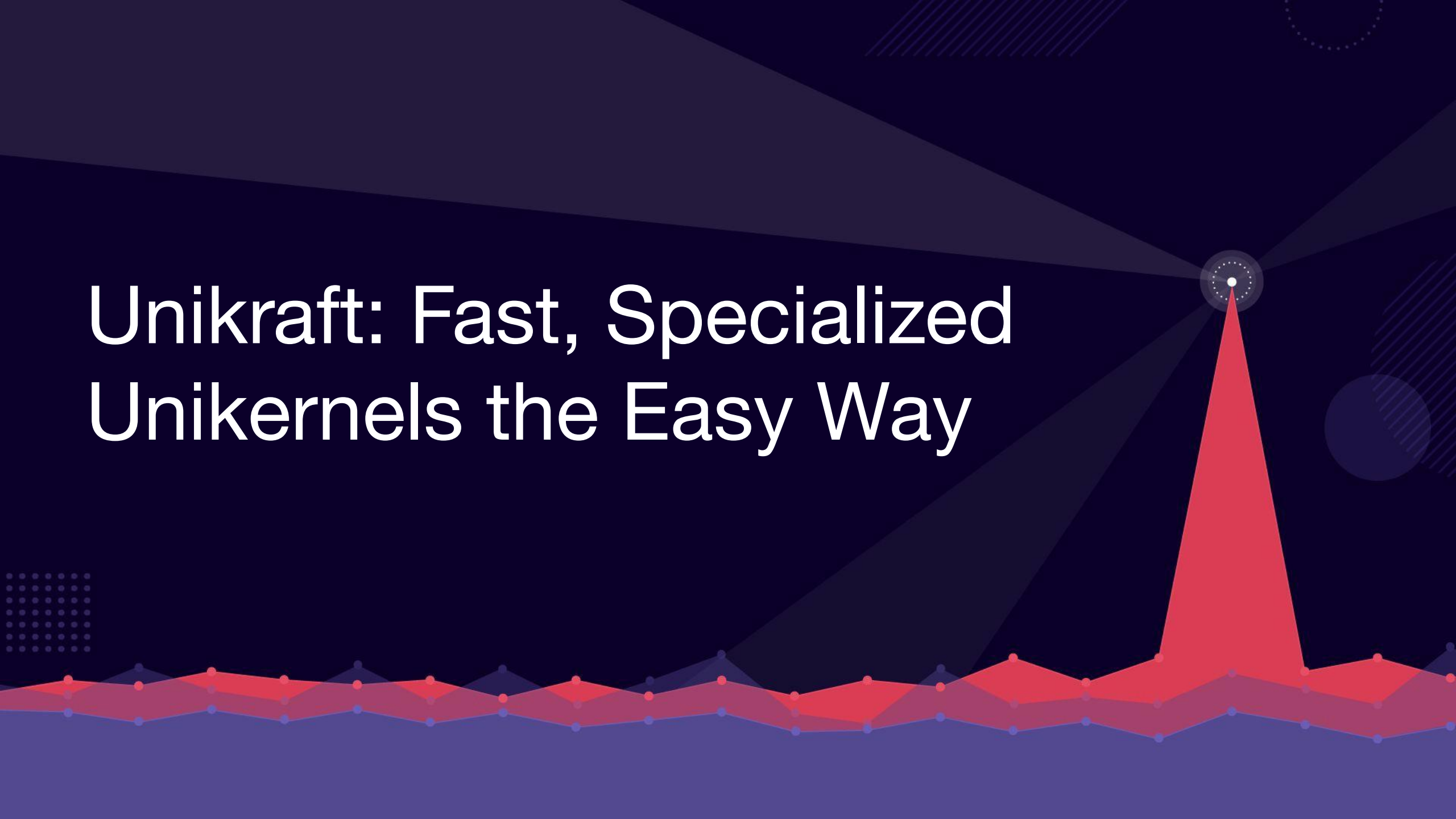
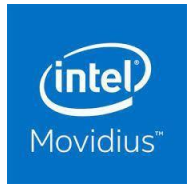
Unikraft: Fast, Specialized Unikernels the Easy Way



