University of BRISTOL

Virtual Machines

Dr Dan Schien – COMSM0010

bristol.ac.uk

Admin

- AWS Educate

Goals

- Understand main concepts underlying Virtual Machines
- Be able to configure a VM on AWS

Backflash - What defines the cloud?

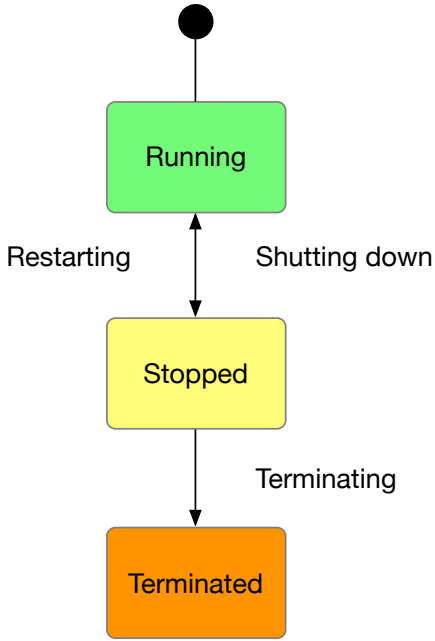
- On-Demand
- Self-Service
- Elasticity
- Metered By-the-hour (by-the-second)

Amazon Elastic Compute Cloud (EC2)

- Unmanaged services – IaaS
- Virtual machines you create, save, reuse
- Networked and connected to storage
- Security Features

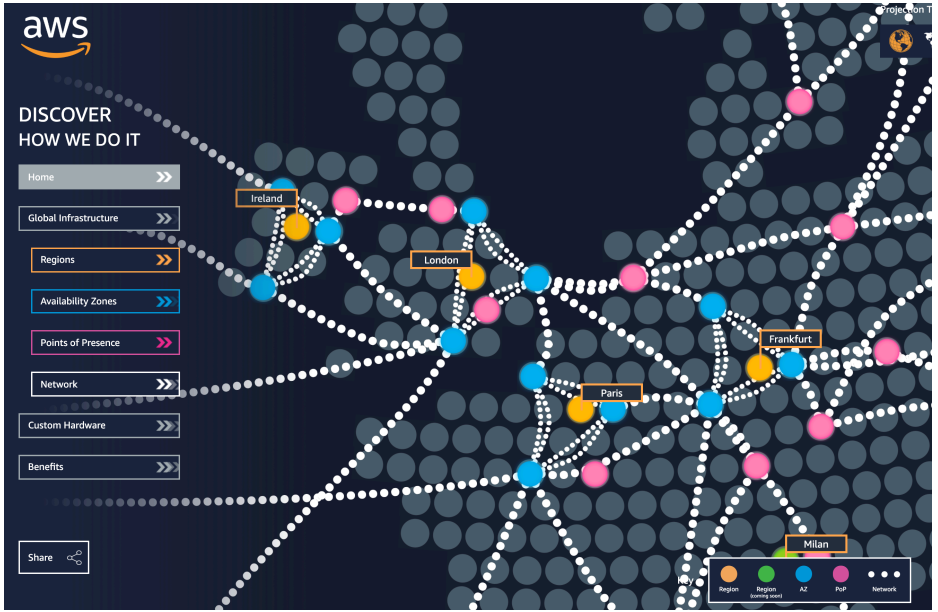
EC2 Instances

- Mostly Xeon processors
- https://aws.amazon.com/ec2/instance-types/
- From free-tier t3.micro (1GB,2CPU, 0.0114$/h) to p3dn.2
  NVIDIA Tesla V100, 96 cores, 768 GB RAM, NVMe SSD
- Different purposes
  - General Purpose (T&M), Compute optimized (C), Storage a
    Memory optimized (R), GPU (P,G)- or FPGA-enabled (F)
- Running AMI
  - Virtual Appliances to create a Virtual Machine

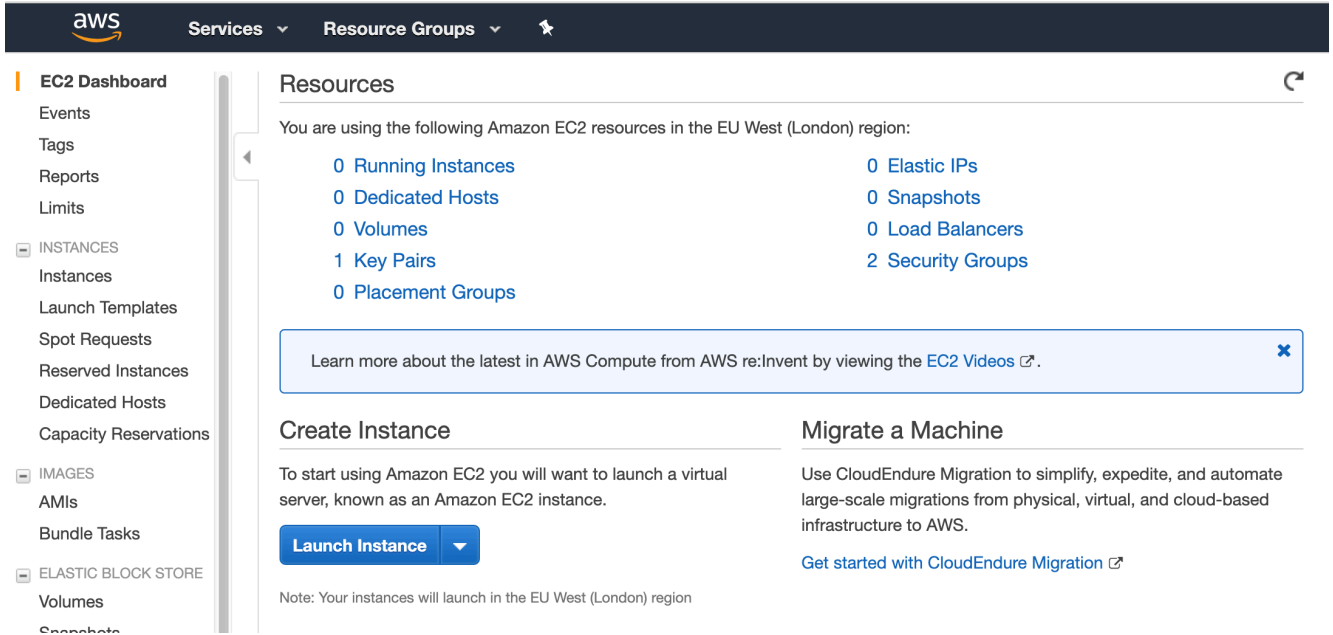


Regions and Availability Zones

- https://www.infrastructure.aws/
- Regions
  - Reach users with low latency
  - Meet location specific regulations
- Availability Zones
  - Each region has AZ for HA
  - Separate DC, with separate PSU

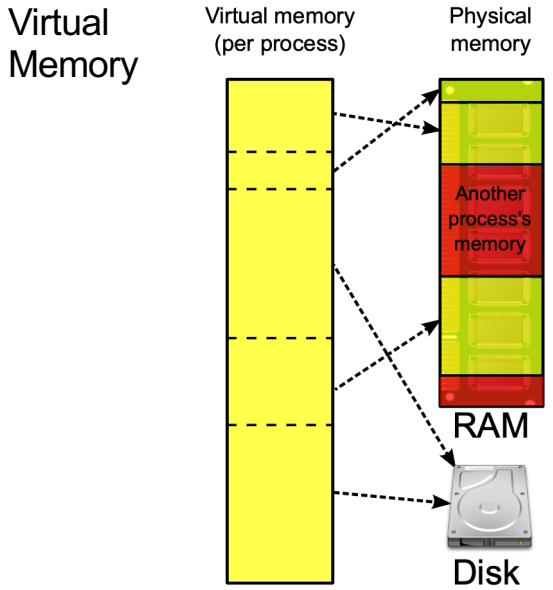


AWS UI https://aws.amazon.com



Virtualisation

Virtualisation



- Abstraction of the storage resources
- Mapping of program (virtual) memory addresses to physical addresses
- Operating system manages
- Hardware support (memory management unit)

Virtual Memory vs Virtual Machine

- Abstraction of the storage resources
- Mapping of program (virtual) memory addresses to physical addresses
- Operating system manages
- Hardware support (memory management unit)

- Abstraction of the storage/process/IO resources
- Mapping of virtual memory addresses/CPU/IO registers to physical addresses
- Operating system manages
- Hardware support (e.g. Intel VT)

Virtualization: the basics

- provision of emulations of specific computer systems, “guest” virtual machines (VMs), running on some underlying physical server hardware: the host machine.
- As far as possible, the specifics of the host server hardware should be hidden from the user: the user (remotely) interacts with the VM as if it was a stand-alone computer.
- Managed by the Hypervisor
- A VM can be paused, moved/copied to another host running the same hypervisor, and then resumed from exactly the point it was suspended at.
  - Enables server consolidation: “compress” VMs to free up servers

Virtualization: the foundations

- Not a new idea.
- Mainframe virtualization in late 1960's
- Popek & Goldberg (1974)
  - >950 citations
- Three goals:
  - Fidelity
  - Performance
  - Safety
- Analyse instruction sets
  - Privileged vs Sensitive
  - (Control-/Behavior-)Sensitive

