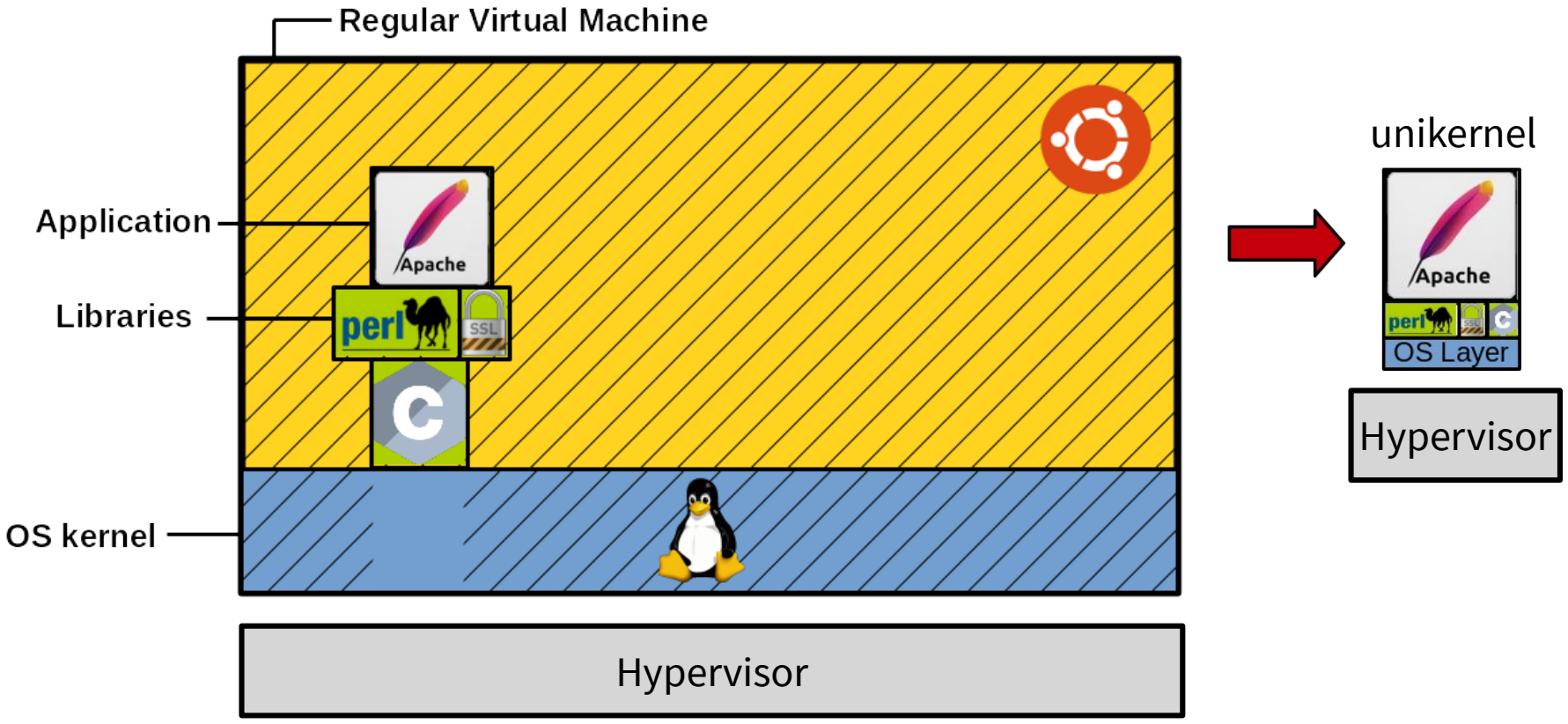
ECE 5984 Virtualization Technologies

Unikernels

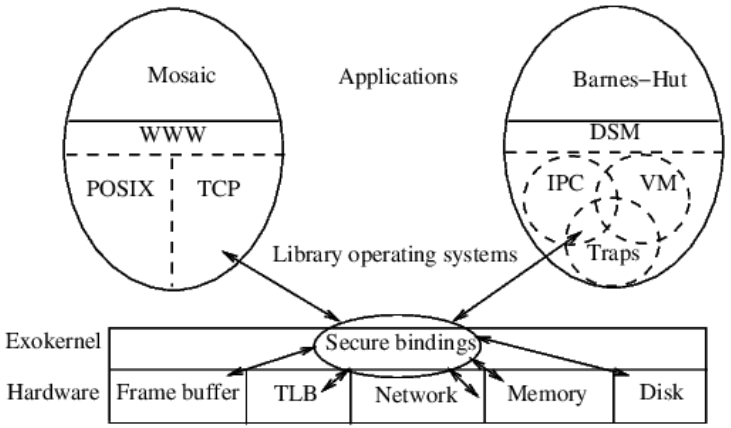
Pierre Olivier

Unikernels


Regular Virtual Machine
Application
Libraries
OS kernel
Apache
perl
SSL
C
Hypervisor
unikernel
Apache
perl
SSL
C
OS Layer
Hypervisor

Library Operating Systems

- **Overheads of regular OS associated with their fundamentals design principles (protection, modularity, generality)**
- **Exokernel: separation of resource protection (exokernel) from management (libOS)**
  - Allows specialization of system services, tailored for each application
  - LibOS accesses the hardware through the exokernel
  - Problem: drivers
- **Unikernels are LibOS for the cloud**
  - The hypervisor plays the role of the Exokernel
  - Drivers: can use PV drivers / direct access through SR-IOV



Engler, Dawson R., and M. Frans Kaashoek. Exokernel: An operating system architecture for application-level resource management. Vol. 29. No. 5. ACM, 1995.

