Virtual Machine Monitors

Lakshmi Ganesh

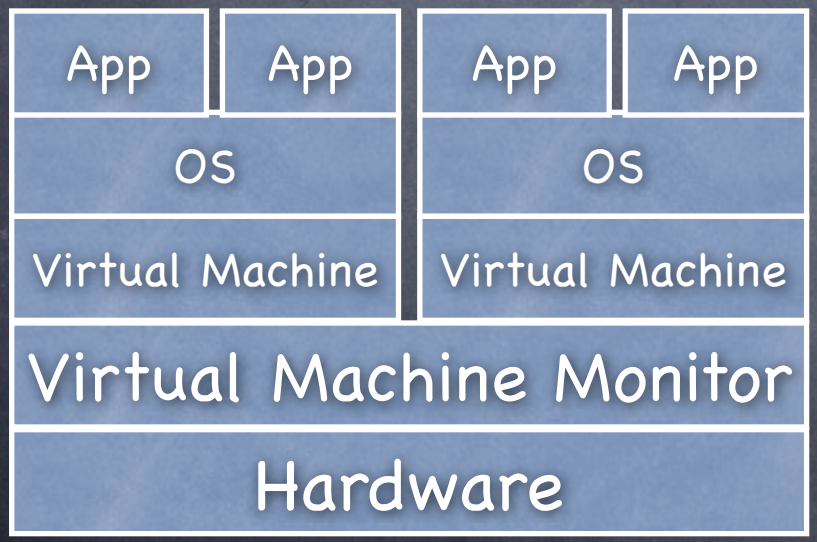
What is a VMM?

- Virtualization: Using a layer of software to present a (possibly different) logical view of a given set of resources
- VMM: Simulate, on a single hardware platform, multiple hardware platforms - virtual machines
  - VMs are usually similar/identical to underlying machine
  - VMs allow multiple operating systems to be run concurrently on a single machine

What is it, really?

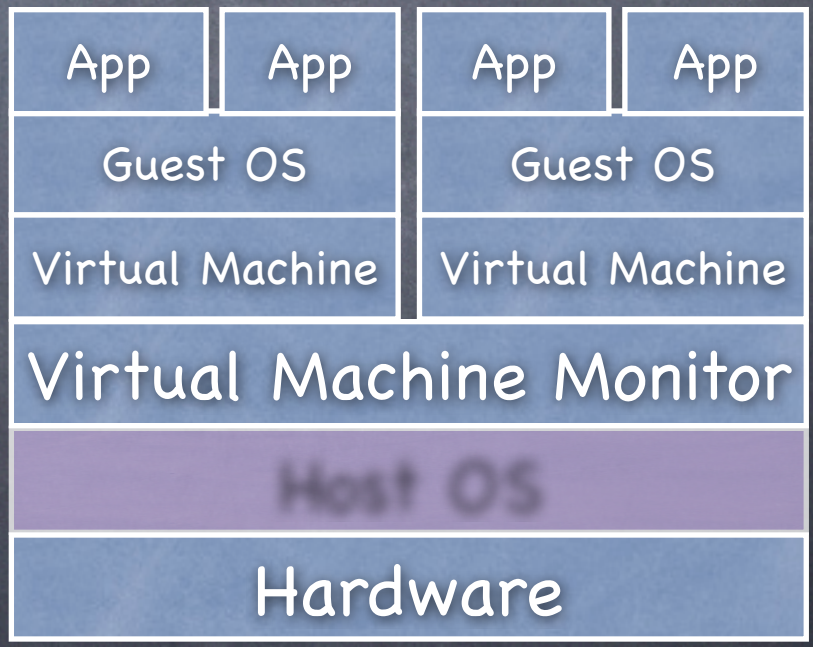
Type 1 VMM:

IBM VM/370, Xen, VMware ESX Server



Type 2 VMM:

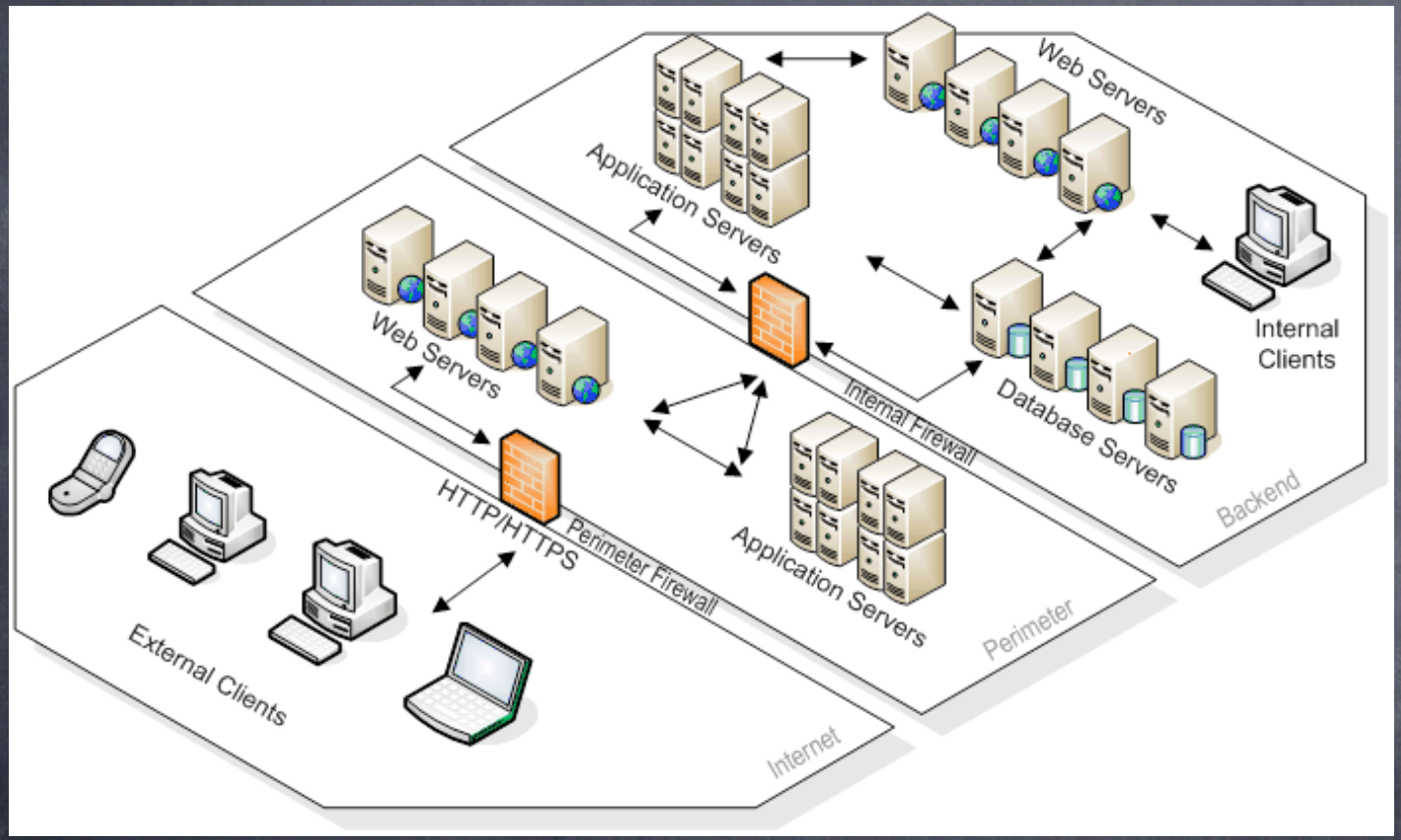
VMware Workstation



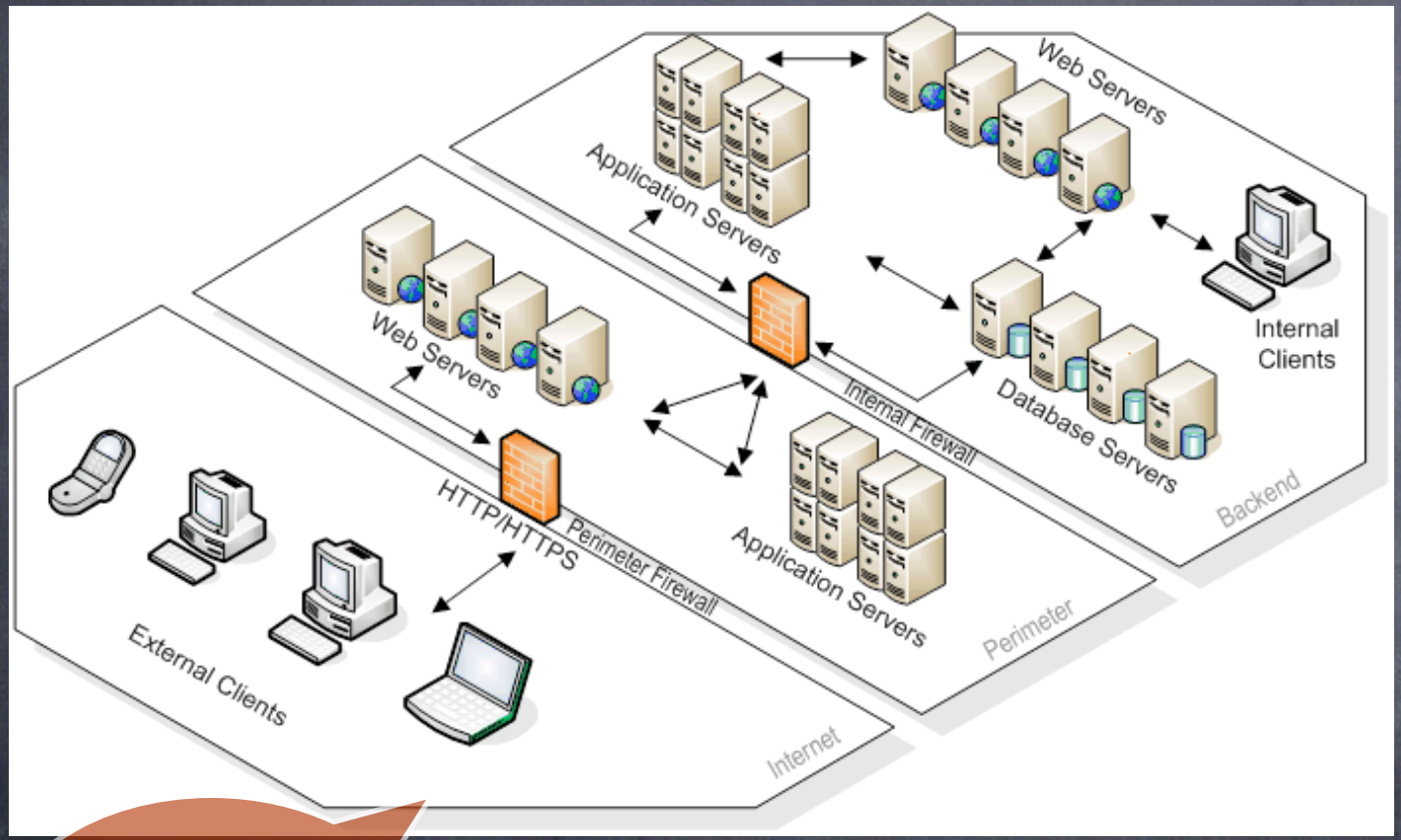
VMMs: Meet the family

- Cousins:
  - Number of instructions executed on hardware:
    - Statistically dominant number: VMM
    - All unprivileged instructions: HVM
    - None: CSIM
- Siblings: VMM subtypes
  - Location of VMM:
    - On top of machine: Type 1 VMM
    - On top of OS (host OS): Type 2 VMM
  - Virtualization approach
    - Full virtualization
    - Paravirtualization

Why is a VMM?

