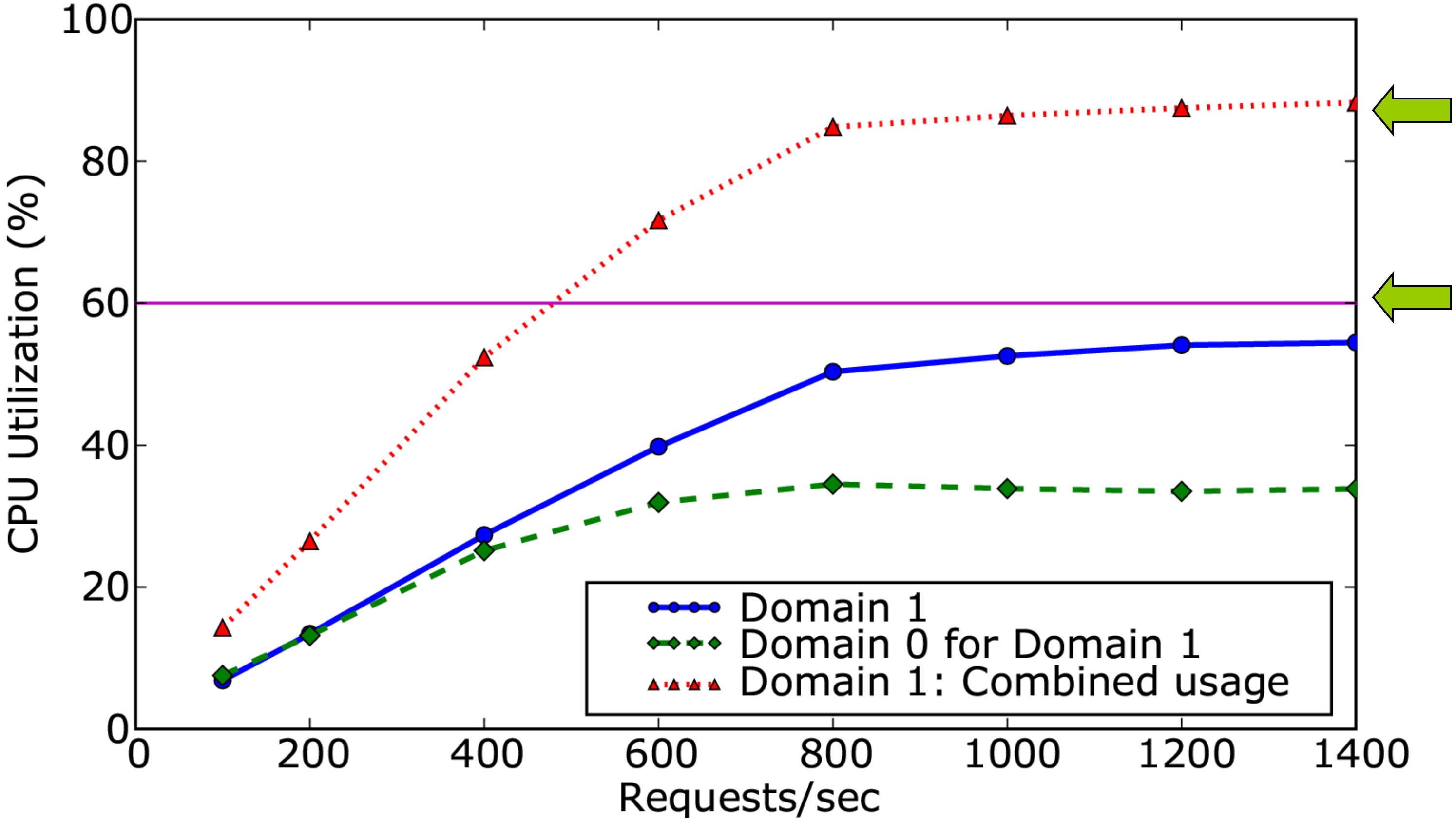
- *How does one control the aggregate resource consumption of a VM (including resources consumed in a driver domain on its behalf)?*
- How does one control the resource consumed by a VM within a driver domain?

Problem: Controlling aggregate CPU

- Example
  - Single CPU system
  - SEDF (Simple Earliest Deadline First) in non work-conserving mode (hard reservations)
  - VM-1: web server, 60%
  - Dom-0: driver domain, 40%
  - How to control aggregate CPU consumption?

General scenario: Two workloads with different characteristics (I/O vs. CPU intensive) are given equal shares. Do they really get equal shares?