19. Fabry, R.S. A user's view of capabilities. *ICR Quart. Rep. 15* (Nov. 1967), ICR, U. of Chicago, Sec. 1C.
20. Fabry, R.S. Preliminary description of a supervisor for a machine oriented around capabilities. *ICR Quart. Rep. 18* (Aug. 1968), ICR, U. of Chicago, Sec. 1B.
21. Fabry, R.S. List-structured addressing. Ph.D. Th., U. of Chicago, 1971.
22. Feustal, E.A. The Rice research computer—a tagged architecture. Proc. AFIPS 1972 SJCC, Vol. 40, AFIPS Press, Montvale, N.J. pp. 369–377.
23. Feustal, E.A. On the advantages of tagged architecture. *IEEE Trans. on Computers C-22*, 7 (July 1973), 644–656.
24. Graham, G.S., and Denning, P.J. Protection—principles and practice. Proc. AFIPS 1972 SJCC, Vol. 40, AFIPS Press, Montvale, N.J., pp. 417–429.
25. Hulton, D. Hardware of the System 250 for communication control. Presented at the Internat. Switching Symp., Cambridge, Mass., June 6-9, 1972, 7 pp.
26. Hamer-Hodges, K.J. Fault resistance and recovery within System 250. Presented at I.C.C. Conf., Washington, D.C., Oct. 1972, 6 pp.
27. Iliffe, J.K. *Basic machine principles*. American Elsevier, New York, 1968.
28. Iliffe, J.K., and Jodeit, J.G. A dynamic storage allocation scheme. *Comput. J. 5* (Oct. 1962), 200–209.
29. Jones, A.K. Protection structures. Ph.D. Th., Carnegie-Mellon U., 1973.
30. Lampson, B.W. On reliable and extendable operating systems. In Techniques in Software Engineering. NATO Science Committee Workshop Material, Vol. II, Sept. 1969.
31. Lampson, B.W. Dynamic protection structures. Proc. AFIPS 1969 FJCC, Vol. 35, AFIPS Press, Montvale, N.J., pp. 27–38.
32. Lampson, B.W. Protection. Proc. 5th Ann. Princeton Conf., Princeton U., Mar. 1971, pp. 437–443.
33. LeClerc, J.Y. Memory structures for interactive computers. Project GENIE document No. 40.10.110, U. of California, Berkeley, 1966.
34. Needham, R.M. Protection systems and protection implementations. Proc. AFIPS 1972 FJCC, Vol. 41, AFIPS Press, Montvale, N.J., pp. 571–578.
35. Organick, E.I. *Computer System Organization—the B5700 B6700 Series*. Academic Press, New York, 1973.
36. Organick, E.I. *The Multics System: An Examination of Its Structure*. MIT Press, Cambridge, Mass., 1972.
37. Saltzer, J.H. Traffic control in a multiplexed computer system. MAC-TR-30, Proj. MAC, MIT, Cambridge, Mass., 1966.
38. Schroeder, M.D. Performance of the GE-645 associative memory while Multics is in operation. Proc. Workshop on System Performance Evaluation, Cambridge, Mass., 1971, pp. 227–245.
39. Schroeder, M.D. Cooperation of mutually suspicious subsystems in a computer utility. Ph.D. Th., MIT, 1972.
40. Sevcik, K.C., et al. Project SUE as a learning experience. Proc. AFIPS 1972 FJCC, Vol. 41, AFIPS Press, Montvale, N. J., pp. 331–338.
41. Shepherd, J. Principal design features of the multi-computer. (The Chicago Magic Number Computer). *ICR Quart. Rep. 19* (Nov. 1968), ICR, U. of Chicago, Sec. 1-C.
42. Sturgis, H.E. A postmortem of a time sharing system. Ph.D. Th., U. of California, Berkeley, 1973.
43. Wilkes, M.V. *Time Sharing Computer Systems*. 2nd ed., American Elsevier, New York, 1972.
44. Wilner, W.T. Design of the Burroughs B1700. Proc. AFIPS 1972 FJCC, Vol. 41, AFIPS Press, Montvale, N.J., pp. 489–497.
45. Wilner, W.T. Burroughs B1700 memory utilization. Proc. AFIPS 1972 FJCC, Vol. 41, AFIPS Press, Montvale, N.J., pp. 579–586.
46. Wulf, W.A., et al. HYDRA: The kernel of a multiprocessor operating system. Carnegie Mellon U., Comput. Sci. Dep. rep., June 1973.
47. Yngve, V.H. The Chicago Magic Number Computer. *ICR Quart. Rep. 18* (Nov. 1968), ICR, U. of Chicago, Sec. 1-B.

412

Communications of

July 1974 Volume 17

Formal Requirements for Virtualizable Third Generation Architectures

Gerald J. Popek
University of California, Los Angeles
and
Robert P. Goldberg
Honeywell Information Systems and
Harvard University

Virtual machine systems have been implemented on a limited number of third generation computer systems, e.g. CP-67 on the IBM 360/67. From previous empirical studies, it is known that certain third generation computer systems, e.g. the DEC PDP-10, cannot support a virtual machine system. In this paper, model of a third-generation-like computer system is developed. Formal techniques are used to derive precise sufficient conditions to test whether such an architecture can support virtual machines.

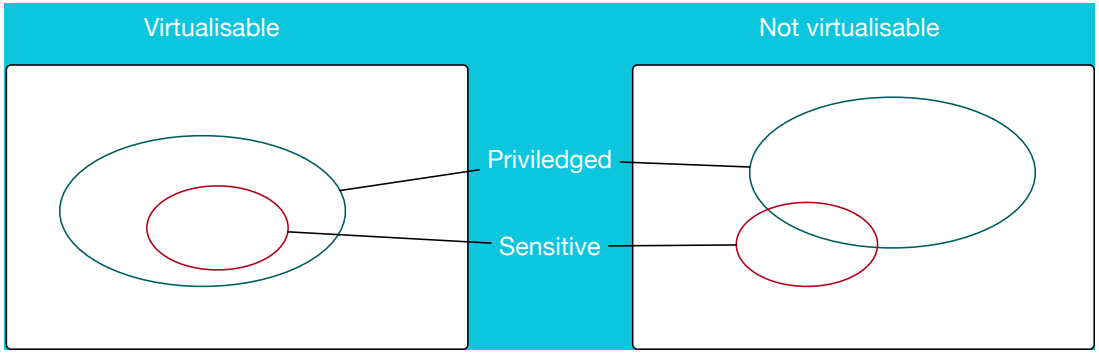
Key Words and Phrases: operating system, third generation architecture, sensitive instruction, formal requirements, abstract model, proof, virtual machine, virtual memory, hypervisor, virtual machine monitor
CR Categories: 4.32, 4.35, 5.21, 5.22

This is a revised version of a paper presented at the Fourth ACM Symposium on Operating Systems Principles, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, October 15–17, 1973.

This research was supported in part by the U.S. Atomic Energy Commission, Contract No. AT(11-1) Gen 10, Project 14 and in part by the Electronic Systems Division, U.S. Air Force, Hanscom Field, Bedford, Massachusetts under Contract Number F19628-70-0217.

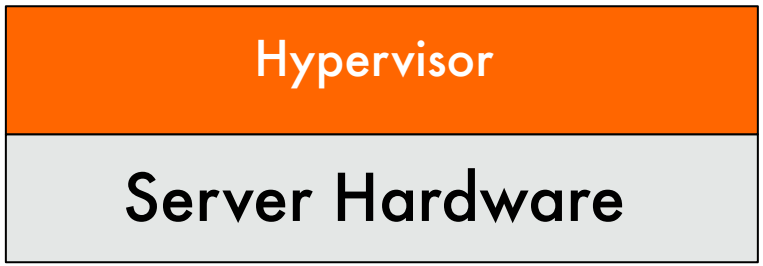
Authors' addresses: Gerald J. Popek, Computer Science Department, University of California, Los Angeles CA 90024; Robert P. Goldberg, Honeywell Information Systems, Waltham, MA 02154.

