

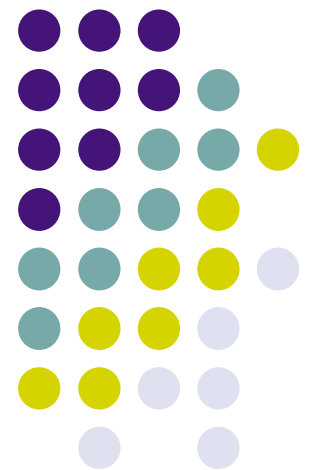
Performance Isolation in Xen

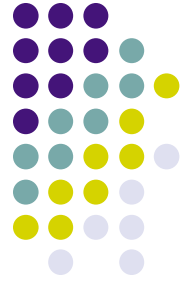
Diwaker Gupta (UC San Diego)

Lucy Cherkasova (HP Labs)

Rob Gardner (HP Labs)

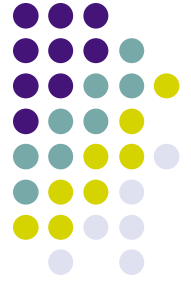
Amin Vahdat (UC San Diego)





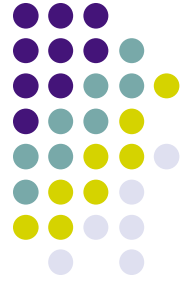
Outline

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



Introduction

- VMs provide *fault isolation*. Enterprise customers want *performance isolation*.
- What is 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*?



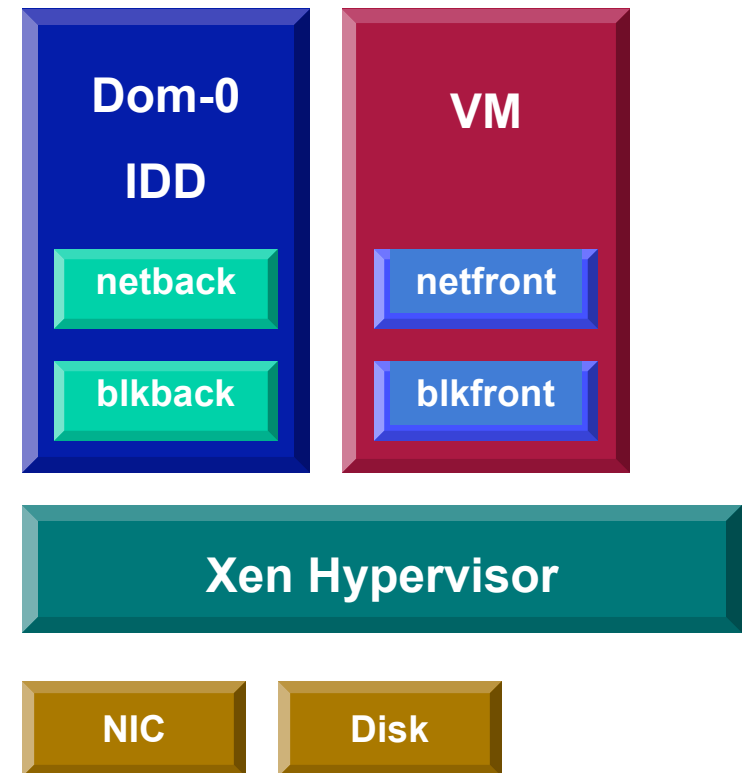
Resource Isolation

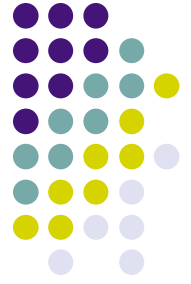
- Common resources: CPU, Disk, Memory, Network
- Spatial (disk, memory) vs. Temporal resources (CPU)
- Partitioning vs. Time sharing
- Quality of Service
 - Availability
 - Cost of access
- CPU is special: now just how much, but also *when?*



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

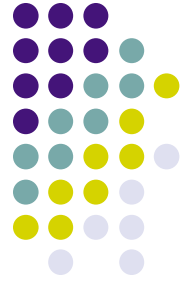




General Strategy

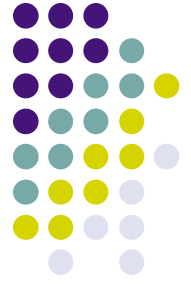
- Measure
 - Profiling tools
- Allocate
 - Modifications to the scheduler
- Control
 - Mechanisms to control resource usage

Our work focuses on CPU and network I/O.



