

Network Migration in Linux

Oren Laadan

orenl@cs.columbia.edu

Network Migration (1)

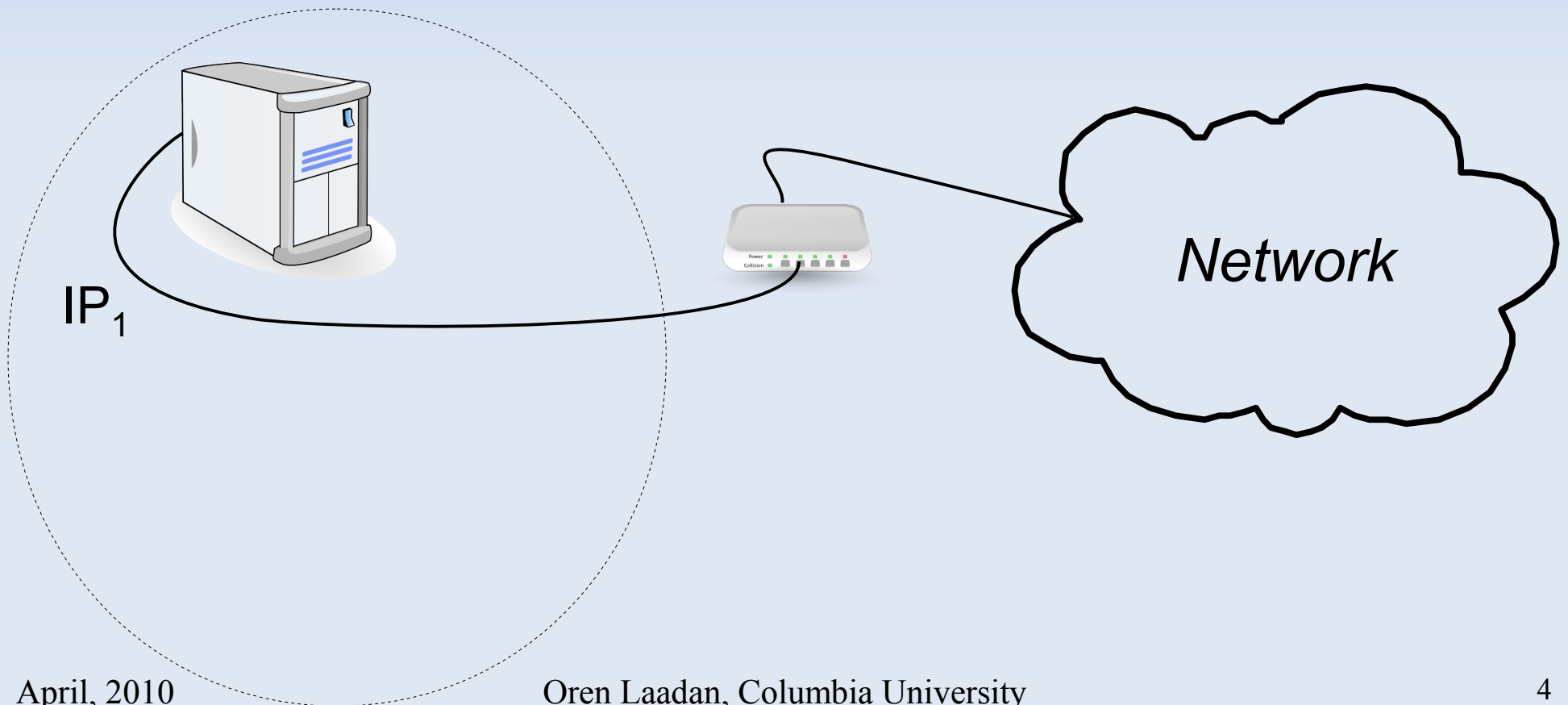
- Goal:
 - transparently migrate a live system between hosts
- Use cases:
 - load balancing
 - fault tolerance
 - green computing
 - Maintenance
 - ...