Aggregate CPU consumption



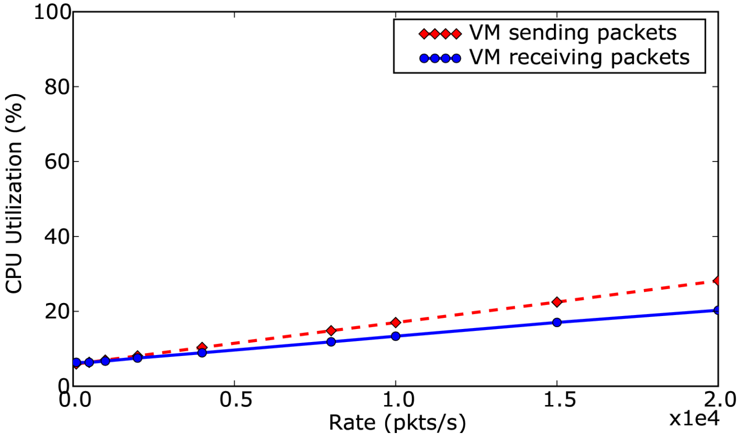
Aggregate

Ideal

Controlling aggregate CPU

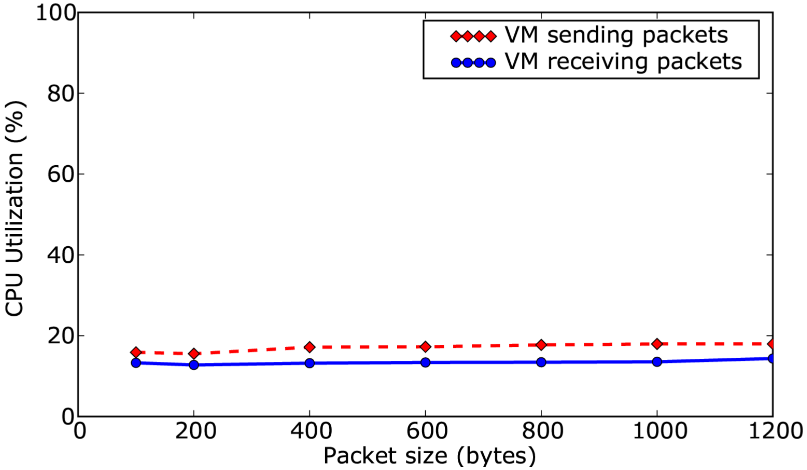
- Goal: allocate CPU shares accounting for **aggregate** CPU consumption
- Steps:
  - Partition CPU consumption in IDD for different VMs
  - Charge this *debt* back to the VM
- Partitioning: timing code paths vs. heuristics
- Heuristic for partitioning: CPU overhead is proportional to the amount of I/O