Profiling Tools

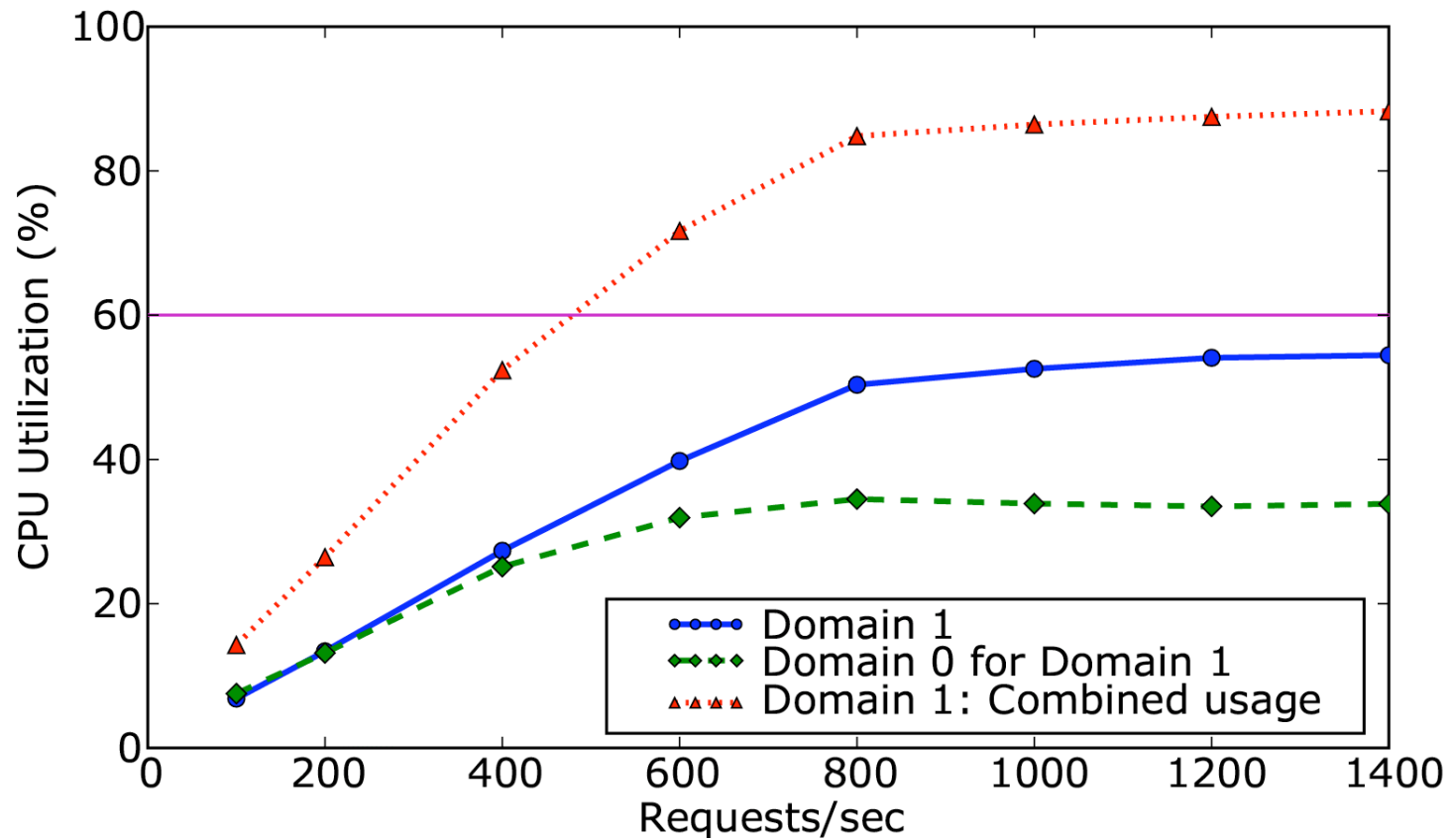
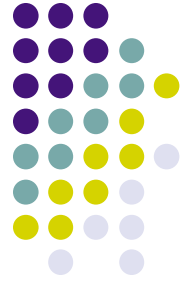
- XenMon
 - Uses trace events – fairly easy to add new metrics in the framework
 - Useful for analyzing schedulers (blocking time, waiting time etc)
 - Metrics *per execution period*
- Other tools
 - libxenstat and XenTop
 - xenoprofile

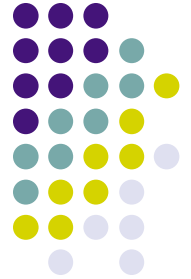


Problem: Accounting in IDD

- Scenario
 - Two enterprise customers: CPU intensive workload and interrupt driven workload (web server)
 - Given equal shares, do they *really* get equal shares?
- Example
 - Single CPU system, SEDF, non work-conserving
 - VM-1: web server, 60%
 - Dom-0: driver domain, 40%
 - How to control aggregate CPU consumption?

Aggregate CPU consumption



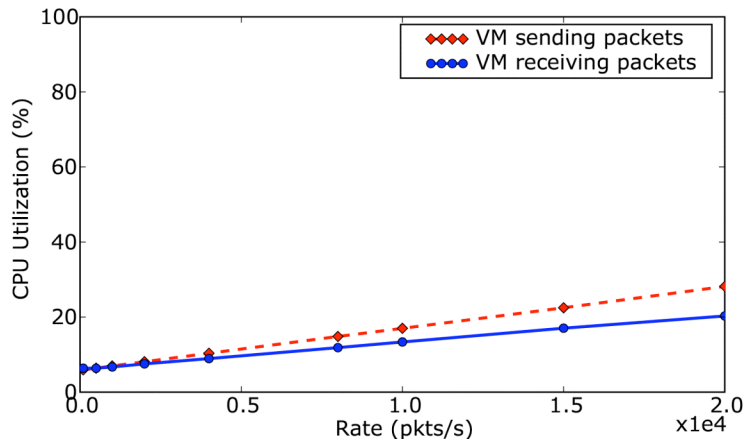


Problem: Accounting in IDD

- 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
- Heuristic for partitioning: CPU overhead is proportional to the amount of I/O