Network Migration (2)

- "Transparently migrate a live system ..."
 - live system: a system with open network connections with other peers
 - migrate: transfer execution state from one host to another host, and continue there
 - transparently: both the system and the network peers should remain unaware

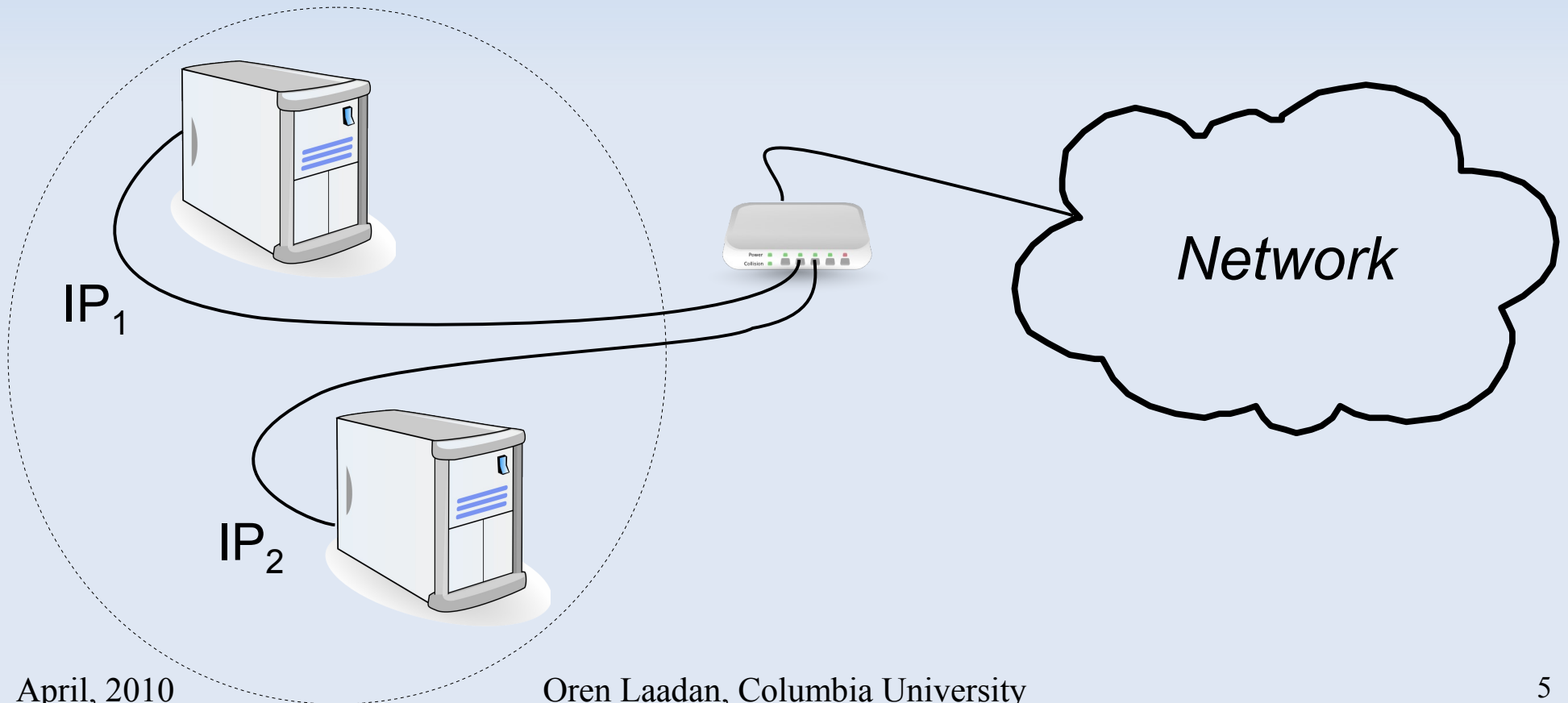
Example: Virtual Machine

- What does it take to migrate a virtual machine while keeping its network connections ?