Specialization = High Performance

software

hardware

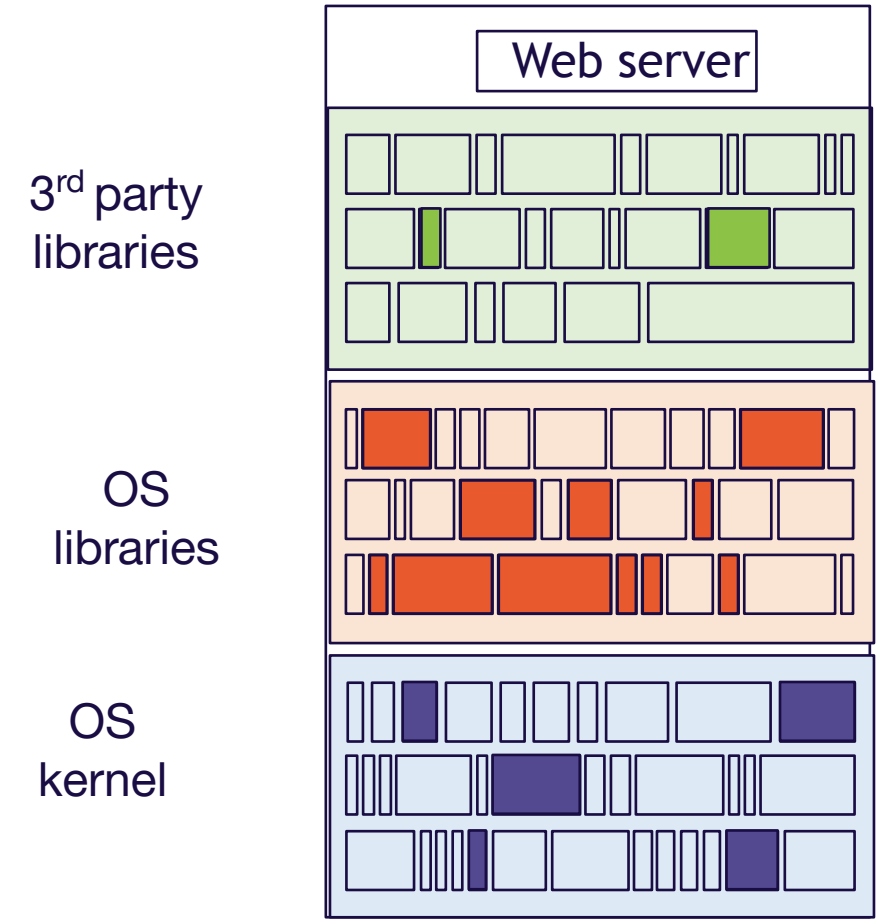


Unikernels = Specialized Virtual Machines

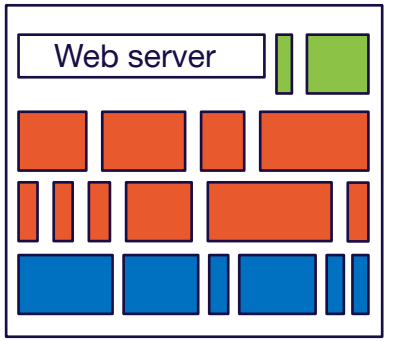
GOALS

- **Easy to build and run**
- **Easy or no app porting**
- **Great performance**

Unikernels in One Slide



General Purpose OS (e.g., Linux)



Unikernel

Unikernel Power

Fast start/stop/migration times

- 10s of milliseconds or less (and as little as 2.3ms)

REFS: LigthVM [Manco SOSP 2017], Jitsu [Madhvapeddy, NSDI 2015]

Low memory footprint

- Few MBs of RAM or less

REFS: ClickOS [Martins NSDI 2014], Unikraft [Kuenzer, Eurosys 2021. Best Paper Award]

High density

- 8k guests on a singlex86 server

REFS: LigthVM [Manco SOSP 2017]

High Performance

- ~300K reqs/sec nginx with a single guest CPU

REFS: Unikraft [Kuenzer, Eurosys 2021. Best Paper Award], Elastic CDNs [Kuenzer VEE 2017]

Security Features

- Small trusted compute base
- Strong isolation by hypervisor
- Per-library isolation

REFS: FlexOS [Lefeuvre HotOS 2021], CubicleOS [Sartakov ASPLOS 2021])