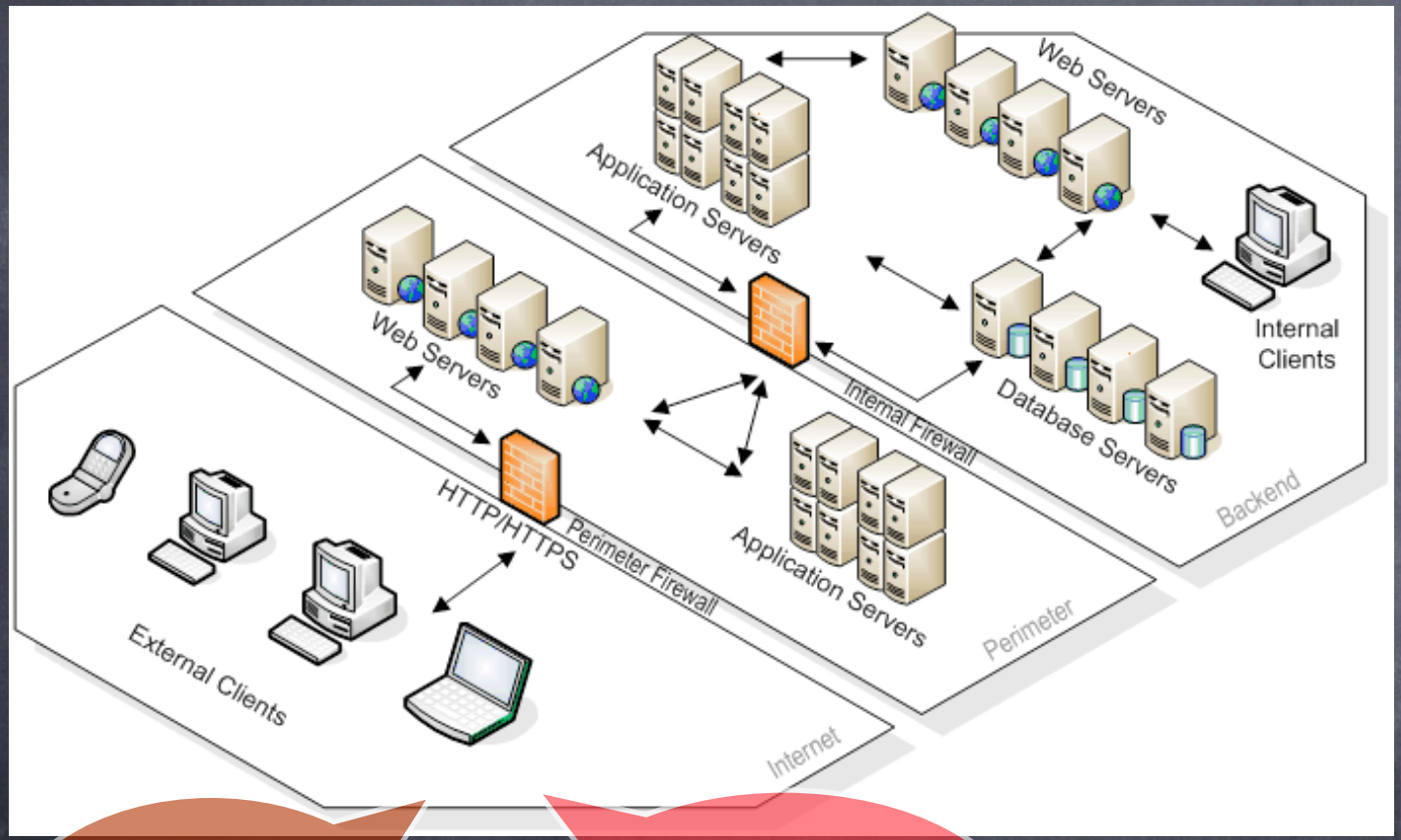

Web Servers
Application Servers
Internal Clients
Web Servers
Internal Firewall
Database Servers
Backend
HTTP/HTTPS
Perimeter Firewall
Application Servers
Perimeter
External Clients
Internet

Why is a VMM?



No more dual booting!

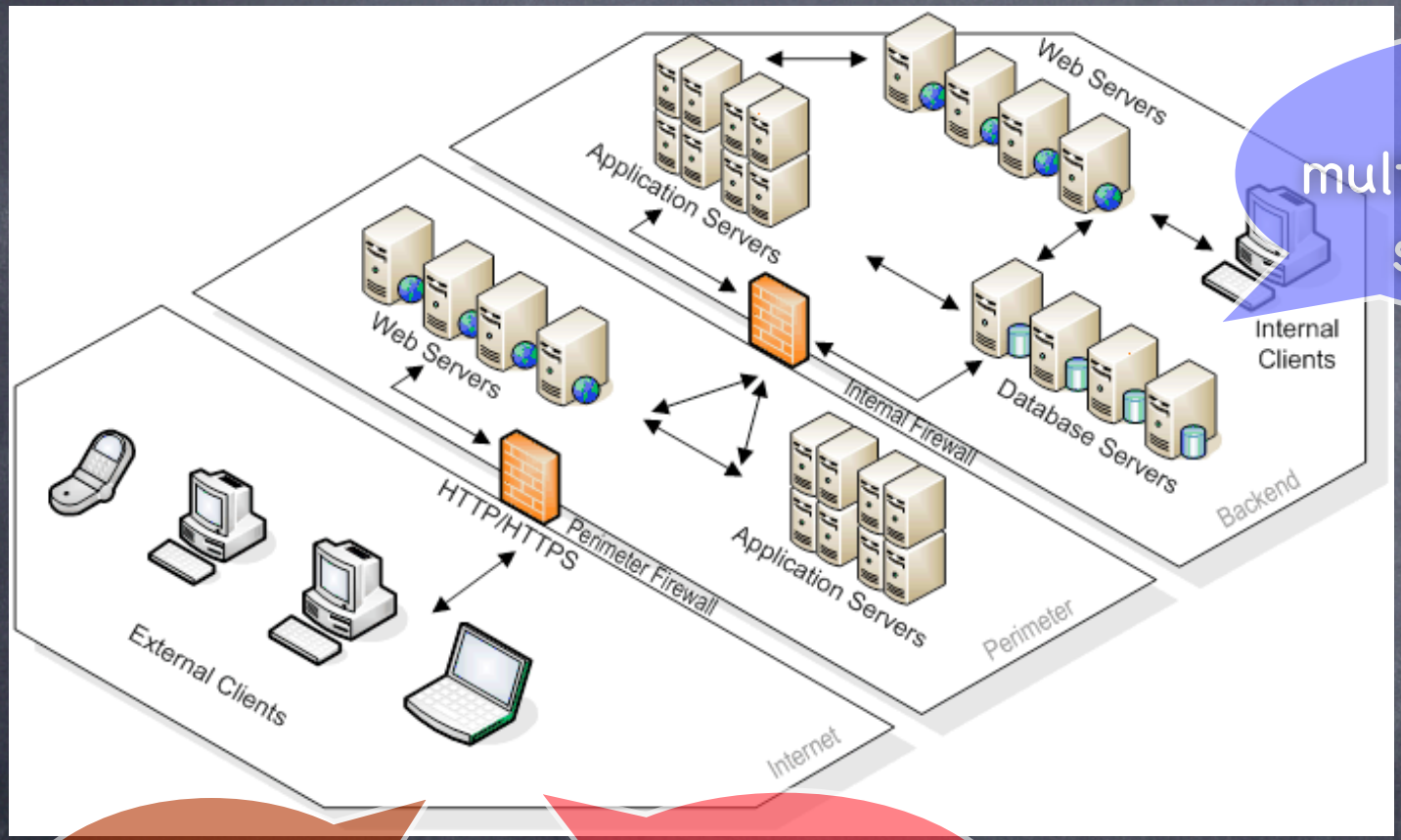
Why is a VMM?



No more dual booting!

Sandbox for testing

Why is a VMM?

