MemX: Supporting Large Memory Workloads in Xen Virtual Machines

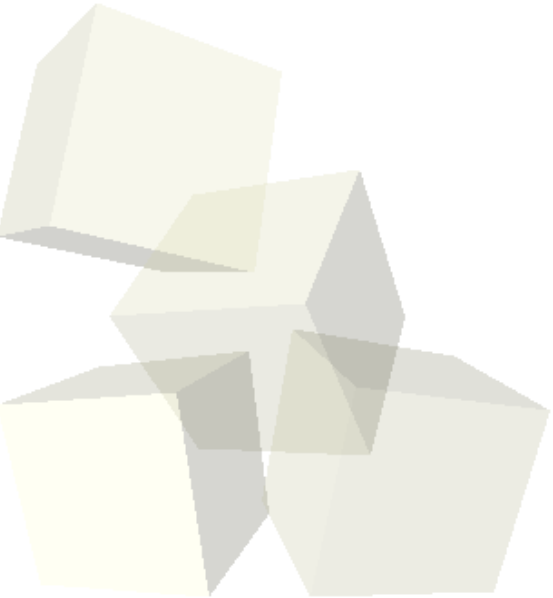
State University of New York at Binghamton

Michael R. Hines & Kartik Gopalan

Talk Outline

- **Motivation:**
  - Why network memory for Virtual Machines?
- **Design:**
  - Predecessor MemX system (non-virtualized)
  - Xen-background
  - MemX + Xen
- **Performance:**
  - Microbenchmarks
  - Application Speedups
- **Final Thoughts**

Motivation



Motivation (1)

- **Increased use of Virtual Machines:**
  - Enterprise, Grid, and Scientific applications
  - Deployed across high-speed clusters
- **Memory & I/O handling:**
  - ***How should VMs off-load memory or I/O usage?***
  - Where should you implement it?
  - How fast can you virtualize the solution?
  - How transparent can it be?
- **Our strategy:**
  - Develop block device tailored to network RAM
  - Find the right place within the VM
  - Ensure scalable solution for all VMs
- **What we're not targeting:**
  - Software DSM & Consistency Protocols
  - Cooperative caching systems

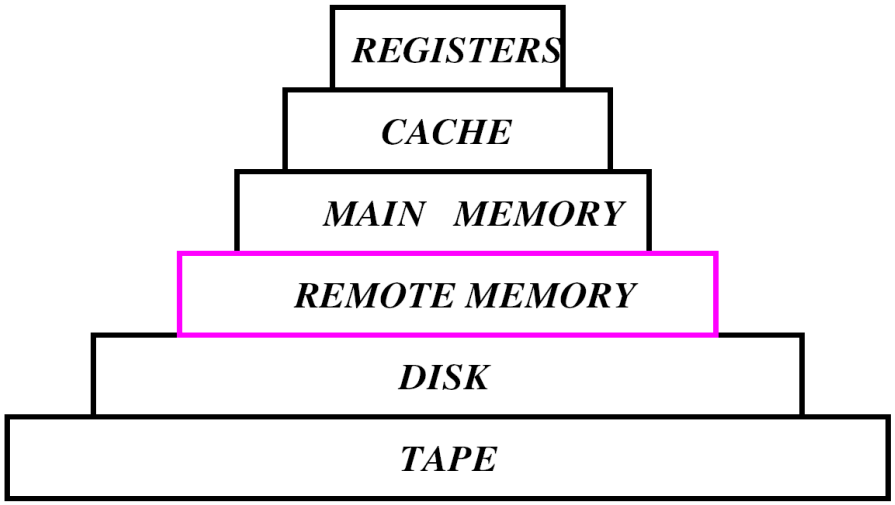
Motivation (3)

- **We need support for:**
  - Legacy Apps
    - Without special APIs, recompiling, or OS modifications
  - Easy VM deployment
    - Choice of virtualization placement
  - Server Maintenance:
    - Without interruption
  - Client Migration:
    - Without interruption

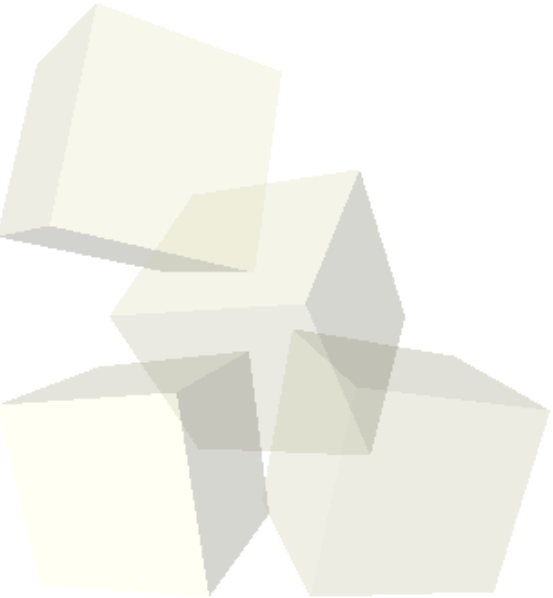
- **MemX:**
  - **Supports all of the above**
  - Exploits growing LAN speeds (commodity GigE)
  - Deployed on under-utilized servers
  - Provides microsecond response-times

Motivation (2)

- **Related Work** (not new):
  - **None designed for VMs**
  - **Rarely used or inactive**
  - [Anderson] [Comer], [Feeley] [Felten], [Koussih], [Flouris], [Stark]
  - **Expensive interconnects**: Myrinet, Infiniband
  - **Application/OS modifications**
  - **Both Micro & Monolithic kernel implementations**

