Virtualization

COS 316 Lecture 16

Amit Levy

Administrativia

- Assignment 4: Object Relational Mapper
    - Due tomorrow (Nov. 5th)

Midterm: Naming Analysis

- Which names and values are right?
  - Any names and values *can* be correct, this is just an analysis tool
  - They have to be consistent with the rest of the framework

Midterm: Naming Analysis

- Allocation:
  - How are names and values chosen?
  - *Not* how those names and values are stored.
- Translation:
  - A function from names to values

Midterm: Naming Analysis

Why do we care?

- Help us understand systems—how they are implemented, how they might perform, what semantics should we expect.
- Help us design systems—if we don't know how to allocate or translate names and values, we probably need to choose different names/values.

Virtualization

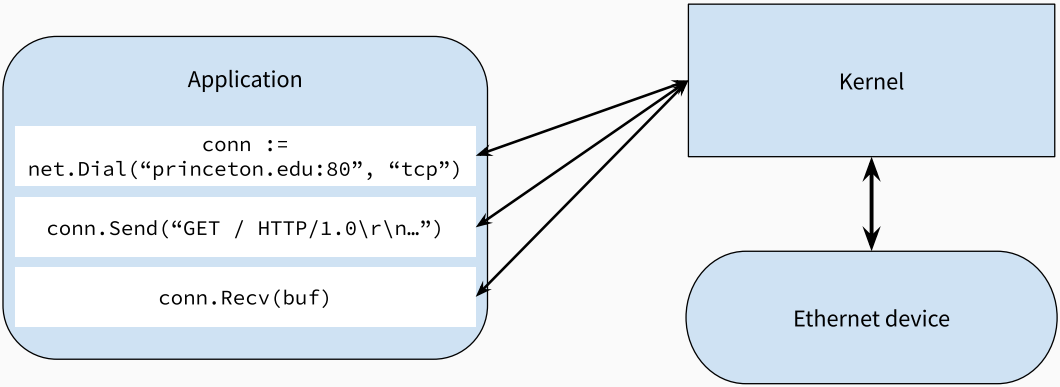
- Virtual machines host multiple “guest” operating systems
- Database transactions allow interleaving requests from multiple clients to remain consistent
    - More on what “consistent” means on Wednesday
- Virtual LAN host isolated local area networks on the same network infrastructure

First: Resource Management through Multiplexing

- Dividing resource by time, or into sub-resources (e.g. TDMA, FDMA, CDMA)
    - How is the resource shareable?
- Provisioning resource division (e.g. congenstion control, CSMA)
    - Who gets which share?
- Framing requests (e.g. Ethernet/IP/TCP headers)
    - Which requests/responses are for me?

Who is responsible for enforcement?

UNIX Network Sockets



Who enforces **carrier-sense multiple access (CSMA)**?

a. Application

b. Kernel

c. Network device (e.g. Ethernet card)

Who enforces **IP header framing**?

a. Application

b. Kernel

c. Network device (e.g. Ethernet card)

Who enforces **TCP congention control**?

a. Application

b. Kernel

c. Network device (e.g. Ethernet card)

Who enforces **TCP header framing**?

a. Application

b. Kernel

c. Network device (e.g. Ethernet card)

Who enforces **HTTP header framing**?

a. Application

b. Kernel

c. Network device (e.g. Ethernet card)

- Network device “virtualizes” physical layer as a bit-stream
- Kernel “virtualizes” ethernet as endpoint in an IP network
- Kernel “virtualizes” IP endpoint as TCP connections

Virtualization as a form of abstraction

Virtualization is the act of presenting a single resource to multiple users as though they each have exclusive access to some resource.

A Plausible Alternative to UNIX Sockets

```
func Send(data []byte, from_addr Addr, to_addr Addr) {
  full_packet := make([]byte, ETHERNET_FRAME_LENGTH)
  copy(packet[:ETHERNET_HEADER_LENGTH + IP_HEADER_LENGTH], myHeader(from_addr, t
  copy(packet[IP_HEADER_LENGTH:], data)
  // OK, lots of details elided
  raw_device := EthernetDevice.Lock()
  raw_device.Send(packet)
  raw_device.Unlock()
}
```

So why use virtualization layers?

So why use virtualization layers?

- Abstraction: Hard to get the details right
- Portability: Details depend on physical, link, network, and transport layers
- *Enforced* resource management

Virtualization as a form of Resource Management

- Fixed vs **arbitrary number of applications**
- **Mandatory** vs cooperative sharing
- **Virtual** vs explicit sharing

Virtual Semantics vs. Performance

A virtualized resource needs to represent the *semantic* behavior of the underlying resource, but not necessarily *performance*.

How do we make virtualization performant?

System has global view of resource demands

- Merge redundant operations from multiple clients
  - Caching, schedule related operations together
- Aggressively interleave operations, but needs to maintain “illusion” of exclusive access.
  - Prof. Lloyd will talk about consistency on Wednesday
- Adapt the interface to make virtualization more efficient
  - Virtual machines & para-virtualization on Monday

References