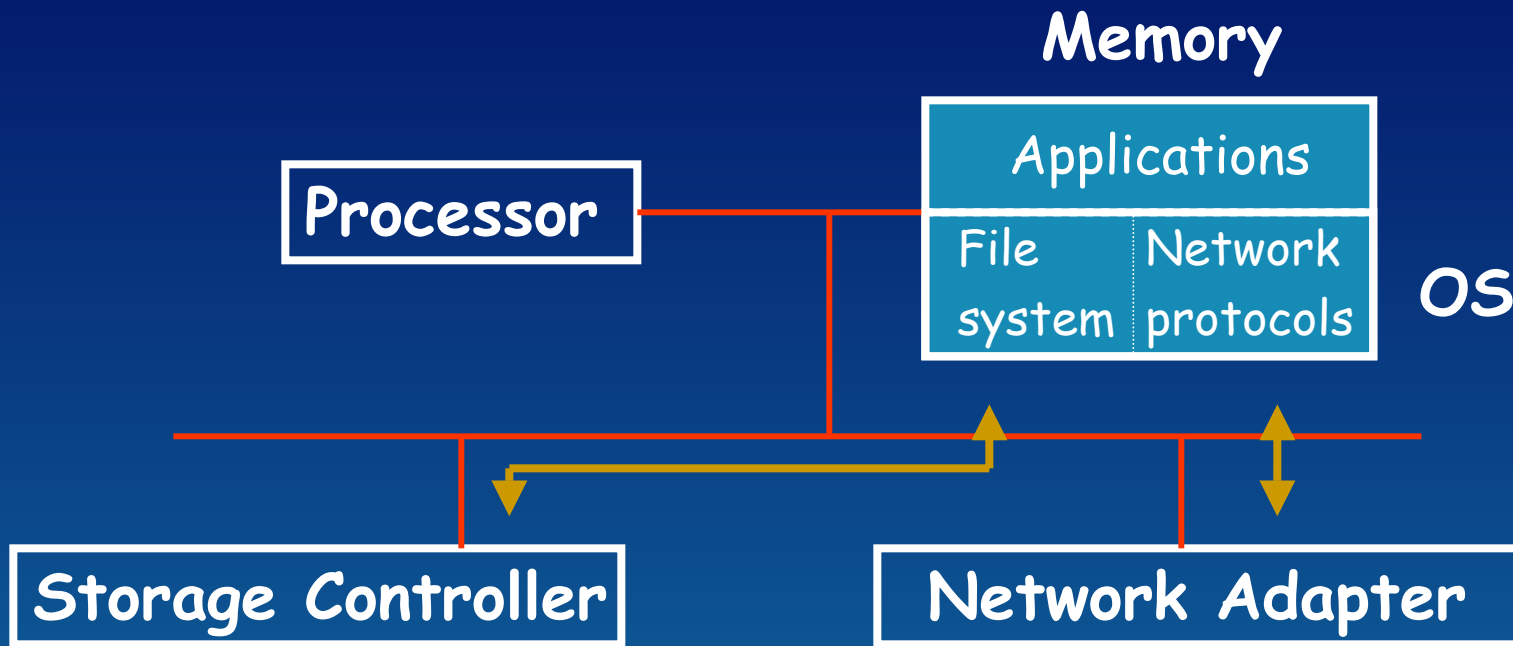


Split-OS: An Operating System for Cluster of Intelligent Devices

The Internet Impact

- ⌘ A new class of applications: network servers
 - ☒ increasing demand for scalable I/O bandwidth and high availability
 - ☒ challenge for OS and I/O architecture
- ⌘ **Networking becomes a first-class resource**
 - ☒ network protocol processing: performance bottleneck
- ⌘ Operating system research
 - ☒ system support for server applications (LRP, IO-Lite, Resource Containers, etc)
 - ☒ intelligent storage support (NASD, Active Disks, ISTORE)
 - ☒ OS-I/O device interaction constrained by the traditional I/O architecture

Traditional Computer System

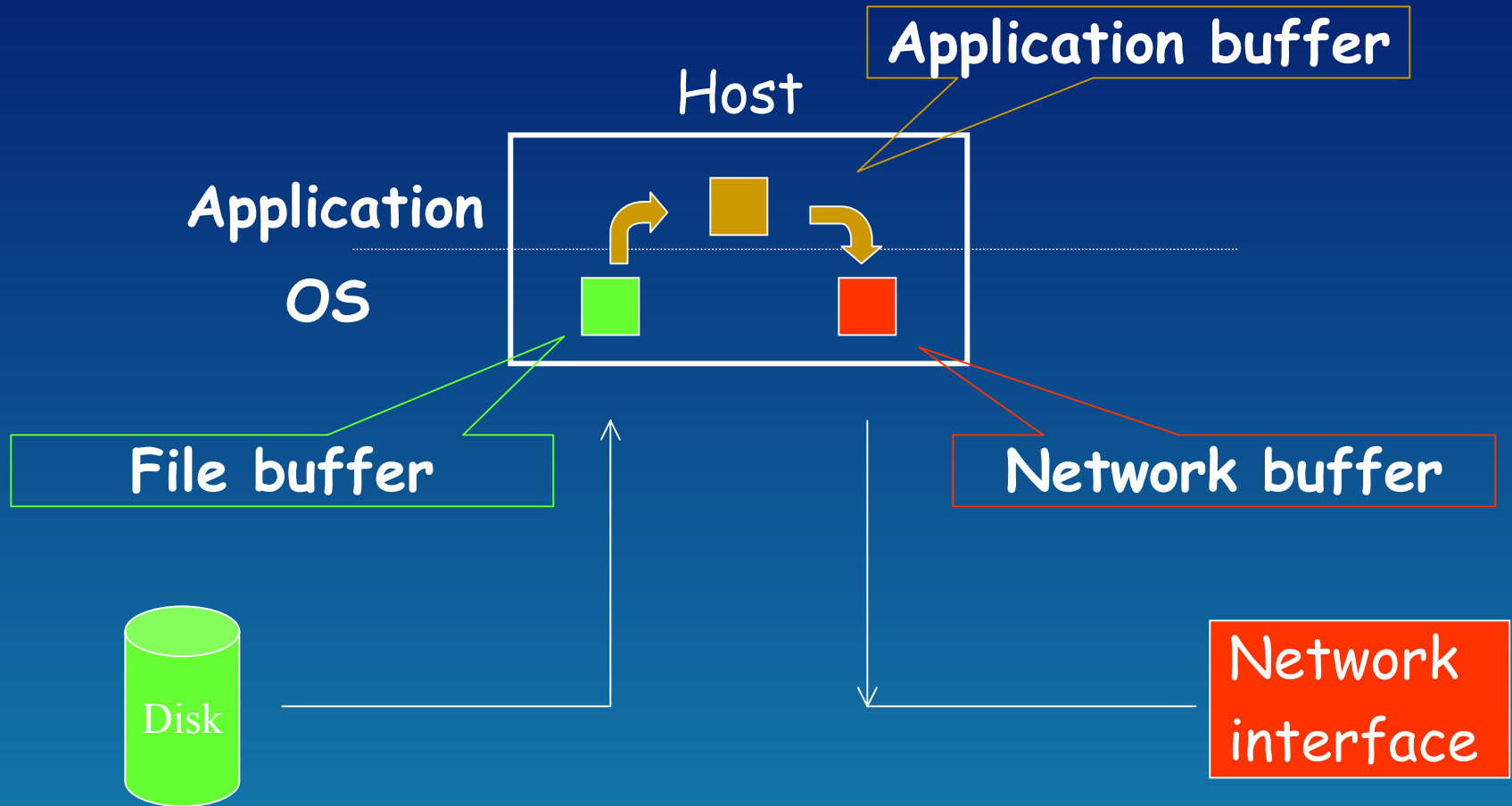


- ⌘ Intelligent host and passive I/O devices
- ⌘ OS executed exclusively on the host along with applications
- ⌘ I/O devices communicate only through the host memory (no direct device-to-device communication)