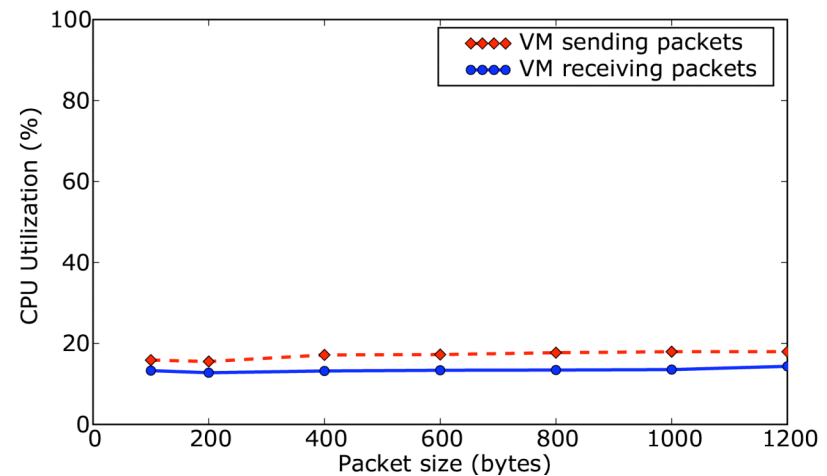
Packet counting in *netback*

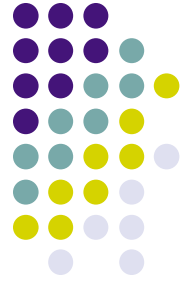


CPU overhead is proportional to rate of packets

CPU overhead is independent of size of packets

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

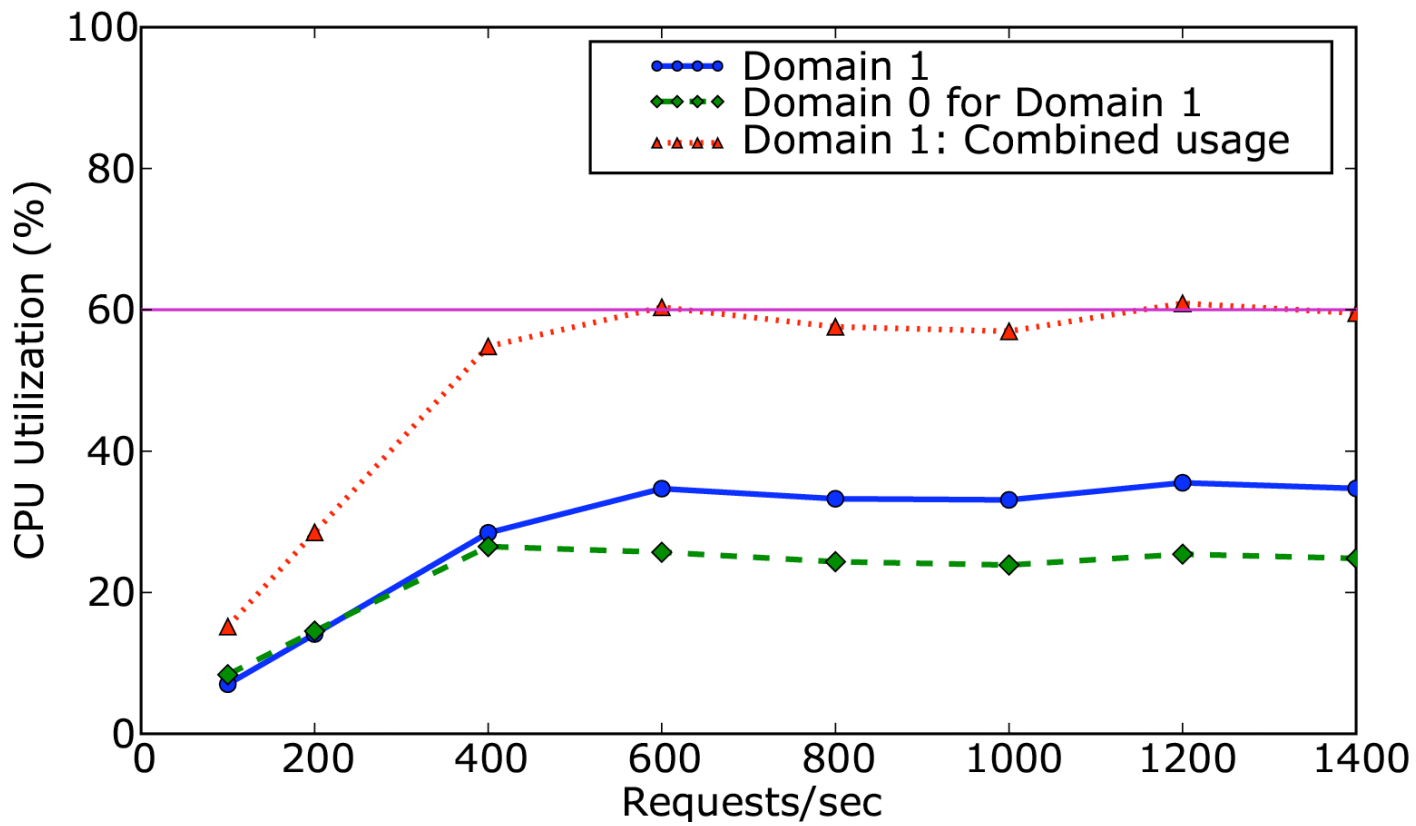
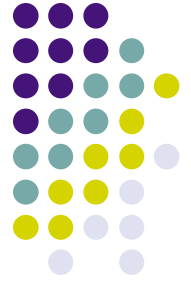


