software defining system devices with the BANANA double split driver model

Dan WILLIAMS, Hani JAMJOOM
IBM Watson Research Center

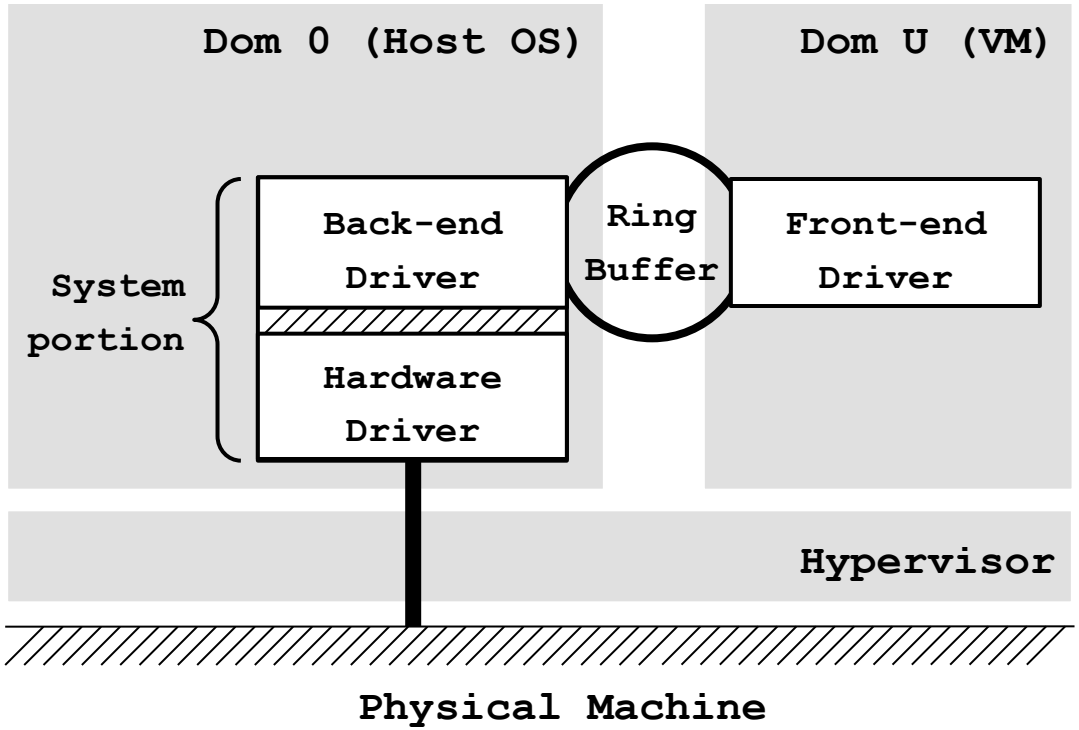
Hakim WEATHERSPOON
Cornell University

Decoupling gives Flexibility

- Cloud's flexibility comes from decoupling device functionality from physical devices
  - aka **"virtualization"**
- Can place VM anywhere
  - Consolidation
  - Instantiation
  - Migration
  - Placement optimizations

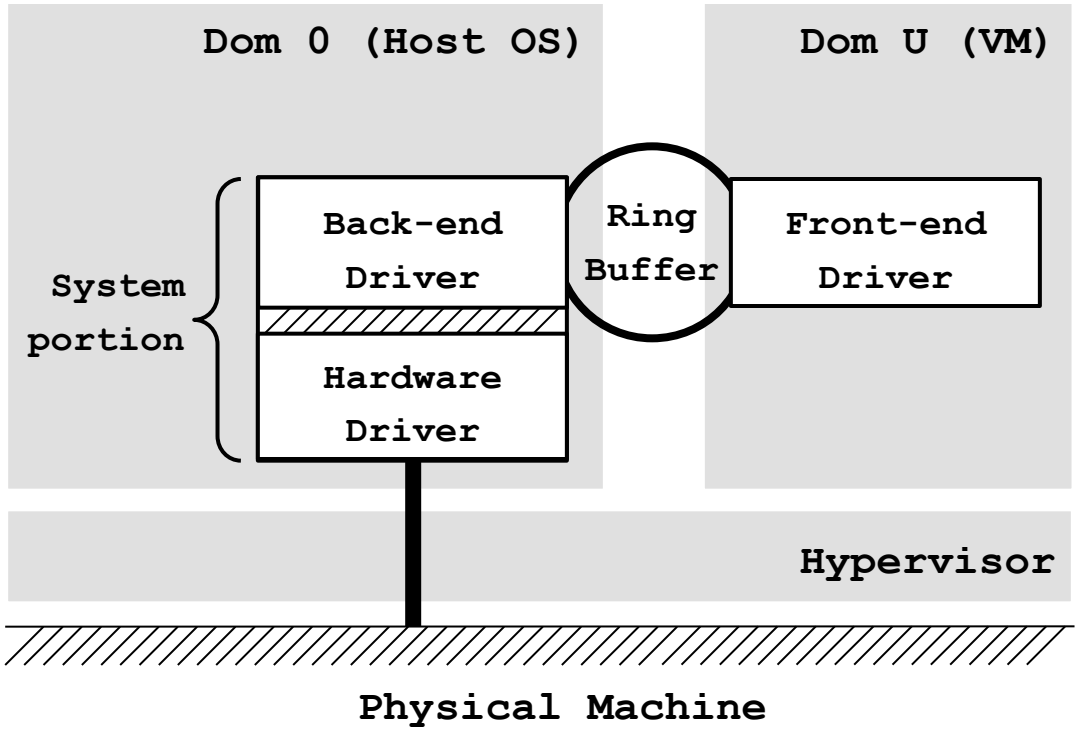
Are all Devices Decoupled?