https://www.sigarch.org/leave-your-os-at-home-the-rise-of-library-operating-systems/

Unikernels
Presentation

- **Application + libraries + thin OS layer compiled into a single binary executed as a kernel guest on top of an hypervisor**
  - Packs at compile time only what is needed
    - Of course libraries ...
    - … but also OS services (filesystem, network stack, drivers, etc.) → libOS ...
    - … as well as configuration
  - Small attack surface
  - Reduced resource usage (disk/RAM)
  - Fast boot/destruction time

Unikernels
Presentation (2)

- **Fundamental properties:**
  - *Single process:* one unikernel → one process
    - No support for `fork` - want to run another process? run another unikernel
    - No need to implement a complex scheduler → rely on the hypervisor VCPU scheduler and avoid redundancy
  - *Single user:* no privilege separation between application and kernel, everything runs with full privileges
    - That's okay there is only a single process per unikernel
  - *Single address-space:* application and kernel share the same address space
    - In combination with the *single user* property, reduces the cost of world switch on system calls → they become common function calls

Unikernels
Presentation (3)

- **Application + libraries + thin OS layer compiled into a single binary executed as a kernel guest on top of an hypervisor**
  - Packs only what is needed
    - Of course libraries …
    - … but also OS services (filesystem, network stack, drivers, etc.) → libOS
    - Small attack surface
    - Reduced resource usage (disk/RAM)
    - Fast boot time

Security, **performance**, and **cost reduction** benefits

- **Fundamental properties:**
  - *Single process:* one unikernel → one process
    - No support for `fork` - want to run another process? run another unikernel
    - No need to implement a complex scheduler → rely on the hypervisor VCPU scheduler and avoid scheduling redundancy
  - *Single user:* no privilege separation between application and kernel, everything runs with full privileges
    - That's okay there is only a single process per unikernel
  - *Single address-space:* application and kernel share the same address space
    - In combination with the *single user* property, reduces the cost of world switch on system calls → they become common function calls

Unikernels
Applications

- **Server applications**
- **Cloud services, on-demand micro-services, SaaS/FaaS**
  - Large monolithic application decomposed into set of single-purpose applications communicating via the network and evolving independently
    - Simplicity, ease of development/deployment, scalability through modularity
- **Network Function Virtualization in distributed environments**
  - Ex: edge computing
- **IoT**
  - High security & low resource consumption demands
- **HPC**
  - High performance demands
  - Reduction of OS interference on compute workload
- **Unikernels are still in a relative 'research' state**

Unikernels
Unikernel models

- **2 main classes: legacy and clean-slate**
- **Legacy:**
  - *Rumprun* (C/C++/Ruby/Go/Python/etc.)
  - *Osv* (C/C++/Java/Lua/Go)
  - *IncludeOS* (C/C++)
  - *HermitCore* (C/C++/Fortran/Go)
  - Etc.
- **Clean-slate → using type-safe languages**
  - *MirageOS* (Ocaml)
  - *HalVM* (Haskell)
  - *LING* (Erlang)

Tools:
- unik: automate porting a legacy application
- solo5/ukvm: minimal hypervisor for unikernels

Unikernels
Current limitations

- **Existing applications support**
  - Clean slate unikernels: need to rewrite the entire application!
  - Legacy unikernels: better but not ideal
    - Still need to recompile
    - No support for fork()
- **Debugging is difficult**
  - Generally no access to tools such as GDB

Unikernels
Links & literature

- **General info about unikernels, multiple models presented: http://unikernel.org/**
- **Important papers:**
  - Manco, Filipe, et al. "**My VM is Lighter (and Safer) than your Container.**" Proceedings of the 26th Symposium on Operating Systems Principles. ACM, 2017.
  - Madhavapeddy, Anil, et al. "**Unikernels: Library operating systems for the cloud.**" Acm Sigplan Notices. Vol. 48. No. 4. ACM, 2013.
  - Madhavapeddy, Anil, et al. "**Jitsu: Just-In-Time Summoning of Unikernels.**" NSDI. 2015.
  - Martins, Joao, et al. "**ClickOS and the art of network function virtualization.**" Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation. USENIX Association, 2014.
  - Tsai, Chia-Che, et al. "**Cooperation and security isolation of library OSes for multi-process applications.**" Proceedings of the Ninth European Conference on Computer Systems. ACM, 2014.
  - Porter, Donald E., et al. "**Rethinking the library OS from the top down.**" ACM SIGPLAN Notices. Vol. 46. No. 3. ACM, 2011.

Unikernels
Links & literature (2)

- MirageOS: https://mirage.io/, https://github.com/mirage
- Rump: http://rumpkernel.org/, https://github.com/rumpkernel/rumprun
- Osv: http://osv.io/, https://github.com/cloudius-systems/osv
- IncludeOS: http://www.includeos.org/, https://github.com/hioa-cs/IncludeOS
- HermitCore: http://www.hermitcore.org/, https://github.com/RWTH-OS/HermitCore
- HalVM: https://galois.com/project/halvm/, https://github.com/GaloisInc/HaLVM
- LING: http://erlangonxen.org/, https://github.com/cloudozer/ling

Unikernels
Links & literature (3)

- **Unikernels profiling**
  - Xen: Schmidt, Florian. "uniprof: A Unikernel Stack Profiler." Proceedings of the SIGCOMM Posters and Demos. ACM, 2017.
  - Xen (MirageOS): http://www.brendangregg.com/blog/2016-01-27/unikernel-profiling-from-dom0.html
  - KVM (HermitCore) – Xray: https://github.com/RWTH-OS/HermitCore/tree/master/usr/xray