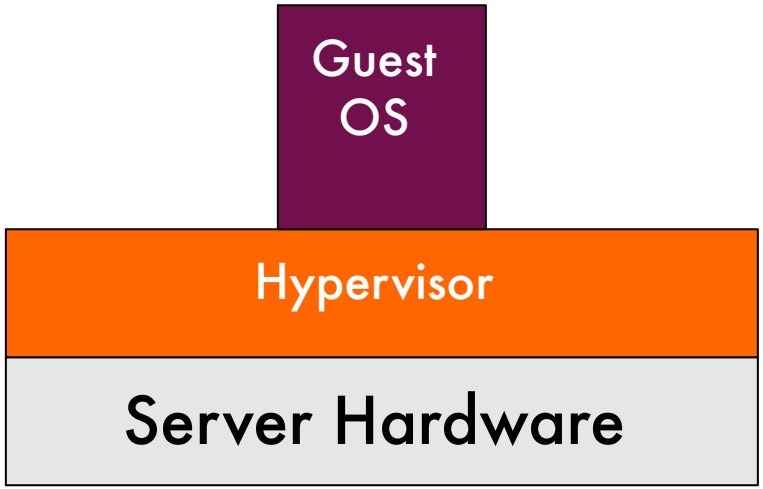
Privileged vs Sensitive Instructions

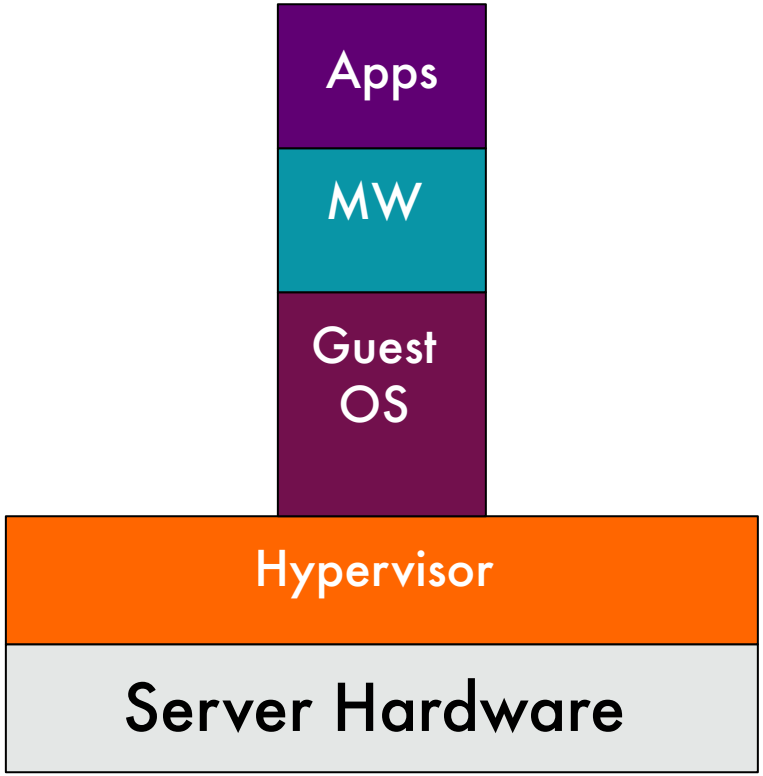


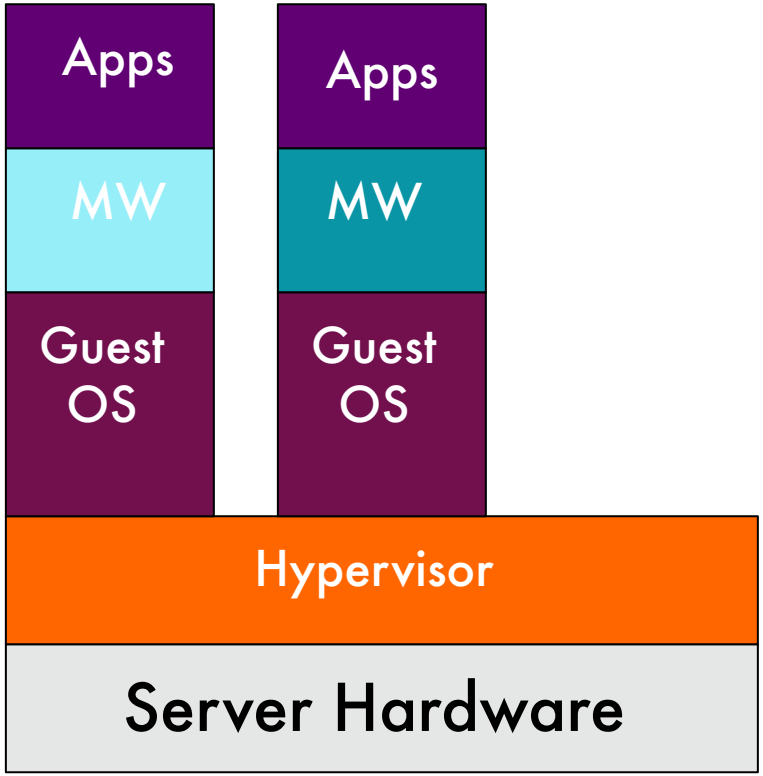
Server Hardware

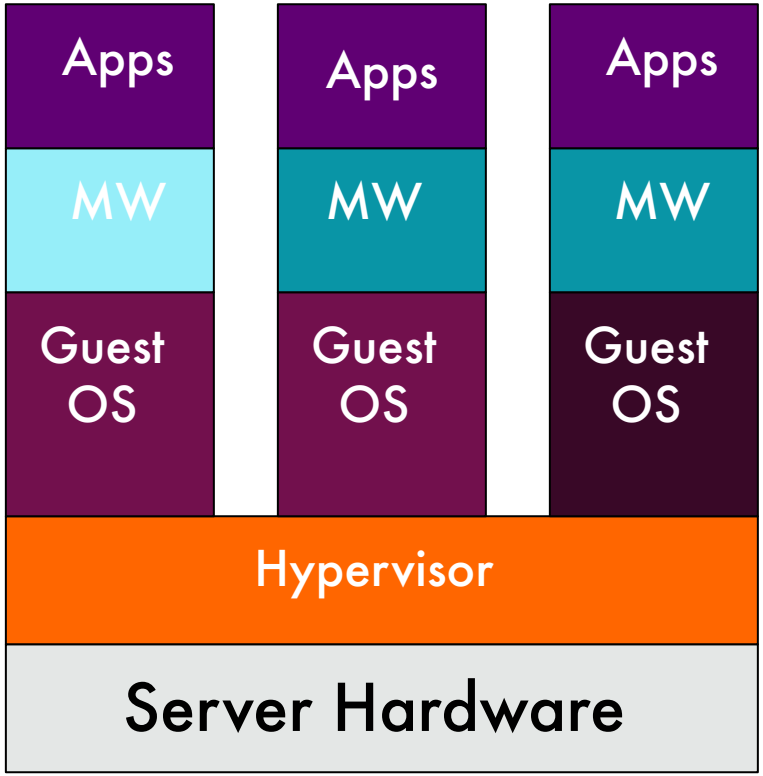
bristol.ac.uk

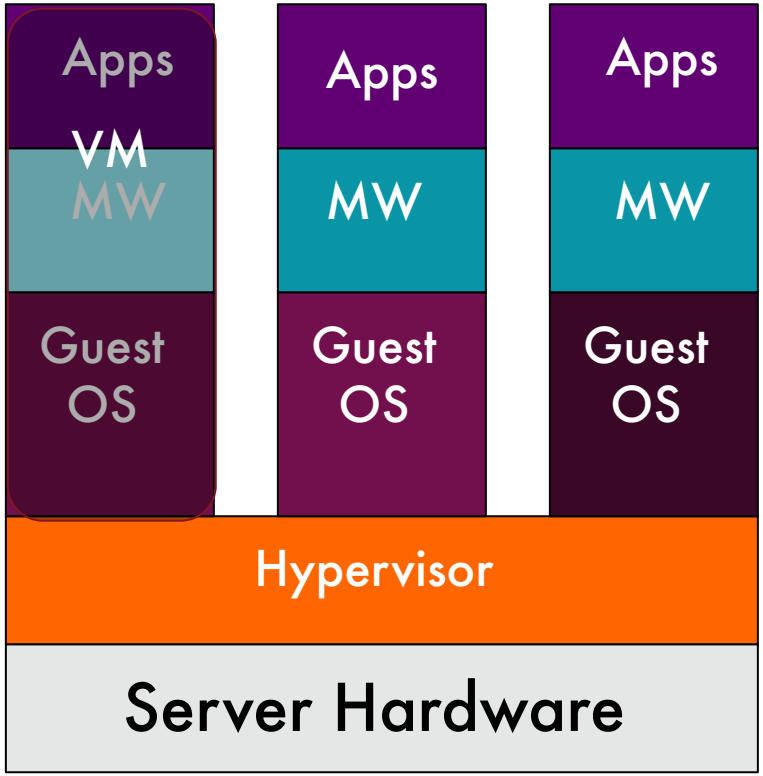

Hypervisor
Server Hardware


Guest
OS
Hypervisor
Server Hardware


Apps
MW
Guest OS
Hypervisor
Server Hardware

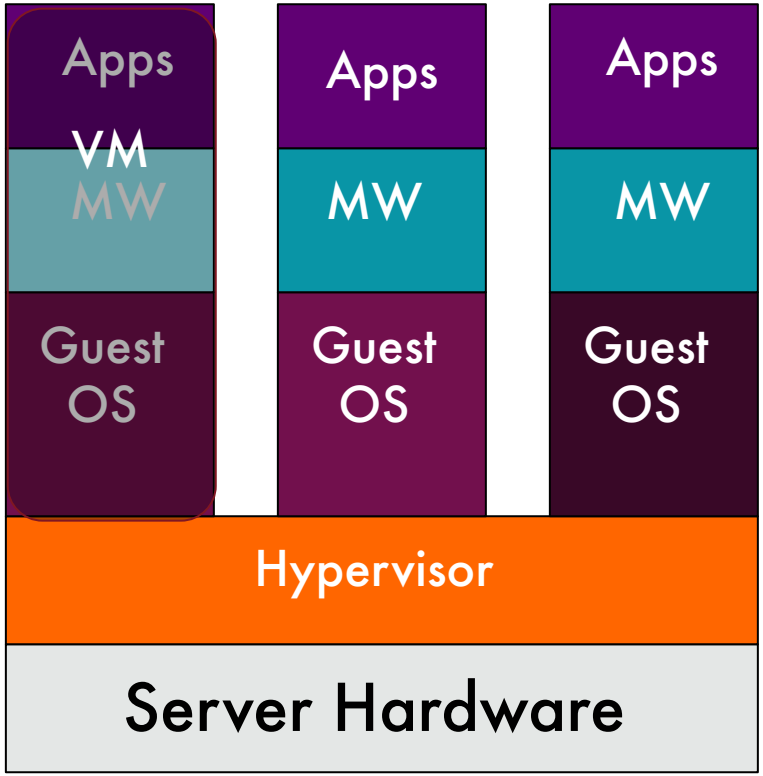

Apps
MW
Guest OS
Apps
MW
Guest OS
Hypervisor
Server Hardware


Apps
Apps
Apps
MW
MW
MW
Guest OS
Guest OS
Guest OS
Hypervisor
Server Hardware

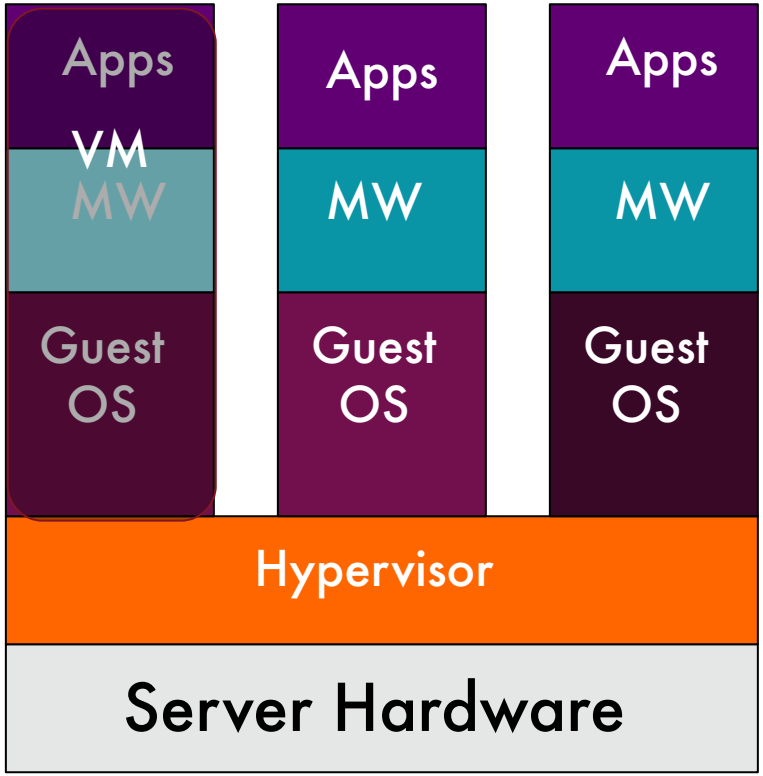


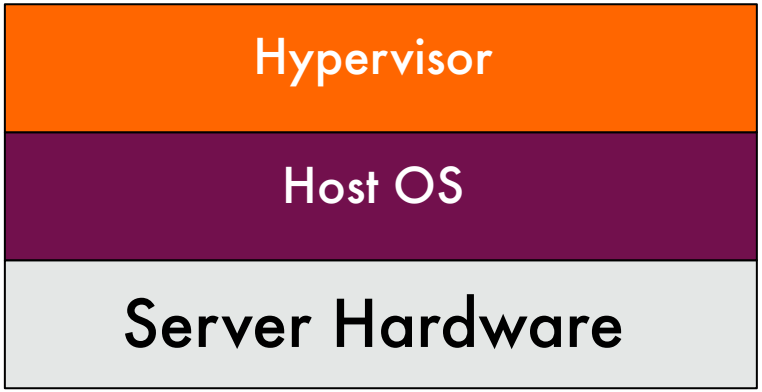
bristol.ac.uk

Type 1 "bare metal"