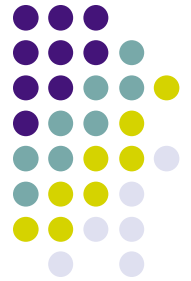


SEDF Debt Collector

- 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 in action

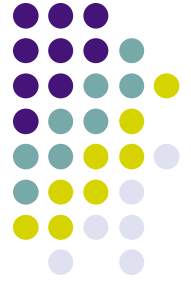




Problem: Accounting in IDD

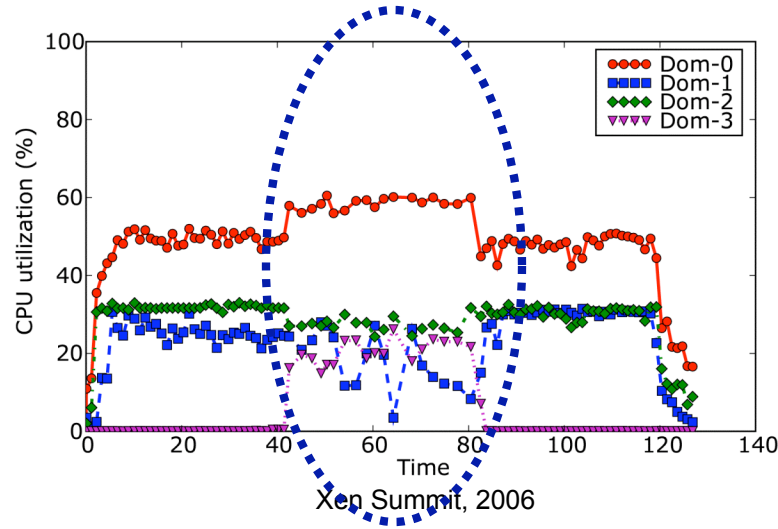
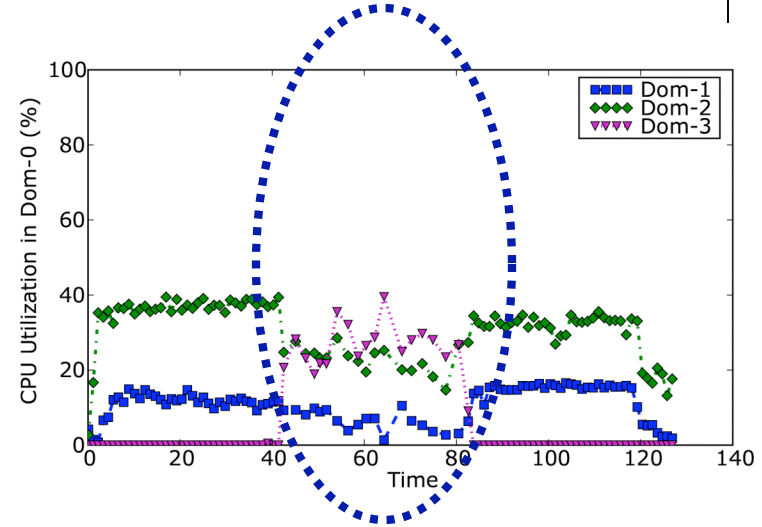
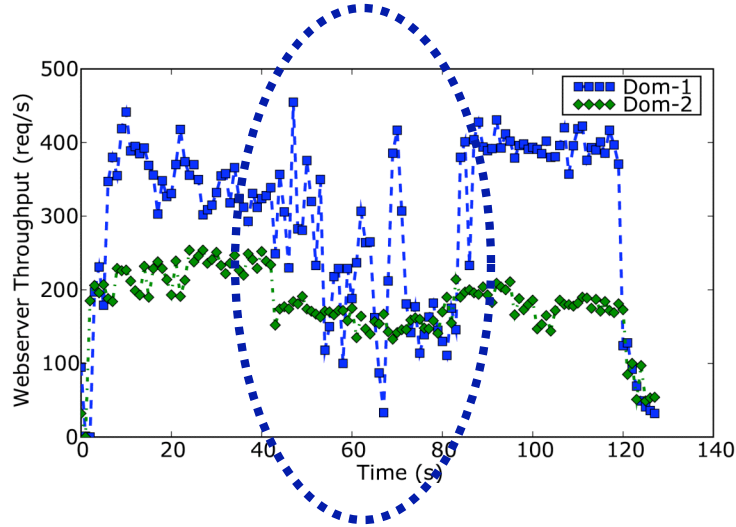
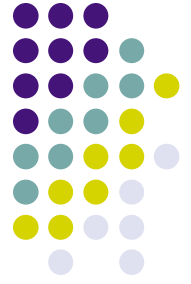
- SEDF-DC addresses problem for SEDF in single processor case
- Idea can be extended to other schedulers (such as Credit)
- Spread debt across multiple execution periods to avoid starvation
- But
 - Debt can still be very high
 - QoS in the driver domain?