Design Principles

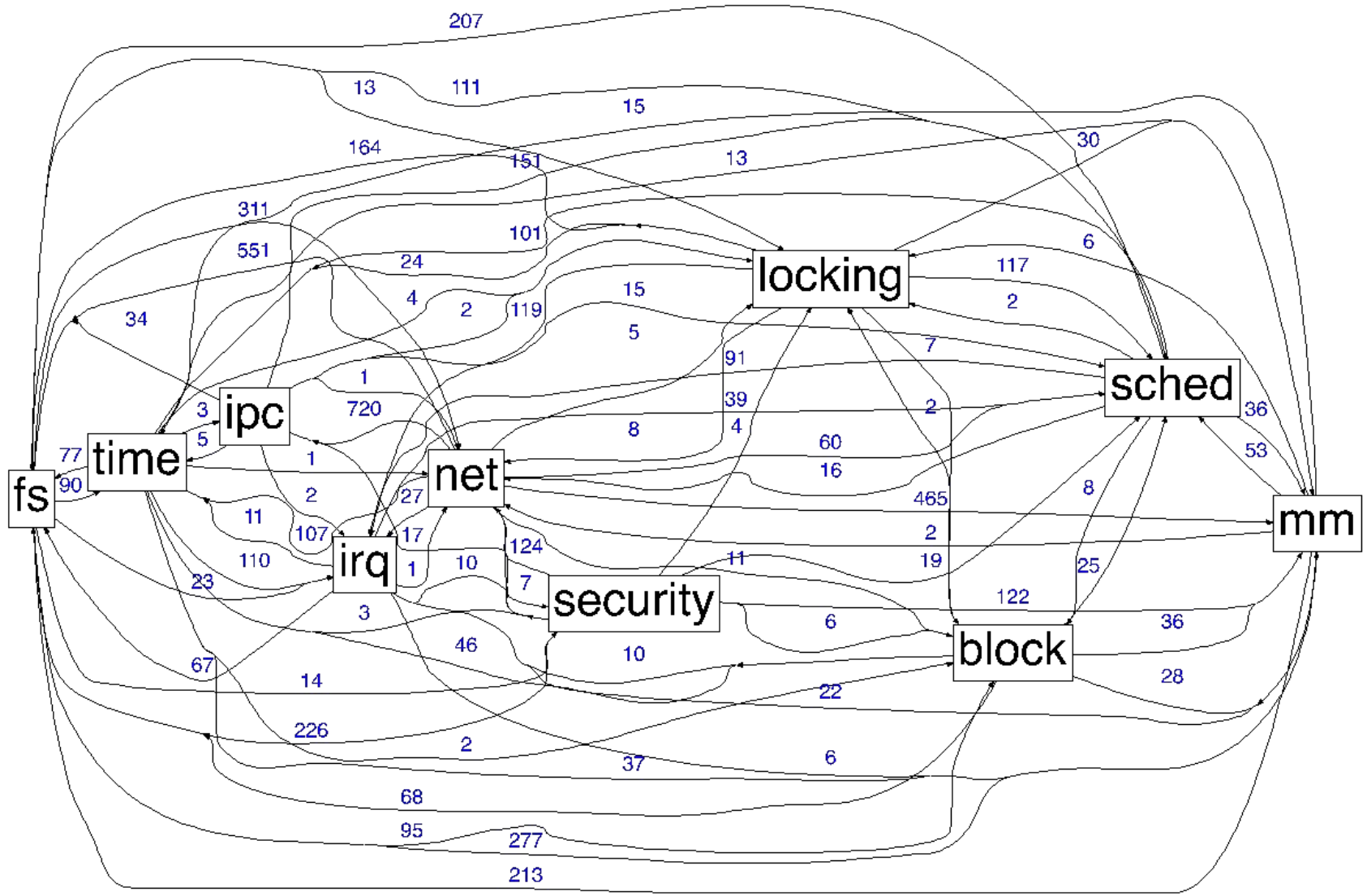
1. Fully modular kernel

2. Provide high performance **specialized APIs**

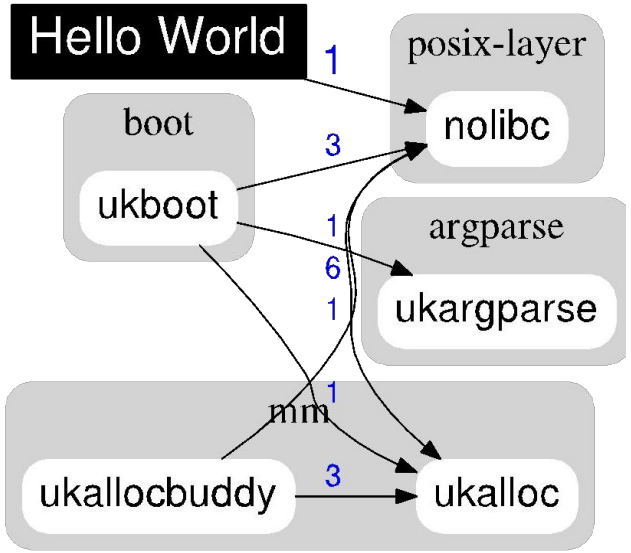
Design Principles

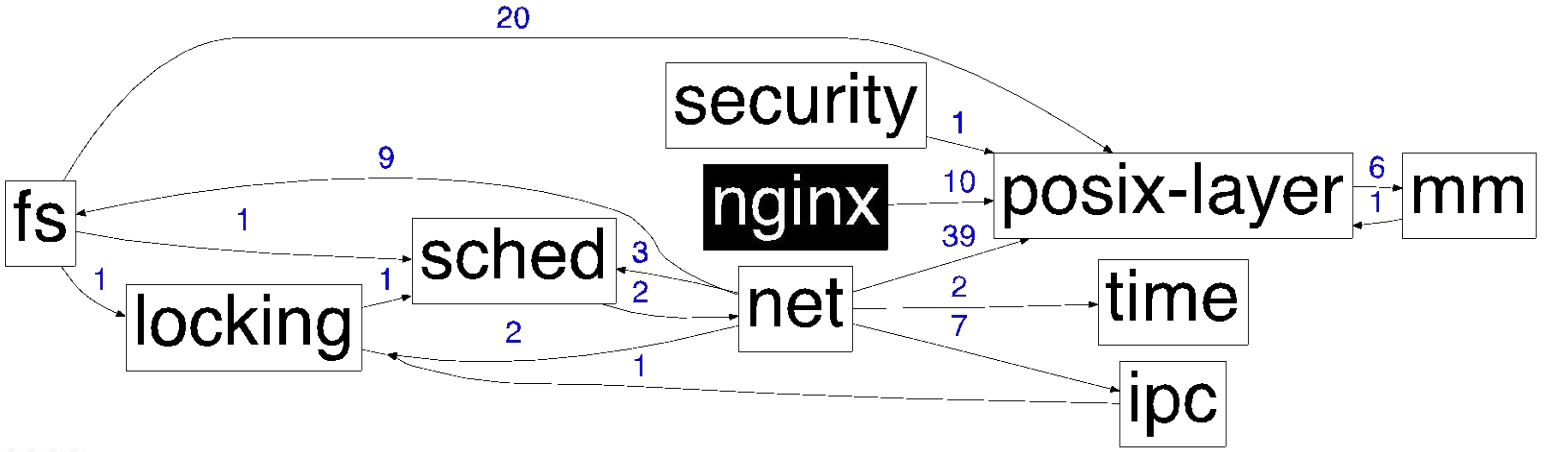
1. Fully modular kernel

Why not Linux?



With Unikraft





Doing it with existing unikernels?