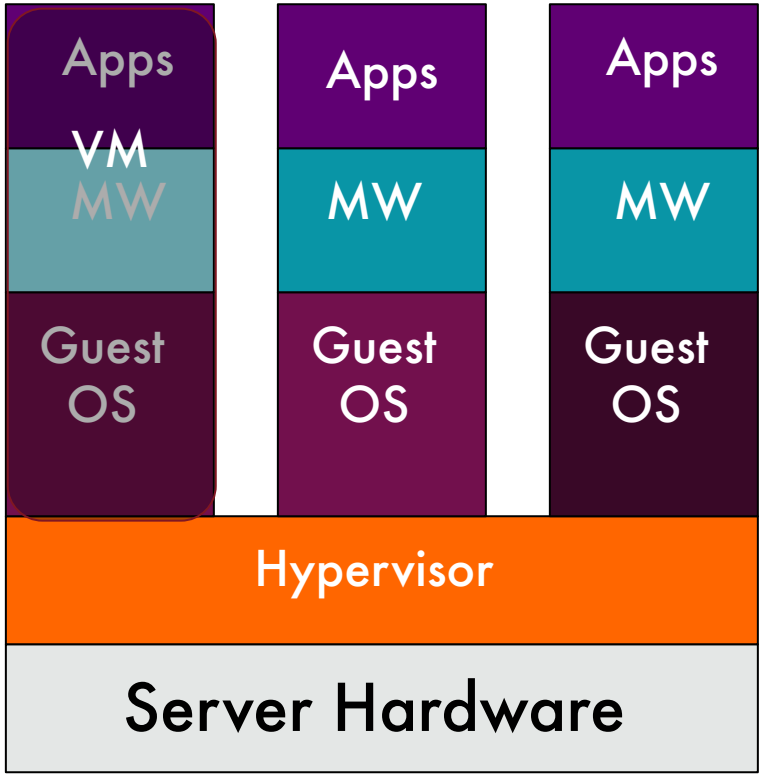


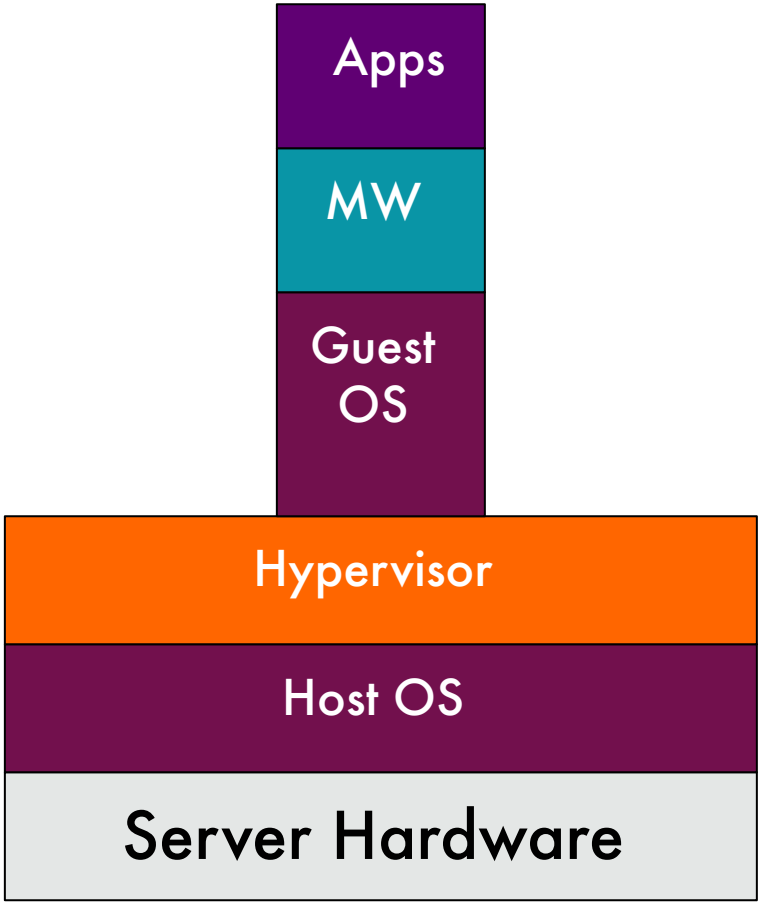
Type 1 "bare metal"



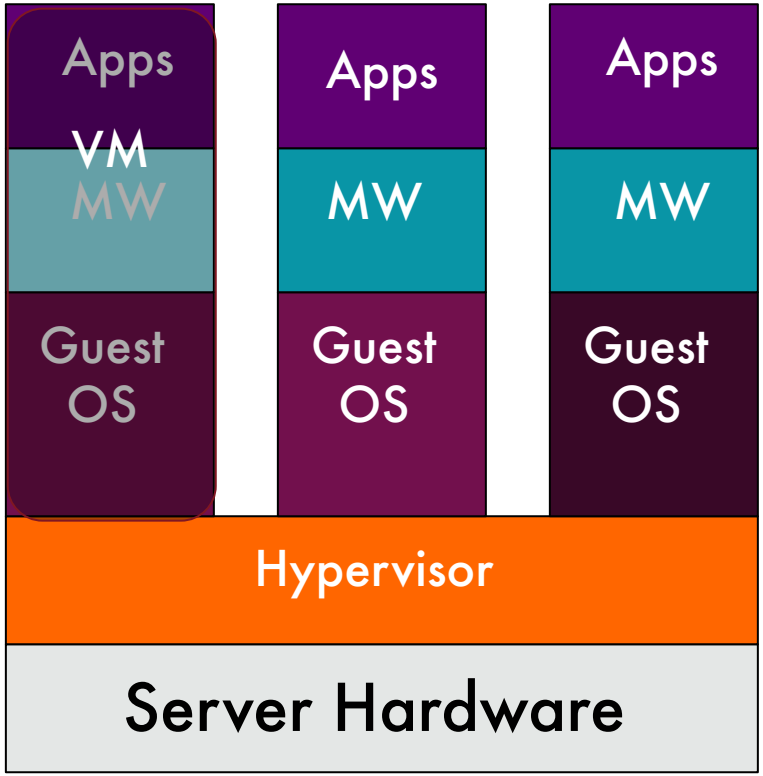


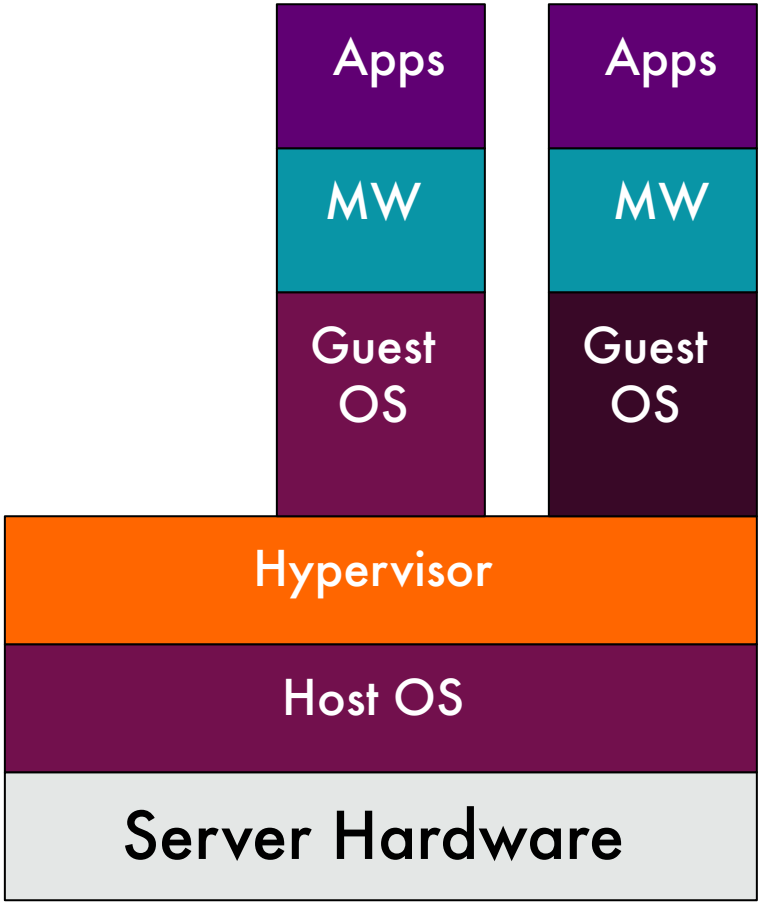
Type 1 "bare metal"