Controlling resource consumption in IDD



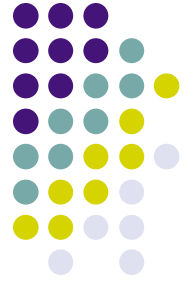
- Scenario
 - SEDF, dual processor machine, non work-conserving mode
 - Dom-1: Web server, 33% on CPU-2 (serving 10KB files)
 - Dom-2: Web server, 33% on CPU-2 (serving 70KB files)
 - Dom-3: File transfer, 33% on CPU-2
 - Dom-0: 60% on CPU-1
- Goal: file transfer in VM-3 should not affect web servers in VM-1 and VM-2

No QoS in IDD



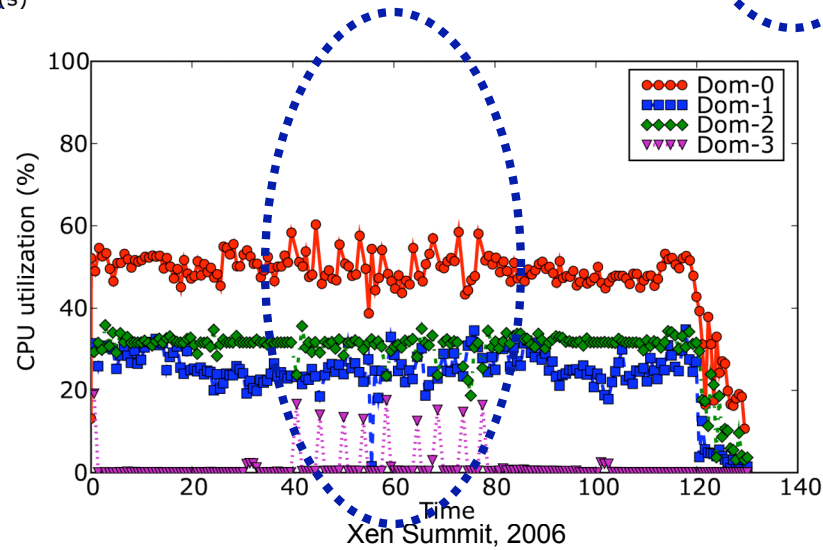
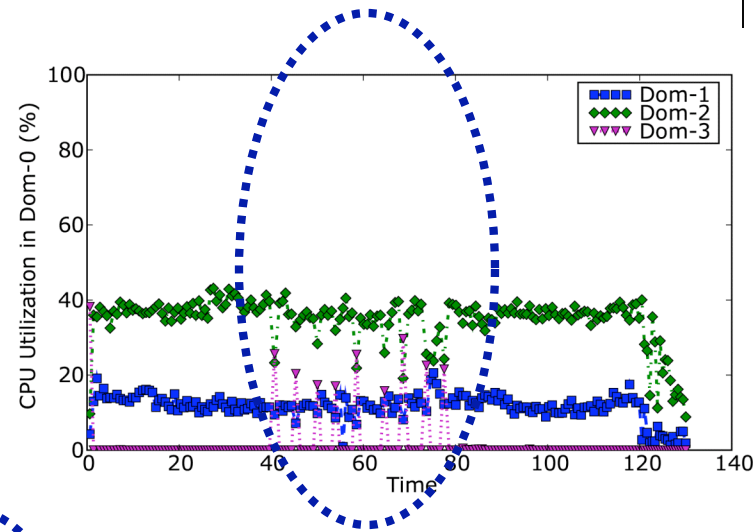
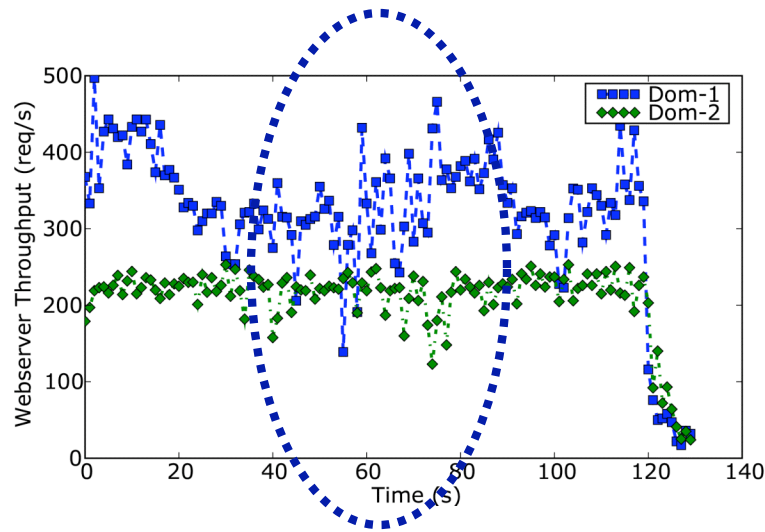
Xen Summit, 2006