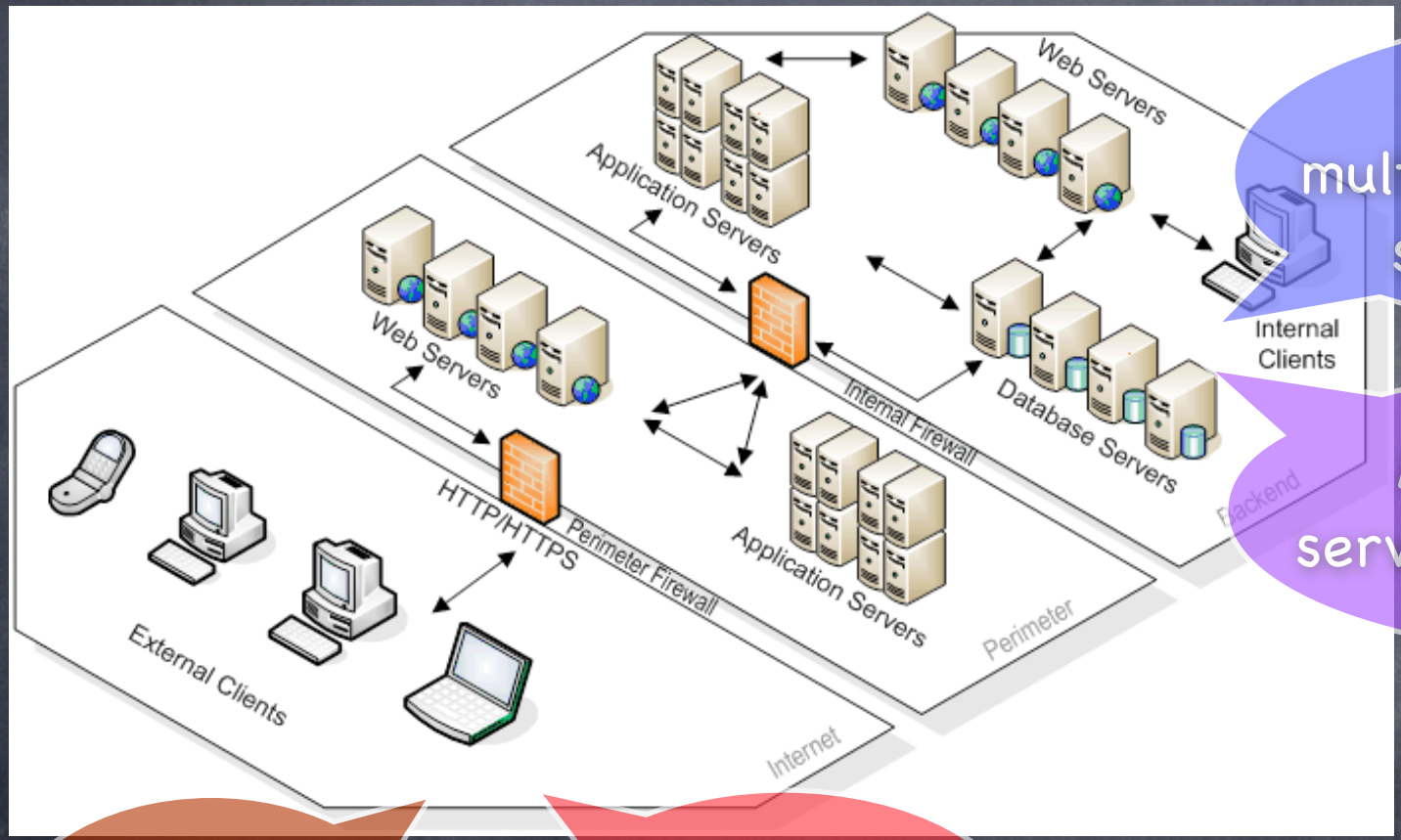


Consolidate multiple servers onto single machine

No more dual booting!

Sandbox for testing

Why is a VMM?



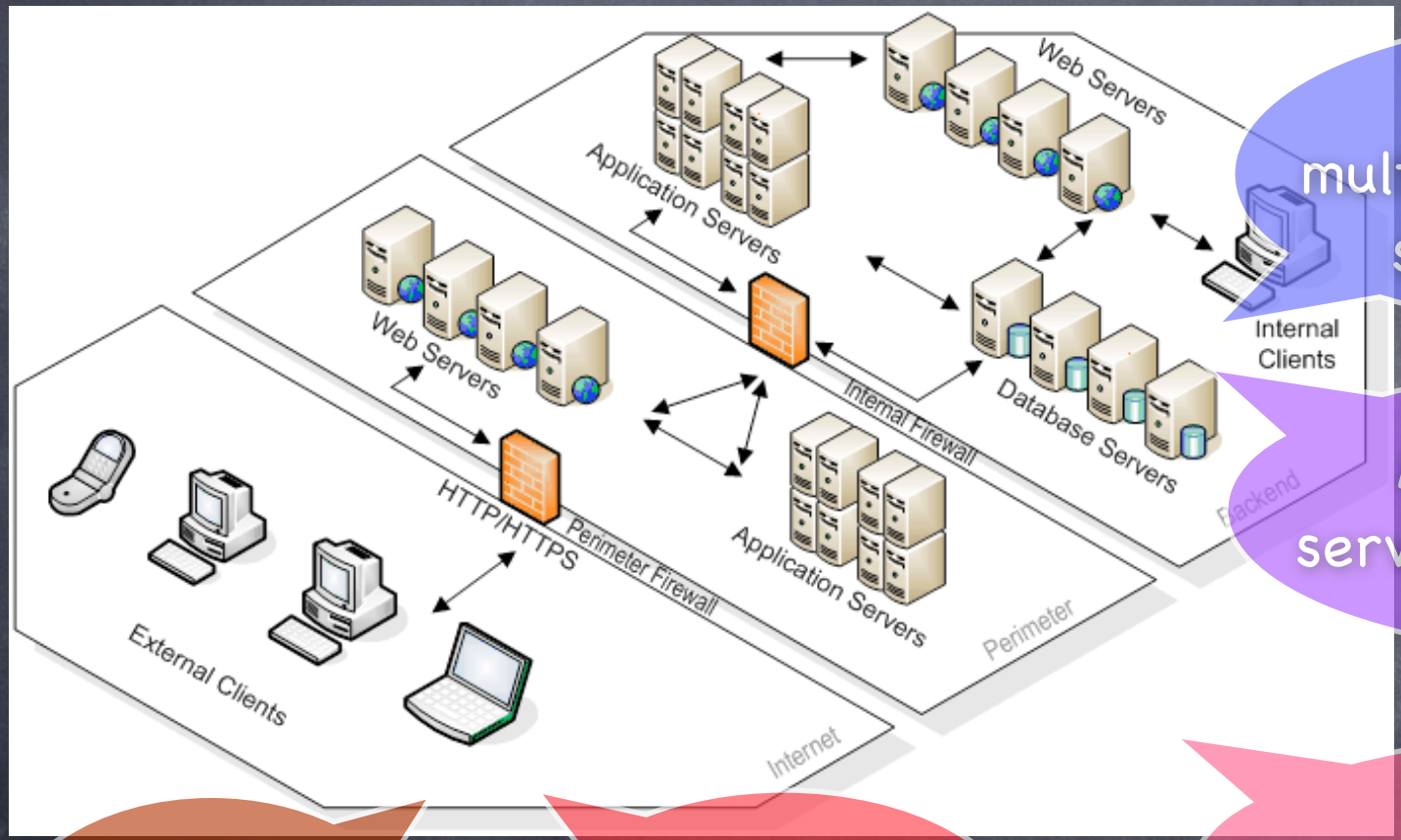
Consolidate multiple servers onto single machine

Add lots more servers - virtual ones!

No more dual booting!

Sandbox for testing

Why is a VMM?



Consolidate multiple servers onto single machine

Add lots more servers - virtual ones!

Flash cloning: adapt number of servers to load

Sandbox for testing

No more dual booting!

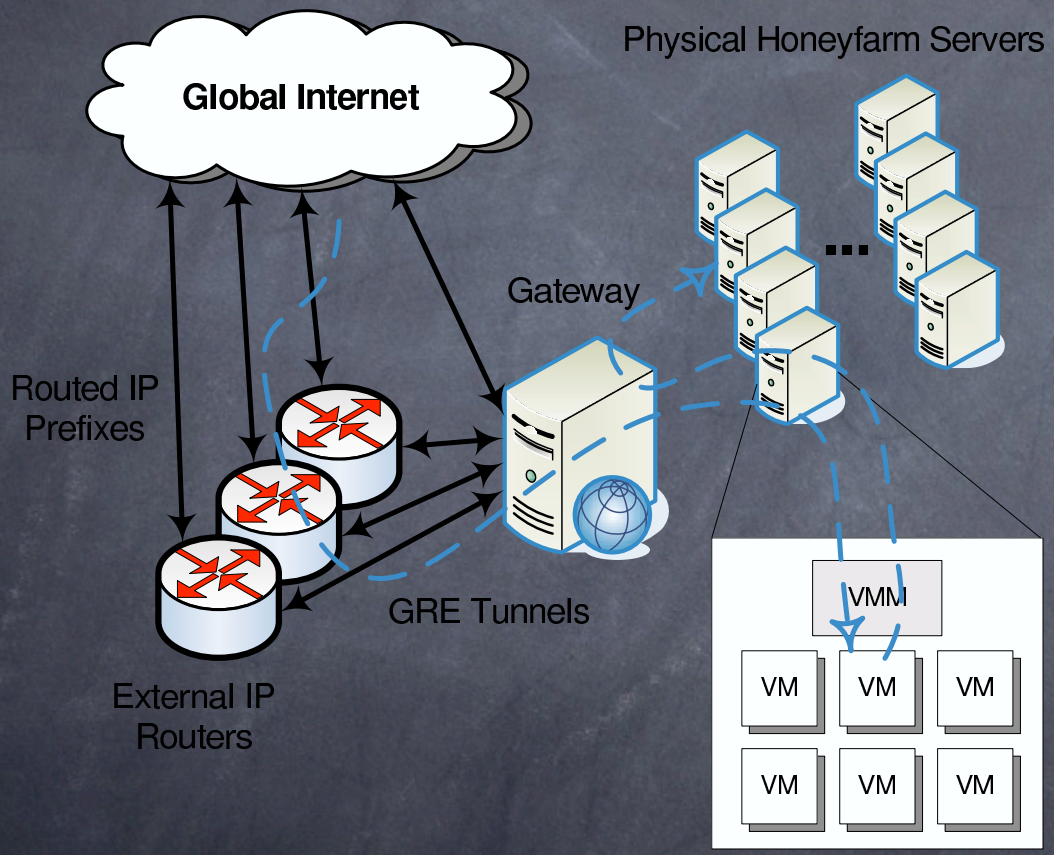
VMMs: Challenges and Design Decisions

- Several warring parameters: what is our goal?
  - Performance: VM must be like real machine!
    - Design Decision: Avoid simulation (Xen, VMware ESX)
    - Design Decision: Type 1 VMM (Xen, VMware ESX)
  - Ability to run unmodified OSes
    - Design Decision: full virtualization (VMware)
  - CPUs non-amenable to virtualization
    - Design Decision: paravirtualization (Xen)