1. Require significant expert work to build

2. They are often not posix compliant

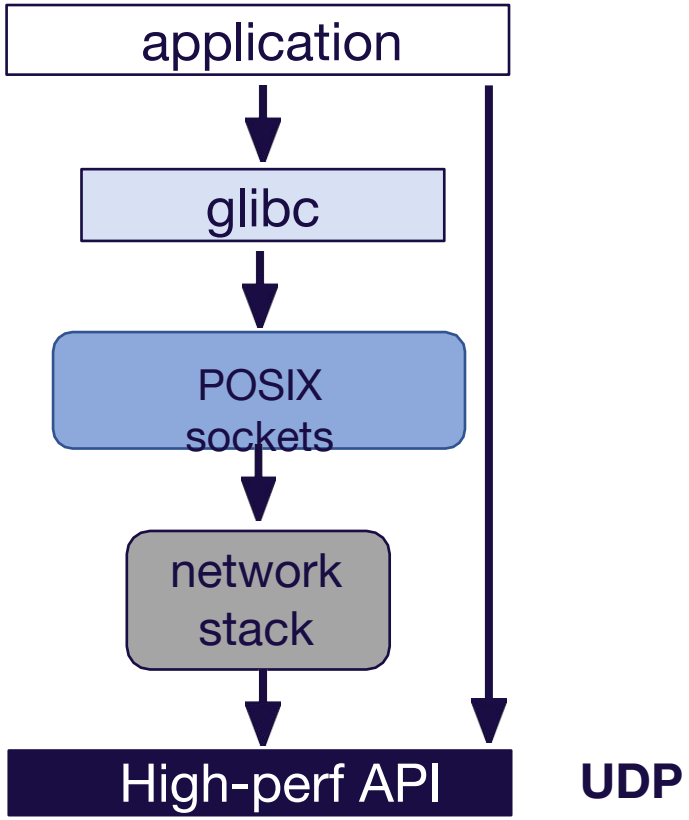
3. The (uni)kernels are *still* monolithic

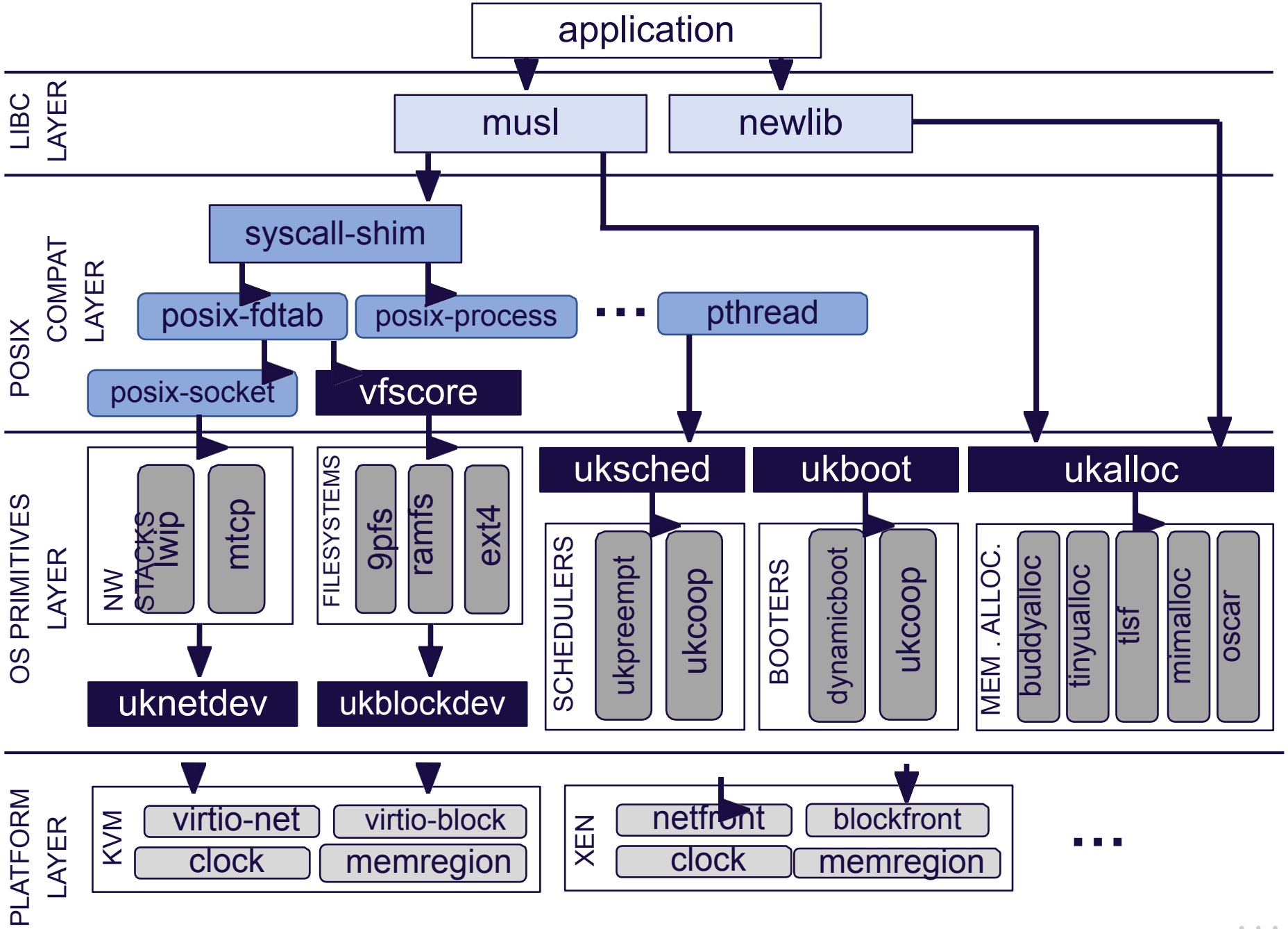
Design Principles

1. Fully modular kernel

2. Provide high performance **specialized APIs**

Specialized API Example




application
LIBC LAYER
musl
newlib
POSIX COMPAT LAYER
syscall-shim
posix-fdtab
posix-process
pthread
posix-socket
vfscore
OS PRIMITIVES LAYER
NW STACKS
lwip
mtcp
FILESYSTEMS
9pfs
ramfs
ext4
uksched
ukboot
ukalloc
SCHEDULERS
ukpreempt
ukcoop
BOOTERS
dynamicboot
ukcoop
MEM . ALLOC.
buddyalloc
tinyualloc
tlsf
mimalloc
oscar
uknetdev
ukblockdev
PLATFORM LAYER
KVM
virtio-net
virtio-block
clock
memregion
XEN
netfront
blockfront
clock
memregion
...