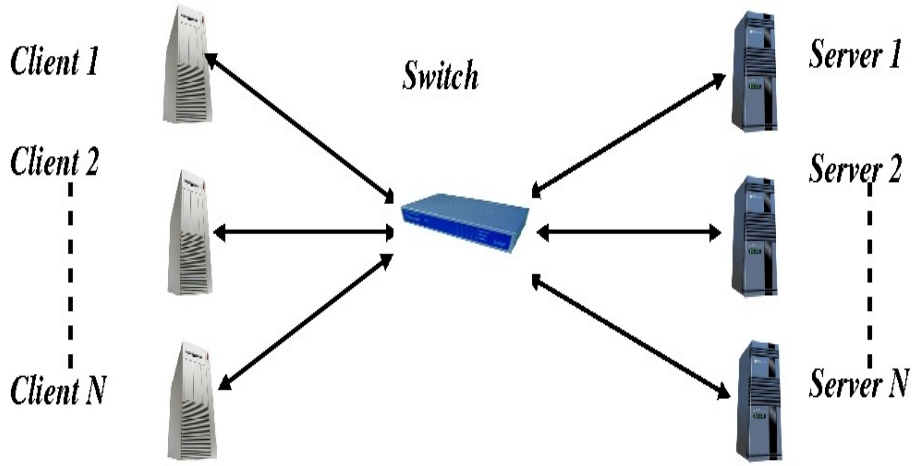


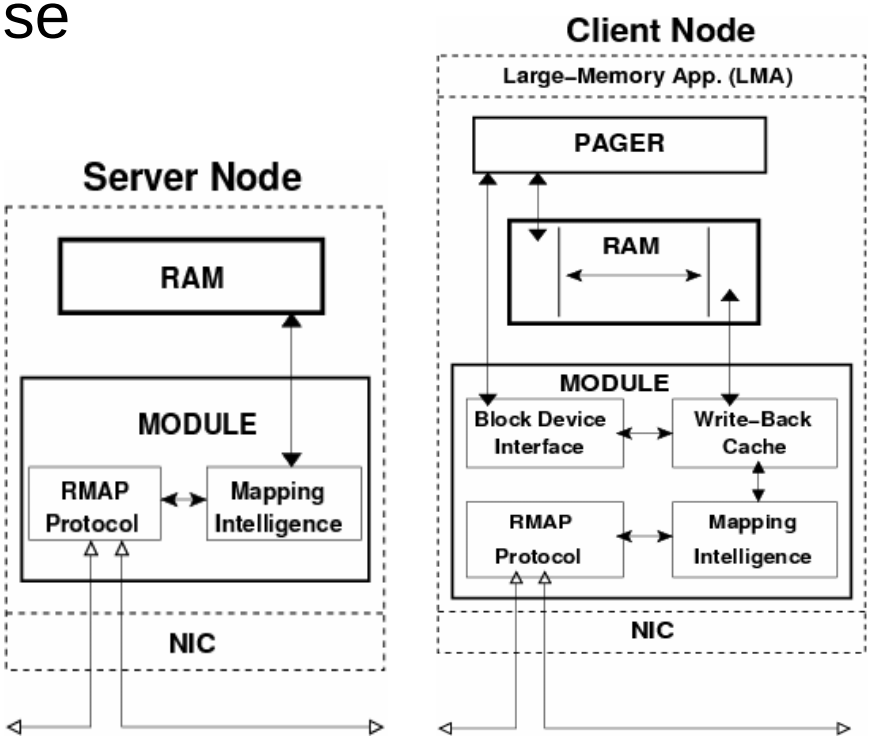
Design Decisions



MemX in Native Linux

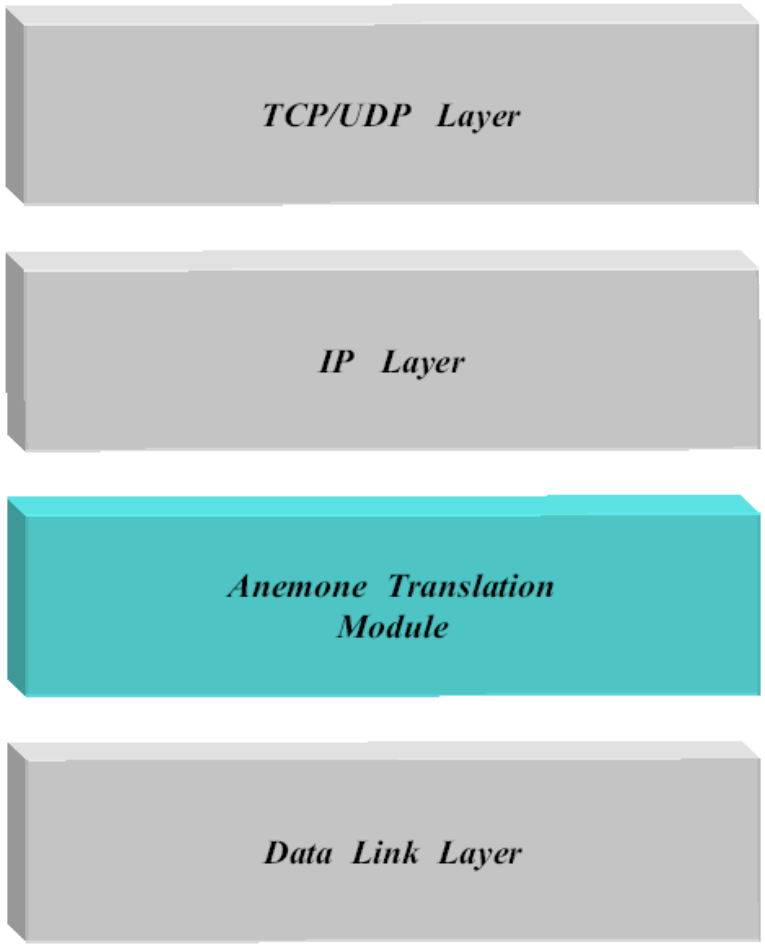
- **Features:**
  - Kernel-module
  - Distributed Resource Discovery
  - Dynamic load-distribution
    - And server-migration
  - Interrupt-mode response
  - 3 modes of operation
    - Block Device
    - Pager
    - mmap
- **RMAP Protocol**
  - TCP/IP Bypass
  - Internal retransmit & fragmentation
  - Low-latency