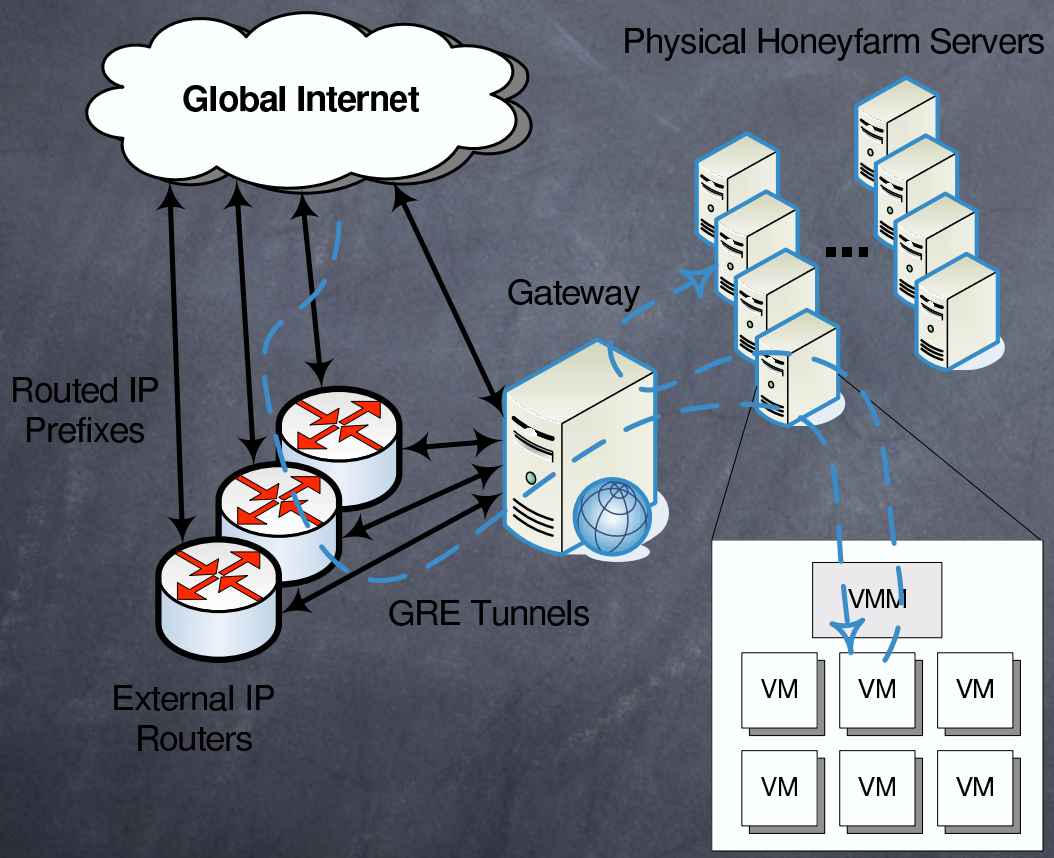
Challenges and Design Decisions (contd.)

- Performance Isolation
  - Design Decision: Virtualize MMU (Xen)
- Scalability: more VMs per machine
  - Design Decision: Memory Reclamation, Shared Memory (Xen, VMware)
- Ease of Installation
  - Design Decision: hosted VMM (VMware WS)
- VMM must be reliable and bug-free
  - Design Decision: Keep it simple: hosted VMM (VMware WS)

Real A Story

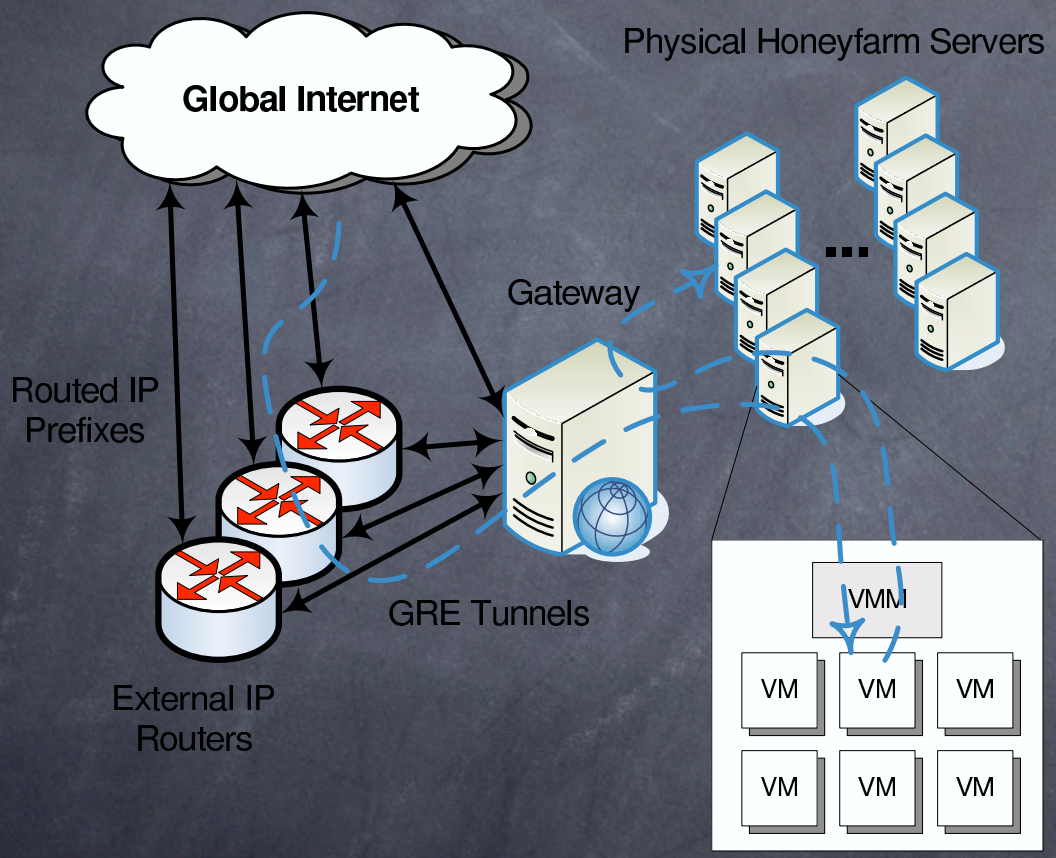

Global Internet
Physical Honeyfarm Servers
Gateway
Routed IP Prefixes
GRE Tunnels
External IP Routers
VMM
VM
VM
VM
VM
VM
VM