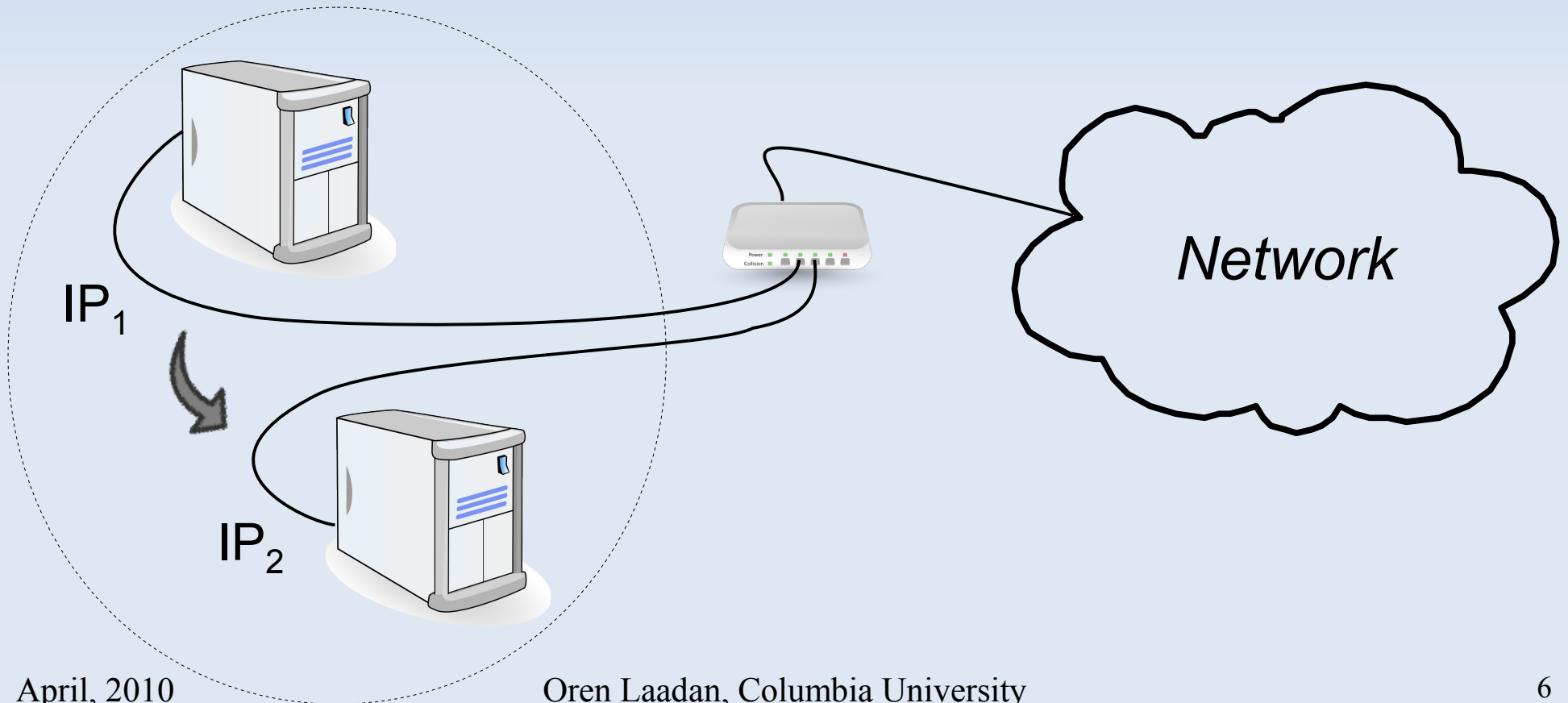
Example: Virtual Machine

- What does it take to migrate a virtual machine while keeping its network connections ?



Example: Virtual Machine

- What does it take to migrate a virtual machine while keeping its network connections ?



Virtual Machine Migration

- Preserve IP through migration
 - only within same IP subnet
- Remap IP ↔ MAC address
 - use gratuitous ARP
- Lost packets ?
 - UDP - don't care
 - TCP - retransmission after timeout

Application Migration

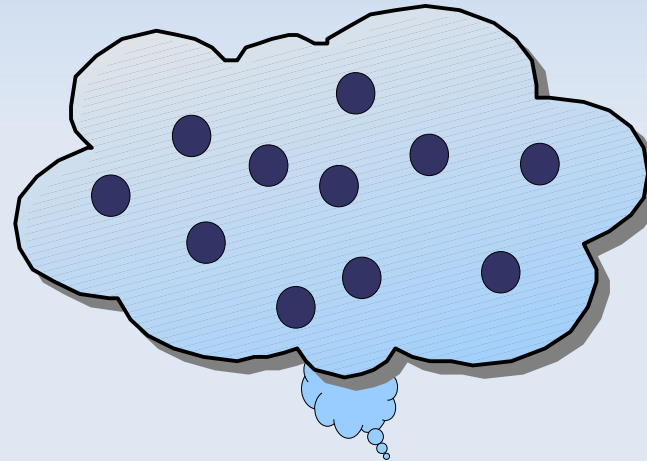
- What if we don't want to move an entire VM ?
- What if we want to move specific applications ?