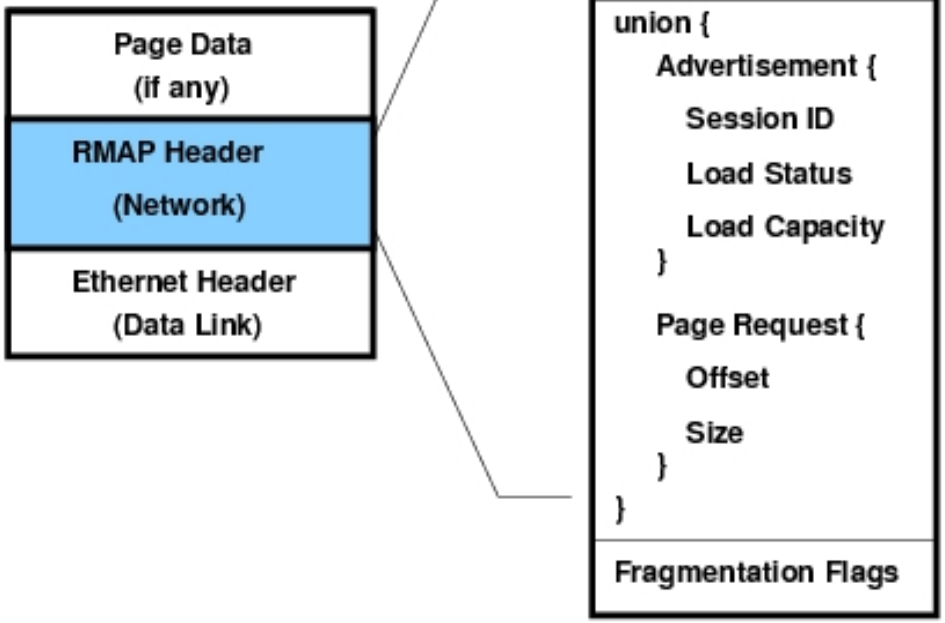




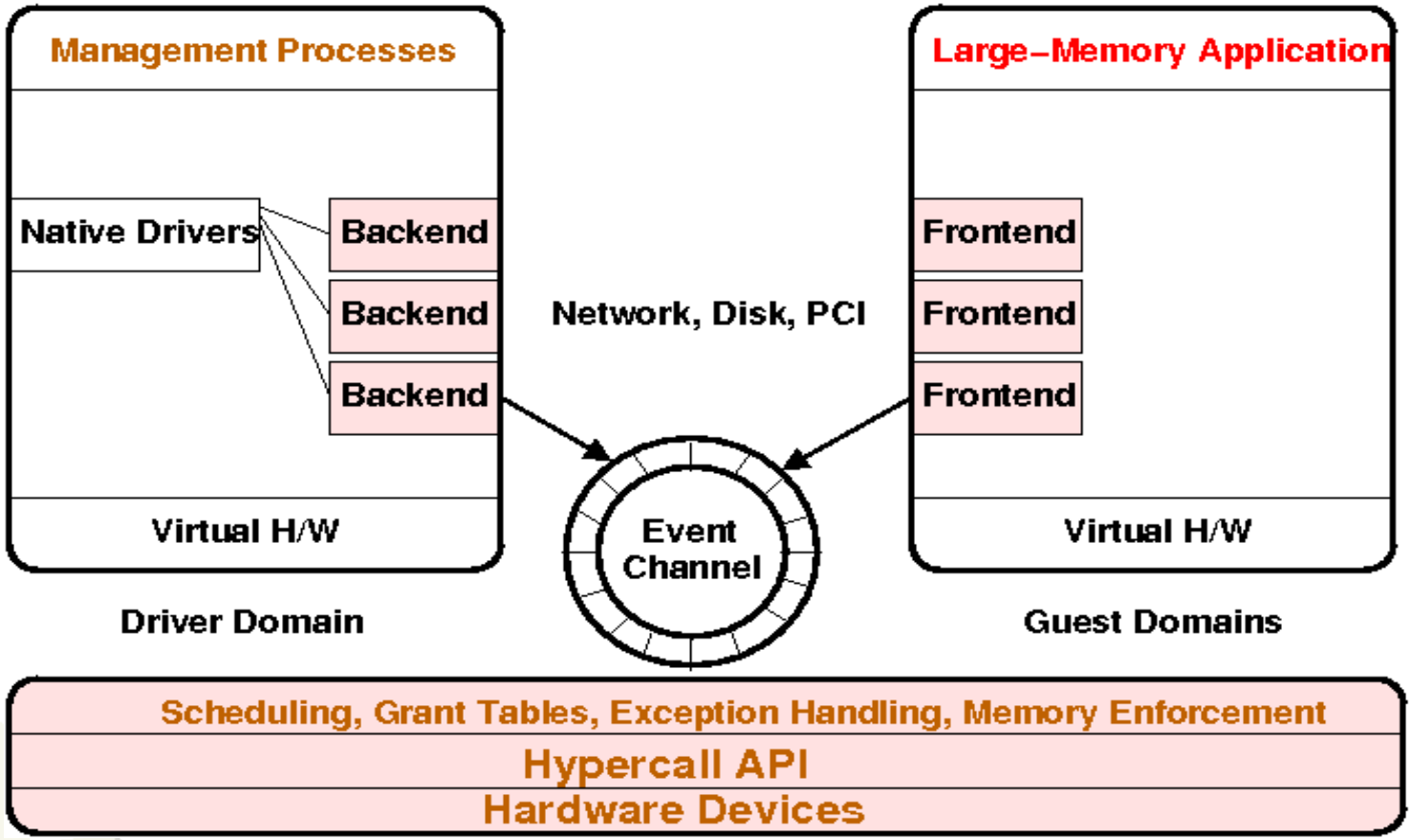
MemX in the Protocol Stack

Protocol Stack



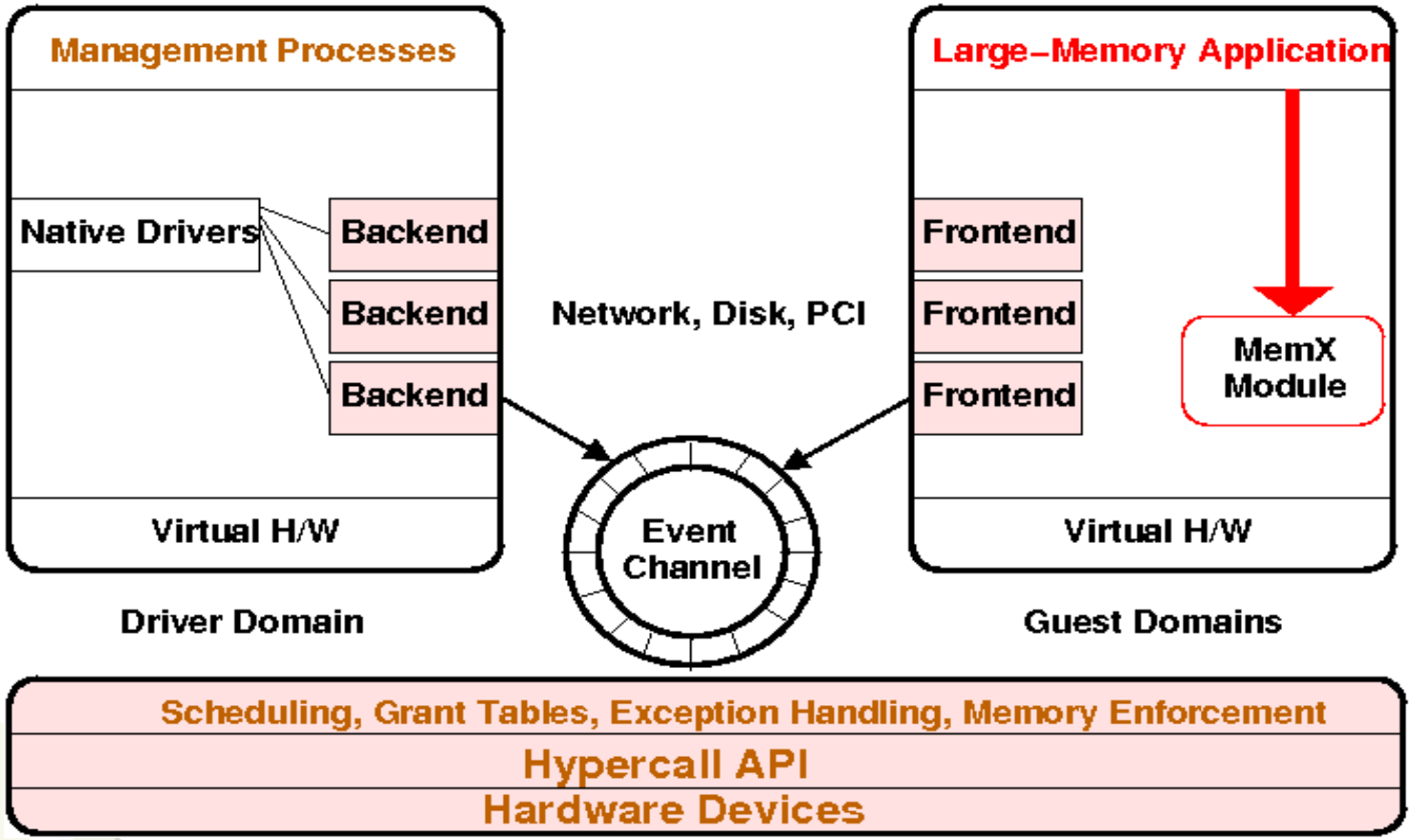


Xen Background



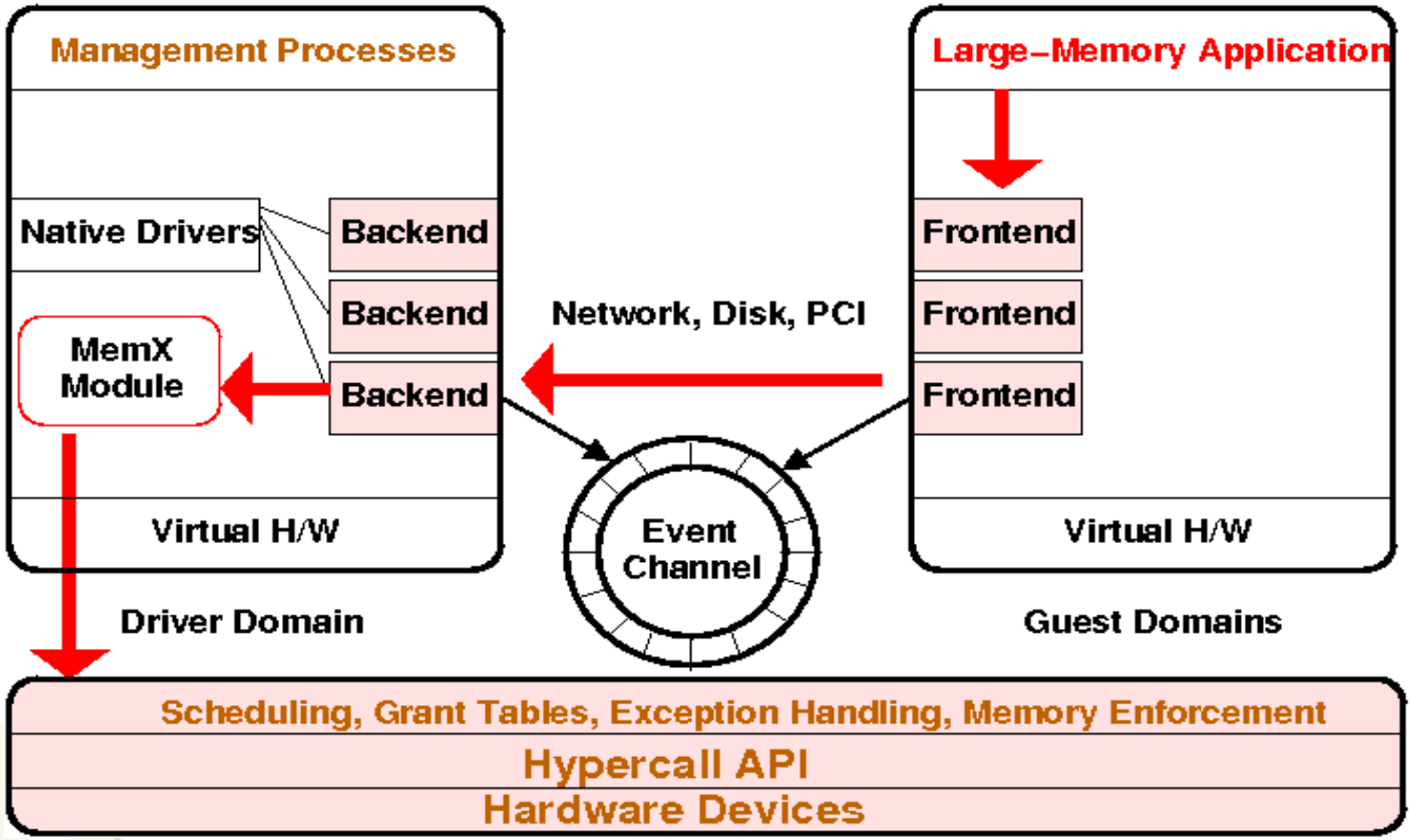
- **Domain I/O signaling: Event-Channels**
  - Used to setup data exchanges
  - Also used for inter-VM IPC
- **Data Exchanges: Grant-Tables**
  - **Sharing:** for synchronous I/O