Real A Story



Each machine must host thousands of VMs

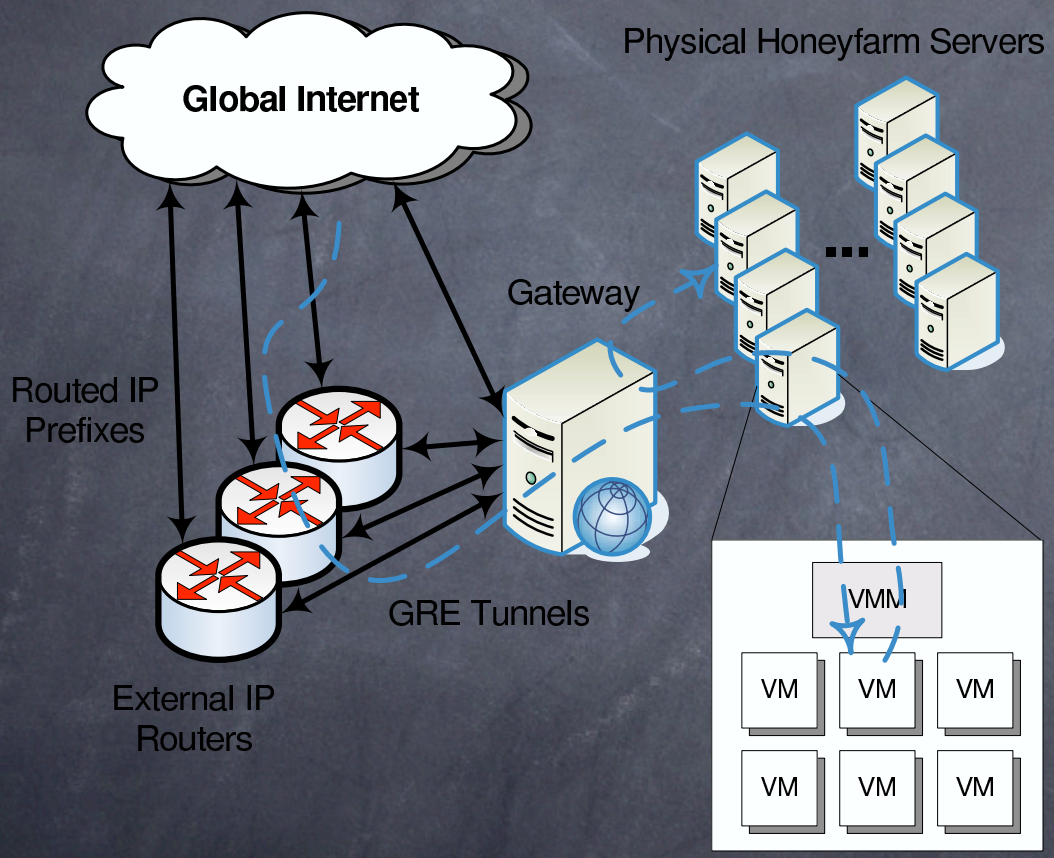
Real A Story



Each machine must host thousands of VMs

Scalability

Real A Story

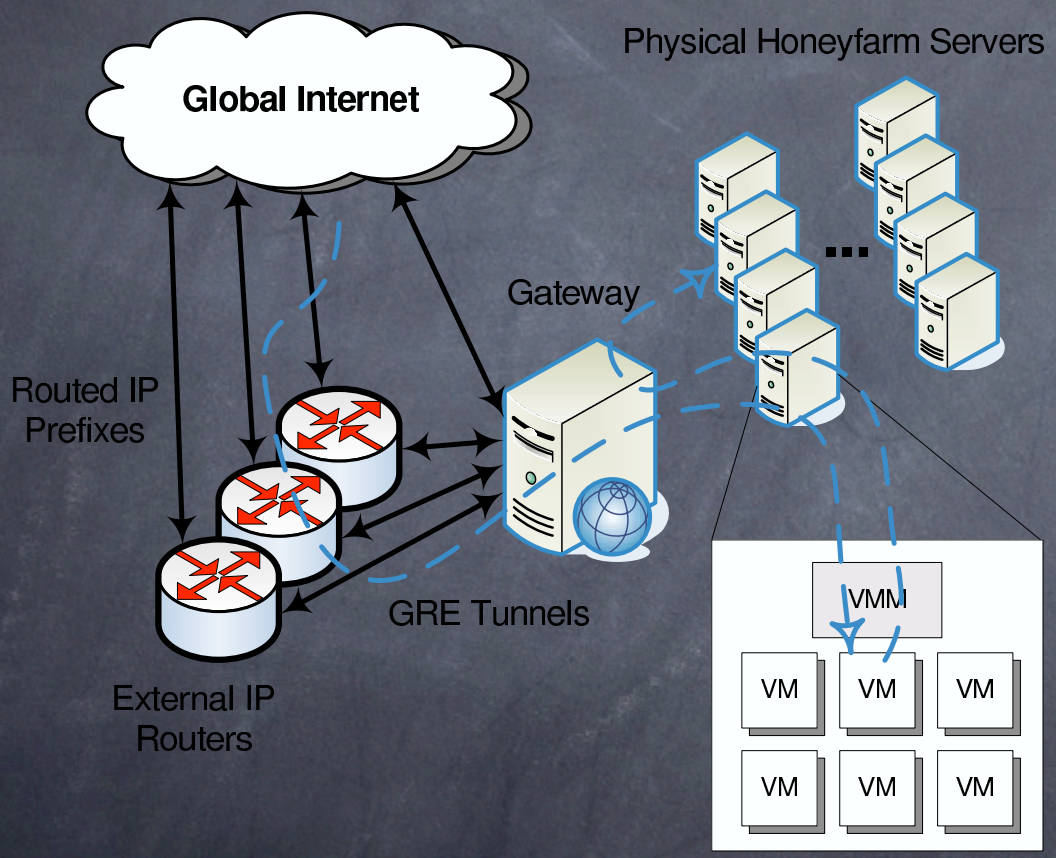


Each machine must host thousands of VMs

Scalability

VMs must run insecure software

Real A Story



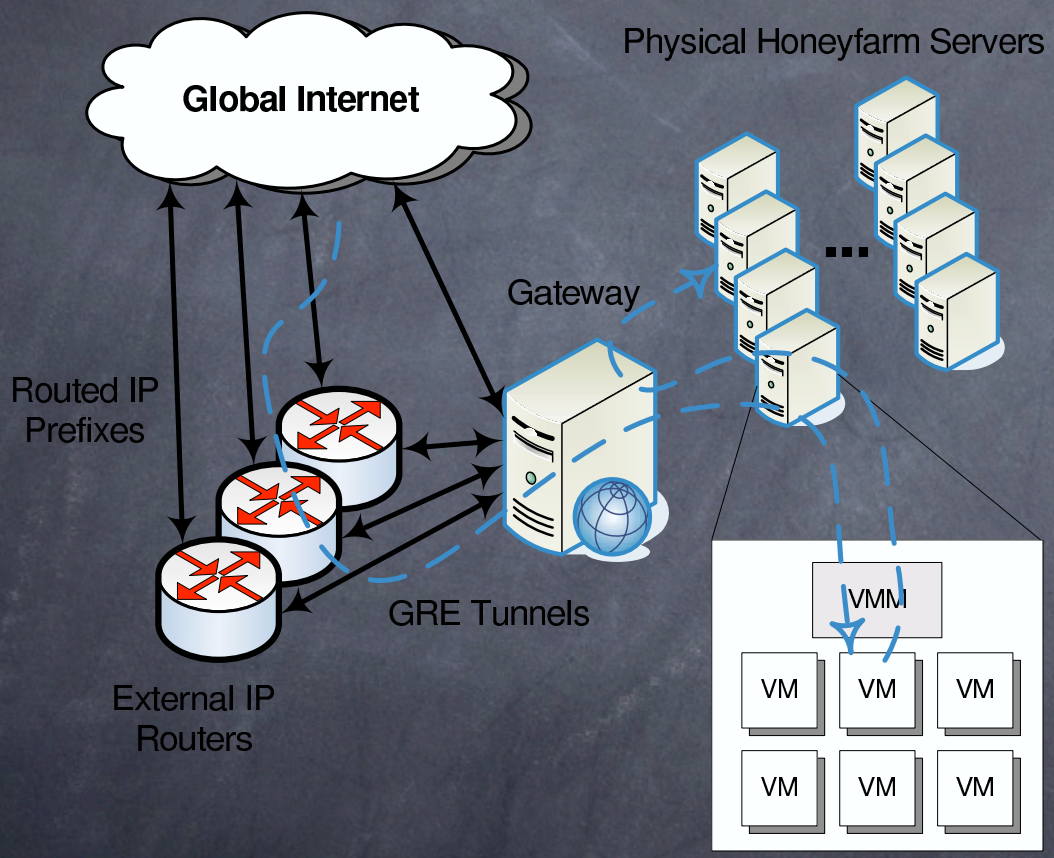
Each machine must host thousands of VMs

Scalability

Fault containment

VMs must run insecure software

Real A Story



Each machine must host thousands of VMs

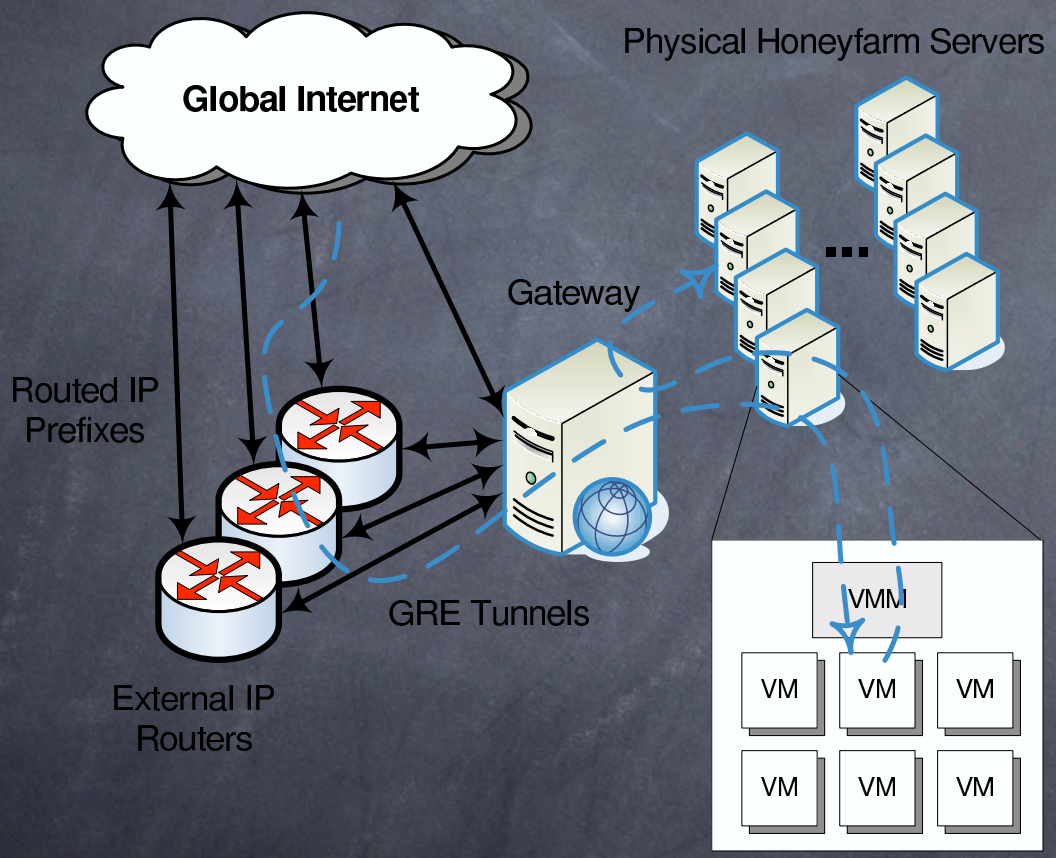
VMM: must send alert when breach occurs

Scalability

Fault containment

VMs must run insecure software

Real A Story



Each machine must host thousands of VMs

VMM: must send alert when breach occurs

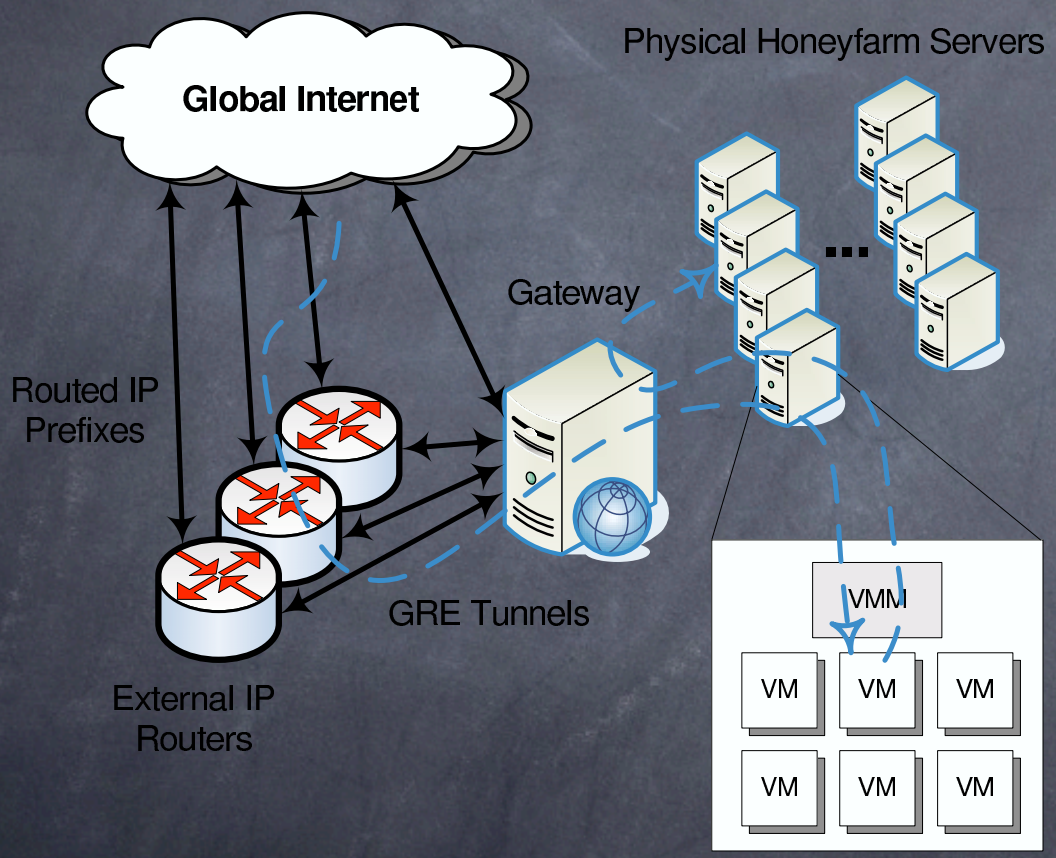
Scalability

Fault containment

Copy-on-write

VMs must run insecure software

Real A Story



Each machine must host thousands of VMs

VMM: must send alert when breach occurs

VM OS must look like native OS to fool malware

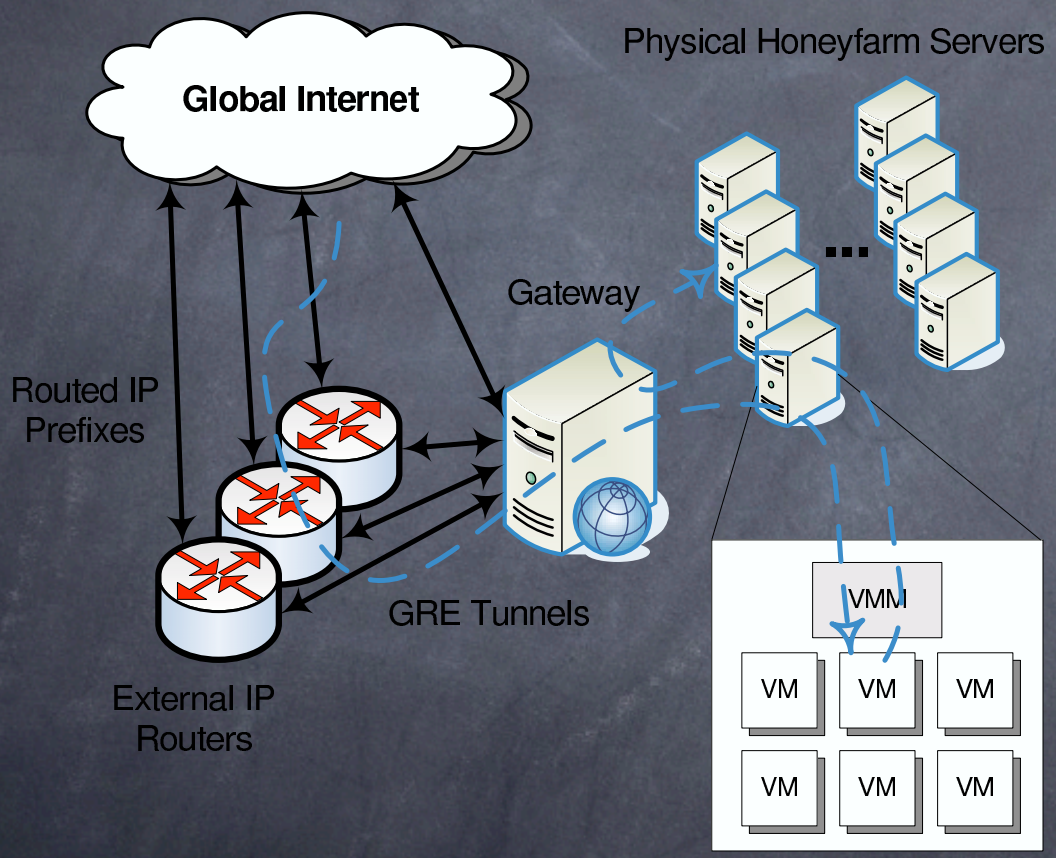
VMs must run insecure software

Scalability

Copy-on-write

Fault containment

Real A Story



Each machine must host thousands of VMs

VMM: must send alert when breach occurs

VM OS must look like native OS to fool malware

VMs must run insecure software

Scalability

Fault containment