Operating System Virtualization

- Virtual execution environment
- Encapsulate entire applications
- Private, virtual namespace:
 - private: isolate and confine dependencies
 - virtual: self containers, decoupled from OS

Virtual Execution Environment

Virtual Execution Environment (VEE)

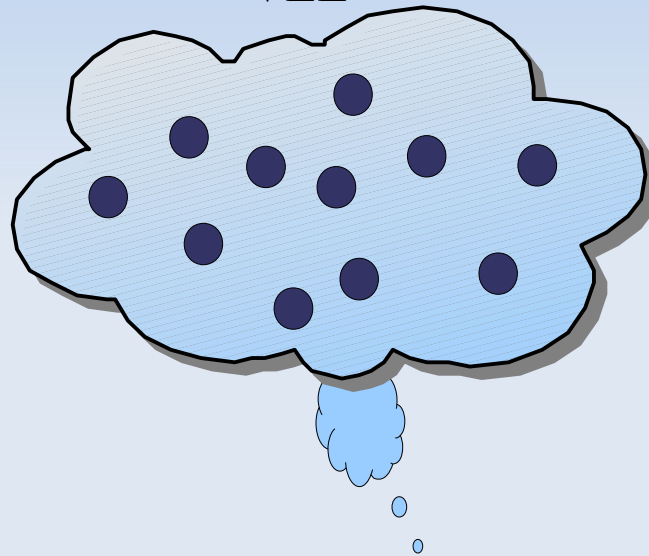


Virtualization Layer

Operating System

Hardware

VEE



getpid() ?

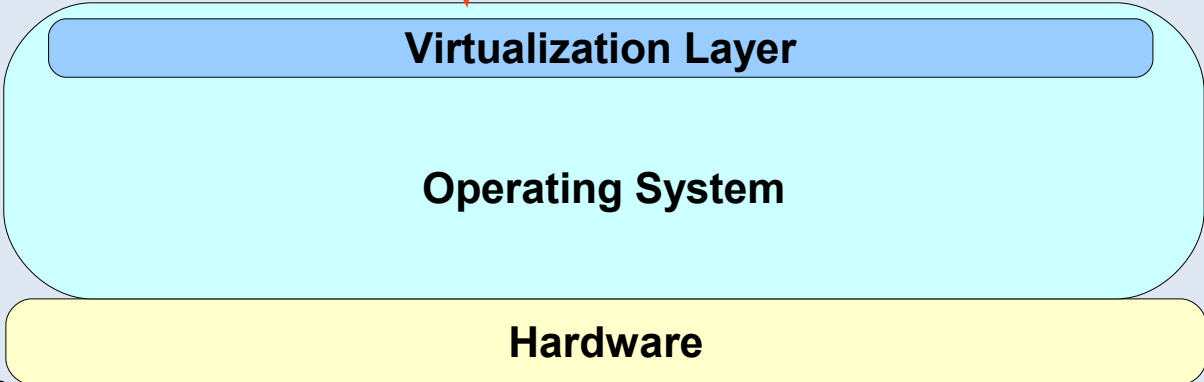
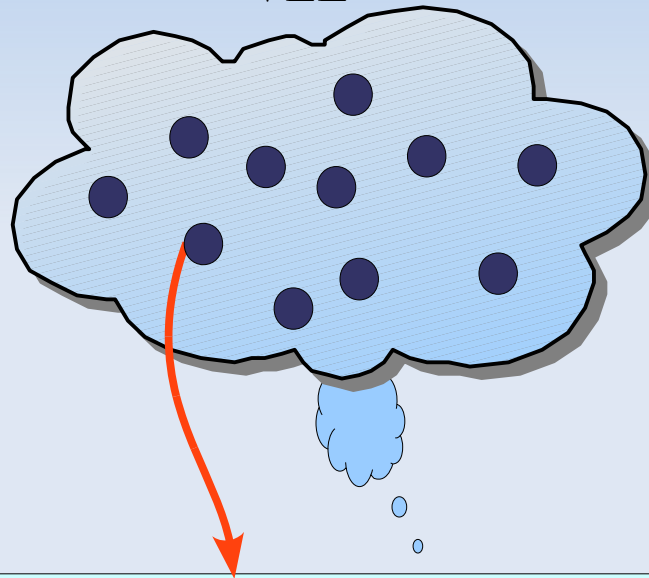
Virtualization Layer

