Conclusion of OS

Operating systems are still very much a current topic

Conclusion of OS

Operating systems are still very much a current topic

We have only scratched the surface — there are many other things a real OS would have to implement

Conclusion of OS

Operating systems are still very much a current topic

We have only scratched the surface — there are many other things a real OS would have to implement

There are still lots of hard problems to solve, such as scheduling

Conclusion of OS

Operating systems are still very much a current topic

We have only scratched the surface — there are many other things a real OS would have to implement

There are still lots of hard problems to solve, such as scheduling

And, as hardware changes, OSs must change, too

Conclusion of OS

Operating systems are still very much a current topic

We have only scratched the surface — there are many other things a real OS would have to implement

There are still lots of hard problems to solve, such as scheduling

And, as hardware changes, OSs must change, too

OSs for low-power devices (in particular mobile phones) are a huge source of research

Conclusion of OS

At the other end of the scale, people are still developing OSs on large supercomputers

Conclusion of OS

At the other end of the scale, people are still developing OSs on large supercomputers

OS *virtualisation* is important in the era of cloud computing

Conclusion of OS

At the other end of the scale, people are still developing OSs on large supercomputers

OS *virtualisation* is important in the era of cloud computing

Where several users (customers) are sharing the same hardware, but each has their own, private OS running their own, private applications

Conclusion of OS

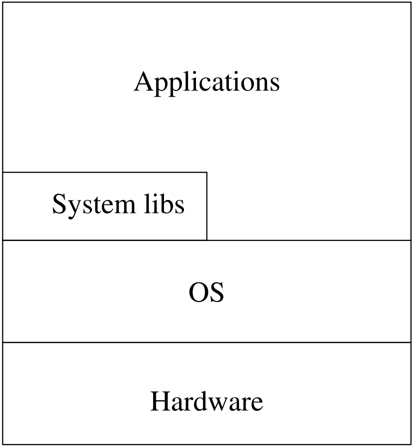
At the other end of the scale, people are still developing OSs on large supercomputers

OS *virtualisation* is important in the era of cloud computing

Where several users (customers) are sharing the same hardware, but each has their own, private OS running their own, private applications

Originally, OSs were the software closest to the hardware: with OS virtualisation, this is no longer necessarily true

Conclusion of OS



Traditional OS

Conclusion of OS

For example, sometimes an application only runs on a specific OS

Conclusion of OS

For example, sometimes an application only runs on a specific OS

But repeatedly rebooting a machine with a different OS every time a user wants to run a different application is not a good approach

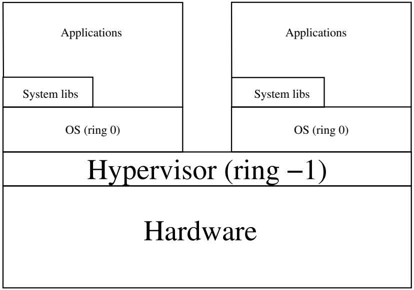
Conclusion of OS

For example, sometimes an application only runs on a specific OS

But repeatedly rebooting a machine with a different OS every time a user wants to run a different application is not a good approach

So the solution is to have multiple, simultaneous OSs on a single machine

Conclusion of OS



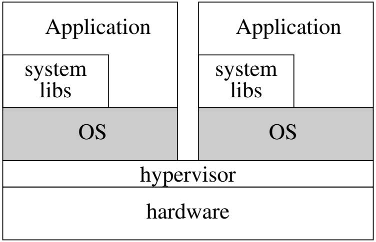
Virtualised OSs

Hypervisors appeared in IBM mainframes in the late 1960s

Conclusion of OS

There are several ways OS virtualisation is done

Conclusion of OS



Bare metal virtualisation has a thin layer, the *hypervisor*, to manage the hardware, allowing each OS to see separate "virtual hardware" which they manage

Conclusion of OS

The OSs can be completely different, e.g., Windows and Linux, and each believe they have the whole machine

Conclusion of OS

The OSs can be completely different, e.g., Windows and Linux, and each believe they have the whole machine

Modern X86 architectures provide a Ring -1 to support this

Conclusion of OS

The OSs can be completely different, e.g., Windows and Linux, and each believe they have the whole machine

Modern X86 architectures provide a Ring -1 to support this

Examples: Xen, Hyper-V

Conclusion of OS

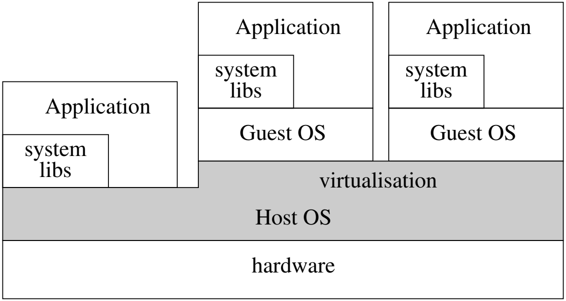
The OSs can be completely different, e.g., Windows and Linux, and each believe they have the whole machine

Modern X86 architectures provide a Ring -1 to support this

Examples: Xen, Hyper-V

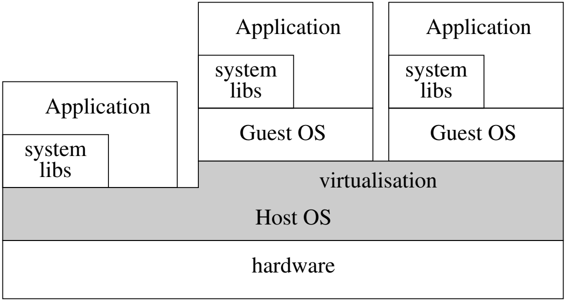
Good for sharing the computer amongst users who have requirements for different OSs