OS-Application Co-habitation

- ⌘ OS "steals" compute cycles and memory from applications
- ⌘ Two protection modes: switching overhead
- ⌘ OS executed asynchronously
 - ☒ interrupt processing overhead
 - ☒ internal synchronization on multiprocessor servers
- ⌘ Cache pollution
- ⌘ Host involved in "service-work"
 - ☒ TCP packet retransmission
 - ☒ TCP ACK processing
 - ☒ ARP request service
- ⌘ Extreme cases are even worse
 - ☒ Receive livelocks
 - ☒ Denial-of-service (DoS) attacks

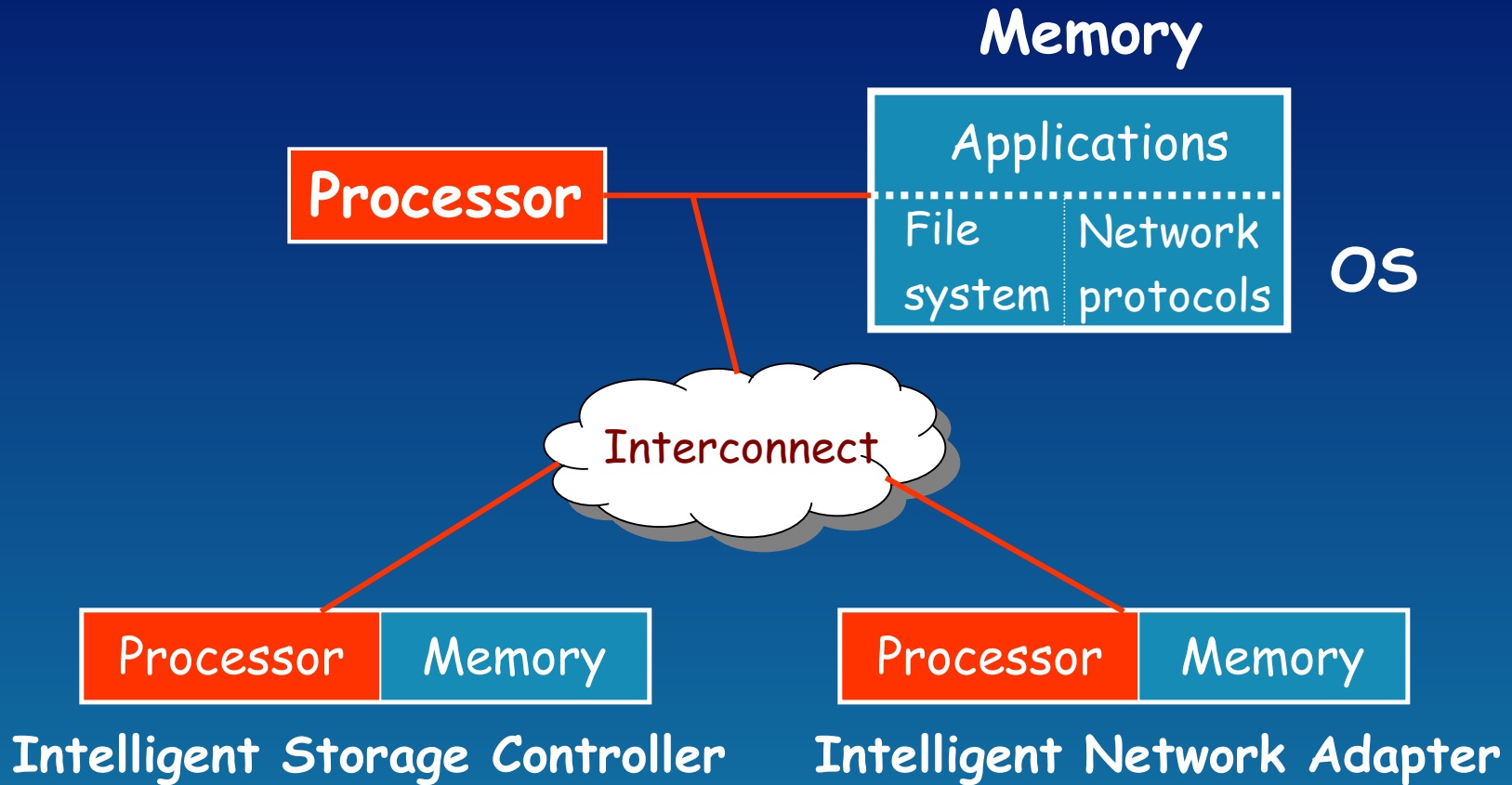
Host mediates data transfer between devices



Split-OS Idea

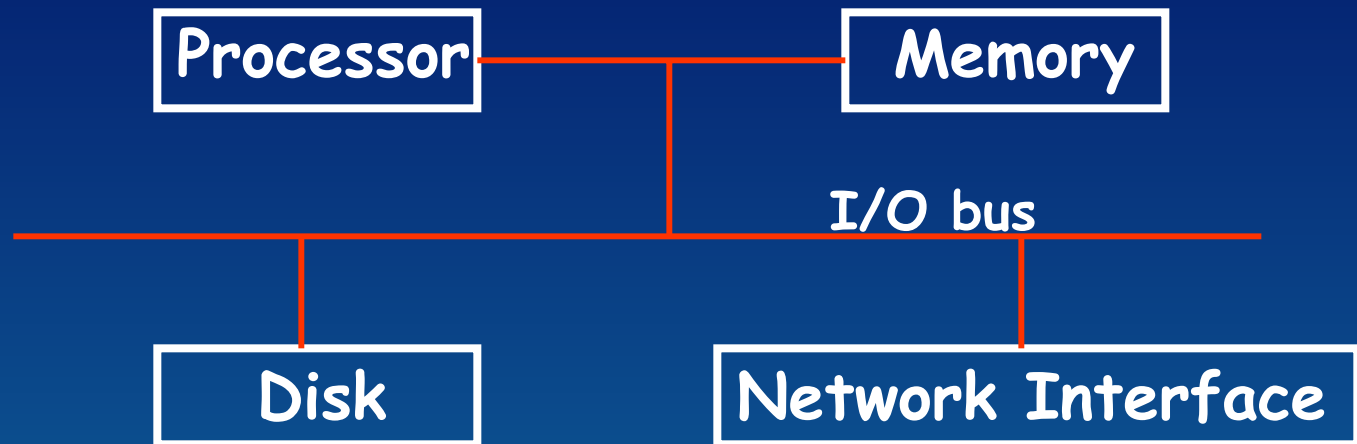
- ⌘ OS functionality split between host(s) and devices
- ⌘ New server architecture: **Cluster of Intelligent Devices**
 - ☒ "intelligent devices" (programmable, active)
 - ☒ non-intrusive communication: InfiniBand
- ⌘ Goals:
 - ☒ offload device-specific functionality from host to devices
 - ☒ eliminate host from device-to-device communication
- ⌘ Novelty
 - ☒ multiple communicating intelligent devices: intelligent storage and intelligent network adapters
 - ☒ combine intelligent devices with non-intrusive communication

Cluster of Intelligent Devices (CID)