  - **Flipping:** for asynchronous I/O

Option 1: MemX-DomU



- Page-flipping occurs for each network transmission
- 3 pages/request for each 1500 byte MTU -> overhead.
- One client module per VM: Simplifies VM migration

Option 2: MemX-DD

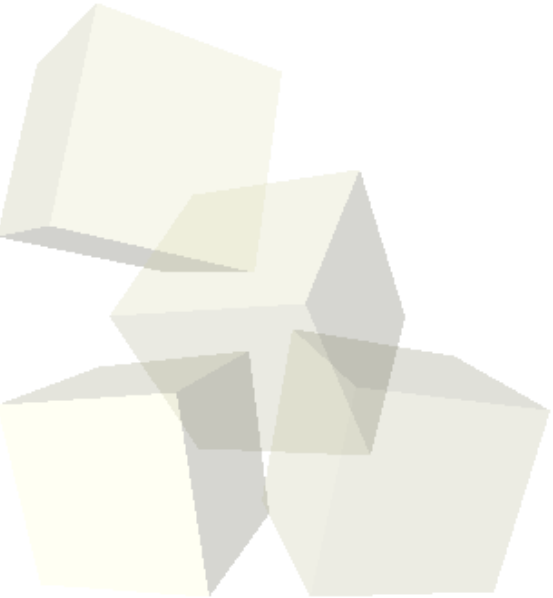


- Less cross-domain traffic: single 4KB shared page
- Client-module can be shared across multiple DomUs
- VM migration more difficult, but not impossible

Other options

- **Option 3: Server in a DomU**
  - VM solely dedicated to contribute memory
  - No significant functional benefit
- **Option 4: Expanding pseudo-physical address space**
  - Requires significant hypervisor changes
  - Functionally equivalent to MemX-DomU and MemX-DD
  - Might be viable with support for nested paging and HVMs.
  - Blurs the the boundaries of Machine/VMM isolation