Packet counting in **netback**



CPU overhead is **proportional to rate of packets**

CPU overhead is **independent of packet size**

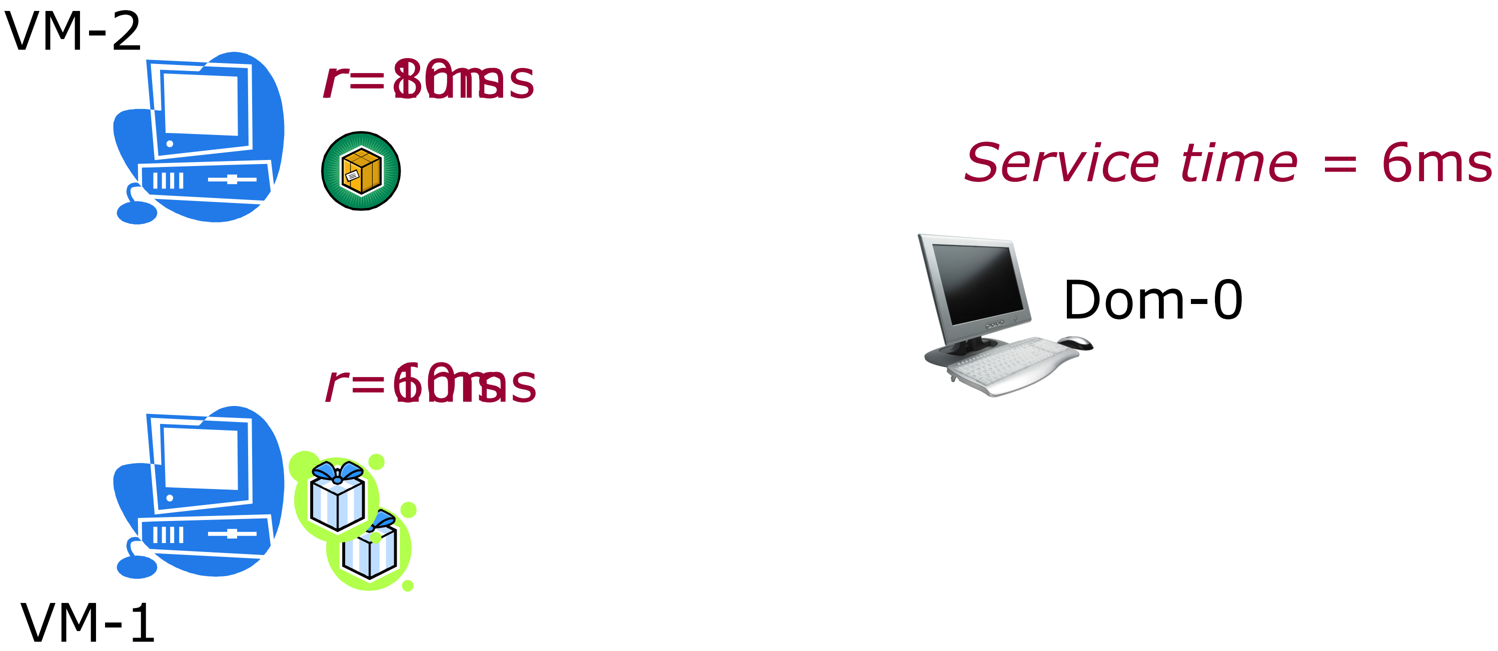


- CPU overhead is different for send and receive paths
- But send:receive cost is *constant*

SEDF Debt Collector (SEDF-DC)

- Count packets corresponding to each VM
- Compute *weighted* packet count (using the send:receive factor)
- Partition CPU consumed by IDD using weighted packet counts
- Charge *debt* of each VM to its CPU consumption in the scheduler