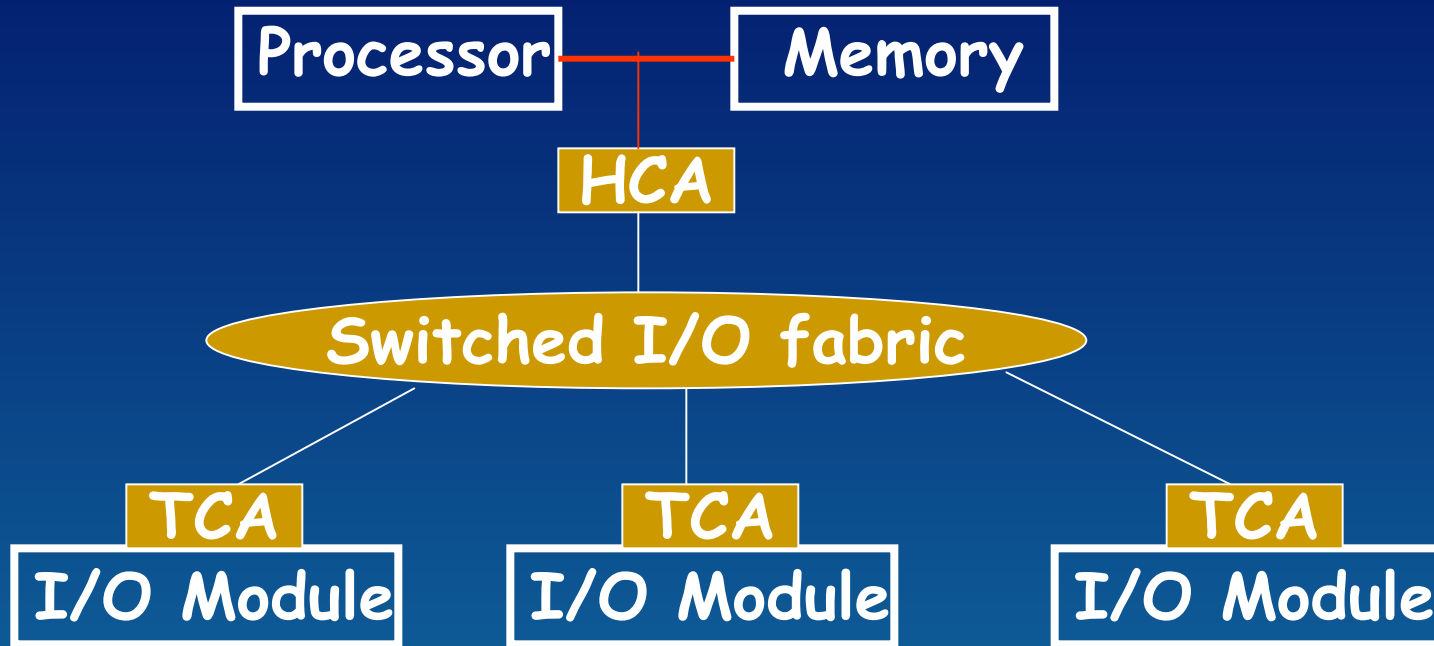
- ⌘ "Active" or "programmable" devices
- ⌘ Applications or/and OS?

Traditional Computing Architecture



- ⌘ shared bandwidth
 - ☒ limited scalability
- ⌘ data transfers between device and memory (DMA)
 - ☒ no protection to erroneous data transfers

InfiniBand I/O Architecture

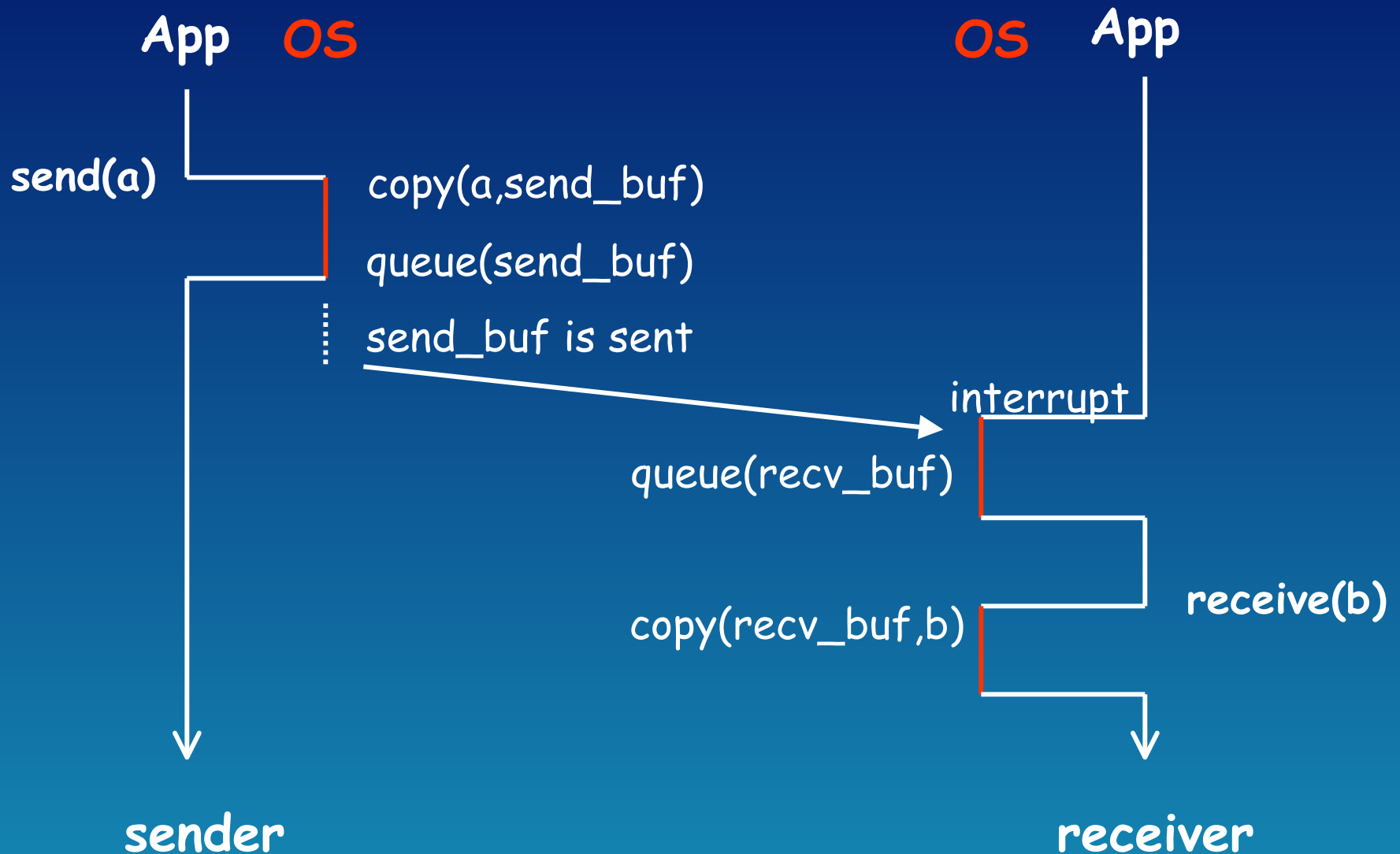


- ⌘ Point-to-point switched-based interconnect
- ⌘ Hardware protocols for message-passing between devices (implemented by **host/target channel adapters**)
 - ☑ send/receive messaging
 - ☑ **protected memory-to-memory communication**