- Today: split driver model
  - Guests don't need device-specific driver
  - System portion interfaces with physical device



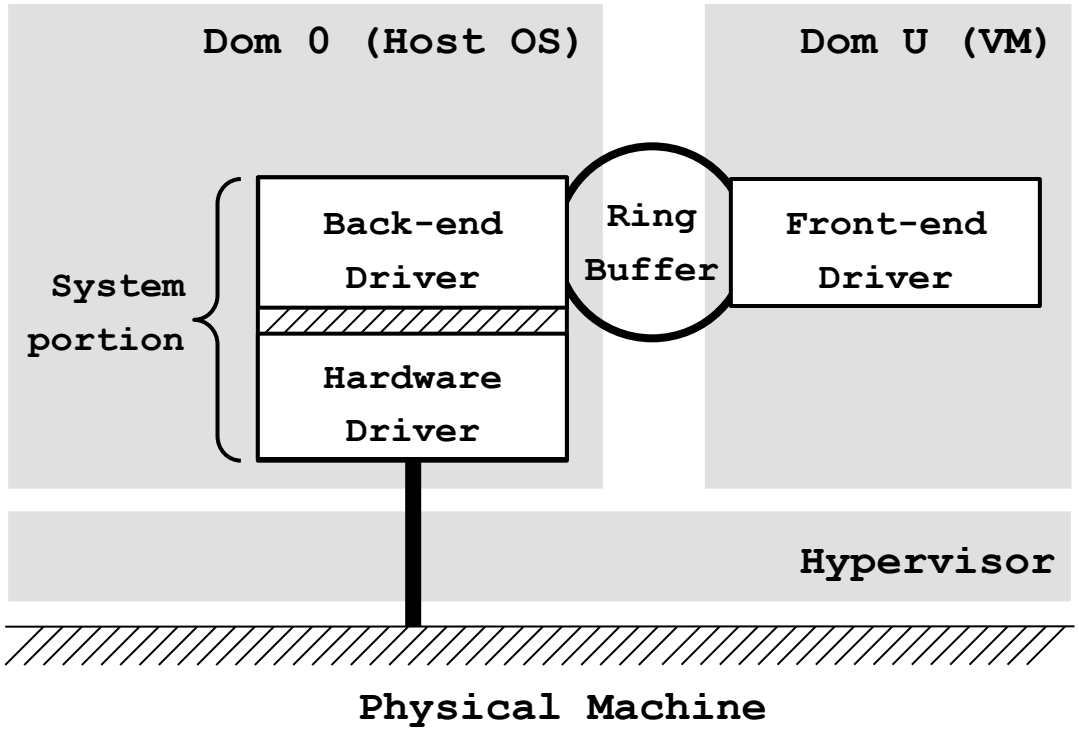
Are all Devices Decoupled?

- Today: split driver model
  - Guests don't need device-specific driver
  - System portion interfaces with physical device
- Dependencies on hardware
  - Presence of device (e.g., GPU, FPGA)
  - Device-related configuration (e.g., VLAN)