Copy-on-write

Minimal OS modification

Case Study: Xen

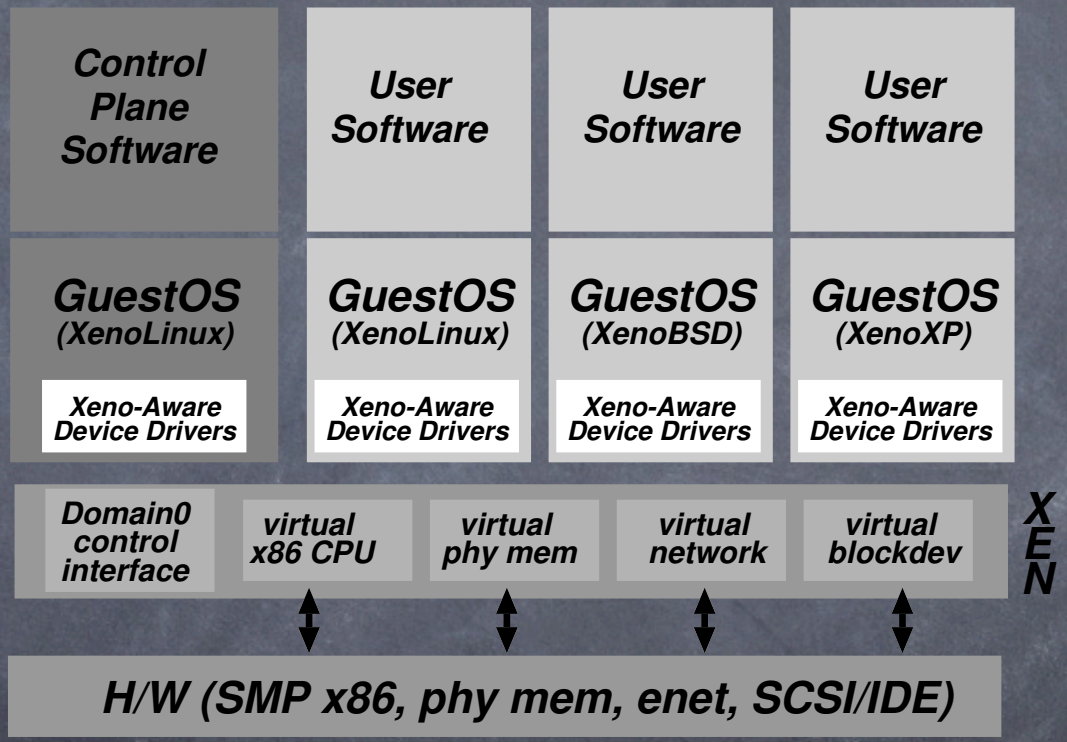


Figure 1: The structure of a machine running the Xen hypervisor, hosting a number of different guest operating systems, including *Domain0* running control software in a XenoLinux environment.

Xen: The case for Paravirtualization

Paravirtualization: When the interface the VM exports is not quite identical to the machine interface

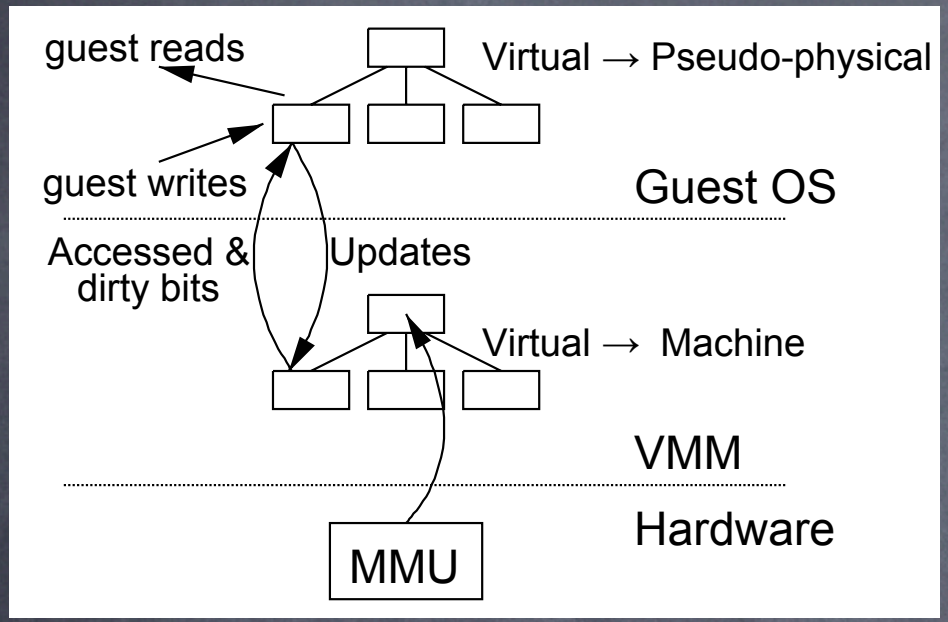
- Full virtualization is difficult
  - non-amenable CPUs, eg. x86
    - Replace privileged syscalls with hypercalls: Avoids binary rewriting and fault trapping
- Full virtualization is undesirable
  - denies VM OSes important information that they could use to improve performance
    - Wall-clock/Virtual time, Resource Availability

Xen: CPU Virtualization

- Xen runs in ring 0 (most privileged)
- Ring 1/2 for guest OS, 3 for user-space
  - GPF if guest attempts to use privileged instr
- Xen lives in top 64MB of linear addr space
  - Segmentation used to protect Xen as switching page tables too slow on standard x86
- Hypercalls jump to Xen in ring 0
- Guest OS may install 'fast trap' handler
  - Direct ring user-space to guest OS system calls

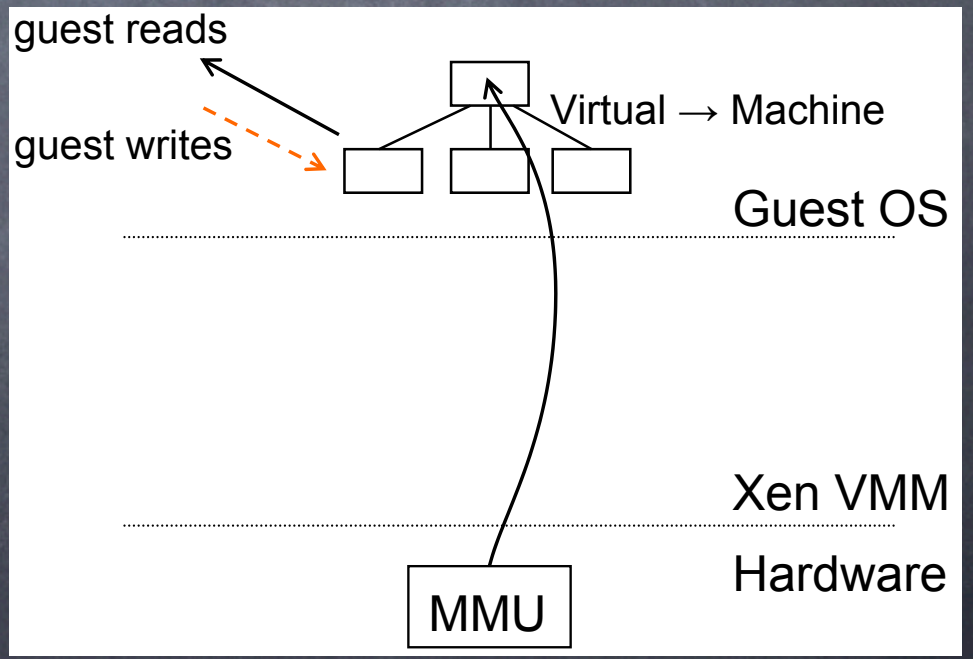
Slide source: Ian Pratt
http://www.cl.cam.ac.uk/research/srg/netos/papers/2004-xen-ols.pdf