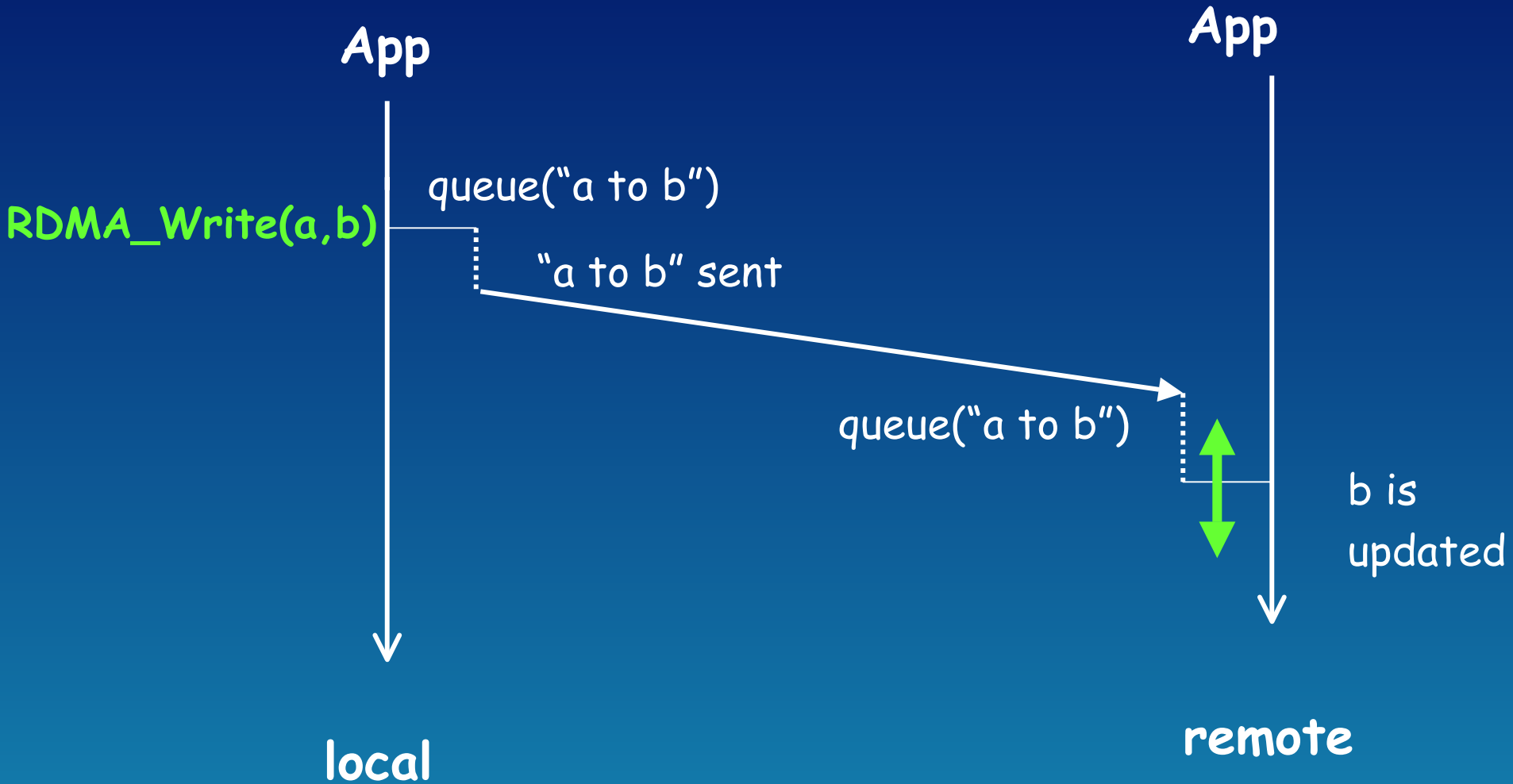
InfiniBand History

- ⌘ 1999: trade association: Compaq, Dell, HP, IBM, Intel, Microsoft and Sun
- ⌘ October 2000: specifications released
- ⌘ Heavily inspired by Virtual Interface Architecture (VIA)
 - ⌘ industry standard for cluster interconnect
 - ⌘ user-level memory-mapped communication
 - ⌘ 1997: Compaq, Intel and Microsoft
- ⌘ VIA is heavily inspired by the university research in memory-mapped communication
 - ⌘ SHRIMP (Princeton)
 - ⌘ U-Net (Cornell)