Controlling resource consumption in IDD

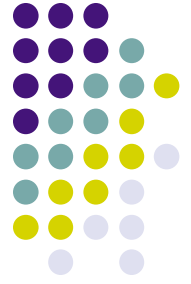


- Problem: No way to control how much CPU each VM consumes in Dom-0
- ShareGuard
 - Periodically monitor CPU usage using XenMon
 - IP tables in Dom-0 turn off traffic for offenders
 - Added similar functionality to *netback*
- Repeated experiment, with VM-3 restricted to 5% CPU in Dom-0

ShareGuard in action

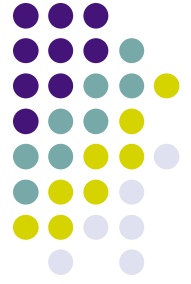


CPU in Dom-0 for Dom-3 is 4.42% over the run



Isolated Driver Domains

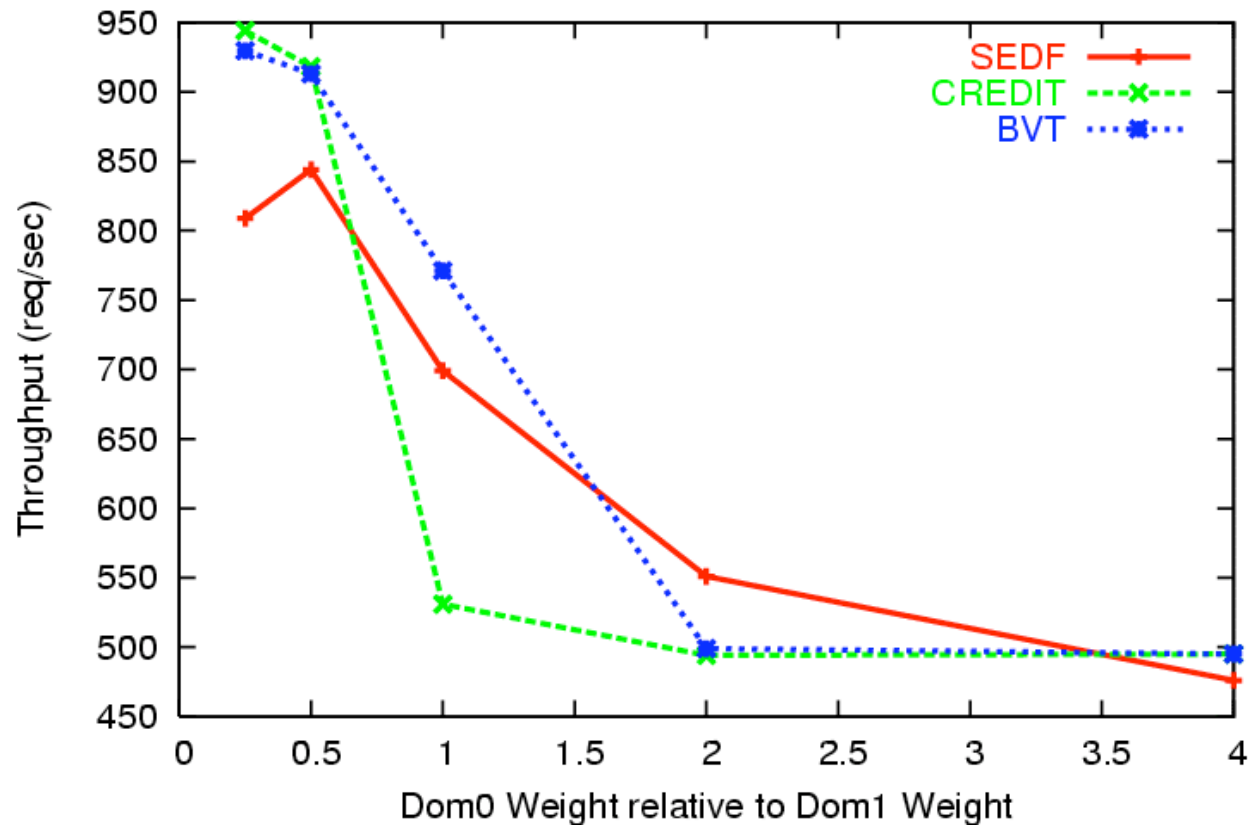
- Are they happening?
- We *need* accurate accounting. But how?
- ShareGuard only works for network I/O. What about disk?
- We've tried
 - Memory page exchanges [USENIX 05]
 - Weighted packet counts
 - Instrumentation?