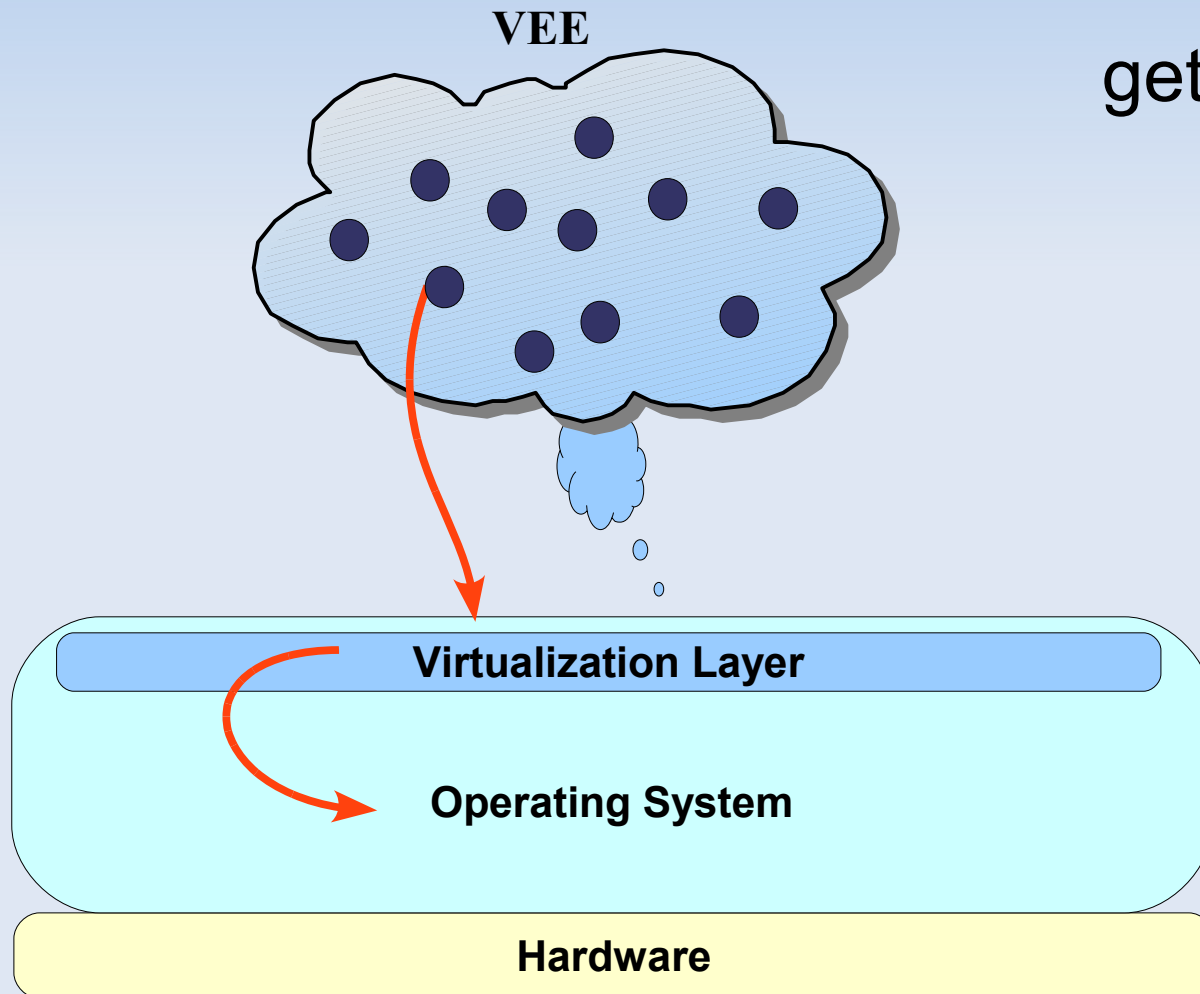
Operating System

Hardware