Xen: MMU Virtualization



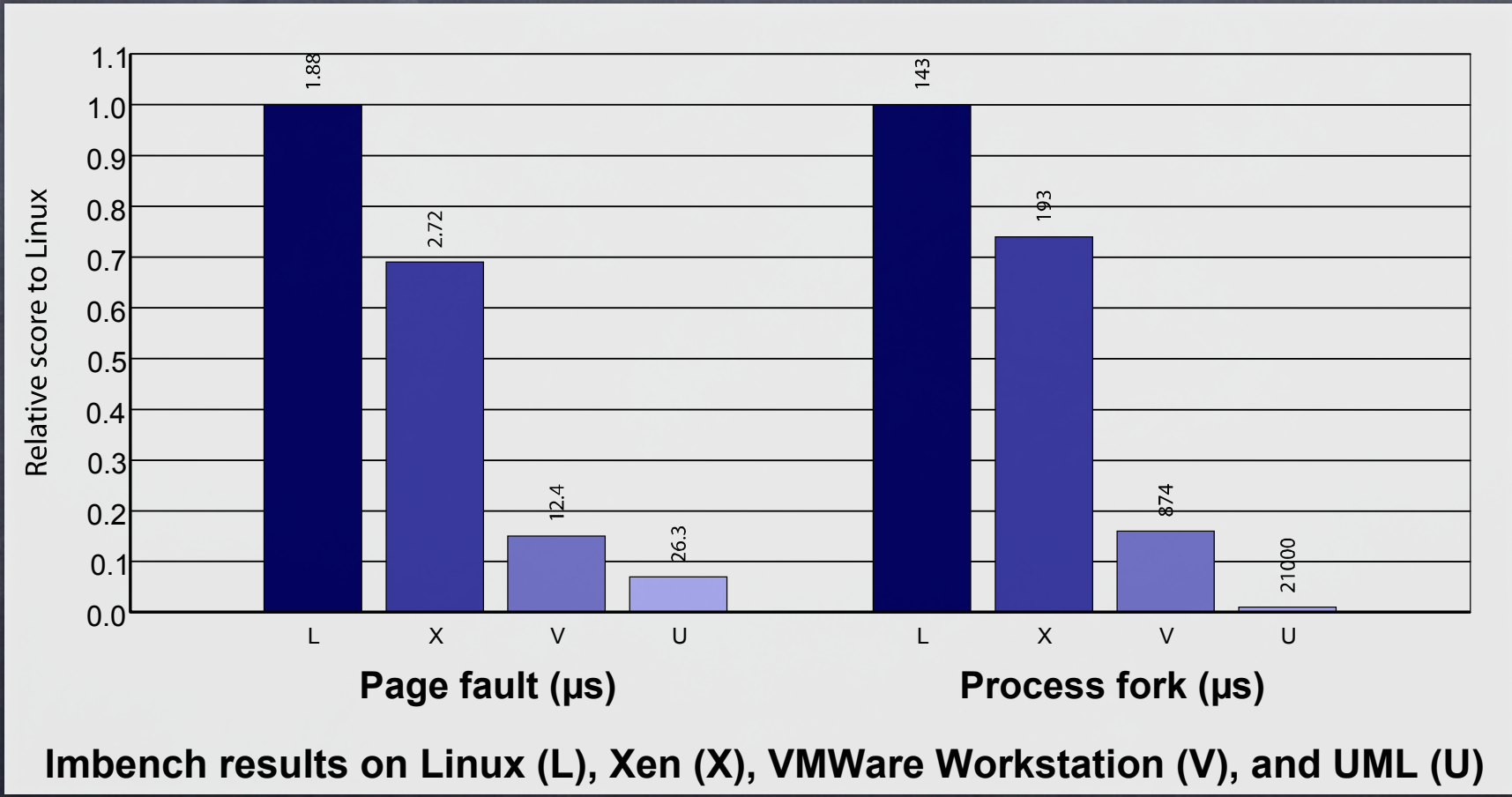
Shadow-mode

Direct-mode



Slide source: Ian Pratt
http://www.cl.cam.ac.uk/research/srg/netos/papers/2004-xen-ols.pdf

MMU Micro Benchmarks



Imbench results on Linux (L), Xen (X), VMWare Workstation (V), and UML (U)

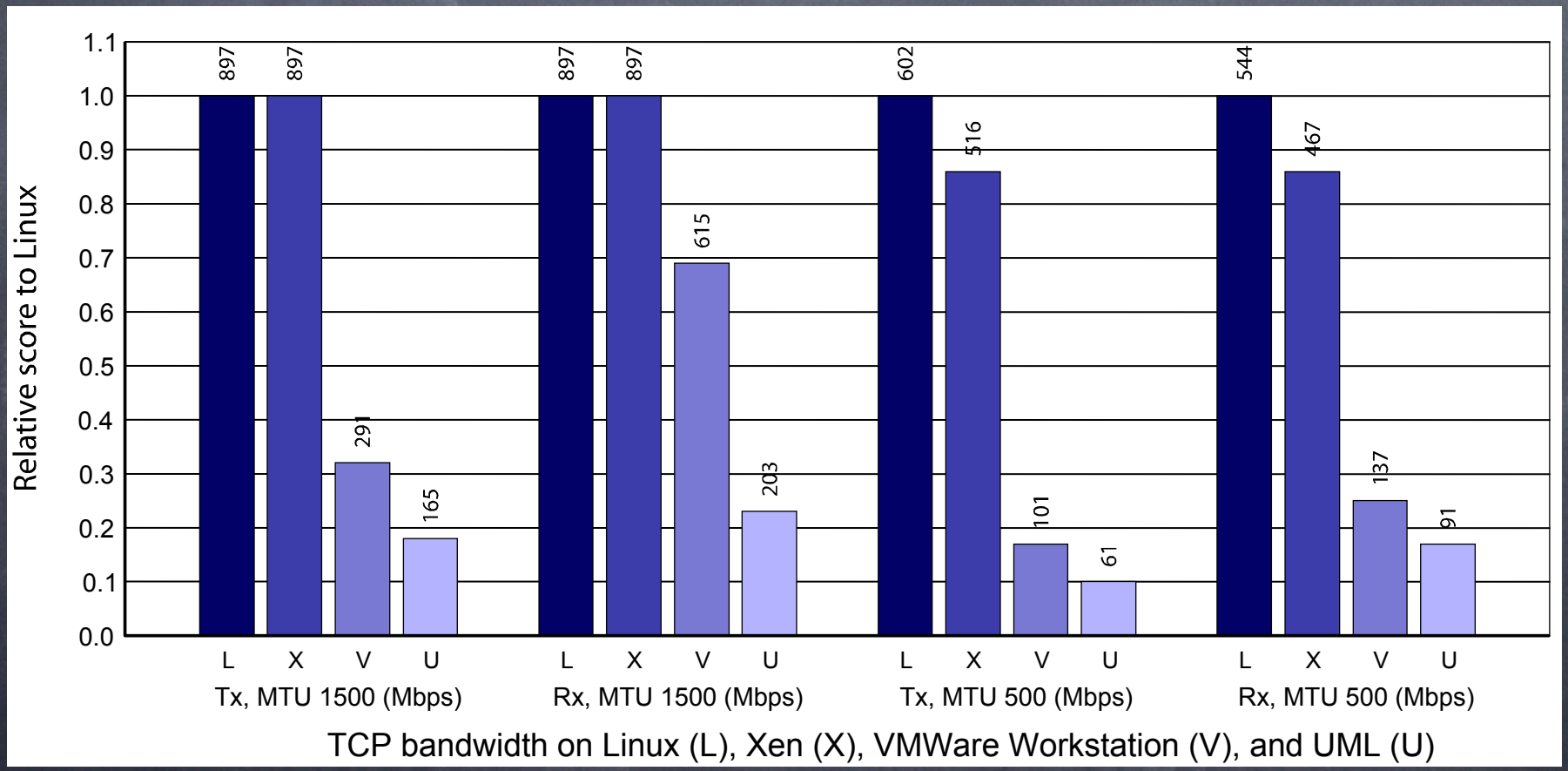
Slide source: Ian Pratt
http://www.cl.cam.ac.uk/research/srg/netos/papers/2004-xen-ols.pdf

Xen: I/O Virtualization

- Device I/O: I/O devices are virtualized as Virtual Block Devices (VBDs)
  - Data transferred in and out of domains using buffer descriptor rings
  - Ring = circular queue of requests and responses. Generic mechanism allows use in various contexts
- Network: Virtual network Interface (VIF)
  - Transmit and Receive buffers
  - Avoids data copy by bartering pages for packets

Slide source: Ian Pratt
http://www.cl.cam.ac.uk/research/srg/netos/papers/2004-xen-ols.pdf

Xen: TCP Results



Slide source: Ian Pratt
http://www.cl.cam.ac.uk/research/srg/netos/papers/2004-xen-ols.pdf

Xen: Odds and Ends

- Copy-on-write
  - VMs share single copy of RO pages
  - Writes attempts trigger page fault
  - Traps to Xen, which creates unique RW copy of page
  - Result: lightweight VMs, can scale well
- Live Migration
  - Within 10's of milliseconds can migrate VMs from one machine to another! (though app dependent)

Xen: Odds and Ends (contd.)

- Live Migration mechanism
  - VM continues to run
    - Pre-copy approach: VM continues to run
    - 'lift' domain on to shadow page tables
      - Bitmap of dirtied pages; scan; transmit dirtied
      - Atomic 'zero bitmap & make PTEs read-only'
    - Iterate until no forward progress, then stop VM and transfer remainder

Xen: Odds and Ends (contd.)