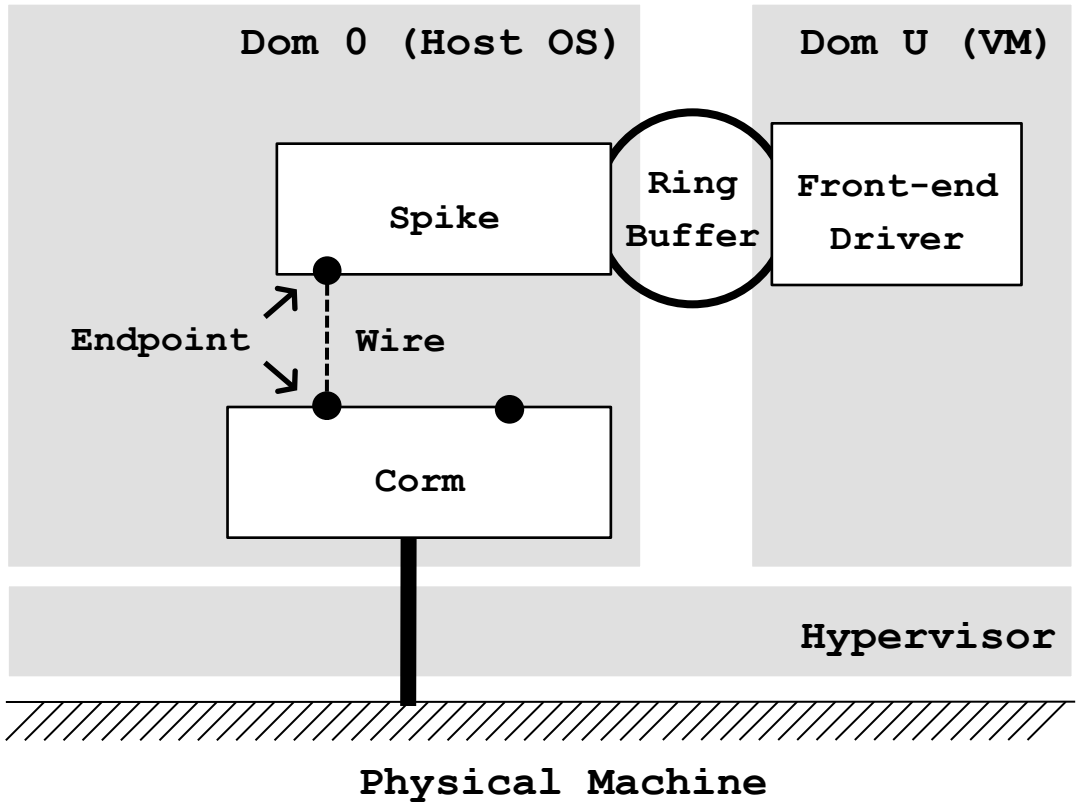
Devices Limit Flexibility

- Split driver model: dependencies break if VM moves
- No easy place to plug into hardware driver
  - System portion connected in ad-hoc way



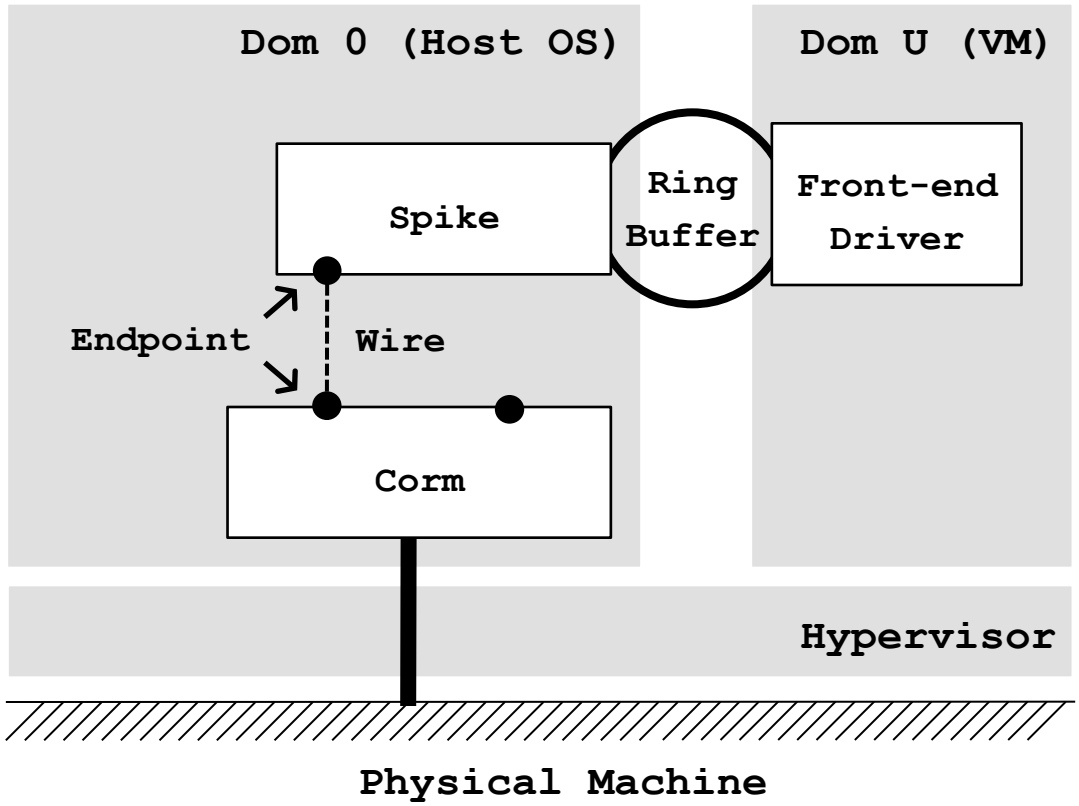
Banana Double-Split

- Clean separation between hardware driver (**Corm**) and backend driver (**Spike**)
- Standard interface between Corm and Spike (**endpoint**)
- Connected with **wires**