GOALS

- **Easy to build and run**
- **Easy or no app porting**
- **Great performance**

BUILD

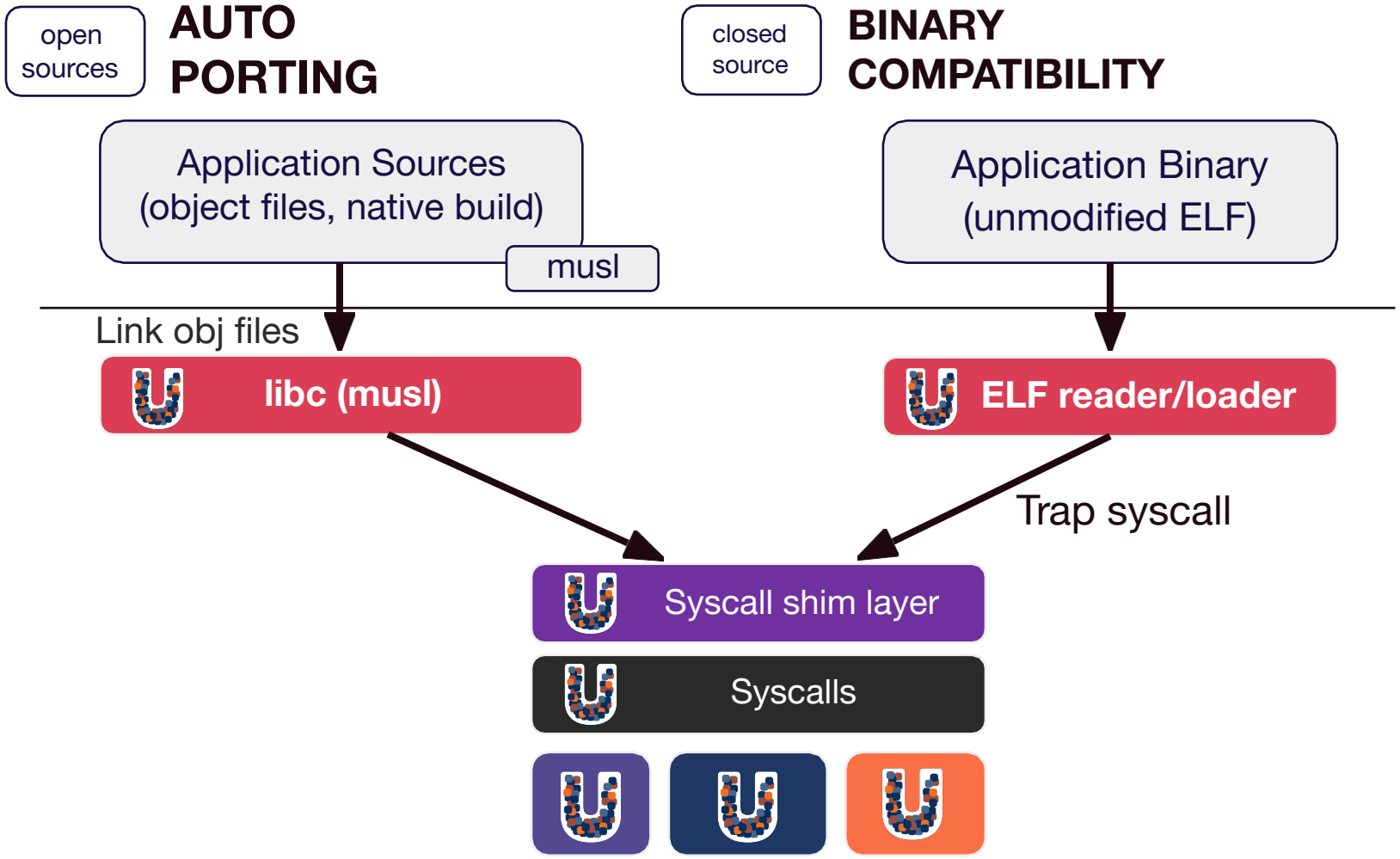
KRAFT

ready

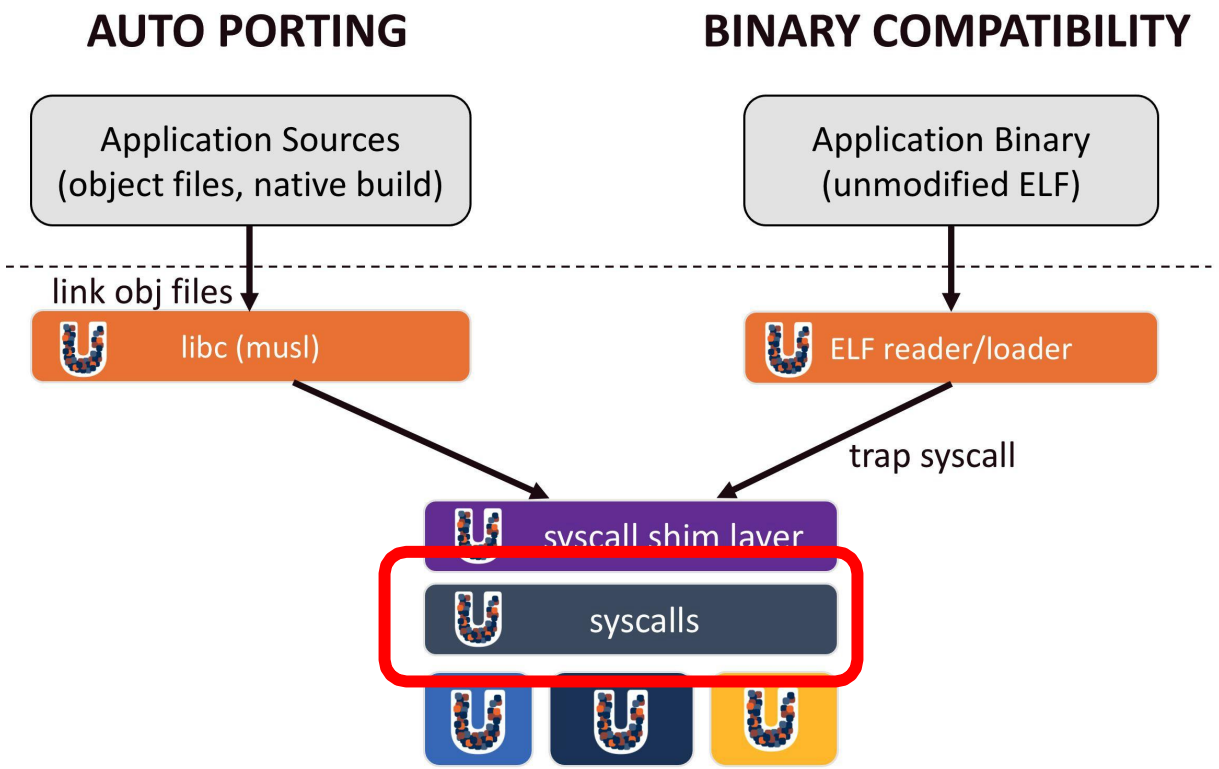
GOALS

- Easy to build and run