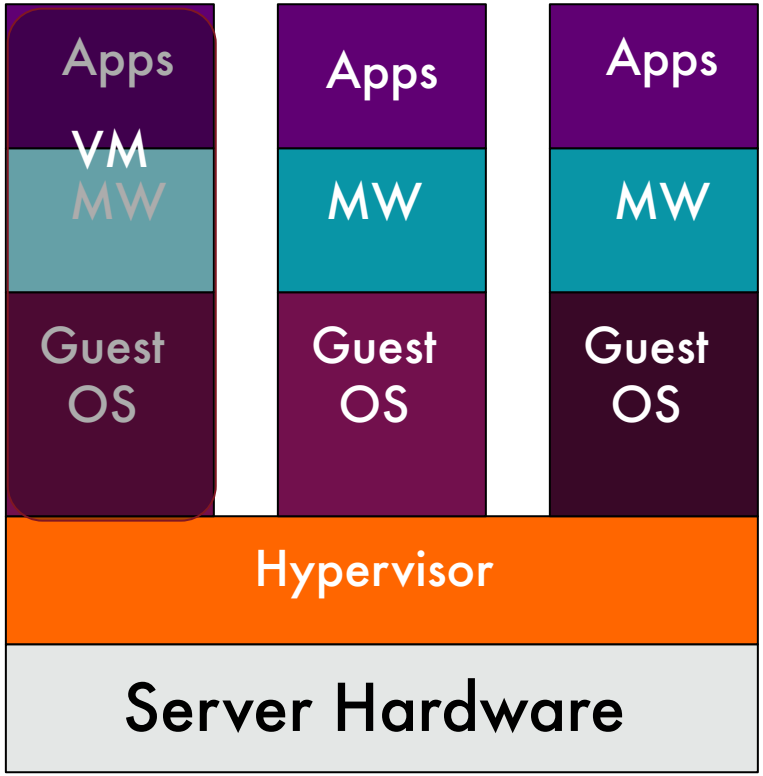


Type 1 "bare metal"

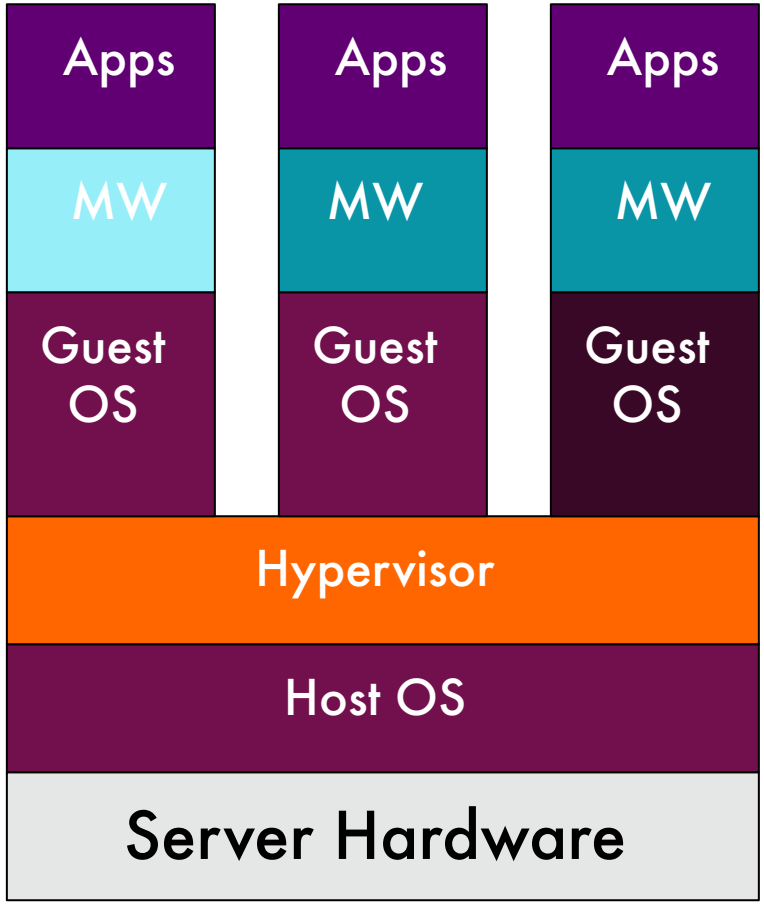




Type 1 "bare metal"



Type 1 "bare metal"



Type 2 "hosted"

More terminology…

- Full Virtualization is a complete or almost-complete simulation of the underlying guest-machine hardware: virtualized guest OS runs as if it was on a bare machine.
- Paravirtualization: the guest OS is edited (and recompiled) to make system calls into the hypervisor API, to execute safe rewrites of sensitive instructions: the hypervisor doesn't simulate hardware.

EC2 Paravirtual (PV) or hardware virtual machine (HVM)

- HVM
  - virtualized set of hardware
  - Can use operating systems without modification
  - Intel virtualisation technology
- PV
  - Requires OS to be prepared
  - Doesn't support GPU Instances

Xen

P. Barham, *et al.* (2003). Xen and the Art of Virtualization. *ACM SIGOPS Operating Systems Review*, 37(5):164-177.

Xen and the Art of Virtualization

Paul Barham*, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris,
Alex Ho, Rolf Neugebauer†, Ian Pratt, Andrew Warfield

University of Cambridge Computer Laboratory
15 JJ Thomson Avenue, Cambridge, UK, CB3 0FD
{firstname.lastname}@cl.cam.ac.uk

ABSTRACT

Numerous systems have been designed which use virtualization to subdivide the ample resources of a modern computer. Some require specialized hardware, or cannot support commodity operating systems. Some target 100% binary compatibility at the expense of performance. Others sacrifice security or functionality for speed. Few offer resource isolation or performance guarantees; most provide only best-effort provisioning, risking denial of service.

This paper presents Xen, an x86 virtual machine monitor which allows multiple commodity operating systems to share conventional hardware in a safe and resource managed fashion, but without sacrificing either performance or functionality. This is achieved by providing an idealized virtual machine abstraction to which operating systems such as Linux, BSD and Windows XP, can be *ported* with minimal effort.

Our design is targeted at hosting up to 100 virtual machine instances simultaneously on a modern server. The virtualization approach taken by Xen is extremely efficient: we allow operating systems such as Linux and Windows XP to be hosted simultaneously for a negligible performance overhead — at most a few percent

1. INTRODUCTION

Modern computers are sufficiently powerful to use virtualization to present the illusion of many smaller *virtual machines* (VMs), each running a separate operating system instance. This has led to a resurgence of interest in VM technology. In this paper we present Xen, a high performance resource-managed virtual machine monitor (VMM) which enables applications such as server consolidation [42, 8], co-located hosting facilities [14], distributed web services [43], secure computing platforms [12, 16] and application mobility [26, 37].

Successful partitioning of a machine to support the concurrent execution of multiple operating systems poses several challenges. Firstly, virtual machines must be isolated from one another: it is not acceptable for the execution of one to adversely affect the performance of another. This is particularly true when virtual machines are owned by mutually untrusting users. Secondly, it is necessary to support a variety of different operating systems to accommodate the heterogeneity of popular applications. Thirdly, the performance overhead introduced by virtualization should be small.

Xen hosts commodity operating systems, albeit with some source

Xen

Free, open-source, hypervisor developed at University of Cambridge.

Released in 2003, open-source Linux Foundation project since 2013.

Modes:

- Paravirtualization: guest OS recompiled with modifications.
- Hardware-assisted virtualization: Intel x86 and ARM extensions.

…Intel architecture is…

- Widespread
- NOT easily virtualizable (17 instructions that violate Popek & Goldberg)

Different approaches to addressing this:

- VMWare do binary-rewrite
- Xen does paravirtualization.

Areas of Virtualisation

- CPU Virtualisation
  - guest to have exclusive use of a CPU for a period of time,
  - CPU state of the first guest is saved, and the state of the next guest is loaded before the control is passed onto it.
- Memory Virtualization
  - Additional layer of indirection to virtual memory
- I/O Virtualization
  - Hypervisor implements a device model to provide abstractions of the hardware

Xen in action

(Based on Fig 1.5 of Chisnall, 2007).

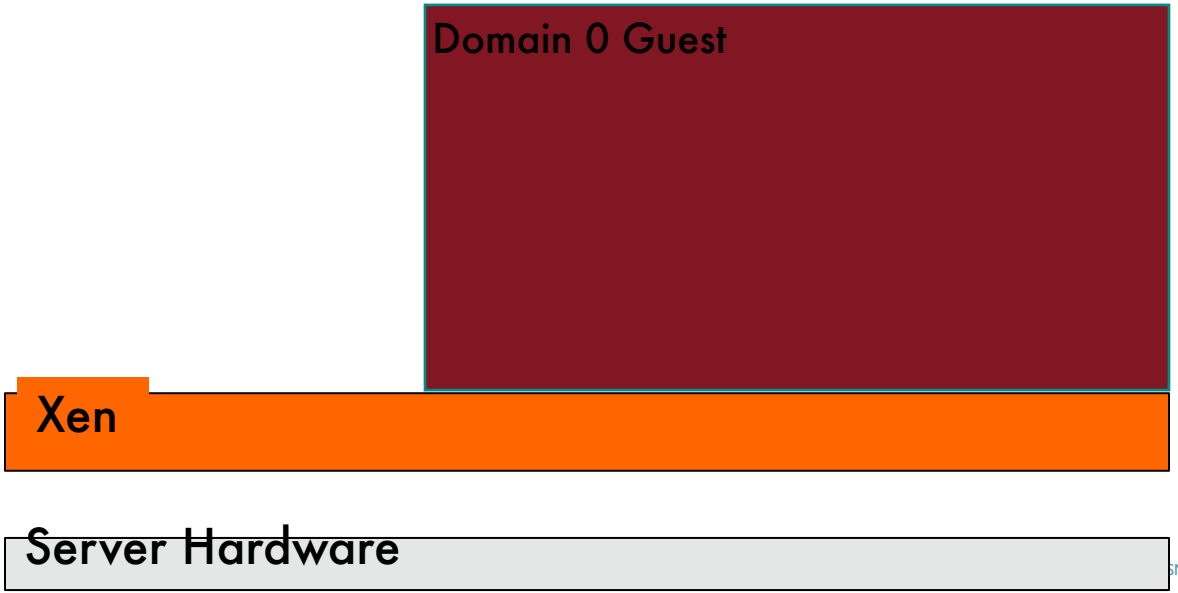
Server Hardware

Xen in action

(Based on Fig 1.5 of Chisnall, 2007).