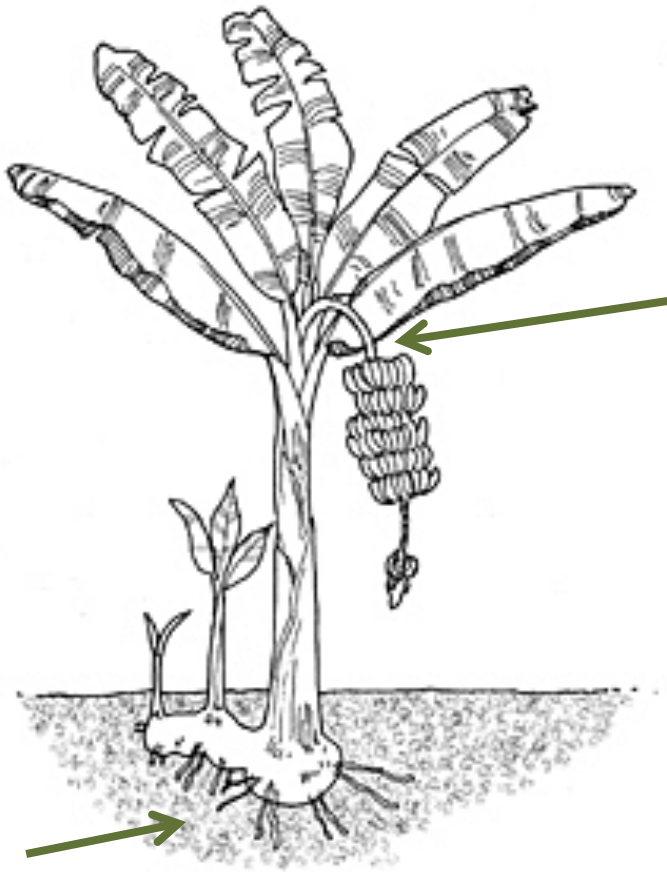


Roadmap

- Banana Double-Split Driver Model
- Implementation
- Experiments
- Related Work
- Summary



(Anatomy of a Banana Plant)