Conclusion of OS



Hosted virtualisation has a normal *host* OS that runs virtualisation code. One or more *guest* OSs run on top of that

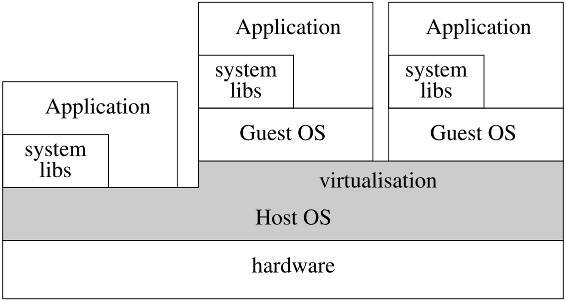
Conclusion of OS



Hosted virtualisation has a normal *host* OS that runs virtualisation code. One or more *guest* OSs run on top of that

Examples: VMWare, VirtualBox, Parallels

Conclusion of OS

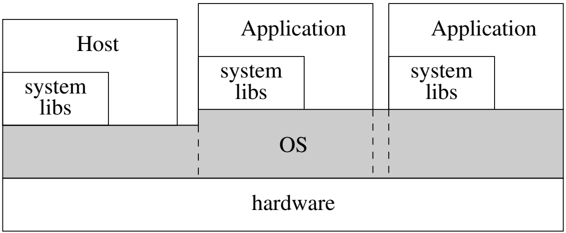


Hosted virtualisation has a normal *host* OS that runs virtualisation code. One or more *guest* OSs run on top of that

Examples: VMWare, VirtualBox, Parallels

Good for when you need sophisticated management of the guest OSs by the host OS, for example in Cloud provision

Conclusion of OS



Not quite OS virtualisation, but with the same target applications is *containers*. The applications share the same OS, but the OS is rigidly partitioned so each container cannot see or influence what is happening in other containers (e.g., CPU limits)

Conclusion of OS

With containers, the applications must run on the same OS kernel, but can have different systems libraries and other software (e.g., RedHat in one and Ubuntu in another)

Conclusion of OS

With containers, the applications must run on the same OS kernel, but can have different systems libraries and other software (e.g., RedHat in one and Ubuntu in another)

We might think of this as a kind of multiple user modes

Conclusion of OS

With containers, the applications must run on the same OS kernel, but can have different systems libraries and other software (e.g., RedHat in one and Ubuntu in another)

We might think of this as a kind of multiple user modes

Examples: Solaris containers, Docker

Conclusion of OS

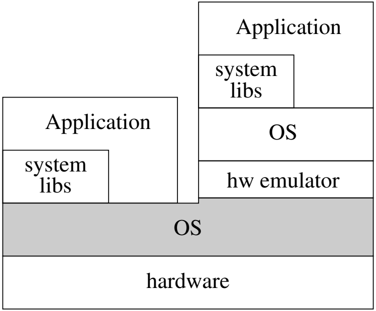
With containers, the applications must run on the same OS kernel, but can have different systems libraries and other software (e.g., RedHat in one and Ubuntu in another)

We might think of this as a kind of multiple user modes

Examples: Solaris containers, Docker

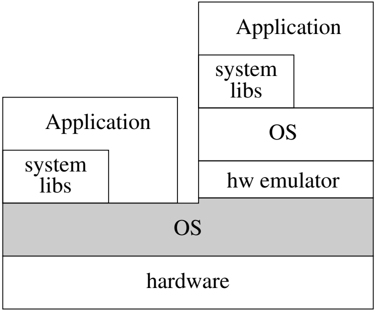
Good for application delivery, where an application needs a lot of specific system library support: so we deliver the systems libraries with the application!

Conclusion of OS



And then there are variants that do *hardware virtualisation* by emulating different kinds of hardware, e.g., we might have our OS running on an ARM emulation running on X86 hardware

Conclusion of OS



And then there are variants that do *hardware virtualisation* by emulating different kinds of hardware, e.g., we might have our OS running on an ARM emulation running on X86 hardware

Or on an X86 emulation on ARM hardware

Conclusion of OS

These emulations are a lot slower than the native hardware, but provide a flexibility to the customer

Conclusion of OS

These emulations are a lot slower than the native hardware, but provide a flexibility to the customer

Examples: Qemu (emulates several kinds of hardware), Bochs (emulates X86)

Conclusion of OS

These emulations are a lot slower than the native hardware, but provide a flexibility to the customer

Examples: Qemu (emulates several kinds of hardware), Bochs (emulates X86)

Exercise Compare with Apple's new Rosetta software that allows Intel code to run on Arm hardware (only user code, though)

Conclusion of OS

Exercise Read up on Cloud Services, Software as a Service (SaaS), Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software Appliances

Conclusion of OS

All of these techniques are applied in cloud computing, where users buy time on a large, remote machine

Conclusion of OS

All of these techniques are applied in cloud computing, where users buy time on a large, remote machine

Welcome to the 1960s!

Conclusion of OS

Exercise On Mars, the autonomous helicopter drone Ingenuity (brought by the lander Perseverance) runs Linux on a 500Hz (not MHz!) processor. Read about this

Exercise Play with an OS you are not familiar with (Mac, Win or Lin or other) and learn the ways it does things. Write, compile and run a program

Exercise Read about the advances in persistent memory: comparable in speed to main memory, but retains data when power cycled like disk (*non-volatile*). What changes would we need from an OS to deal with such a technology?