SEDF-DC Example



VM-2

r=10ms

r=8ms

Service time = 6ms

Dom-0

r=10ms

r=6ms

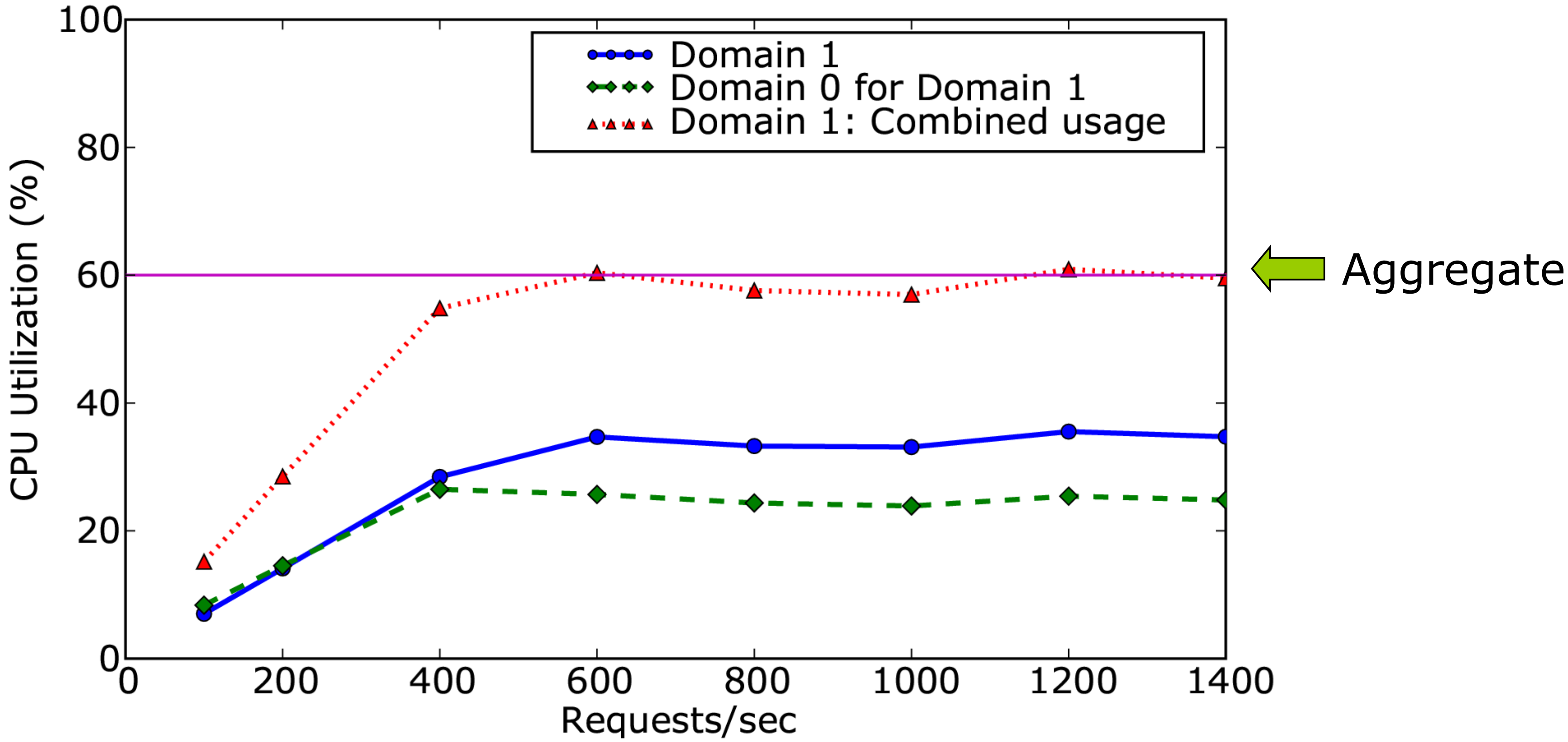
VM-1

t=0: Both VM-1 and VM-2 have remaining time 10ms

t=10ms: Dom-0 ran for 6ms to service VM traffic

SEDF-DC reduces remaining time of VM-1 by 2ms and VM-2 by 4ms respectively

SEDF-DC in action



SEDF-DC Summary

- SEDF-DC addresses problem for SEDF in single processor case
- Idea can be extended to other CPU schedulers in Xen (such as Credit)
- Spread debt across multiple execution periods to avoid starvation

But still no QoS in the driver domain

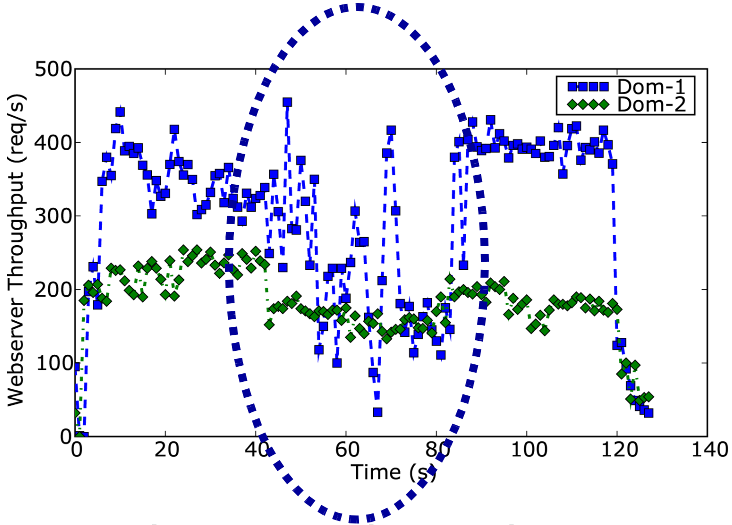
Two concrete problems

- How does one control the *aggregate* resource consumption of a VM (including resources consumed in a driver domain on its behalf)?
- *How does one control the resource consumed by a VM within a driver domain?*

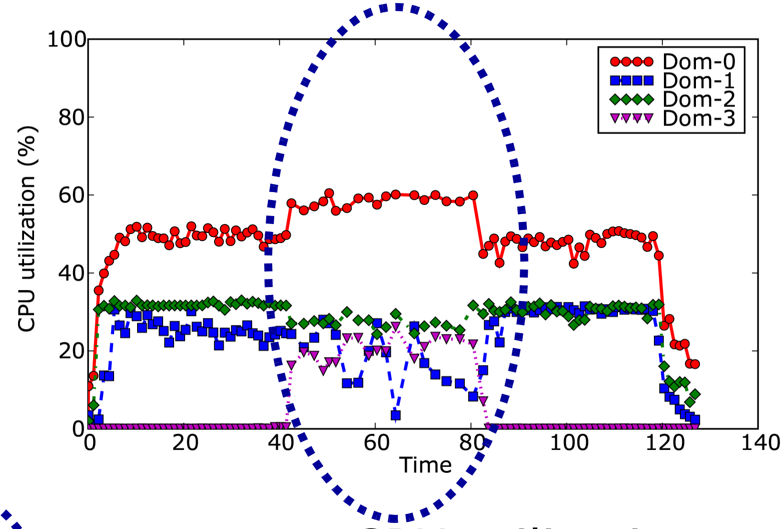
Problem: Controlling resource consumption in driver domain

- Scenario
  - SEDF, dual processor machine, non work-conserving mode
  - Dom-1: Web server, 33% on CPU-2 (10KB files)
  - Dom-2: Web server, 33% on CPU-2 (100KB files)
  - Dom-3: File transfer, 33% on CPU-2
  - Dom-0: 60% on CPU-1
- File transfer begins 20s into the experiment
- Goal: file transfer in VM-3 should not affect web servers in VM-1 and VM-2