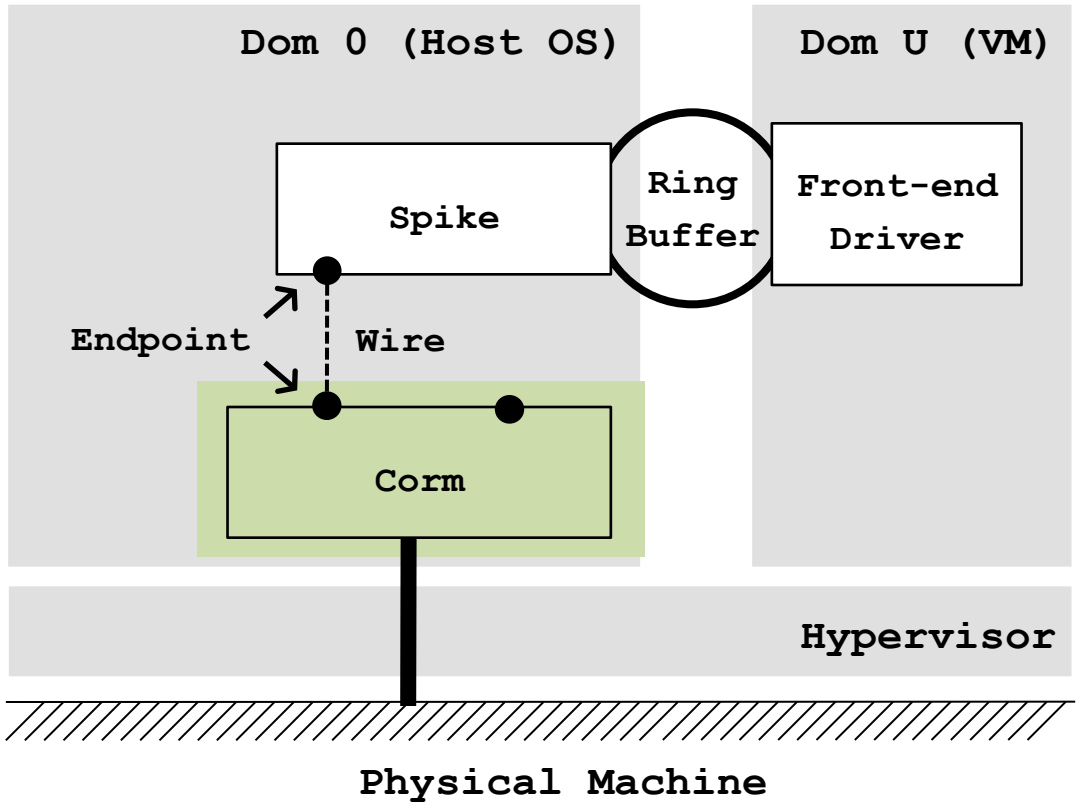


Corms

- One Corm per physical device

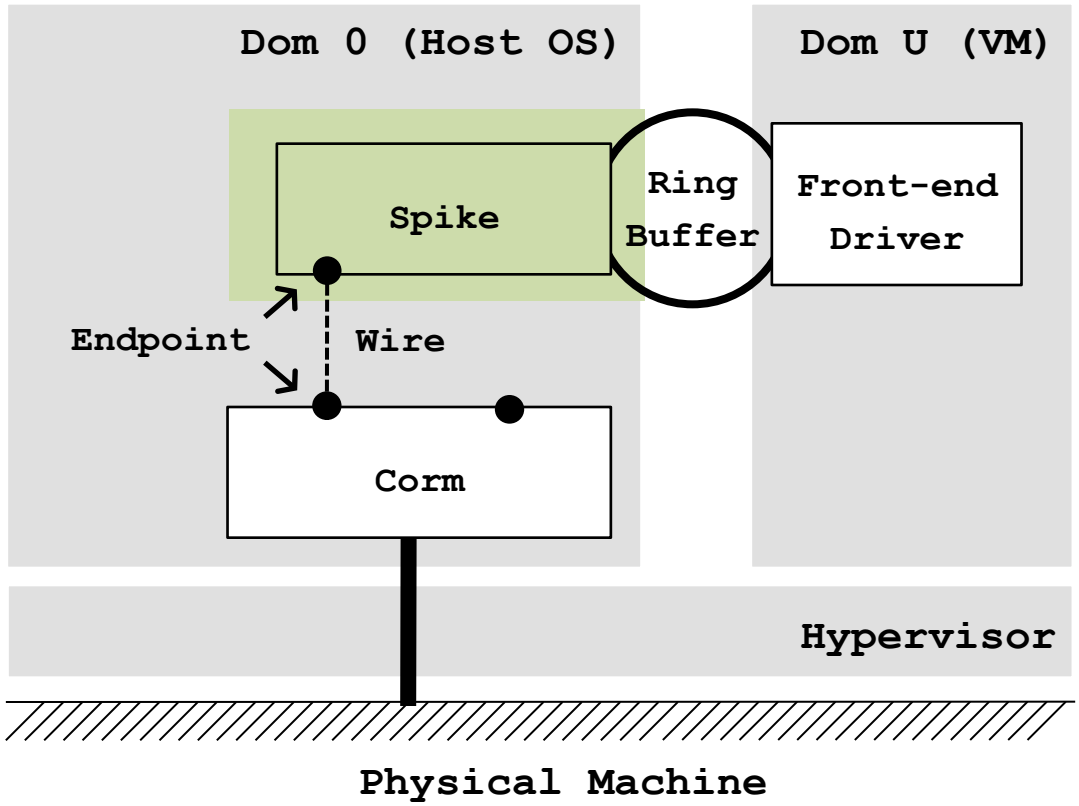
- Expose one or more **endpoints**
  - Virtual NIC-like interfaces

- Connect to one or more **Spikes**



Spikes

- One Spike per front-end driver
- Expose an **endpoint**
- Attach to a **Corm**
- **No need to share machine!**