Performance Results



Performance Evaluation

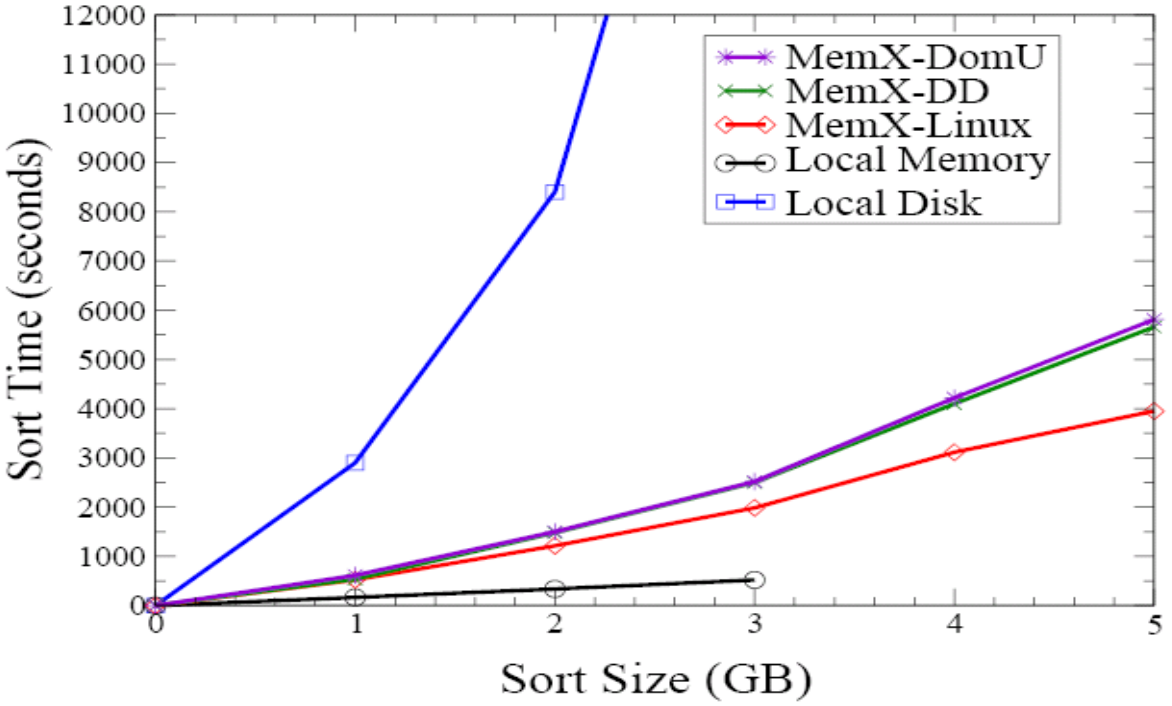
- **8-node testbed**
  - 4GB of DRAM / node:
    - → 24 GB of collective usable cluster-wide memory
  - Client memory limited to 512 MB
  - Dual-core 2.8 GHz processor
  - Broadcom Ethernet NIC

- **Disk Baselines:**
  - 160GB SATA Disk
  - 7200 RPM, 16MB on-disk cache
  - 8ms average seek time

	MemX-Linux	*MemX*-Dom0	*MemX*-DD	*MemX*-DomU	Virtualized Disk
Kernel RTT	85 usec	95 usec	95 usec	115 usec	8.3 millisec