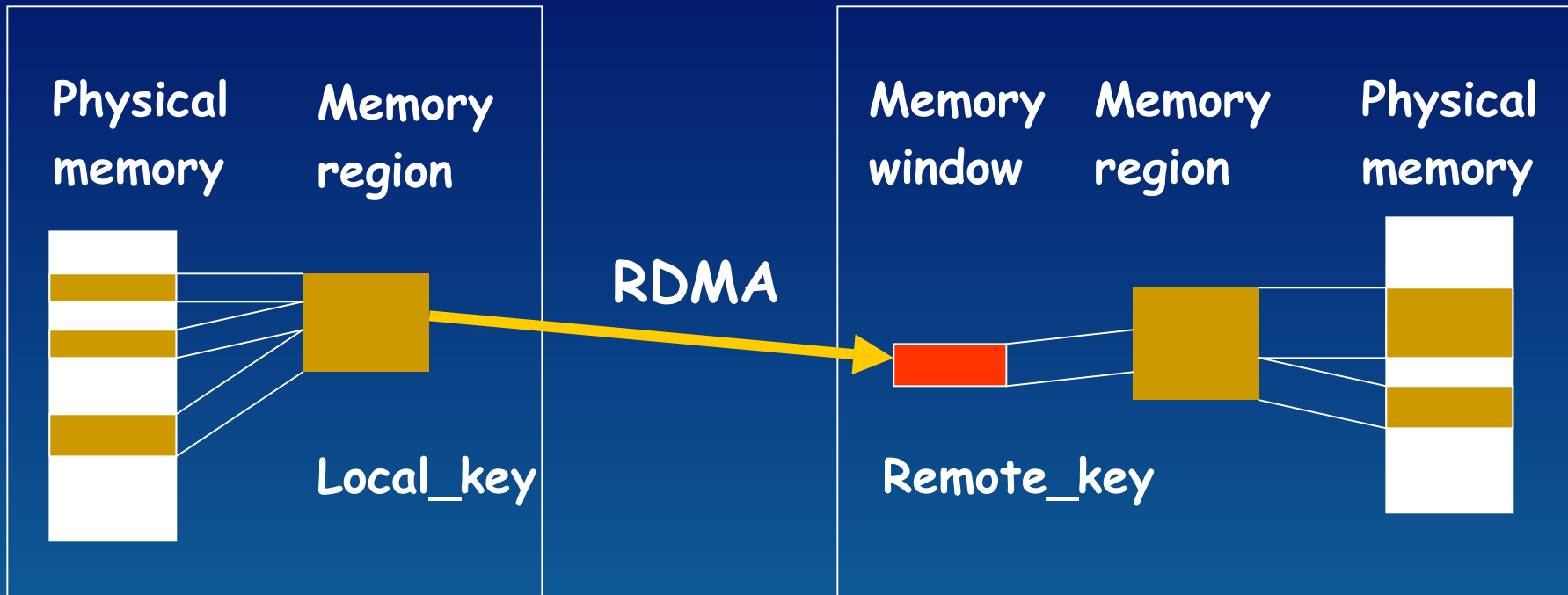
Traditional Send/Receive Communication



Non-Intrusive Communication



Non-Intrusive Communication in IB

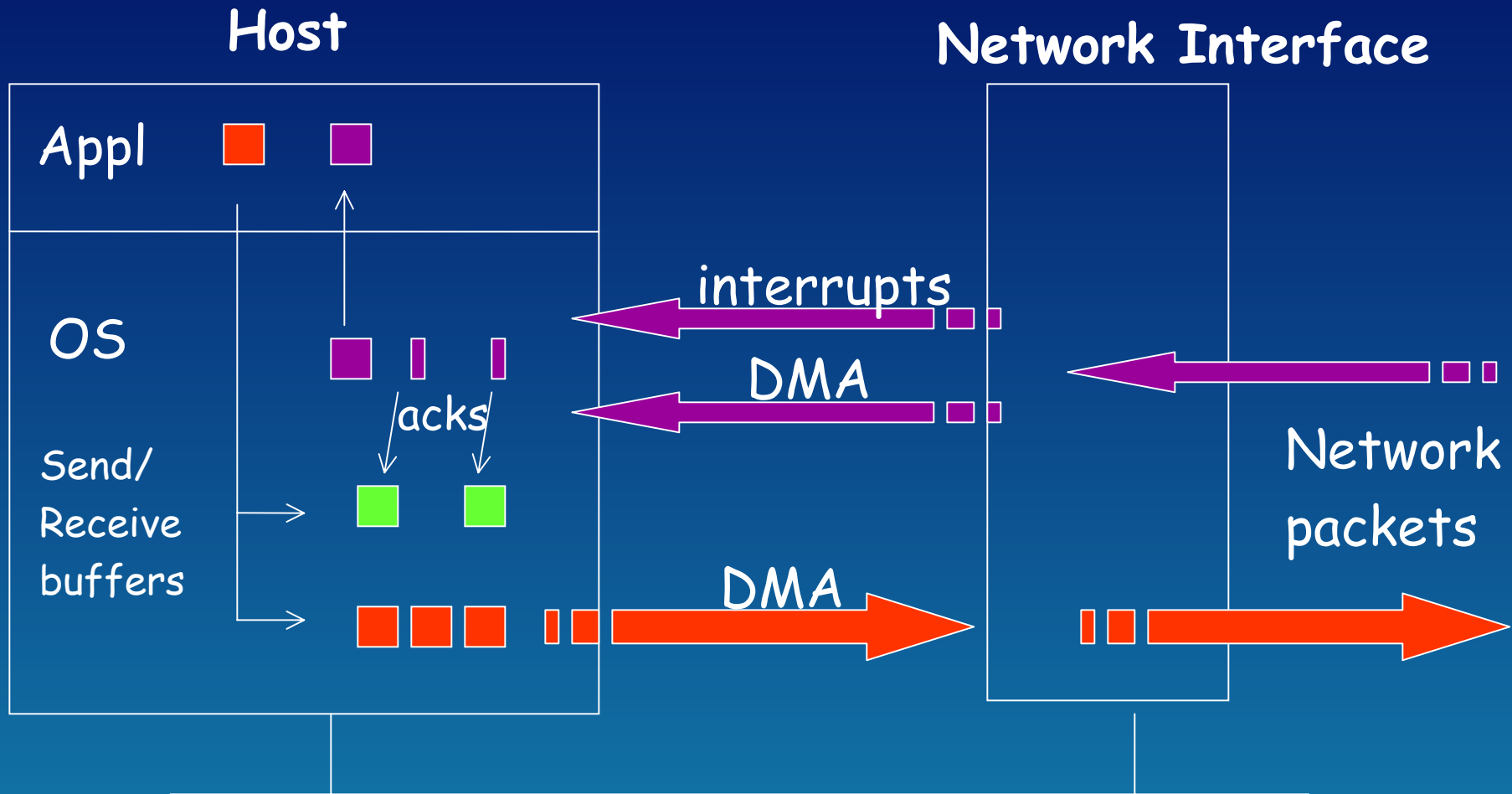


- ⌘ Memory region: virtually contiguous area of memory registered with the channel adapter (L_key)
- ⌘ Memory window: protected remote access to a specified area of the memory region (R_key)
- ⌘ **Remote DMA {L_key, R_key}**: read, write, atomic ops
- ⌘ No remote software intervention

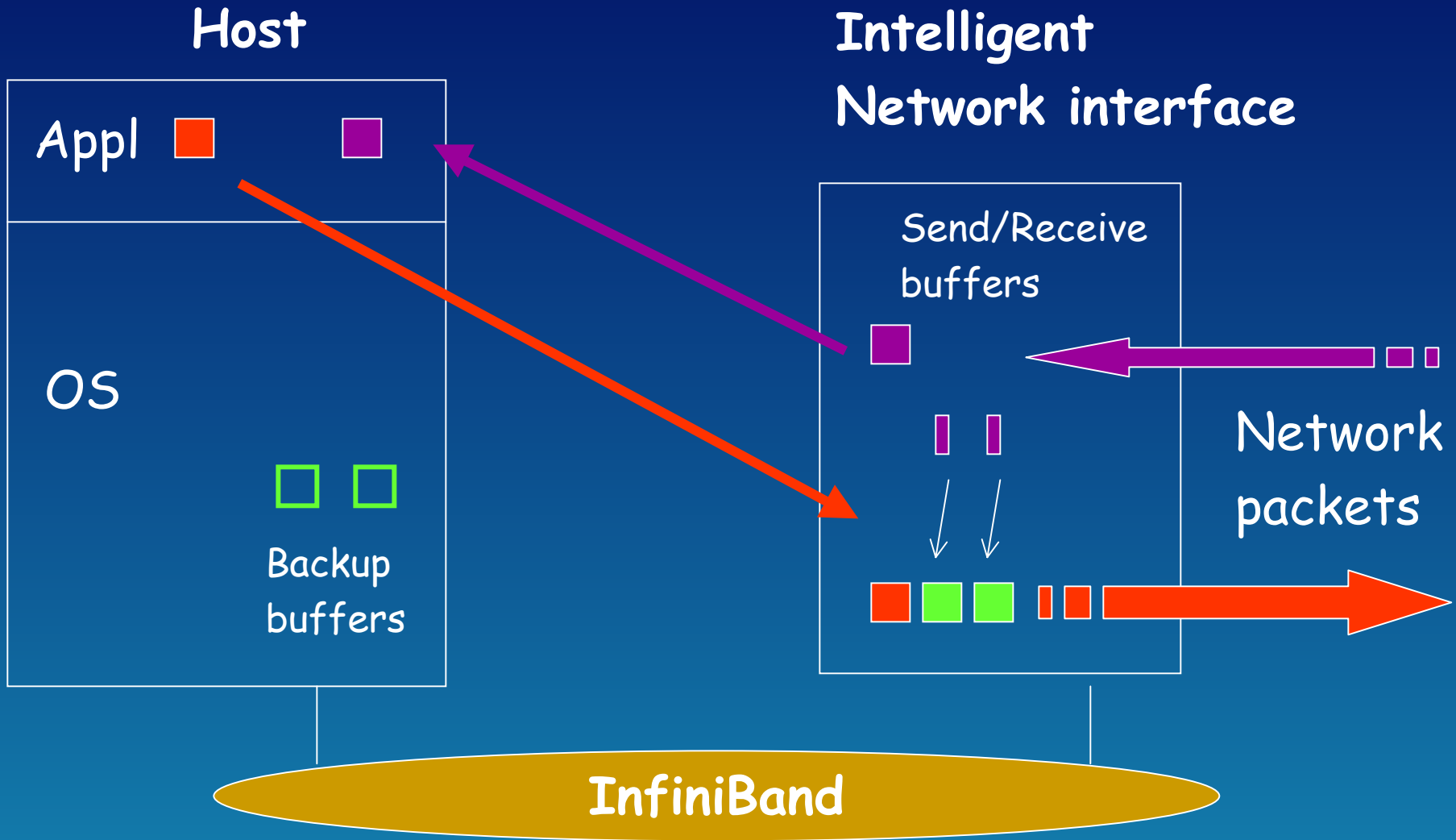
Split-OS

- ⌘ Execute network protocols in the intelligent network interface
- ⌘ Execute the file system manager in the intelligent storage
- ⌘ Design Issues
 - ☑ light-weight split vs heavy-weight split
 - ☑ direct device-to-device communication
 - ☑ reliability issues: cooperative devices
 - ☑ how to use the idle cycles of intelligent devices
- ⌘ Non-intrusive communication
 - ☑ application-to-device communication: bypass OS
 - ☑ device-to-device: bypass host