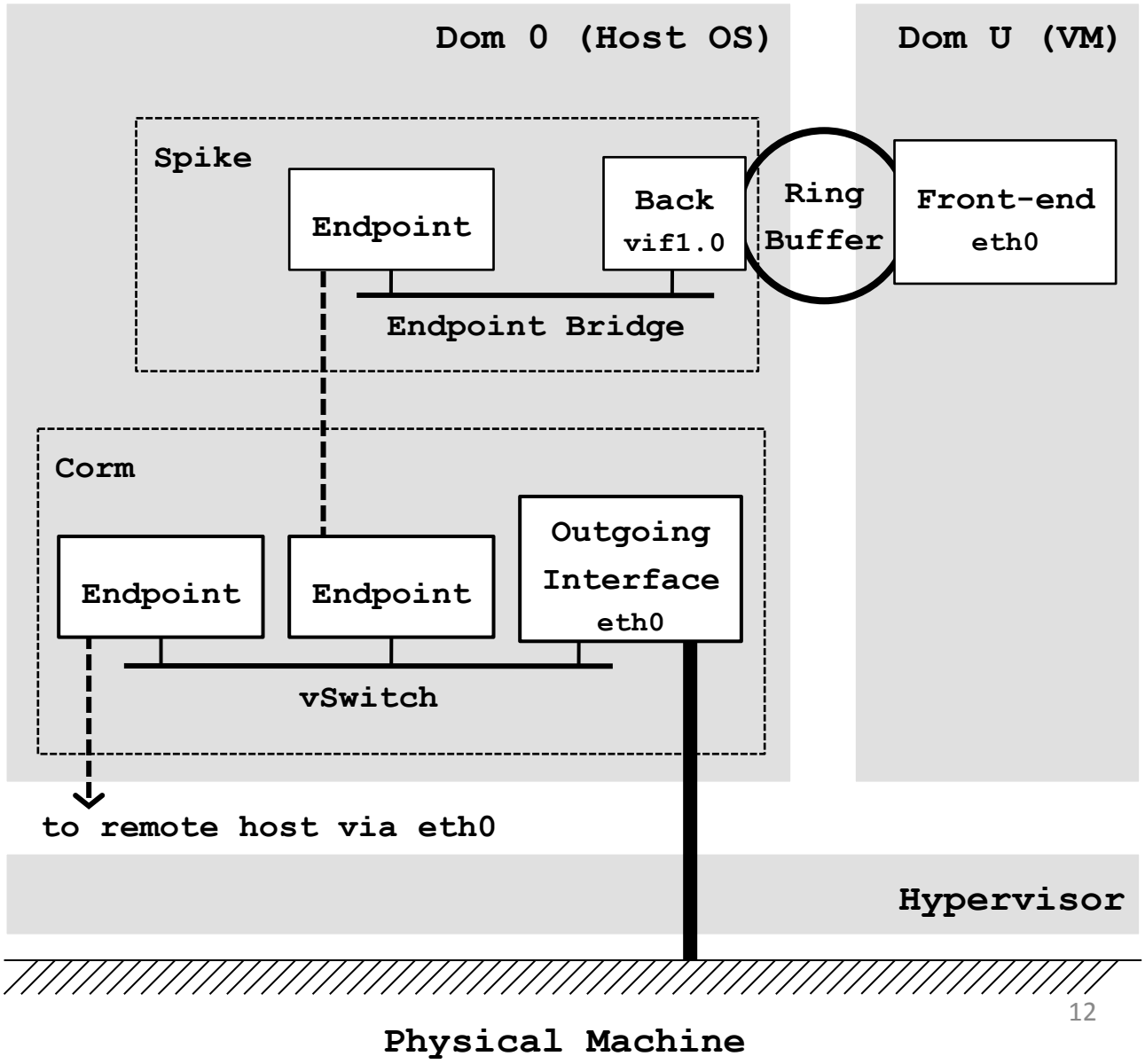


Wires

- Connections between endpoints
  - E.g., tunnel, VPN, local bridge
- Each hypervisor contains endpoint controller
  - Advertises endpoints (e.g., for Corms)
  - Looks up endpoints (e.g., for Spikes)
  - Sets wire type
  - Integrates with VM migration
- Simple interface
  - **`connect/disconnect`**

Implementation

- Xen network devices
- Endpoint exposes **`netdev`**
- Grafted to existing devices via **endpoint bridge**



Implementation

- Types of wires
  - Native (bridge)
  - Encapsulating (in kernel module)
  - Tunneling (Open-VPN based)
- `/proc` interface for configuring wires
- Integrated with live migration

Initial Experimentation

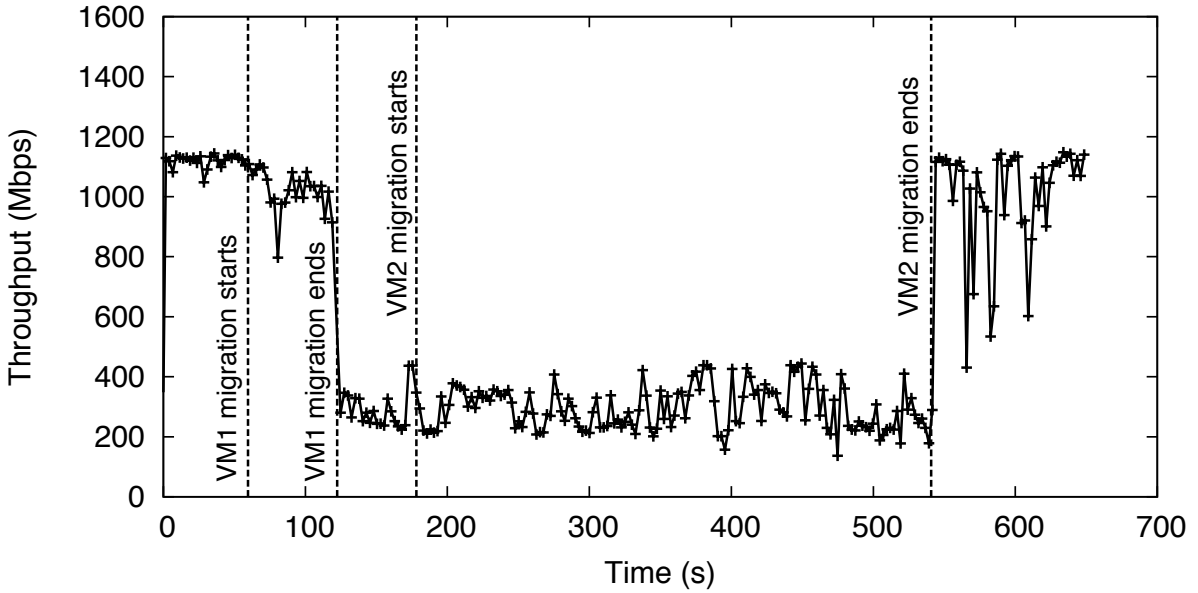
- Ultimate test of location independence
  - Cross-cloud live migration
  - Keep using the same Corm
- Banana Wire Performance