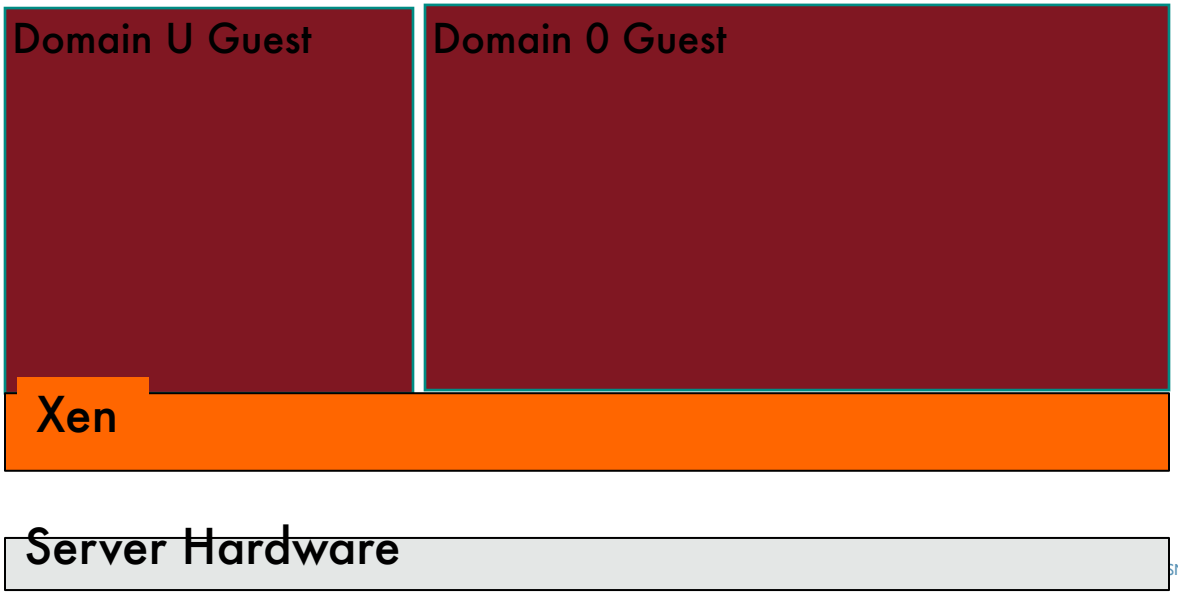
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



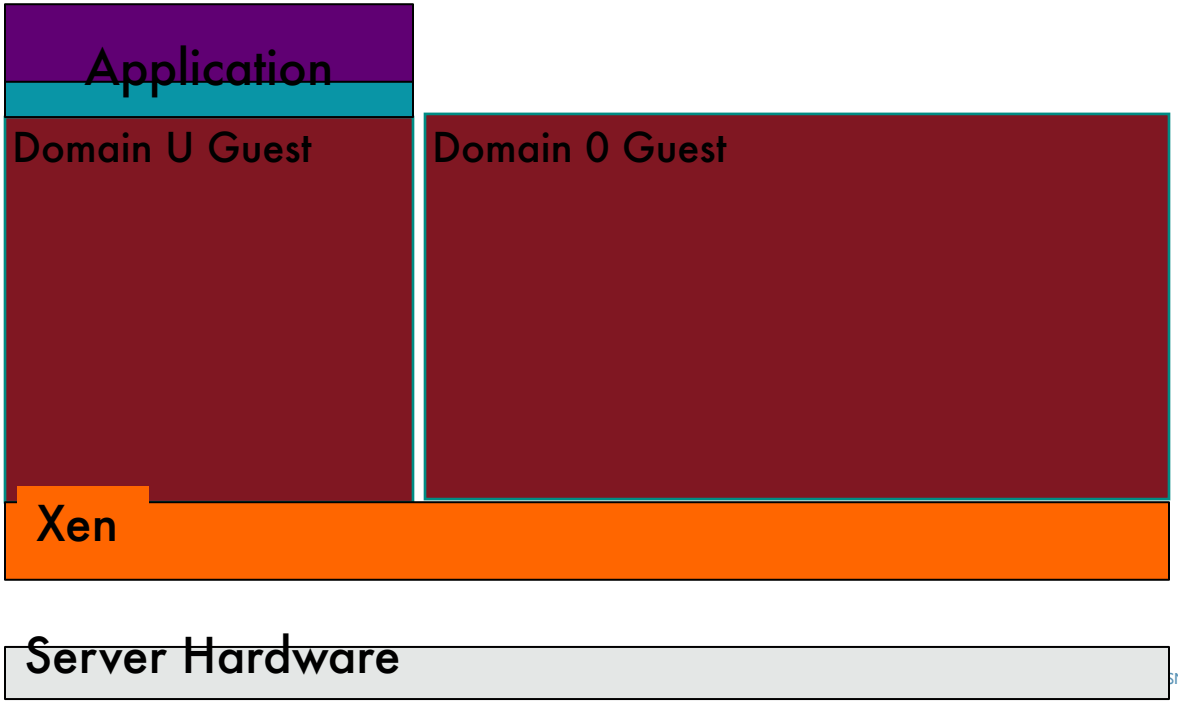
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



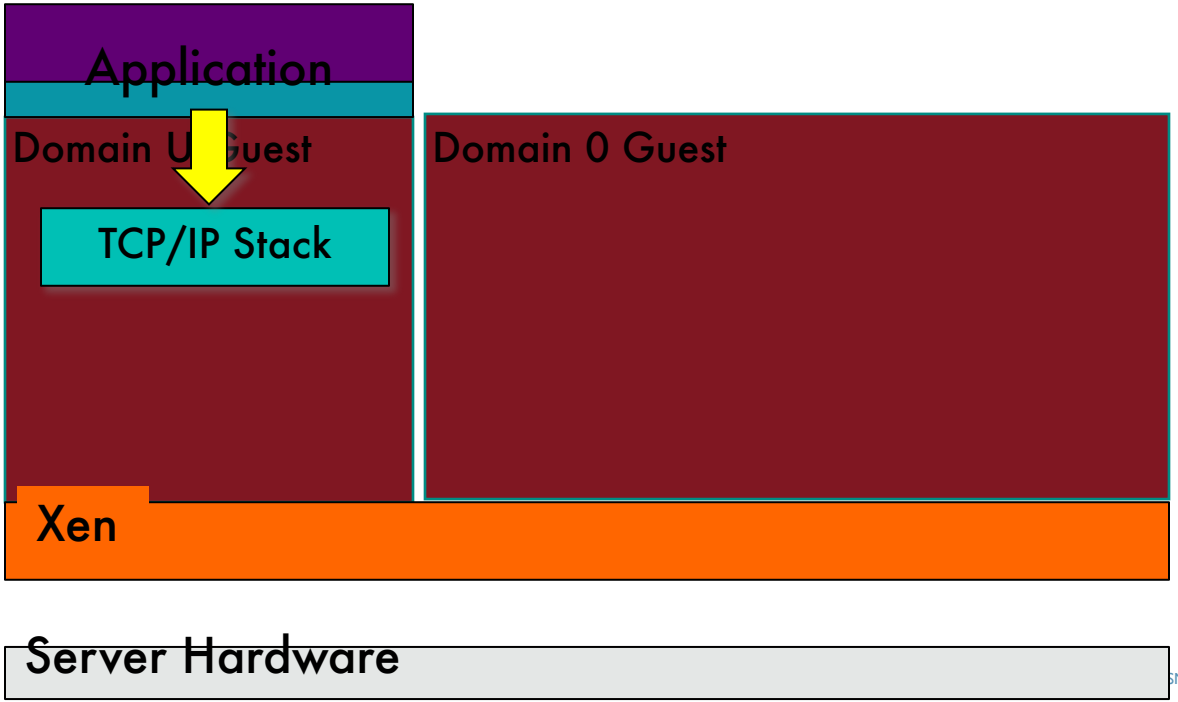
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



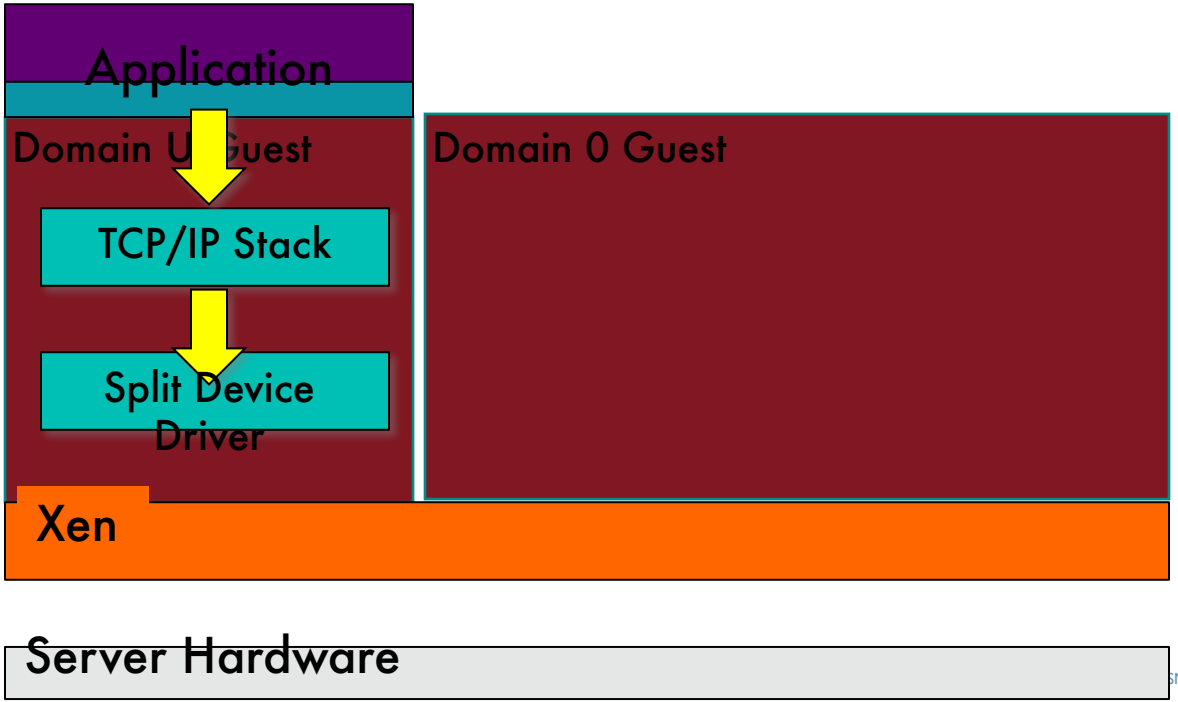
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