- Memory Reclamation
  - Over-booked resources
  - How to reclaim memory from a VMOS?
    - VMware ESX Server: balloon process
    - Xen: balloon driver

Xen: Scalability

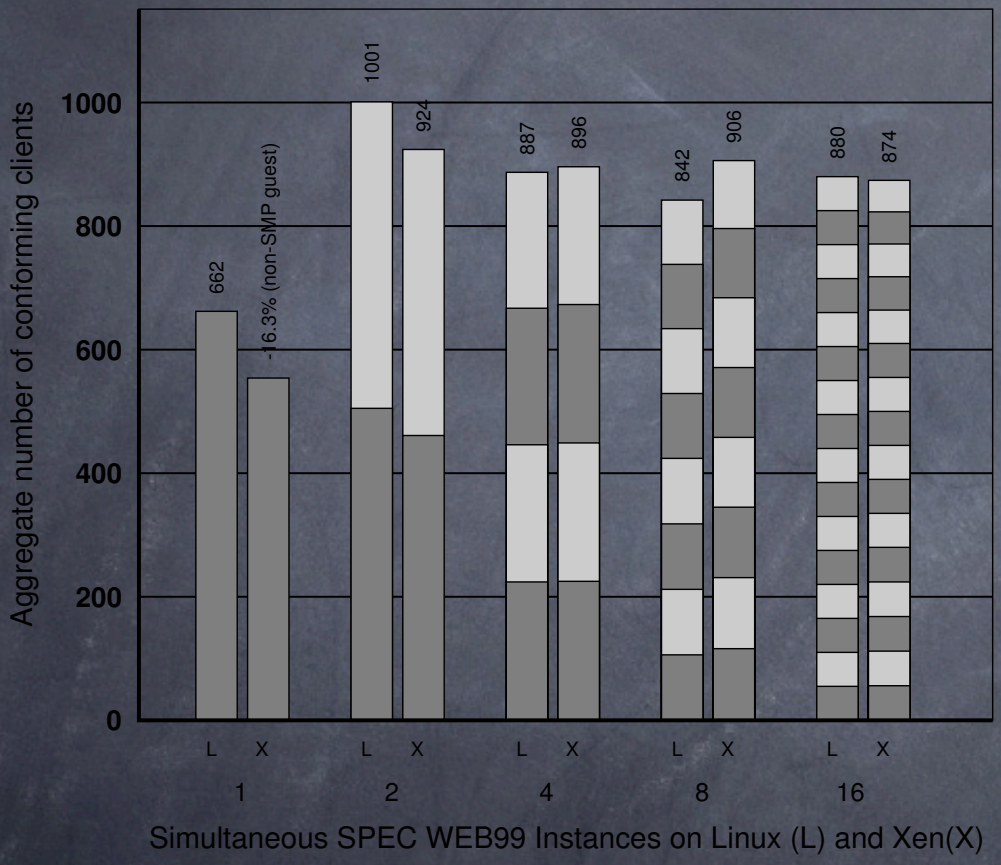


Figure 4: SPEC WEB99 for 1, 2, 4, 8 and 16 concurrent Apache servers: higher values are better.

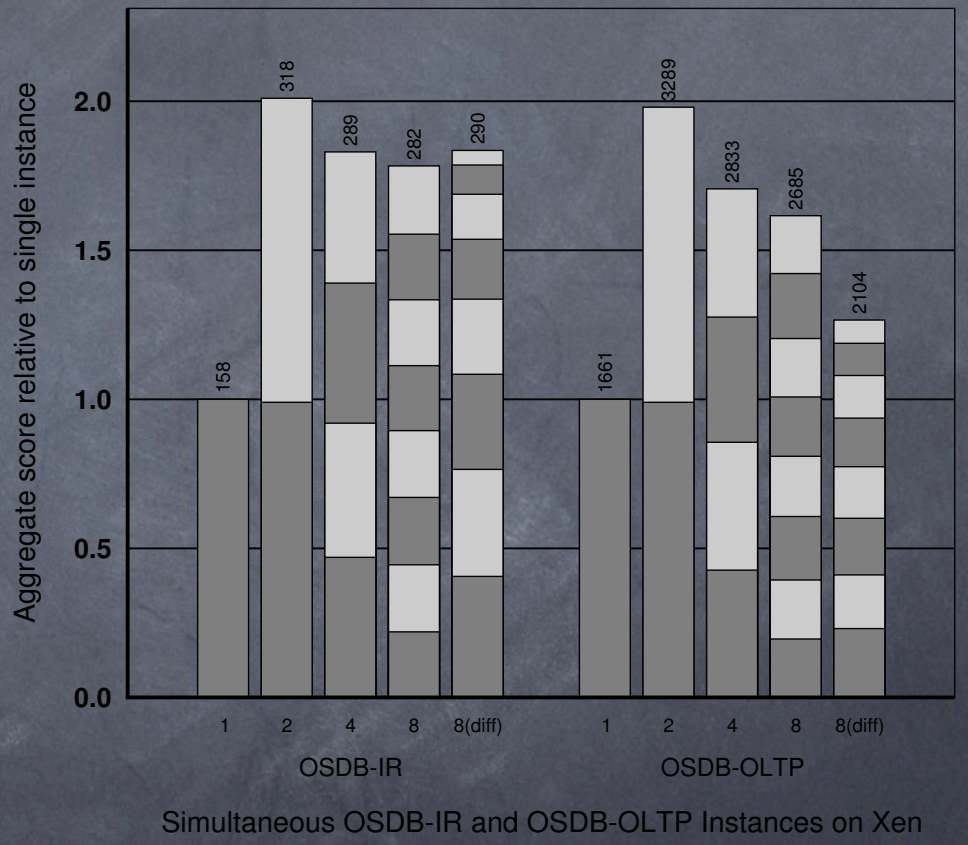


Figure 5: Performance of multiple instances of PostgreSQL running OSDB in separate Xen domains. 8(diff) bars show performance variation with different scheduler weights.

VM vs. Real Machine

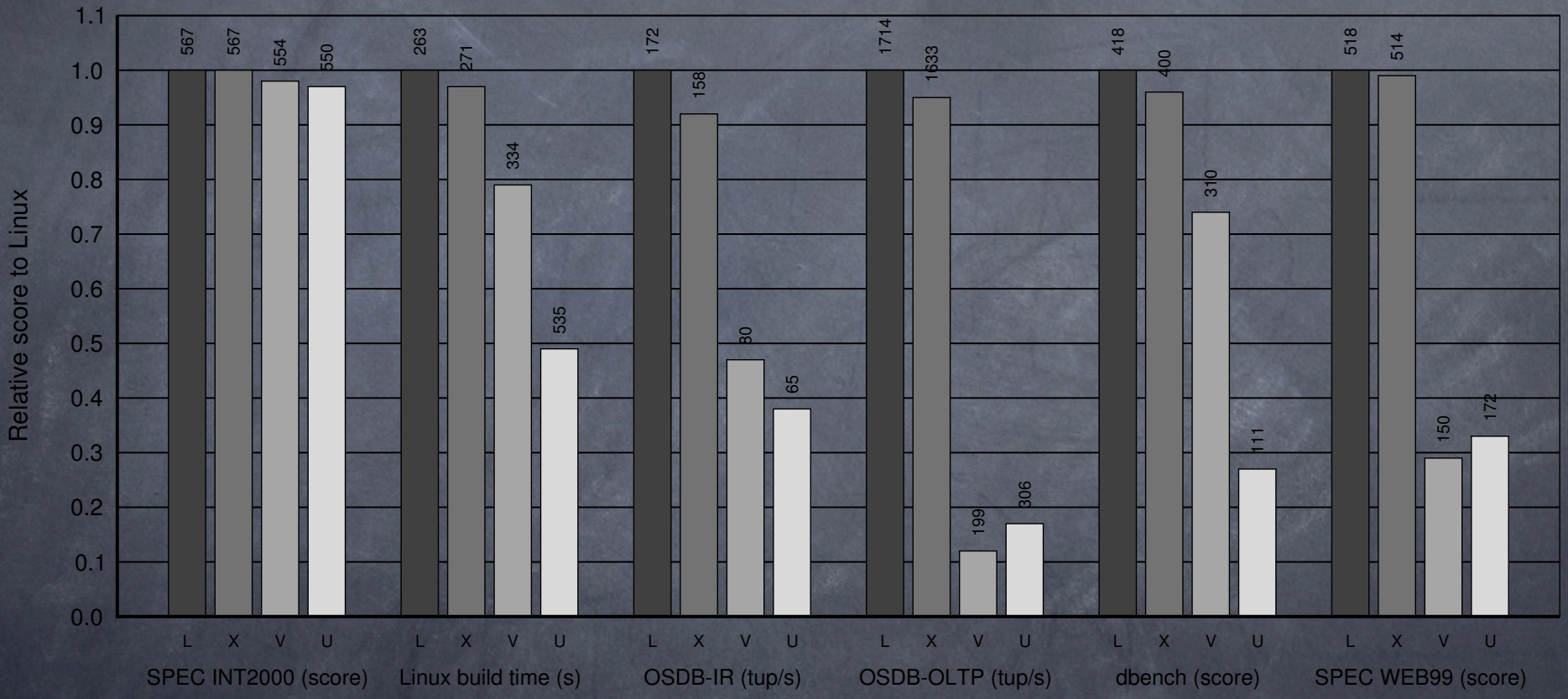


Figure 3: Relative performance of native Linux (L), XenoLinux (X), VMware workstation 3.2 (V) and User-Mode Linux (U).

Things to think about

- Xen only useful for research settings?
  - OS modification is a BIG thing
  - Xen v2.0 requires no modification of Linux 2.6 core code
- Why Xen rather than VMware for honeyfarms?
  - Is performance key for a honeypot?
  - It's free :-)
- Great expectations for VMMs: but how realistic/useful are they?
  - Mobile applications,
- VMMs are not new... they have been resurrected
  - what further directions for research?

Conclusion

- VMMs have come a long way
  - Started out as multiplexing tools back in the '60's
  - Resurrected and made-over to suit a wide range of applications
- VMMs today are
  - Fast
  - Secure
  - Light-weight
- VMMs have taken the (research?) world by storm

Thank you!

:-)

Extra slides

Why full virtualization is difficult

- Modern CPUs are not designed for virtualization
- Full virtualization requires the CPU to support direct execution
  - Privileged instructions, when run in unprivileged mode MUST trigger a trap
  - The x86 has upto 17 sensitive instructions that do not obey this rule!
  - Eg: the SMSW instruction
    - stores machine status word into general purpose register
    - first bit = PE (Protection Enable: Protected Mode when set, real mode when clear)
    - if VMOS checked PE bit when in real mode, it would incorrectly read it as Protected Mode