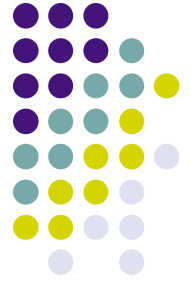


Allocating resources for IDD

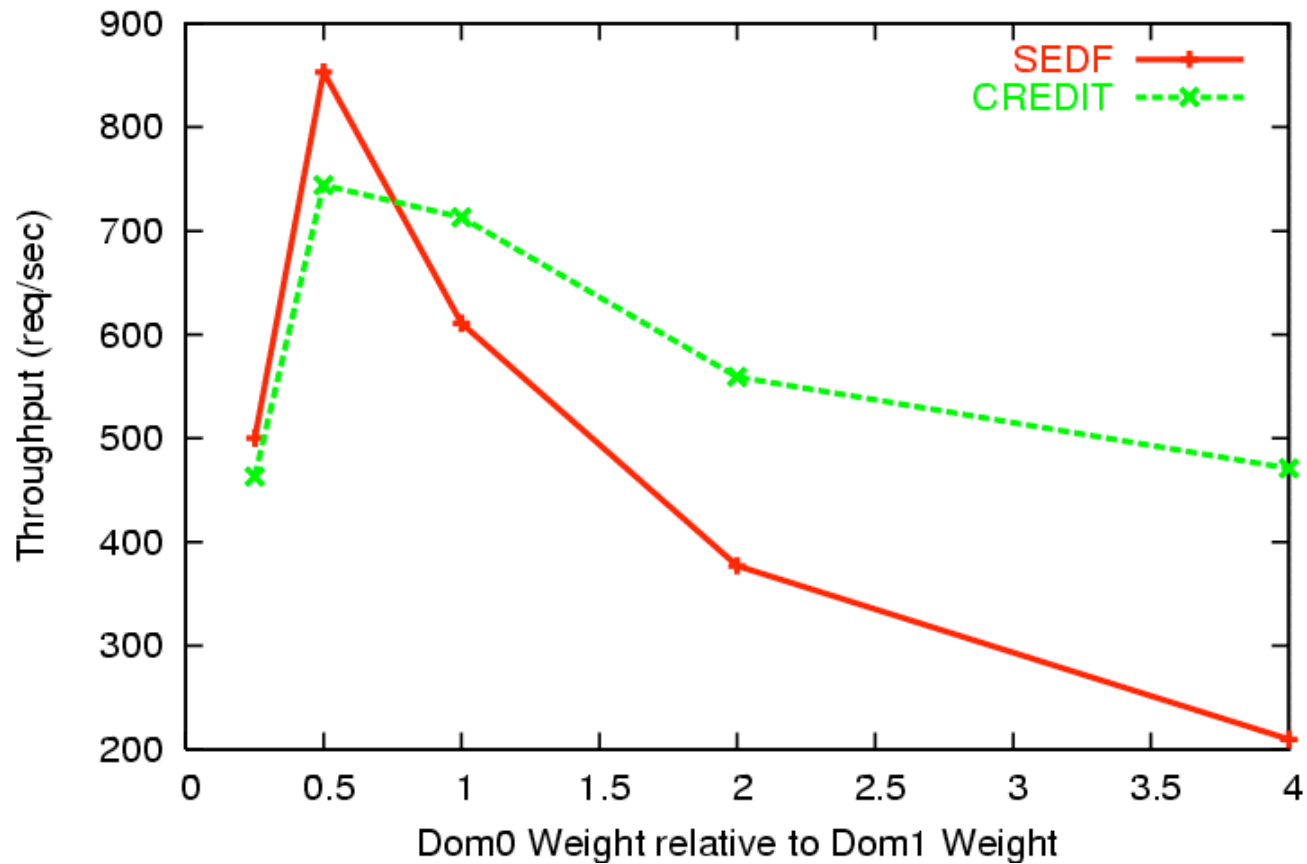
- IDDs are critical for I/O performance
- Scheduling parameters have significant impact
- Different schedulers need different tuning
- Example: on a uni-processor machine, for a web server under load, is it better to give more weight to the VM or to Dom-0?

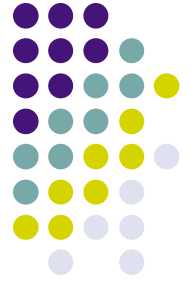
Work Conserving





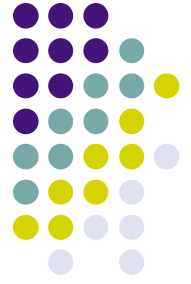
Non work conserving





Other challenges

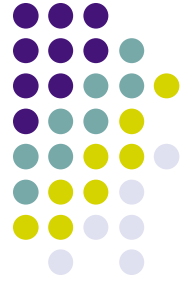
- Separating costs in presence of multiple drivers
- CPU partitioning for other kinds of I/O traffic
- Isolation of low level resources (PCI bus bandwidth, L1/L2 caches etc)
- Choosing and configuring the right scheduler



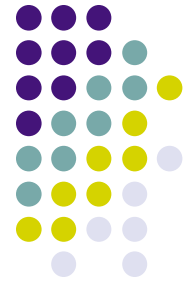
Conclusion

- Xen doesn't have good performance isolation
- Mantra: Measure, Allocate, Control
- XenMon, SEDF-DC, ShareGuard are steps in this direction
- More work needed for SMP, non-network I/O, multiple back-ends
- Does the Xen community care about performance isolation?

Thanks!