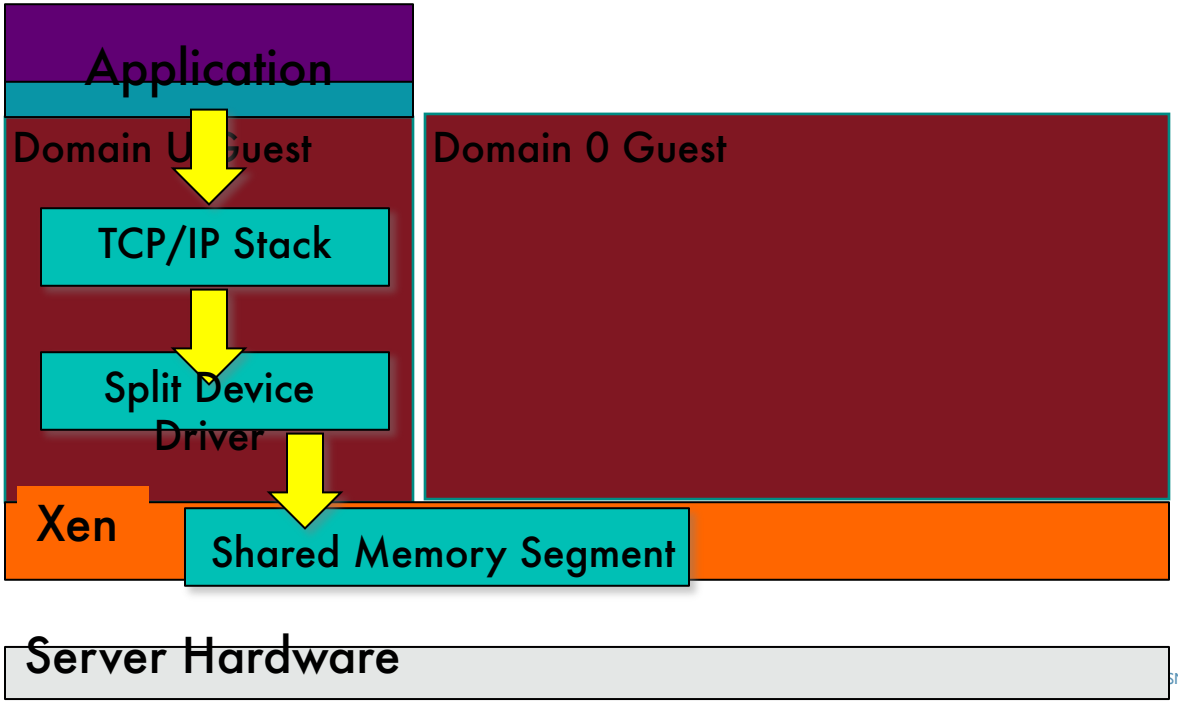
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



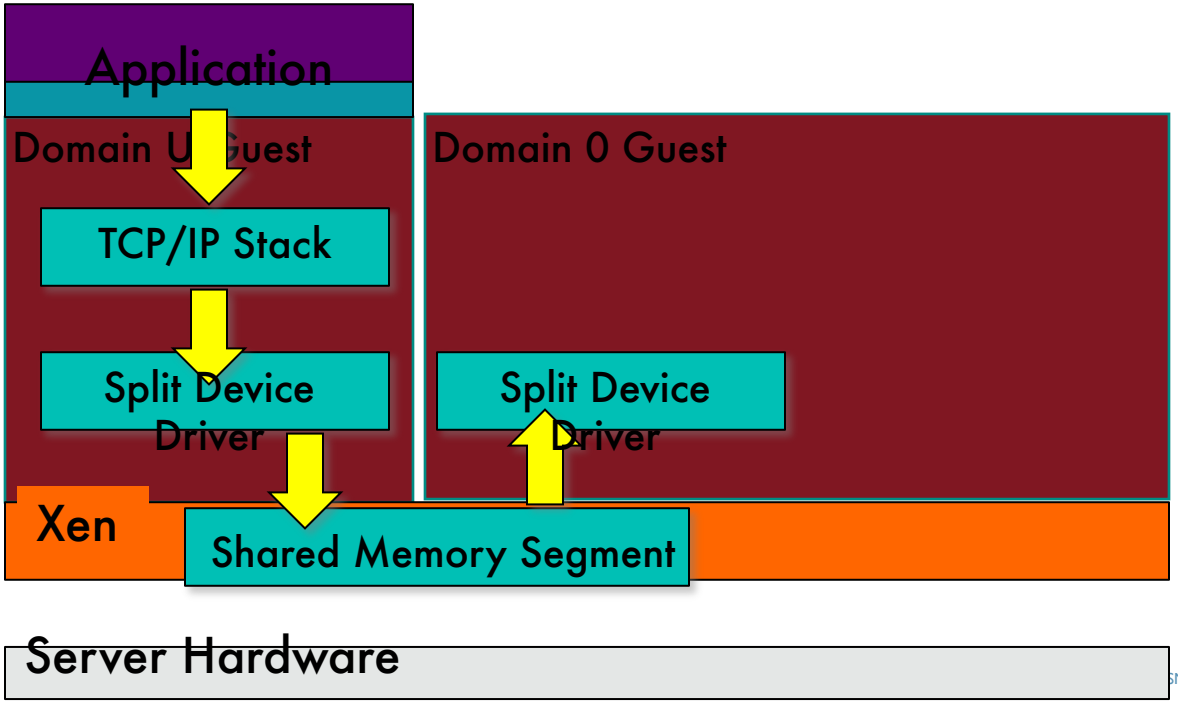
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



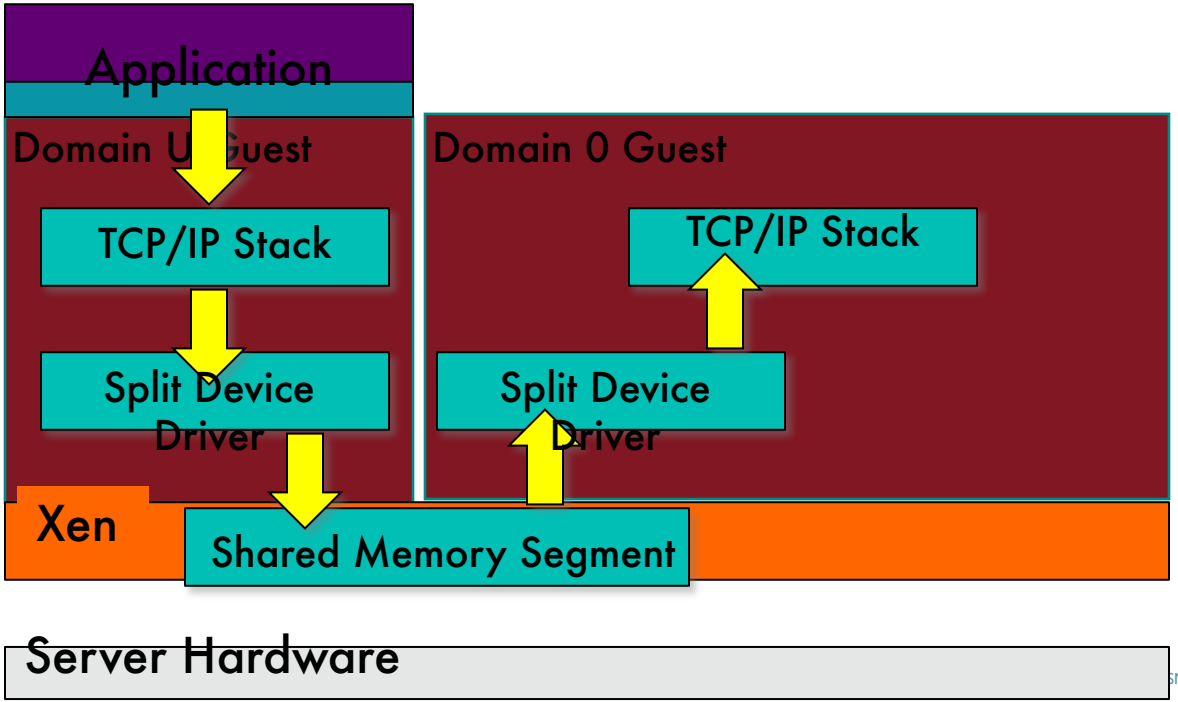
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