- **Easy or no app porting**
- Great performance

POSIX Compatibility – Two Approaches



What about syscall support?

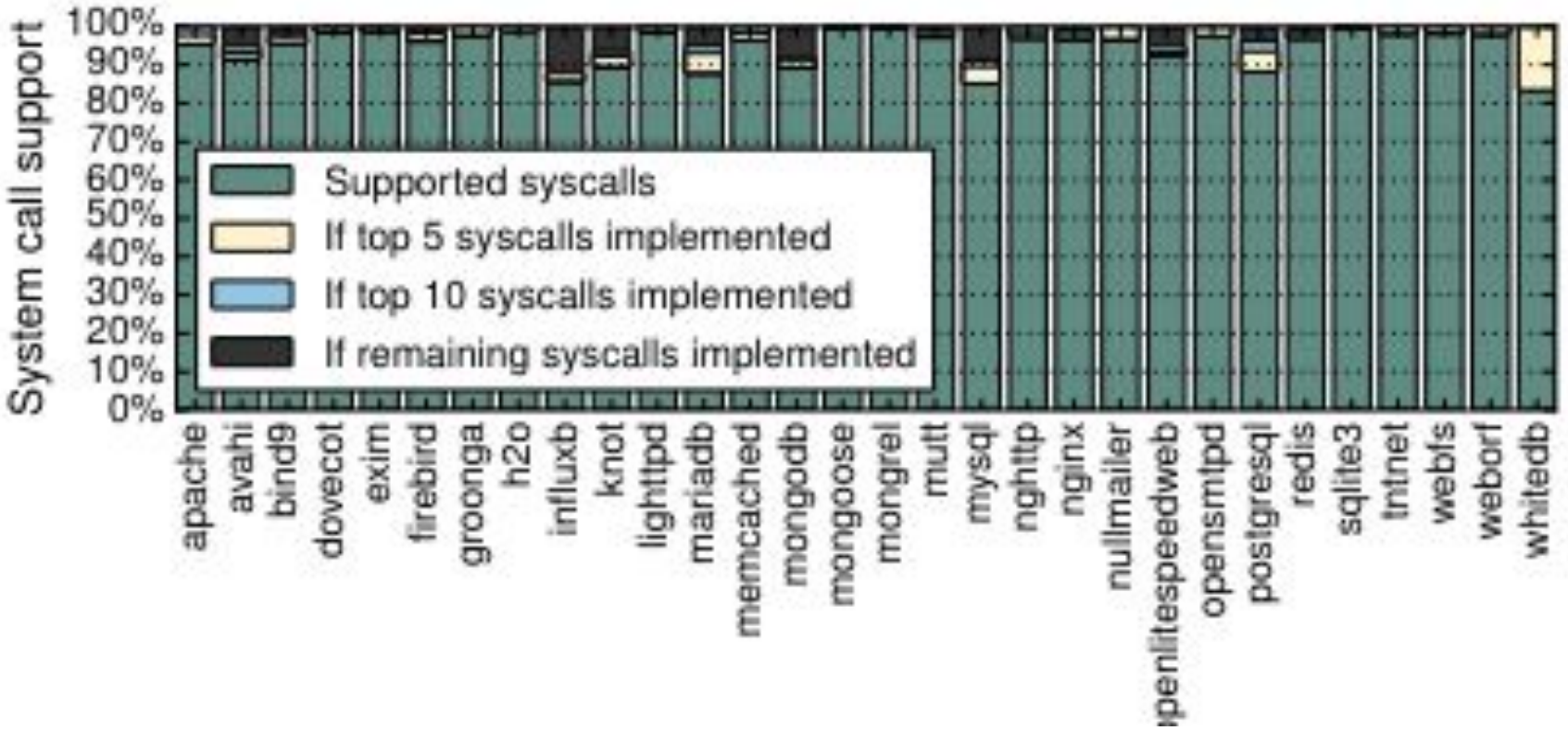


Syscall Support

Linux: ~350 syscalls

Unikraft: 160 syscalls (so far)