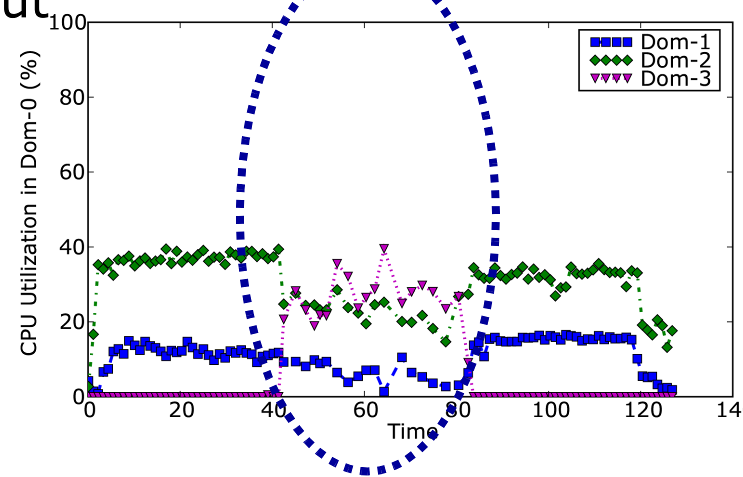
No QoS in driver domain



Webserver throughput



CPU utilization

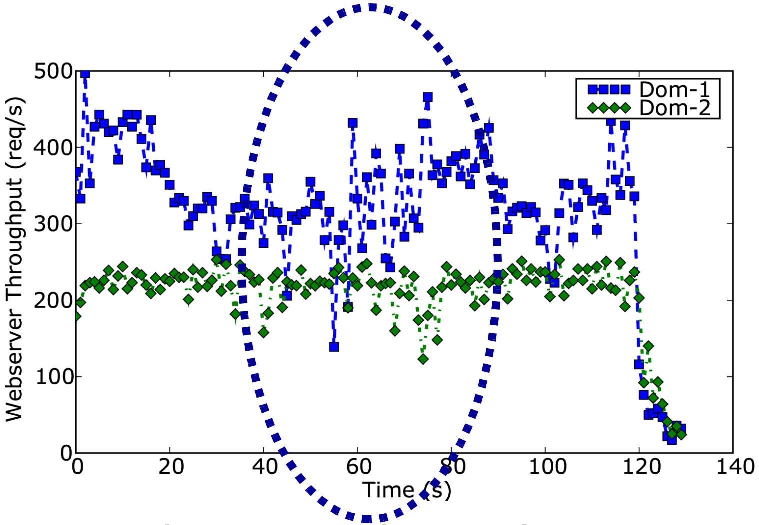


Dom-0 CPU utilization

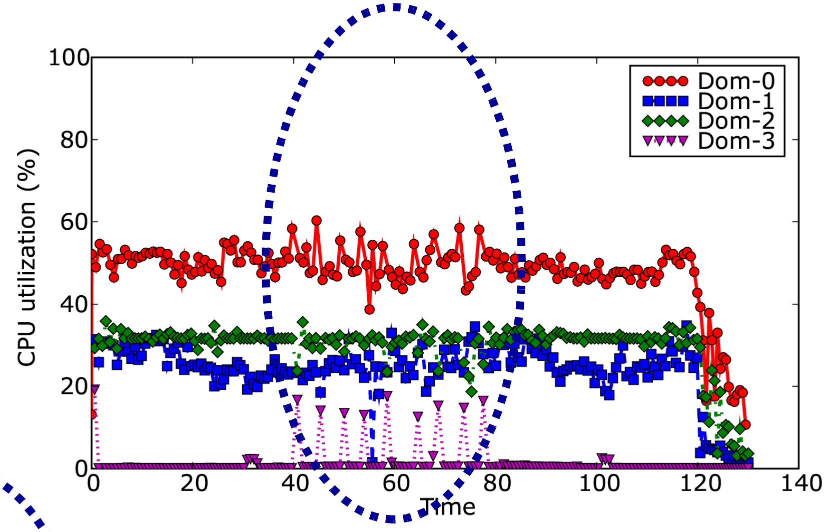
Providing Qos in driver domains

- Problem: No way to control how much CPU each VM consumes in Dom-0
- ShareGuard
  - Periodically monitor CPU usage using XenMon
  - IP tables in Dom-0 turn off traffic for offenders
  - Added similar functionality to *netback*
- Repeated experiment, with VM-3 restricted to 5% CPU in Dom-0

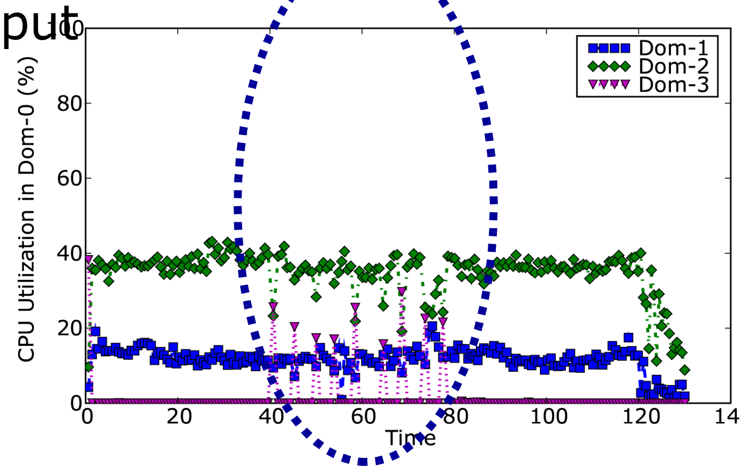
ShareGuard in action



Webserver throughput



CPU utilization



Dom-0 CPU utilization

CPU in Dom-0 for Dom-3 is 4.42% over the run

The big picture

- Both SEDF-DC, ShareGuard depend on XenMon
- ShareGuard only works for network I/O, SEDF-DC is workload agnostic
- ShareGuard is independent of the CPU scheduler
- ShareGuard is intrusive (actively blocks traffic) whereas SEDF-DC is more passive and transparent