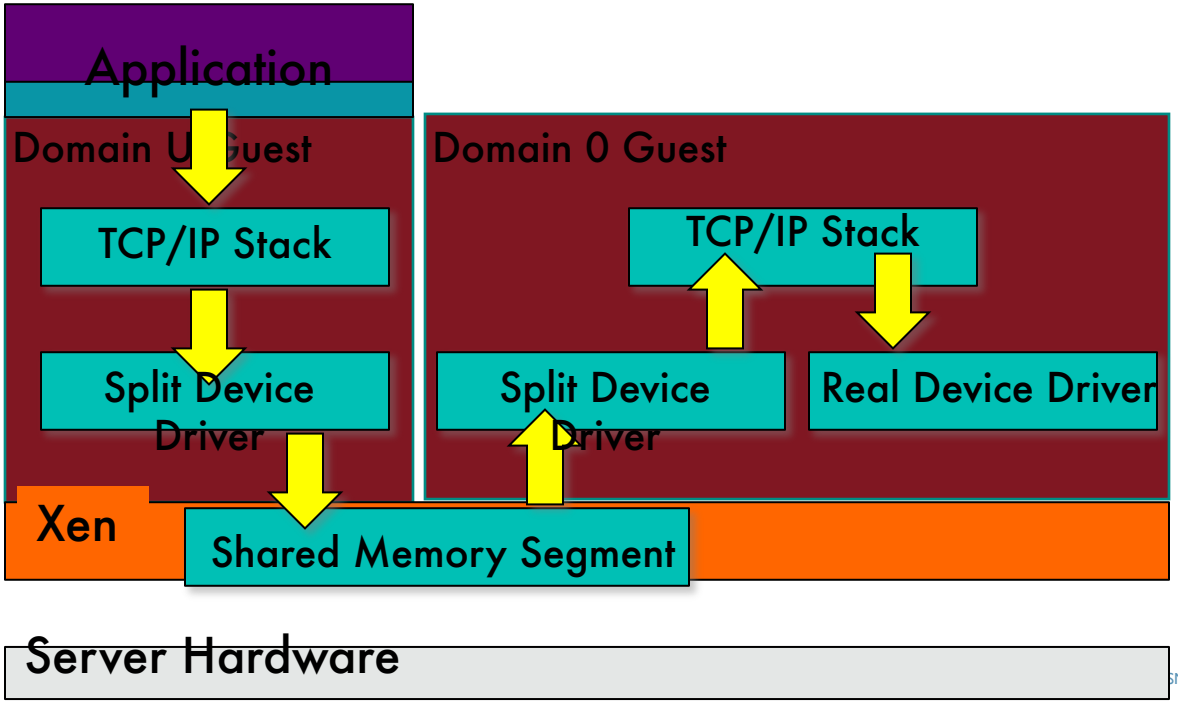
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



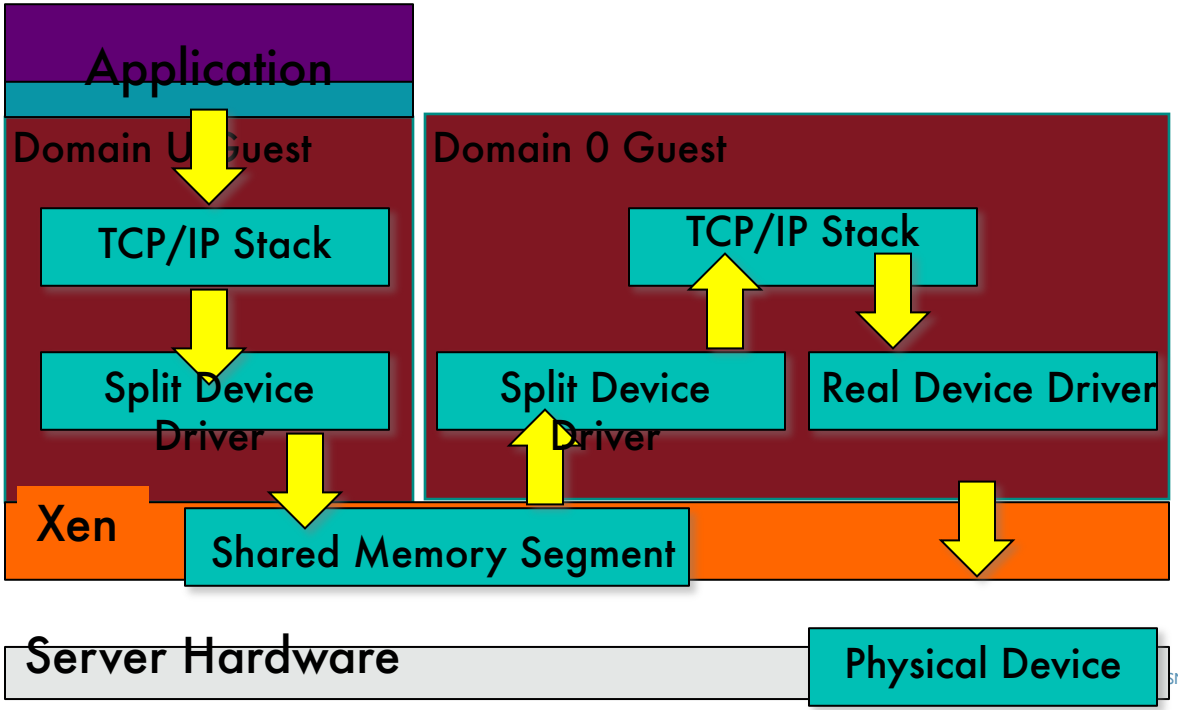
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



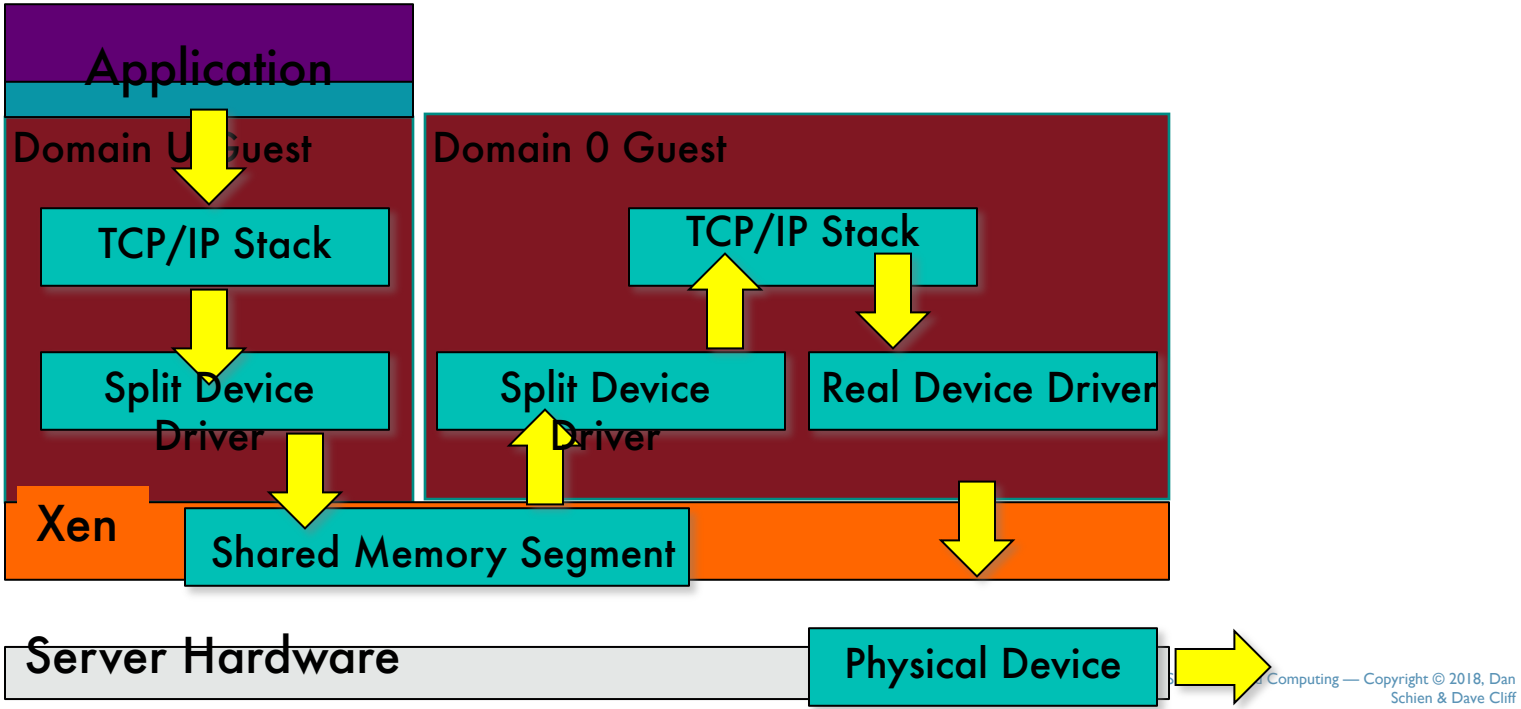
Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



Xen in action

(Based on Fig 1.5 of Chisnall, 2007).



KVM

- Converts Linux into a type-1 hypervisor
- memory manager, process scheduler, input/output (I/O) stack from linux
- VM is implemented as a regular Linux process
- KVM requires CPU virtualization extensions (Intel VT or AMD-V)

Nitro Hypervisor

- Based on KVM
- Since 2017 all new instances (C5, I3, R5)
- Hypervisor mainly provides CPU and memory isolation for EC2 instances.
- Nitro cards for VPC networking and EBS storage
  - Can handle NVMe SSD (non-volatile memory express) for instance and net storage, transparent encryption
  - Nitro Hypervisor not involved in tasks for networking and storage
  - In OS Elastic Network Adapter driver
  - Security groups implemented in the NIC
- See https://aws.amazon.com/ec2/faqs/ for differences to Xen

Back to EC2

EC2 Config Options

- EBS or Attached
  - Network block storage
  - Snapshots to S3
  - Replicated within AZ
  - Boot and data volumes

- ENA (Elastic Network Adapter)
  - Enhanced Networking (faster)
  - Replaced virtualized network interface drivers

Cloud-init

- cloudinit.readthedocs.io
- Vendor neutral system to initialize cloud VMs
- Run by provided metadata from the cloud and initialize the system accordingly
- Setting up the network and storage devices to configuring SSH access key
- User Data
  - Linux script to initialize additional services (install packages, configure)

Further Reading

- AMI
  - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/virtualization_types.html
- KVM
  - https://mkdev.me/en/posts/virtualization-basics-and-an-introduction-to-kvm

Review

- VMs provide access to infrastructure in the Cloud, on demand, self-serve, metered
- Hypervisor isolates guests from hardware
- Different Virtualisation approaches
- Xen, KVM