Networking in Conventional OS



Networking in Split-OS with InfiniBand

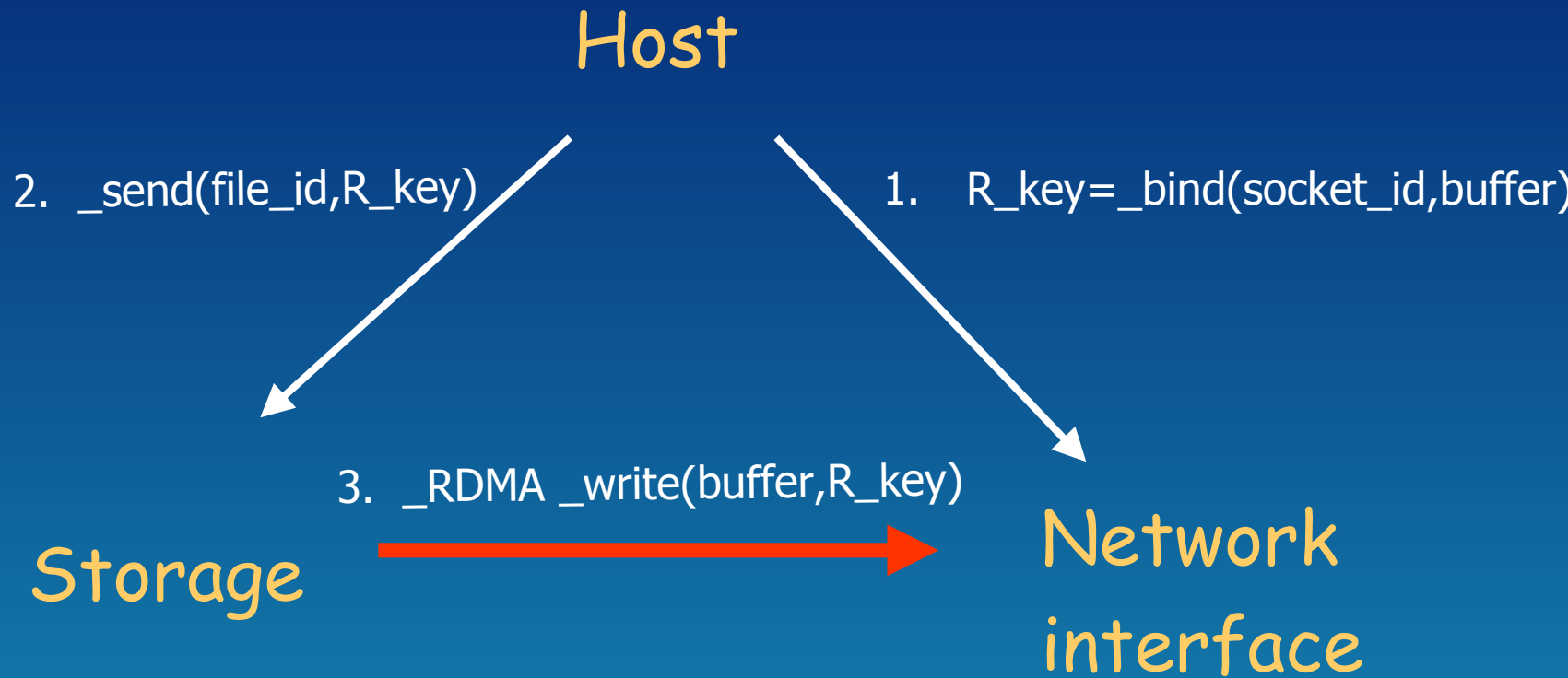


Split-Networking

- ⌘ Minimum overhead on the host: communication between application and network interface
- ⌘ Retransmission and ack processing handled in the intelligent network interface
- ⌘ Interrupts are eliminated
 - ☒ Receive livelock is avoided (no interrupts)
- ⌘ DoS attacks can be absorbed in the network interface
- ⌘ **Send and receive buffers kept in the network interface as long as possible**
 - ☒ retransmission buffers may be evicted and written back to the host (non-intrusively using RDMA)
 - ☒ receive buffers can be eagerly transferred to the host or discarded if overflow

Direct Device-to-Device Communication

- ⌘ Conflict with caching in the host memory
- ⌘ Extended API: `transfer(file_id, socket_id, size)`



Receive

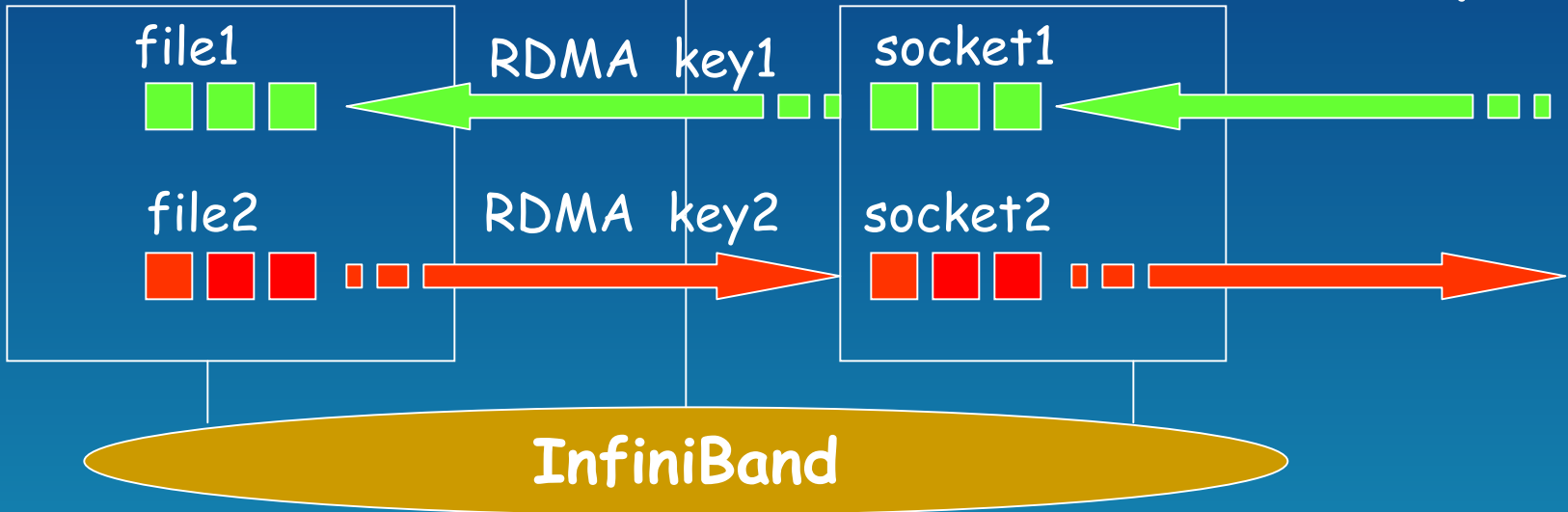
Send

Host

| | | |
|--|--|------|
| <code>f1=creat(file1)</code> <code>transfer(socket1,f1)</code> | <code>f2=open(file2)</code> <code>transfer(f2,socket2)</code> | Appl |
| <code>key1=_bind(file1)</code> <code>_RDMA (socket1, key1)</code> | <code>key2=_bind(socket2)</code> <code>_RDMA (file2, key2)</code> | OS |