Syscall Support
Top 30 Debian Popcon Apps



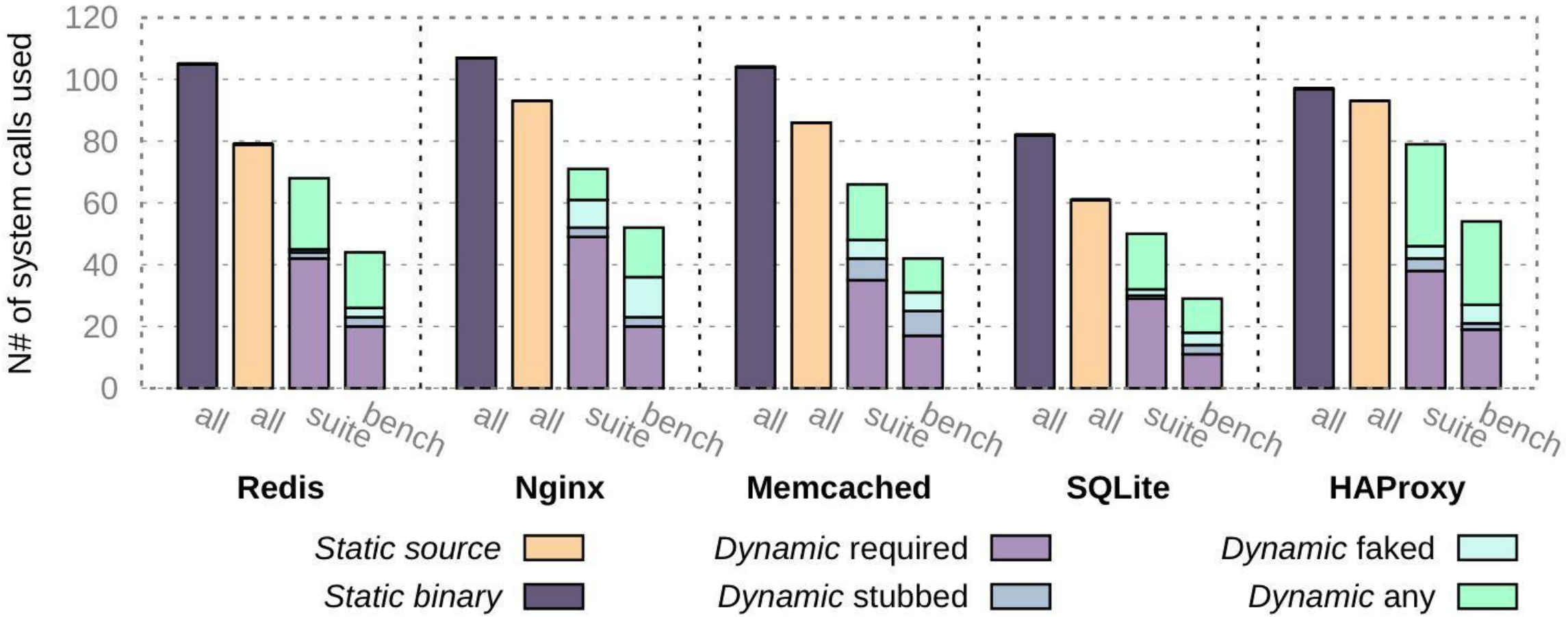
160 syscalls currently supported

Apps are Resilient to Missing Syscalls

```
if (getrlimit(RLIMIT_NOFILE,&limit) == -1) {
    serverLog(LL_WARNING,"Unable to obtain the current NOFILE"
        "limit (%s), assuming 1024 and setting the max clients"
        "configuration accordingly.", strerror(errno));
    server.maxclients = 1024-CONFIG_MIN_RESERVED_FDS;
}
```

(Redis)