Conclusion

- Performance isolation is crucial in multi-user environments
- Current I/O model in Xen breaks performance isolation
- Mantra: Measure, Allocate, Control
- XenMon, SEDF-DC, ShareGuard are steps in this direction
- Hardware support will (hopefully) enable more comprehensive solutions

Thanks!

Questions?

http://sysnet.ucsd.edu/~dgupta
dgupta@cs.ucsd.edu

Resource Isolation

- Common resources: CPU, Disk, Memory, Network
- Spatial (disk, memory) vs. Temporal resources (CPU)
- Partitioning vs. Time sharing
- Quality of Service
  - Availability
  - Cost of access
- CPU is special: now just how much, but also *when?*

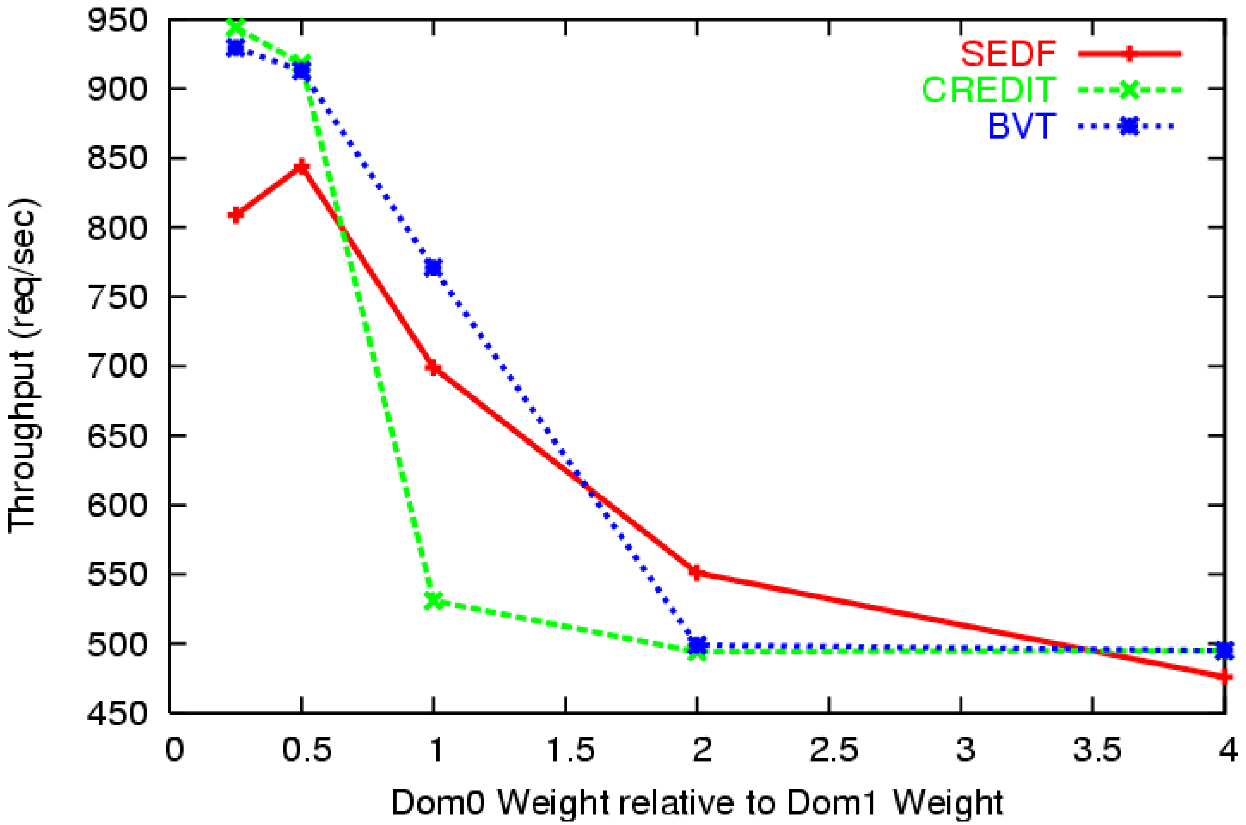
Isolated Driver Domains

- Are they happening?
- We *need* accurate accounting. But how?
- ShareGuard only works for network I/O. What about disk?
- We've tried
  - Memory page exchanges [USENIX 05]
  - Weighted packet counts
  - Instrumentation?