Intelligent Storage

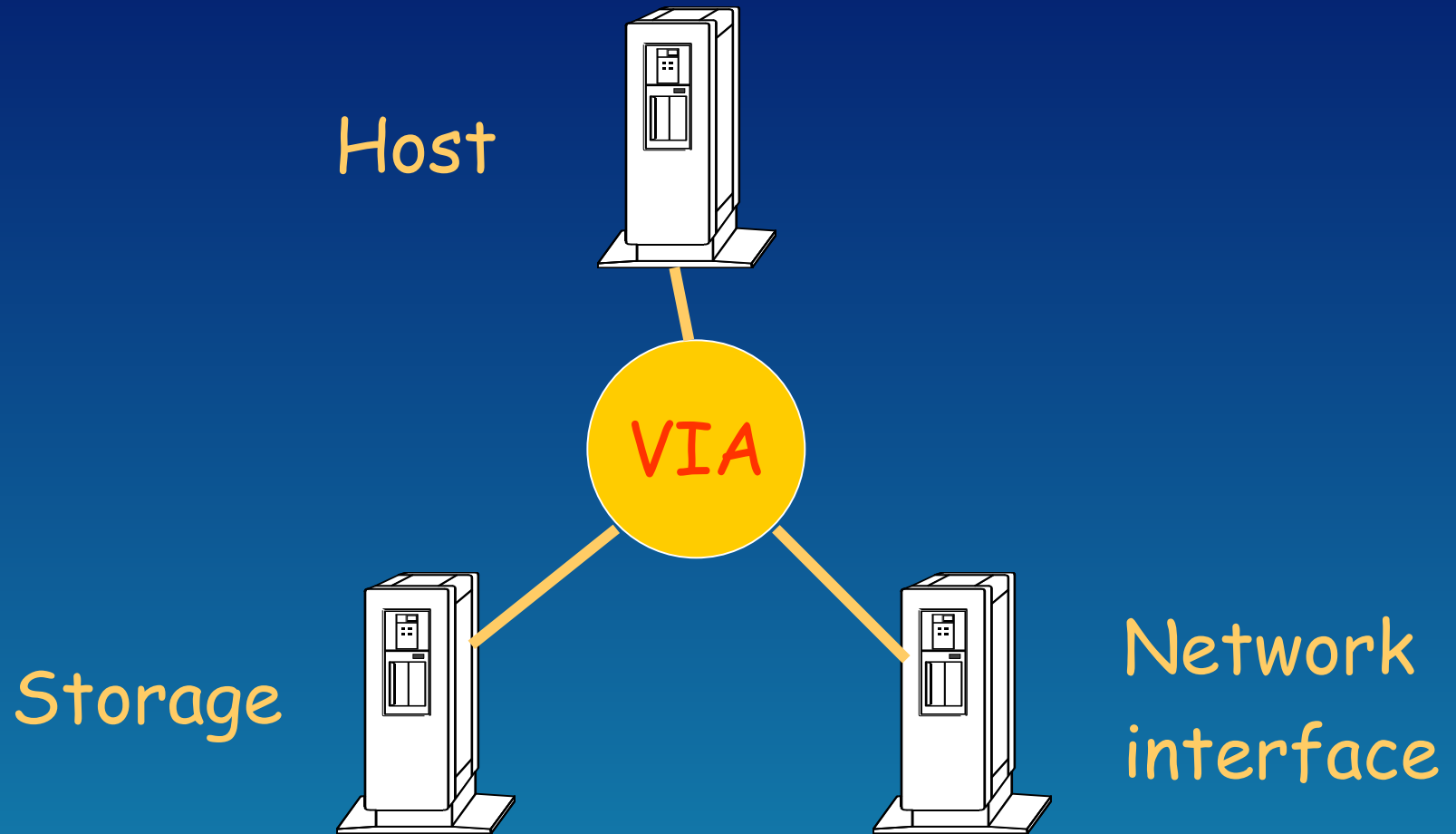
Intelligent Network interface



Cooperative Devices

- ⌘ Logical virtual devices on top of several physical devices
- ⌘ Coherent global state using non-intrusive communication
- ⌘ Cooperative storage
 - ☑ location-independent file naming
 - ☑ cooperative caching
- ⌘ Cooperative networking
 - ☑ shared TCP state: virtual network interface
 - ☑ single/multiple IP addresses: Stream Control Transmission Protocol
 - ☑ network load balancing and failover

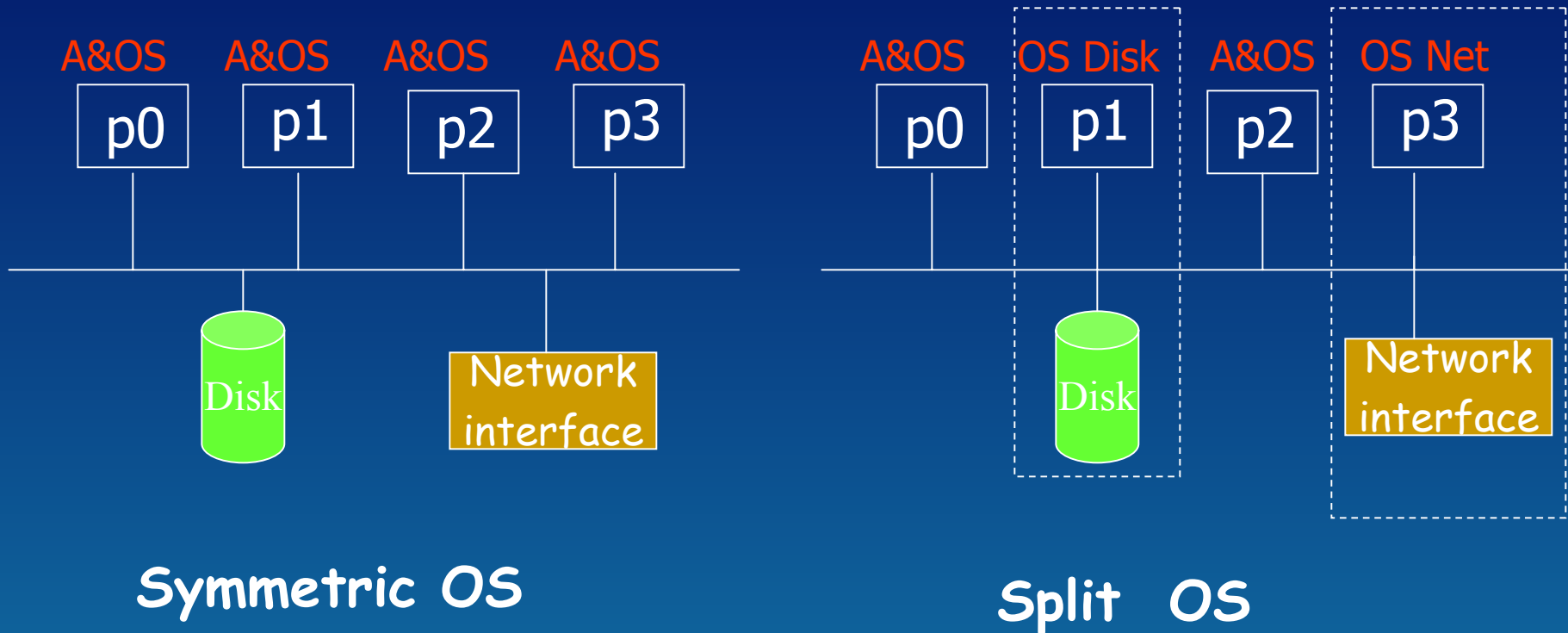
CID Testbed: Cluster of PCs over VIA



Prototype

- ⌘ Cluster of PCs
- ⌘ VIA interconnect
- ⌘ Modified Linux kernel
- ⌘ Challenges
 - ☒ translate IB communication (host-to-device and device-to-device) into VIA user-level communication
 - ☒ emulate the IB remote operation using remote-write exclusively (the only non-intrusive operation most VIA adapters provide)
- ⌘ Approach:
 - ☒ dual processor PC as intelligent devices
 - ☒ use one processor to emulate the IB adapter
- ⌘ Plan to use IB switches (and devices) when they become available