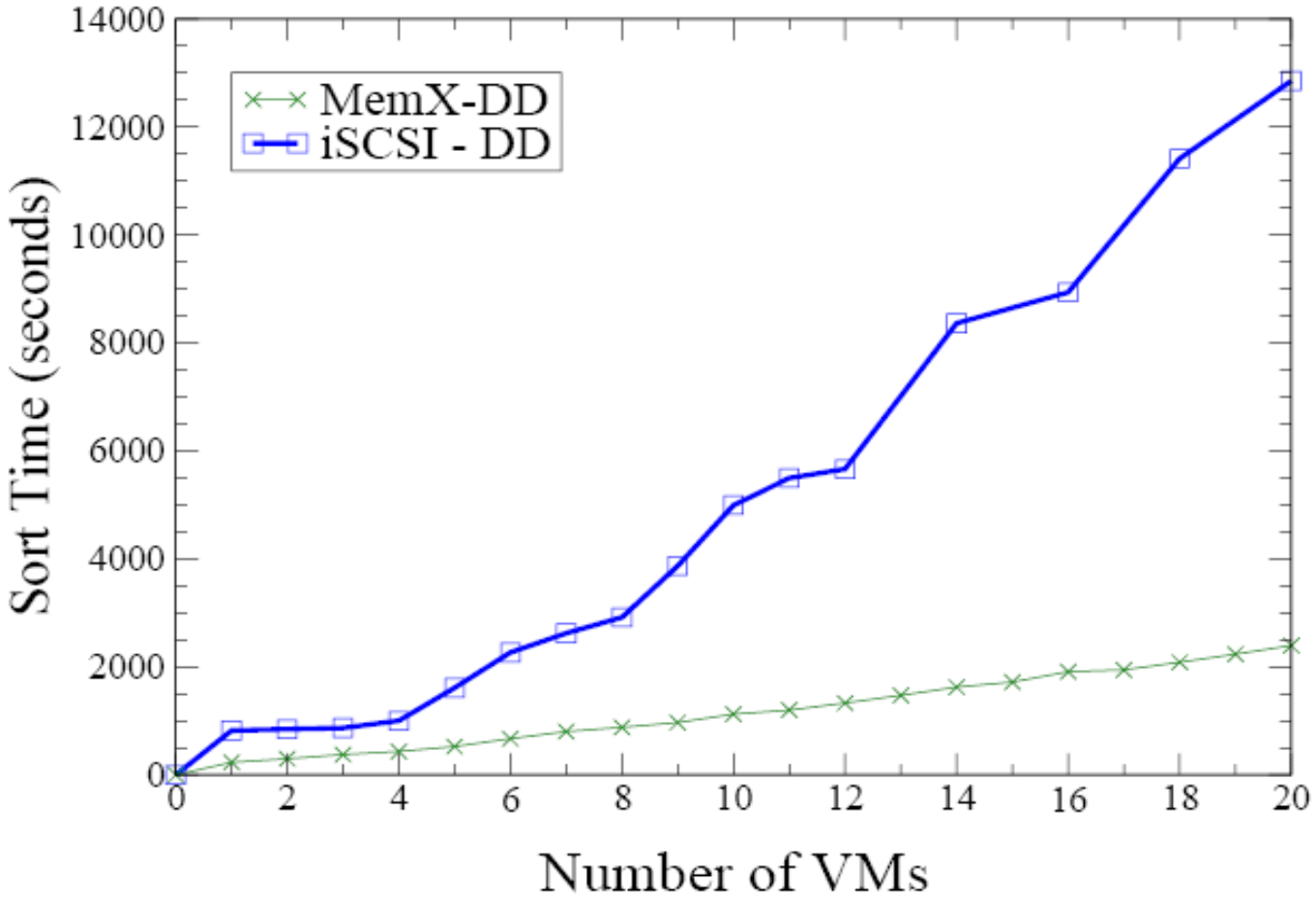
Completion Time vs. Problem Size

Appli-cation	Mem Size	Client Mem	*MemX*-Linux	*MemX*-DD	Virtual Disk
Quicksort	5GB	512 MB	65 min	93 min	> 10 hrs
Povray	(6GB)	512 MB	48 min	61 min	> 10 hrs
Povray	(13GB)	1 GB	93 min	145 min	> 10 hrs
IS-NPB	(6GB)	512 MB	83 min	126 min	> 10 hrs
DC-NPB	(10GB)	1 GB	152 min	217 min	> 10 hrs
sql-bench	(4GB)	512 MB	114 min	208 min	> 10 hrs
NS2	(5GB)	1 GB	175 min	228 min	> 10 hrs



MemX / iSCSI comparison

- **Experiment:**
  - 4 iSCSI nodes, one disk per node, 1 4GB MemX client
  - 400 MB Quicksort per VM, 100 MB memory per VM

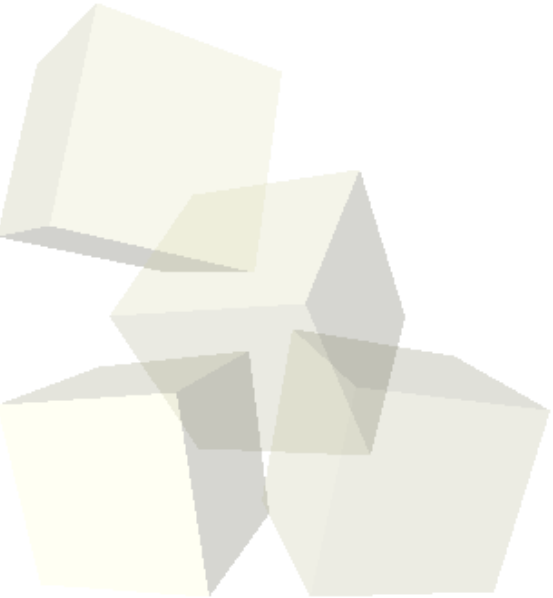


Conclusions

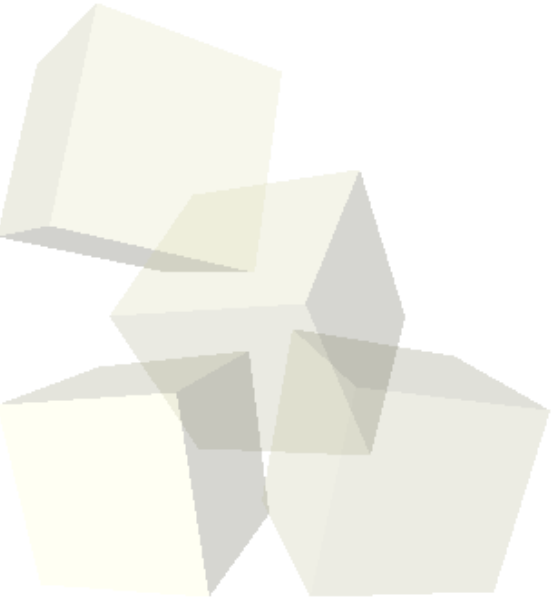
- **MemX can support large-memory Apps for Virtual Machines**
  - Distributed & Scalable
- **Supports the latest VM technology**
  - Xen & Linux
  - Migration, Maintenance and Consolidation still possible
- **Pluggable & Transparent**
  - Uses a custom protocol for latency
  - Uses kernel modules for efficiency and transparency
- **Has many modes-of-operation**
  - Driver domain, VM domain, and regular Linux
  - Files, Paging, and memory-mapping
- **Provides for 2-orders of magnitude speedup**

Thank you!