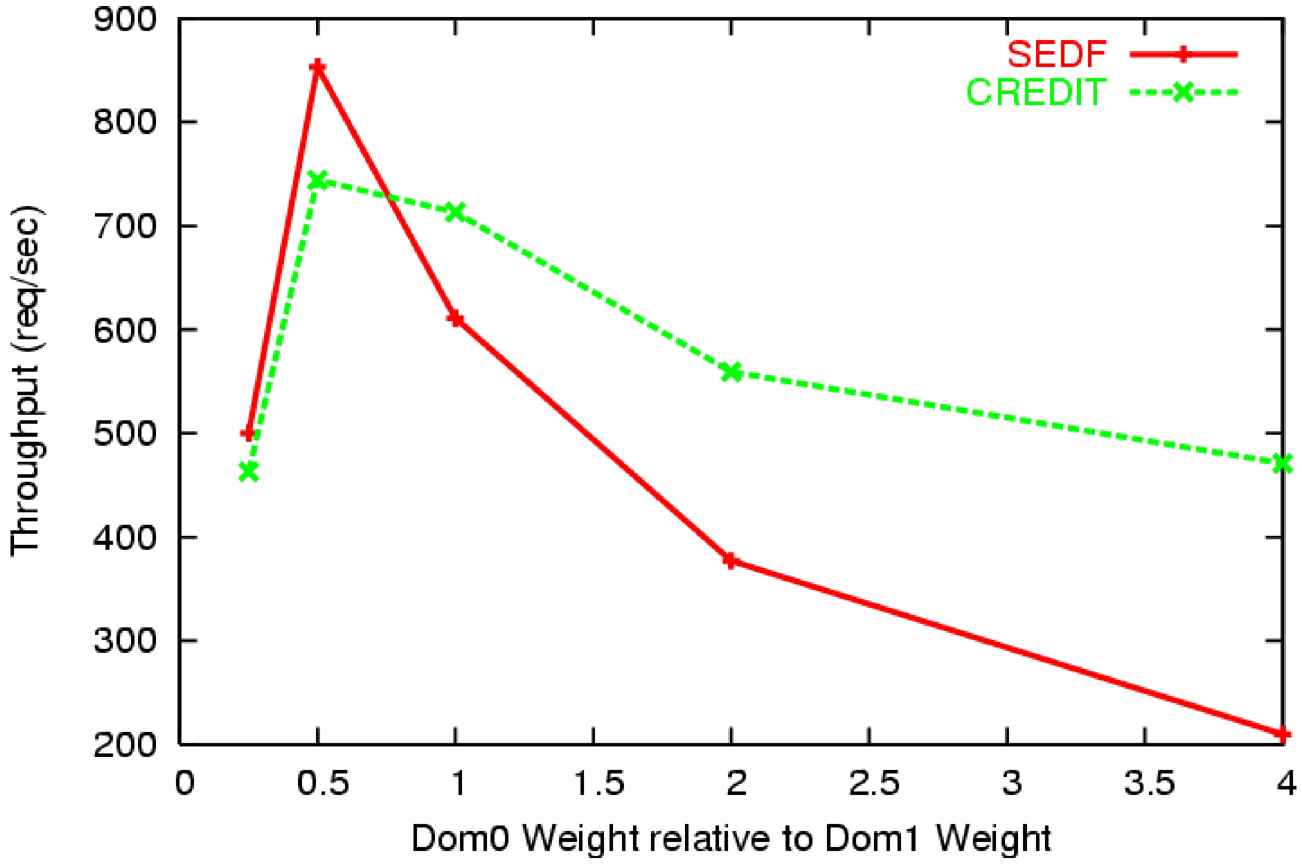
Allocating resources for IDD

- IDDs are critical for I/O performance
- Scheduling parameters have significant impact
- Different schedulers need different tuning
- Example: on a uni-processor machine, for a web server under load, is it better to give more weight to the VM or to Dom-0?