Split-OS for multiprocessor servers

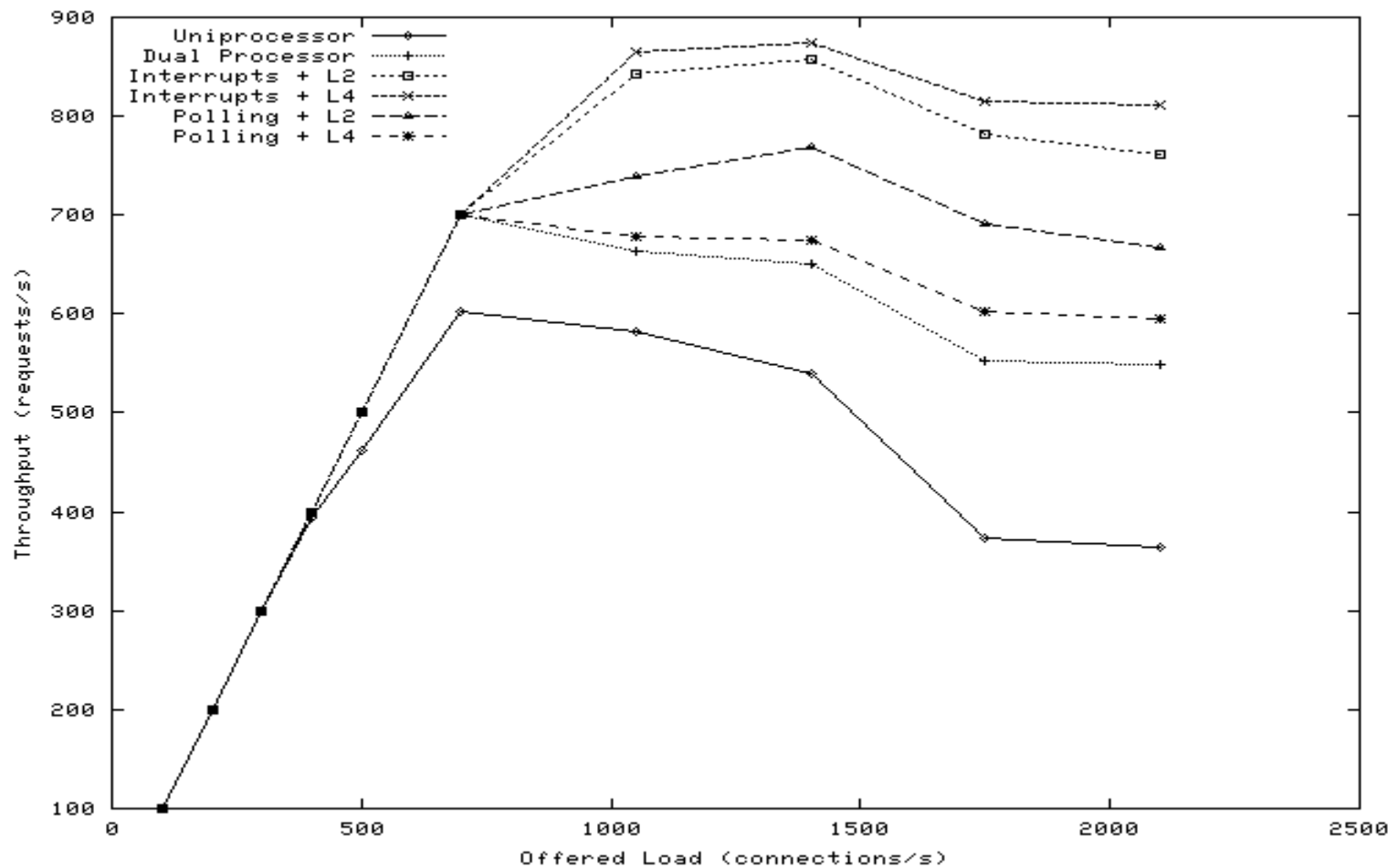


- ⌘ Intelligent device = device + dedicated processor
- ⌘ I/o dedicated processor(s)
 - ☑ networking protocols: Ethernet driver (L2) or TCP/IP (L4)
 - ☑ kernel mode
- ⌘ Interrupts or polling

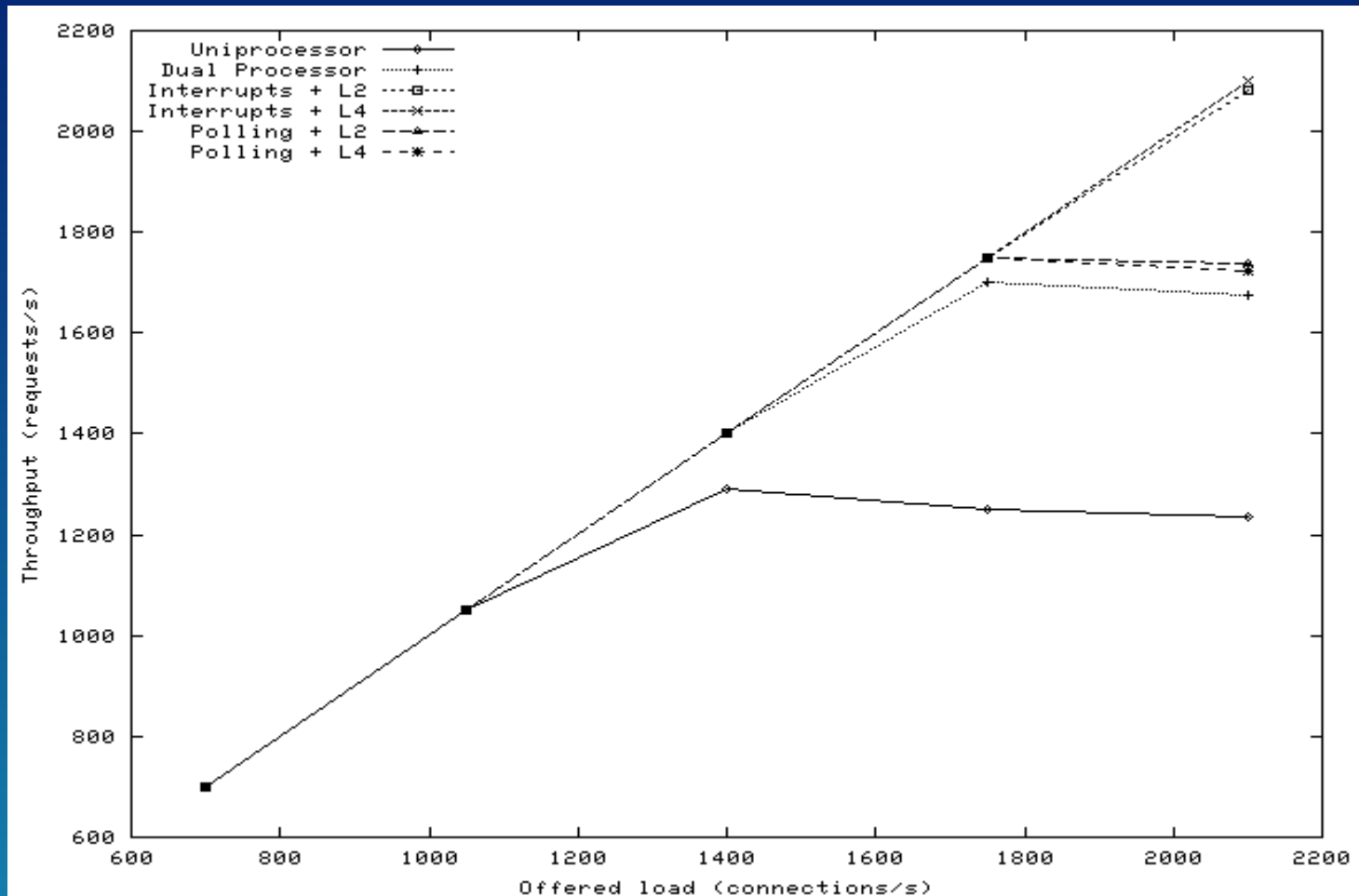
Experiments

- ⌘ One processor dedicated to networking processing
- ⌘ Four modified Linux-2.2.14 kernels
 - ☒ Dedicated processor for L2 with interrupts
 - ☒ Dedicated processor for L4 with interrupts
 - ☒ Dedicated processor for L2 with polling
 - ☒ Dedicated processor for L4 with polling
- ⌘ Conventional uni and dual-processor OS for comparison
- ⌘ A 300 MHz dual-processor server
- ⌘ Apache and Flash web servers
- ⌘ Synthetic workload: 4KB document requests

Apache Throughput



Flash Throughput



Summary

- ⌘ Operating system project
 - ☑ addresses a real problem: network server performance
 - ☑ benefit from new hardware technologies
- ⌘ InfiniBand technology assimilation plan
 - ☑ nothing changes except the drivers
 - ☑ make server applications IB-aware
 - ☑ **modify the OS**
- ⌘ CID is our invention: intelligent devices + IB (non-intrusive communication)
 - ☑ hope to convince the industry