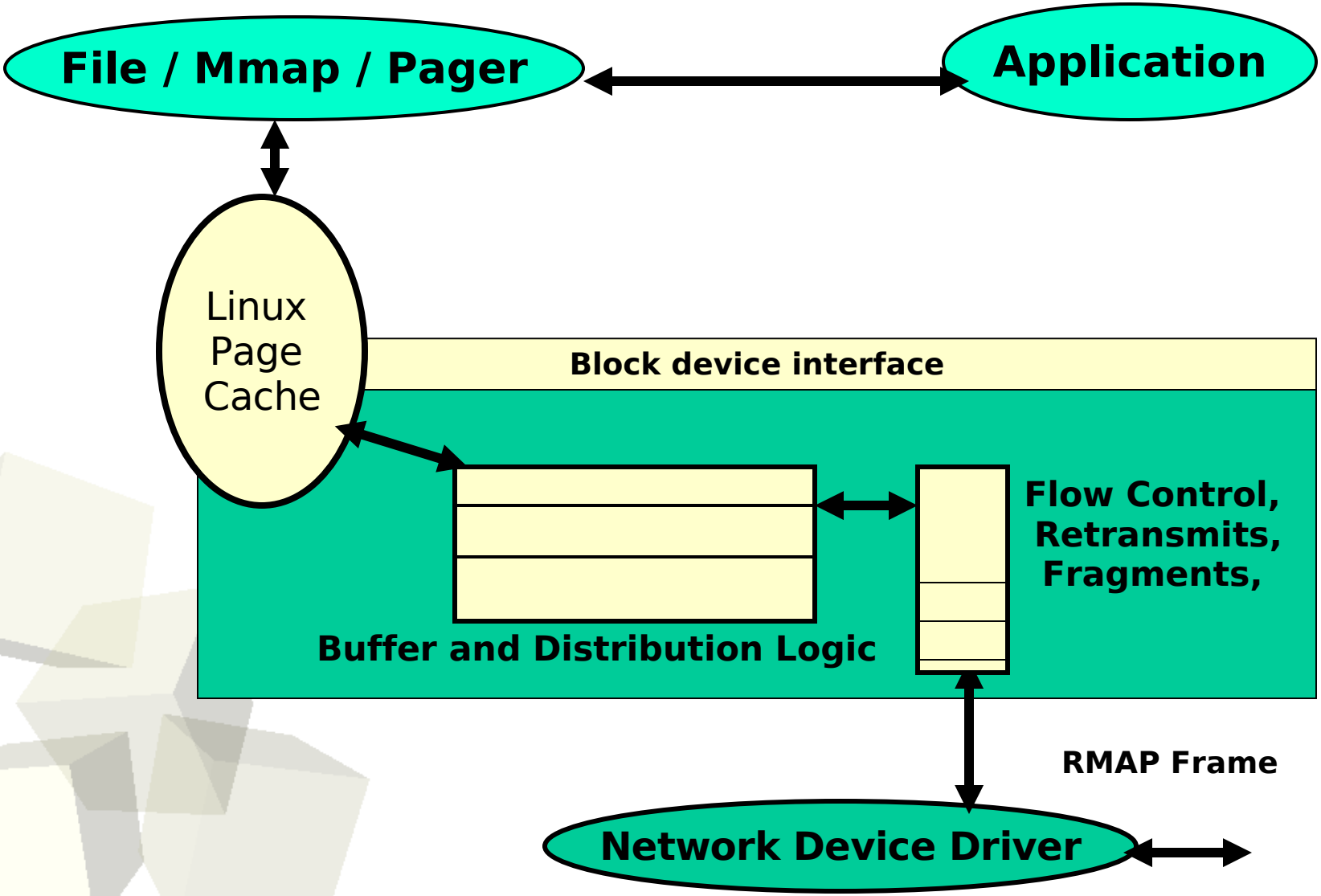
Questions?



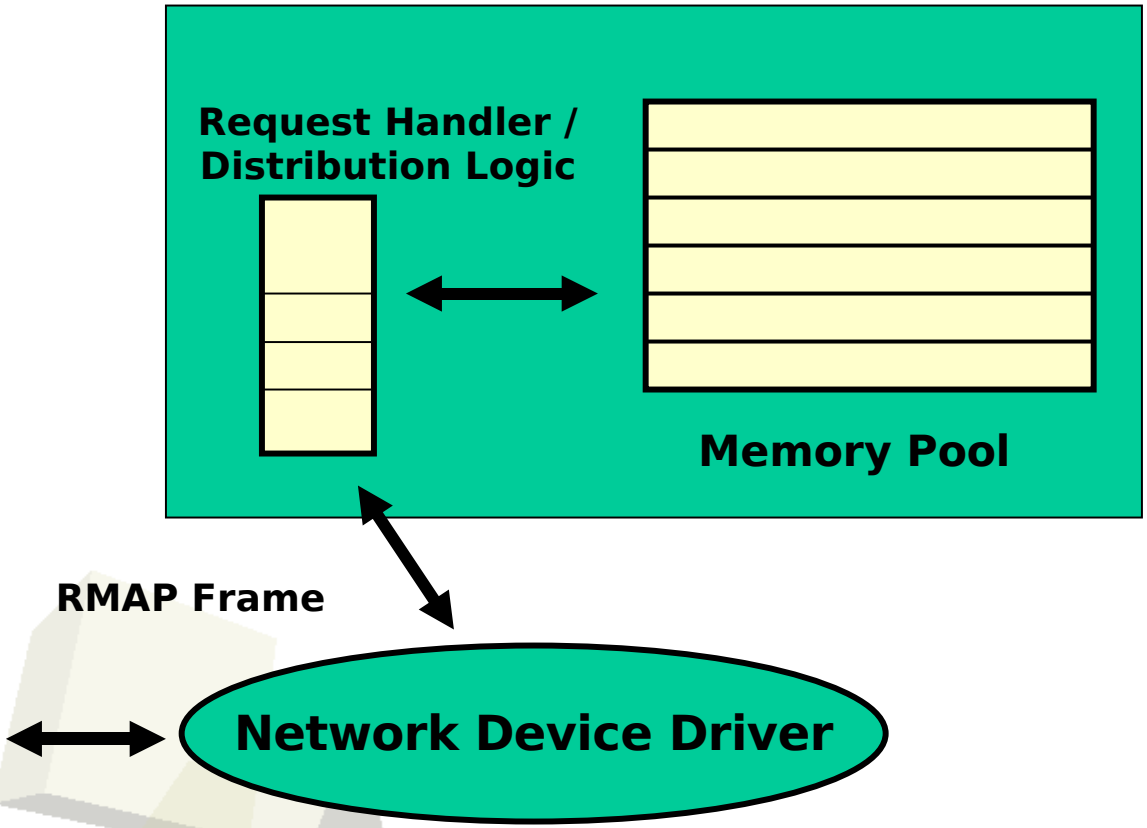
Backup Slides



Client-side Module



Server-Side Module



- Operates in interrupt context
  - For fast servicing
- Grabs memory on demand
- Passively provides/stores pages upon request