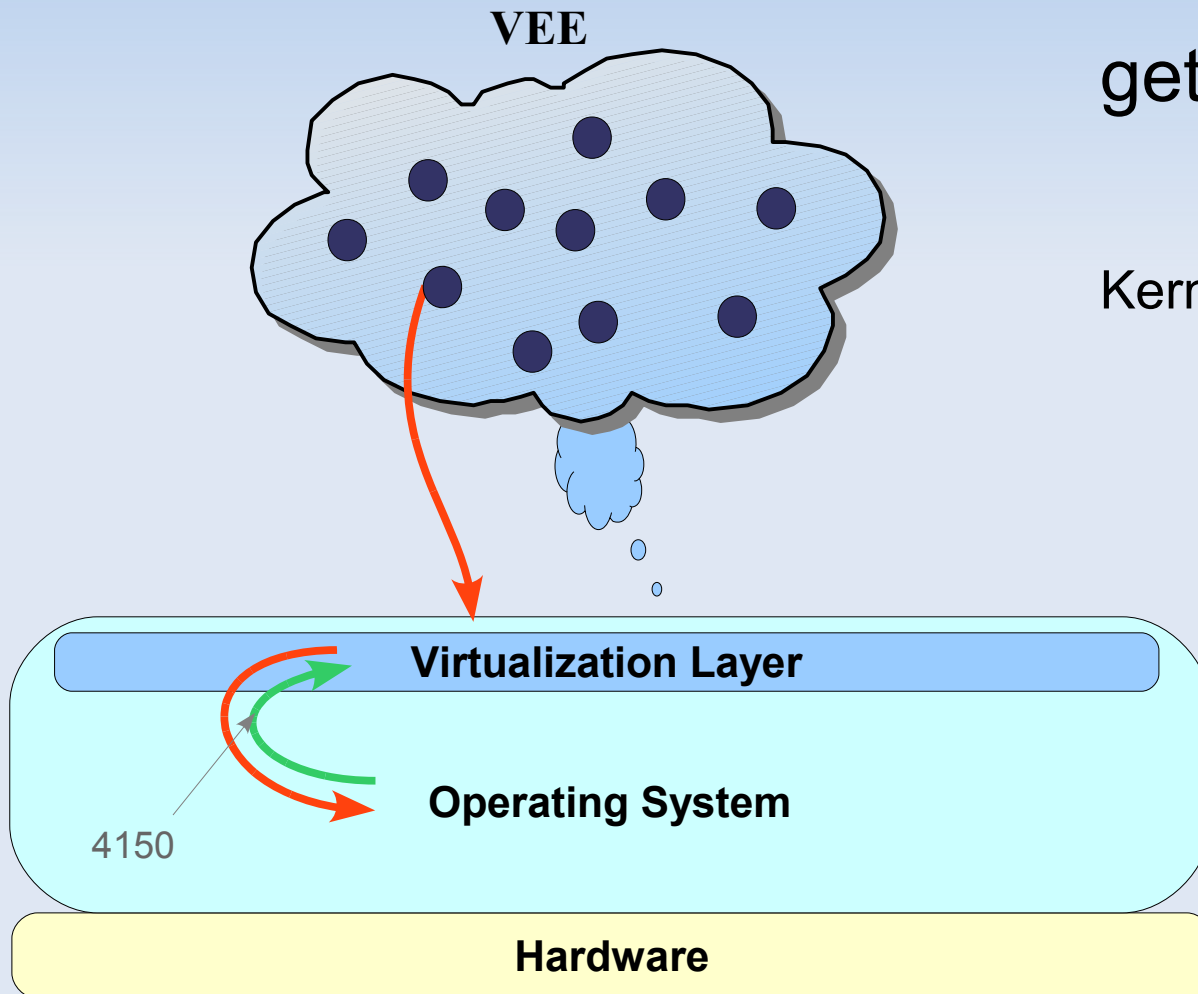
VEE

getpid() ?