Work Conserving



Non work conserving



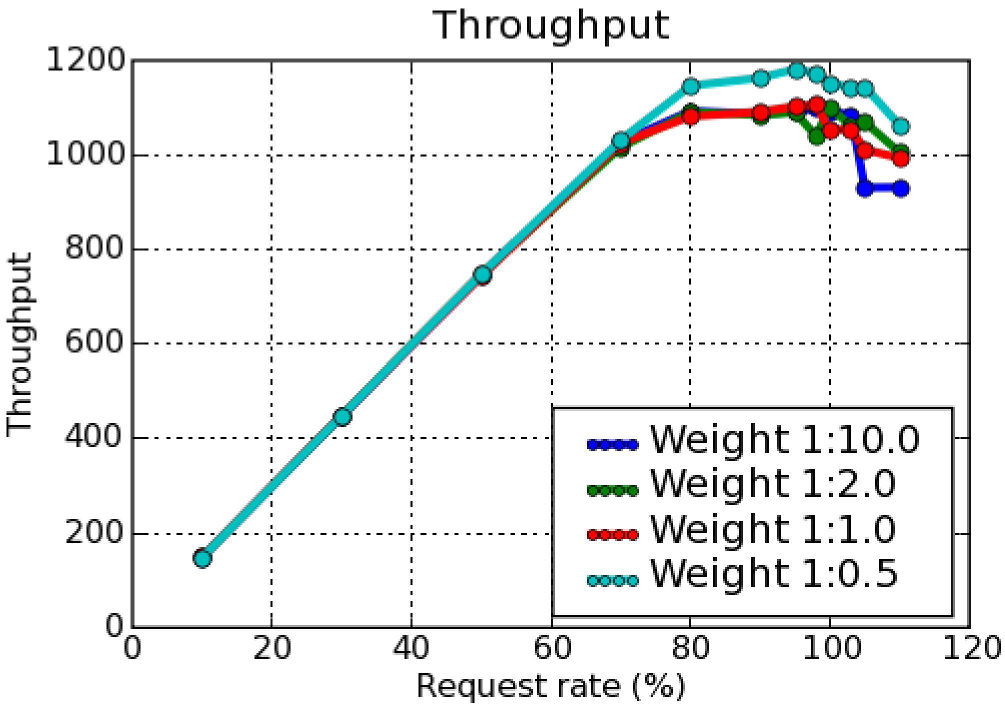
Other challenges

- Separating costs in presence of multiple drivers
- CPU partitioning for other kinds of I/O traffic
- Isolation of low level resources (PCI bus bandwidth, L1/L2 caches etc)
- Choosing and configuring the right scheduler

The tale of 3 schedulers

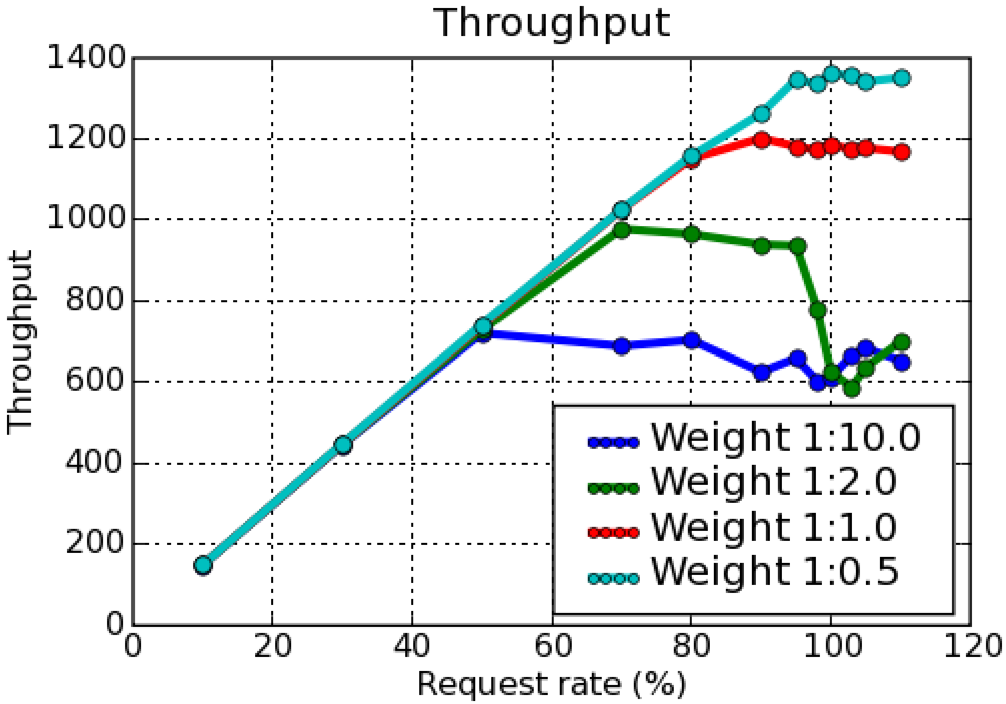
- Three schedulers in less than two years
- Do end users care?
- Schedulers have demonstrated performance problems
- Questions
  - Which scheduler to use?
  - How to configure parameters?
  - Should IDDs be treated specially?

SEDF



Not very sensitive to Dom-0 weights

BVT



Higher weight actually performs worse! Lower weight is better

Outline

- Background and Motivation
- Controlling aggregate CPU consumption
- QoS in the driver domain
- Configuring scheduler parameters
- Conclusion