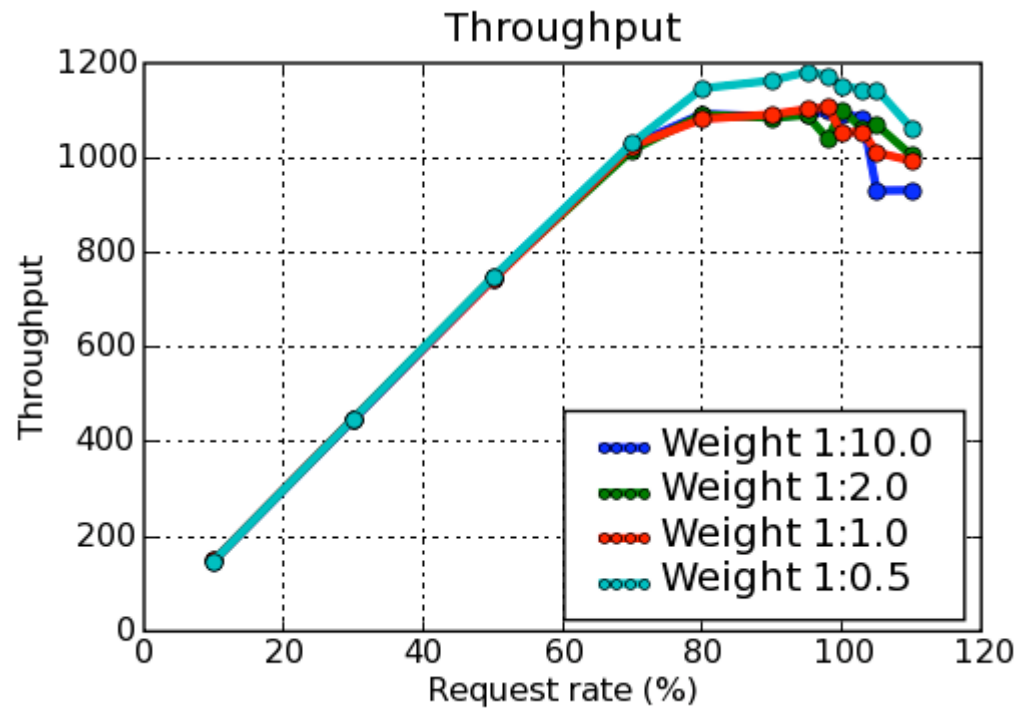
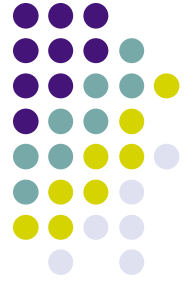
Questions?



The tale of 3 schedulers

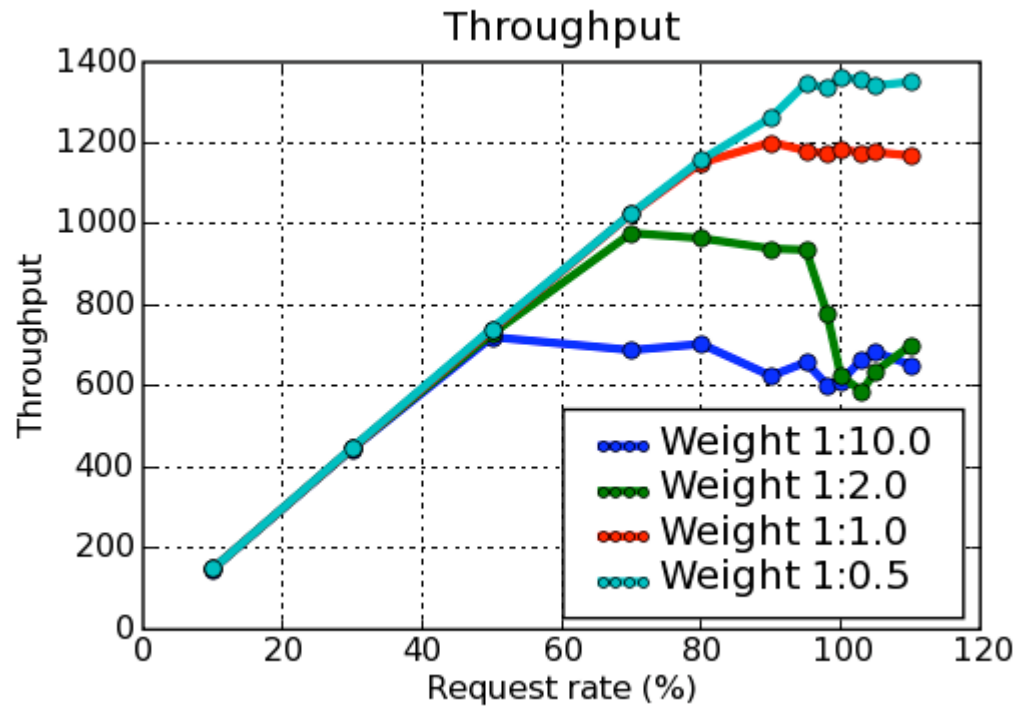
- Three schedulers in less than two years
- Do end users care?
- Schedulers have demonstrated performance problems
- Questions
 - Which scheduler to use?
 - How to configure parameters?
 - Should IDDs be treated specially?

SEDF



Not very sensitive to Dom-0 weights

BVT



Higher weight actually performs worse! Lower weight is better