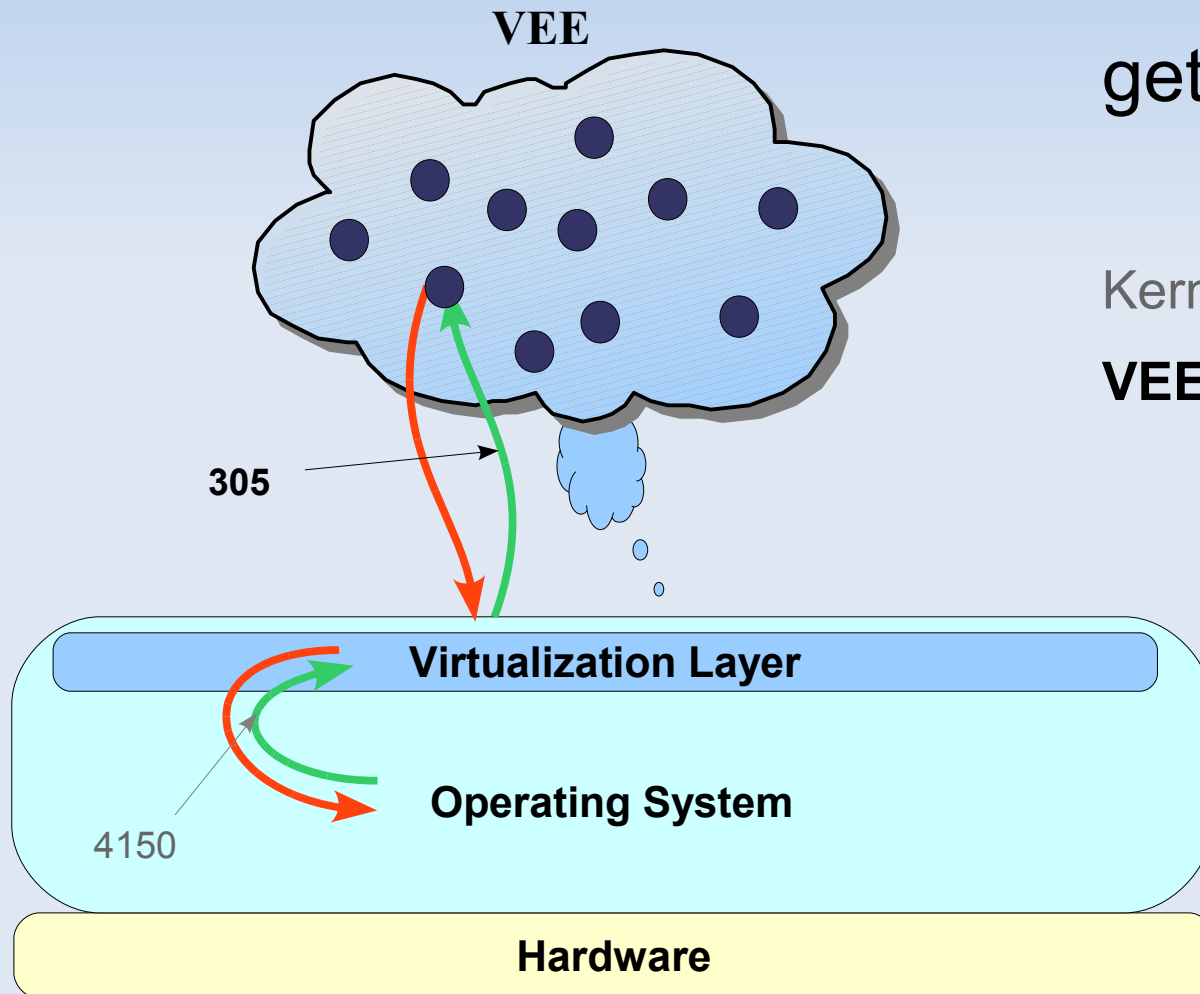


getpid() ?



getpid() ?

Kernel: 4150 (real)