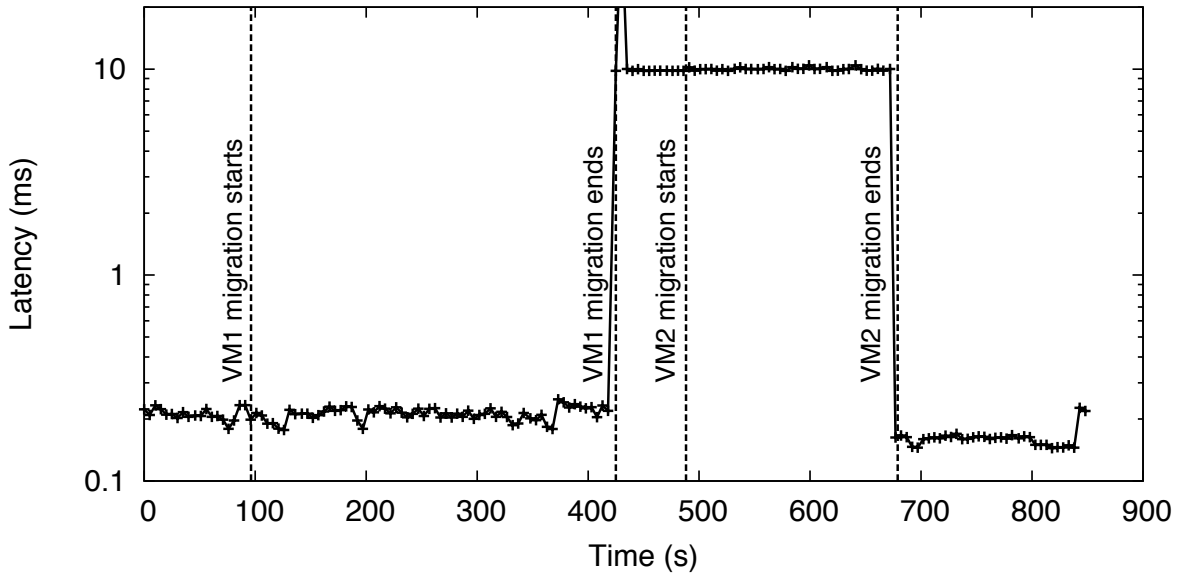
Setup

- Amazon EC2 and local resources
  - EC2 (4XL): 33 ECUs, 23 GB memory, 10 Gbps Ethernet
  - Local: 12 cores @ 2.93 GHz, 24 GB memory, 1Gbps Ethernet
- **Xen-blanket** for nested virtualization
  - Dom 0: 8 vCPUs, 4 GB memory
  - PV guests: 4 vCPUs, 8 GB memory
- Local NFS server for VM disk images
- **`netperf`** to measure throughput and latency
  - 1400 byte packets

Cross-cloud Live Migration

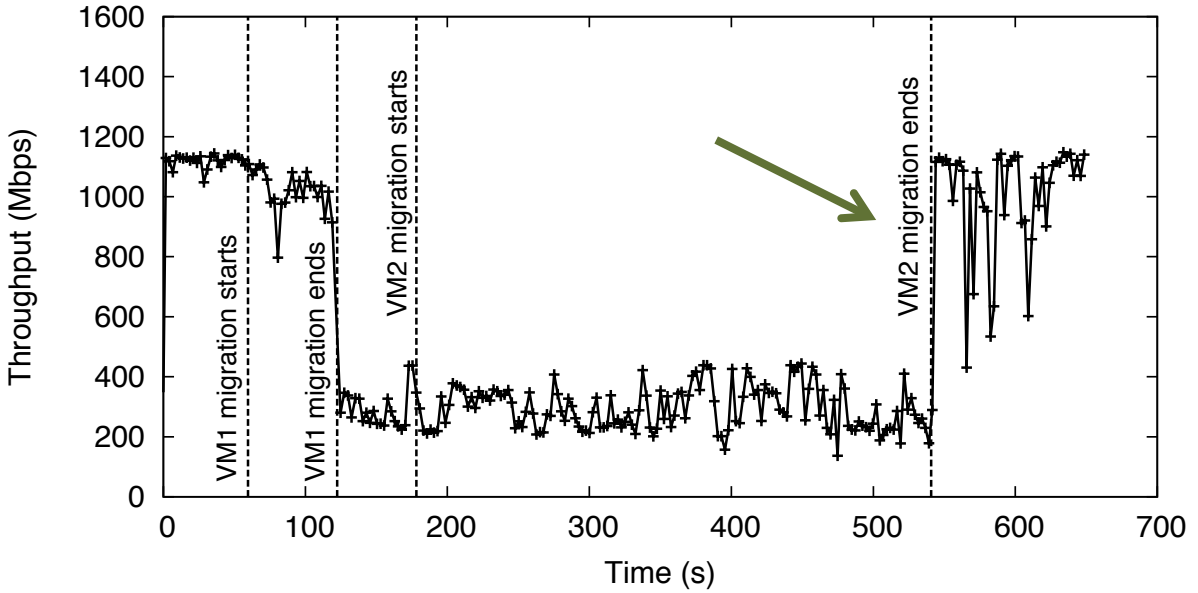
- Continuous **`netperf`** traffic between VMs
- **It works!**