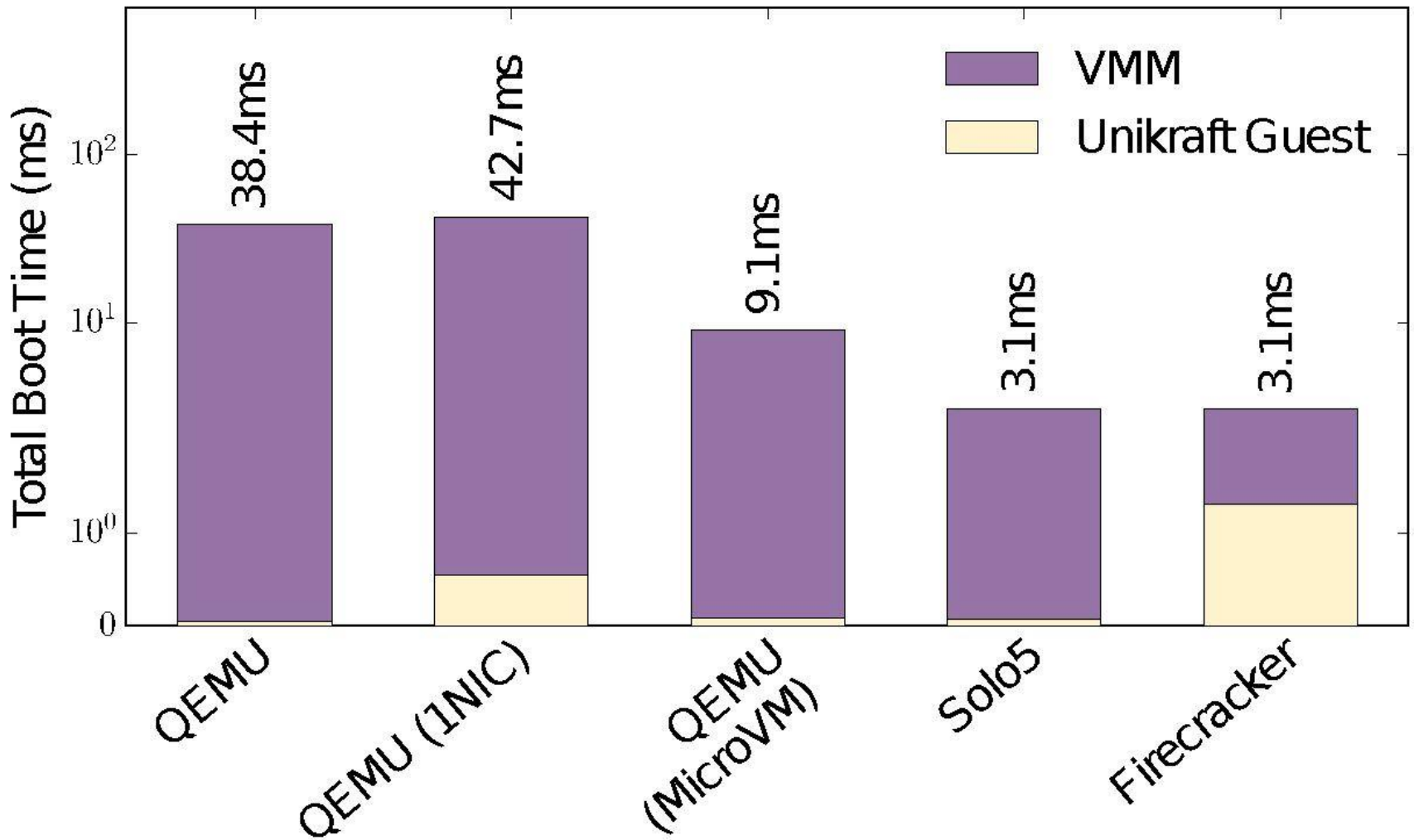
Dynamic Analysis: What we Really Need



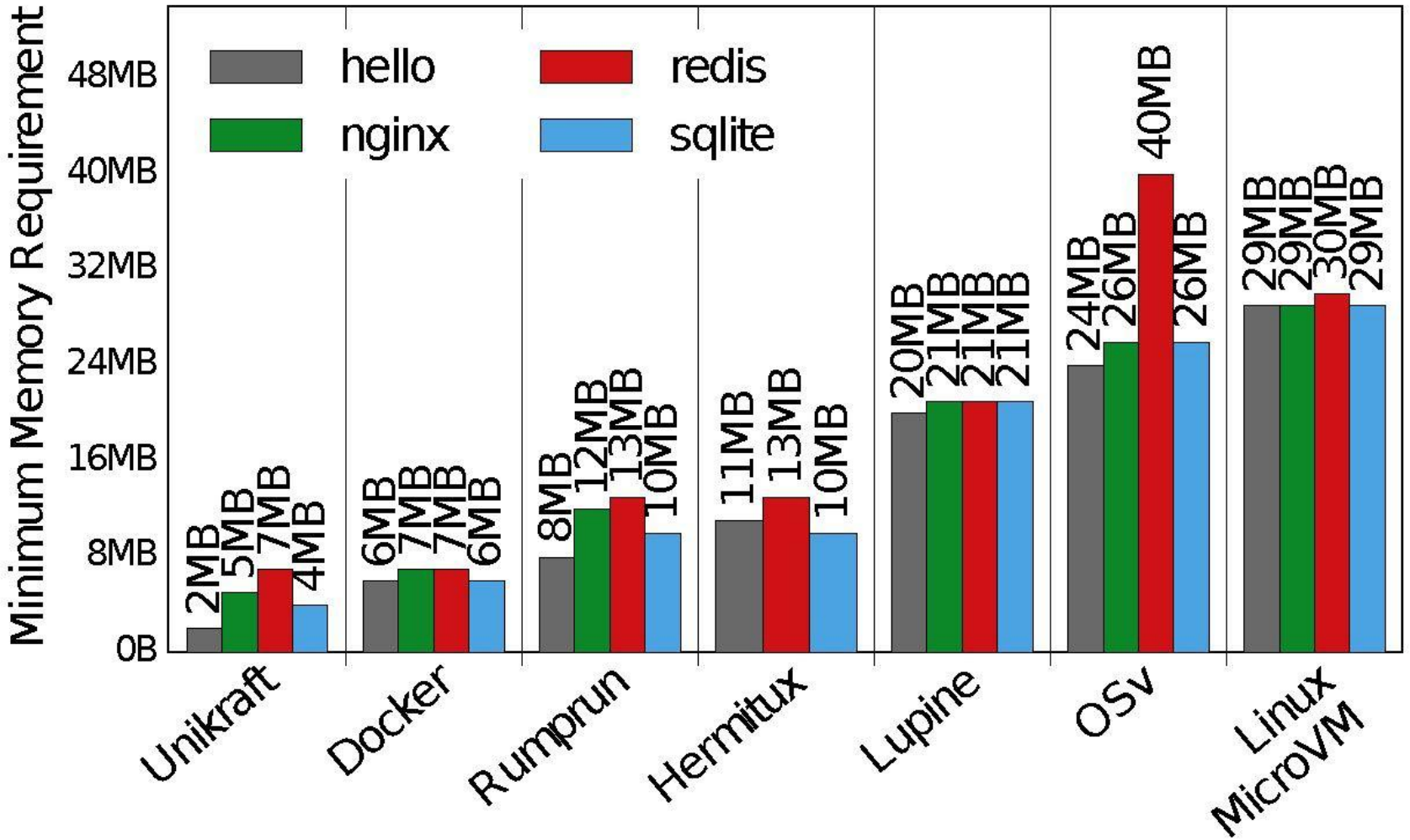
GOALS

- Easy to build and run
- Easy or no app porting
- **Great performance**

Unikraft Boot Times



Minimum Memory Requirements



nginx Throughput