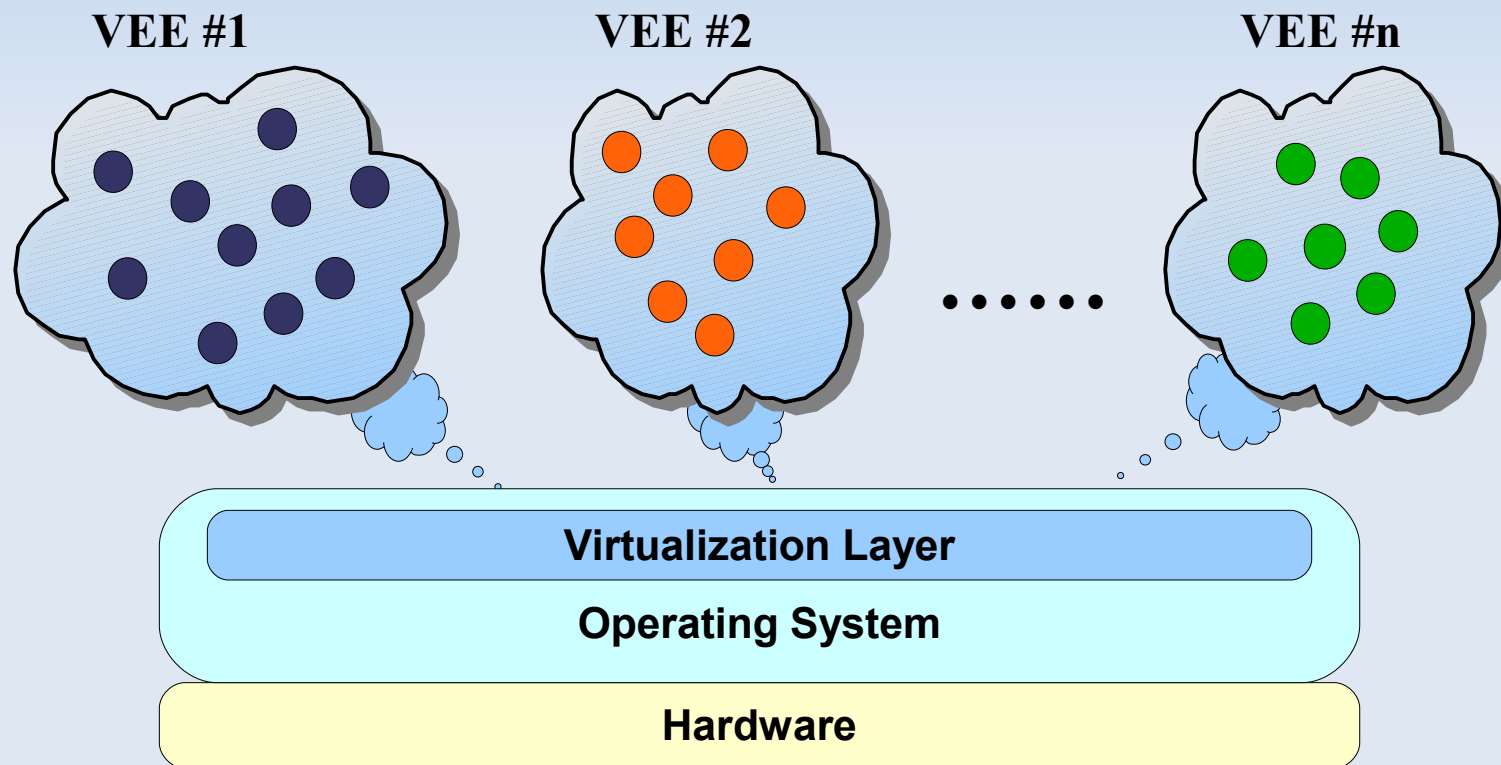


getpid() ?

Kernel: 4150 (real)

VEE: 305 (virtual)

Private Namespace



VM vs Application Migration

- Virtual Machine
 - entire operating system state
 - includes network stack and drivers
- Application Container
 - only designated application(s)
 - specific network connections
 - only IP layer and above
 - exclude driver state

Application Migration

- Preserve IP through migration
 - only within same IP subnet
- Remap IP ↔ MAC address
 - use gratuitous ARP
- Lost packets ?
 - UDP - don't care
 - TCP - retransmission after timeout
- **But also ... ?**

Application Migration

- Reconstruct network connections
 - rebuild sockets
 - reconnect to peers
 - transparently

Network Migration

Source (checkpoint)

1. block network
2. save state
3. transfer state
- 4.
5. kill application
6. kill network
7. unblock network

Target (restart)

- block network
- setup network
- receive state
- rebuild state
- gratuitous ARP
- unblock network

Newtork Migration (1)

- Step 1: Block the network
 - only block for migrating container
 - other containers remain unaffected
 - how do we do that ?

Newtork Migration (1)

- Netfilter and IPTables
- Block the network
 - drop all packets to/from container IP
- Unblock the network
 - remove the above rule ...

TCP Retransmissions

- What happens if migration takes long enough such that packets are lost ?
- How to resume full network speed when migration completes ?

TCP Retransmissions

- What happens if migration takes long enough such that packets are lost ?
- How to resume full network speed when migration completes ?
- Cheat ...
 - Publish 0-window prior to checkpoint
 - Publish (restore) original after restart

Newtork Migration (2)

- Step 2: Save Network State
 - what do we need to save ?

Newtork Migration (2)

- Step 2: Save Network State
 - newtork properties: IPs, netmask, broadcast, ...
 - sockets used by applications: type, protocol, ...
 - lingering (unreferenced) sockets with buffers
 - half-baked connections (received, not accepted)
 - netfilter and iptables configuration

Network Migration (4)

- Step 4: Restore Network State
 - how do we do this ?
 - hidden caveats ?

Network Migration (4)

- Step 4: Restore Network State
- Brute force...
 - conceptually simple
 - deep understanding of kernel stack
 - duplicate existing kernel code

Restore Socket: Fuerza Bruta

- Quick code walk-through...

Questions ?

Thank You !

orenl@cs.columbia.edu