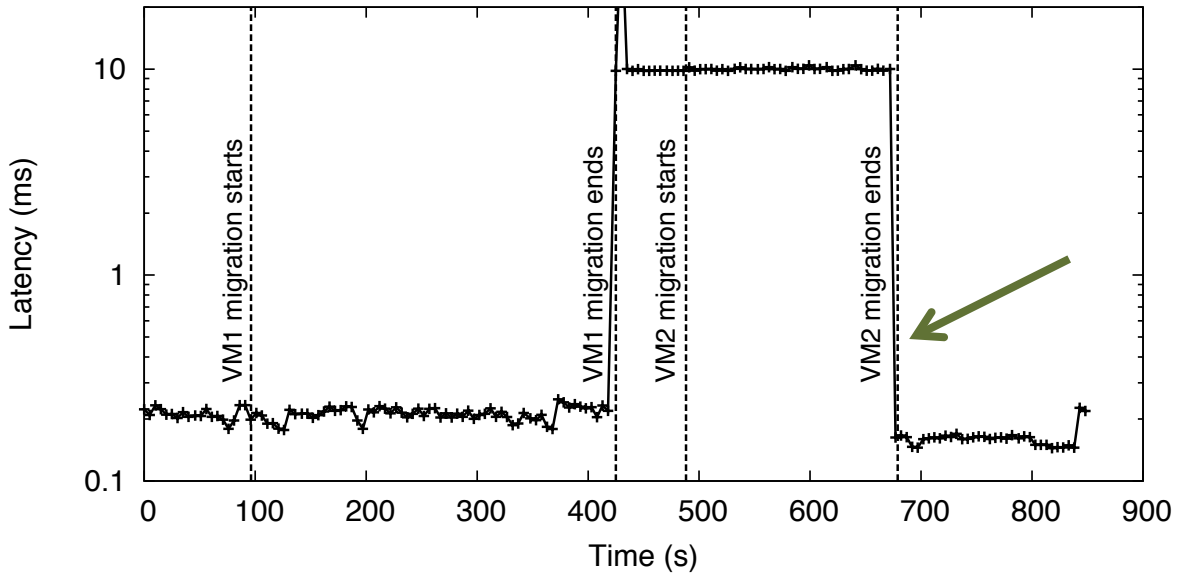




Corm Switching

- Dependency eliminated after second VM migrated
- Switching Corms can recover performance





Migration Numbers

- Experimental setup has overhead
  - Nesting increases downtime by 43%
- Flexibility to do such extreme migration has a cost
  - Migrating Banana endpoints introduces a further 30% overhead
- Cross-cloud live migration had 2.8s downtime

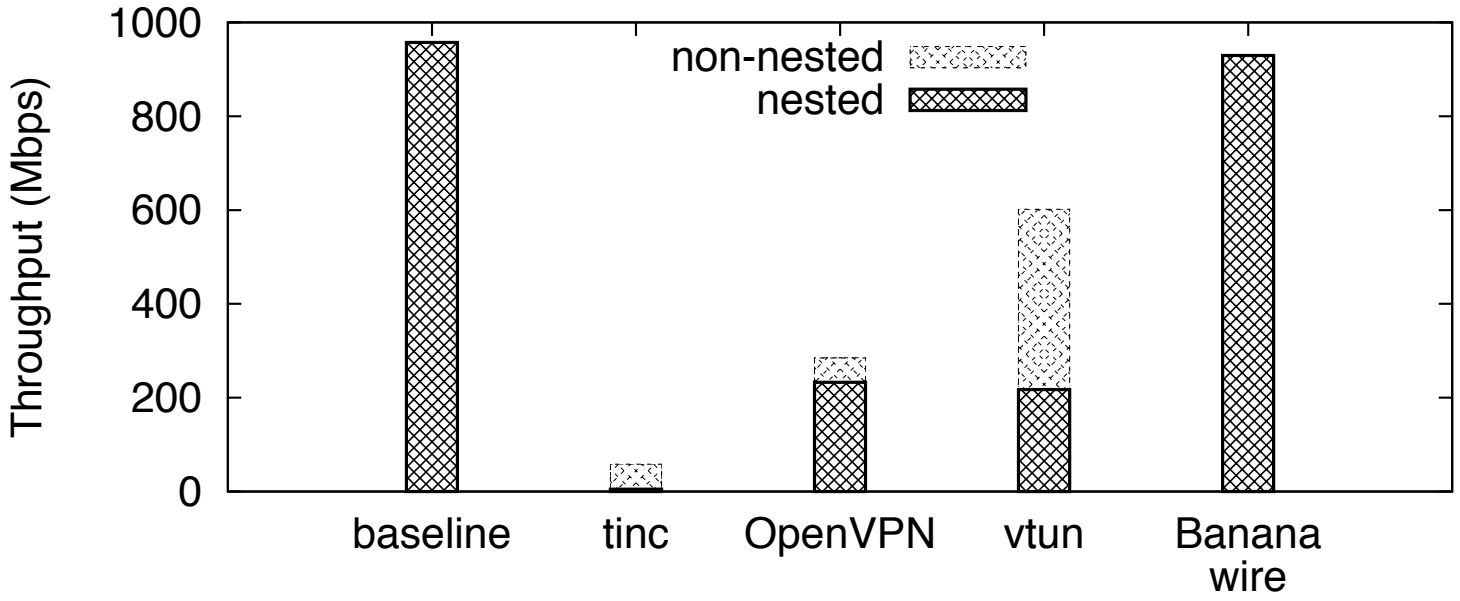
	Downtime	Duration
Banana	1.3 [0.5]	20.13 [0.1]
None (nested)	1.0 [0.5]	20.04 [0.1]
None (not nested)	0.7 [0.4]	19.86 [0.2]

Mean [w/std. dev] local to local live VM migration (s)

	Downtime	Duration
Local to Local	1.3 [0.5]	20.13 [0.1]
EC2 to EC2	1.9 [0.3]	10.69 [0.6]
Local to EC2	2.8 [1.2]	162.25 [150.0]

Mean [w/std. dev] local to local live VM migration (s)

Encapsulating Wires



- UDP throughput experiment
- Using kernel module is especially important in nested environment
  - Factor of 1.55 improvement becomes factor of 3.28

Related Work

- Other cut points
  - Block tap drivers (block devices)
- Continuous access
  - Netchannel
- Specialized solutions
  - Nomad, USB/IP
- Virtual networking
  - VL2, NetLord, VIOLIN, others

Summary

- Banana Double-Split driver model decouples virtual devices from physical
  - Provides location independence
  - Mappings are software-defined
- So far, implemented network device
  - Demonstrated cross-cloud live migration
- Future directions
  - Controller to manage mappings/re-wirings?
  - Corm switching?
  - Other devices?

Thank You