A practical solution for GNU/Hurd's lack of drivers: NetBSD's rumpkernel framework

Damien Zammit

FOSDEM'22

Overview

- Key terms
- Microkernel vs. Monolithic kernel models
- What is rump?
- How does GNU/Hurd use rump?
- Live Demo!

Key terms

- GNU/Hurd
  - original Free Software operating system
  - copyleft licensed
  - started in 1980s
  - not quite finished!
  - uses a microkernel model
- NetBSD
  - original internet operating system
  - permissively licensed
  - based on Berkeley Software Distribution (BSD)
  - runs on pretty much anything!
  - uses a monolithic kernel model

What is a monolithic kernel model?

- all drivers in a single privileged process

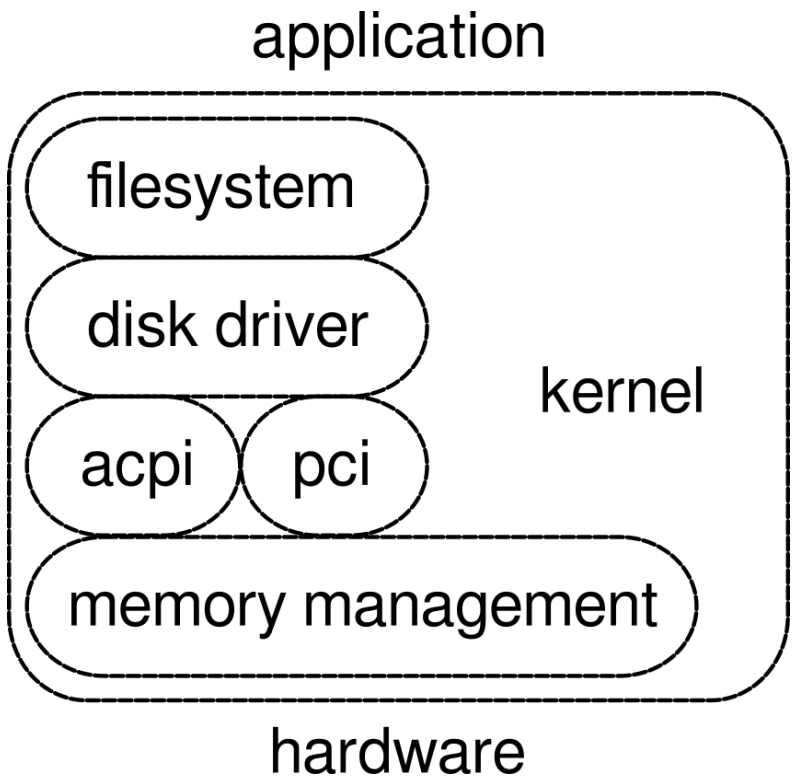


Figure: eg, Linux kernel

What is a microkernel model?

- all drivers are broken into separate user processes
- only minimal components remain in the kernel

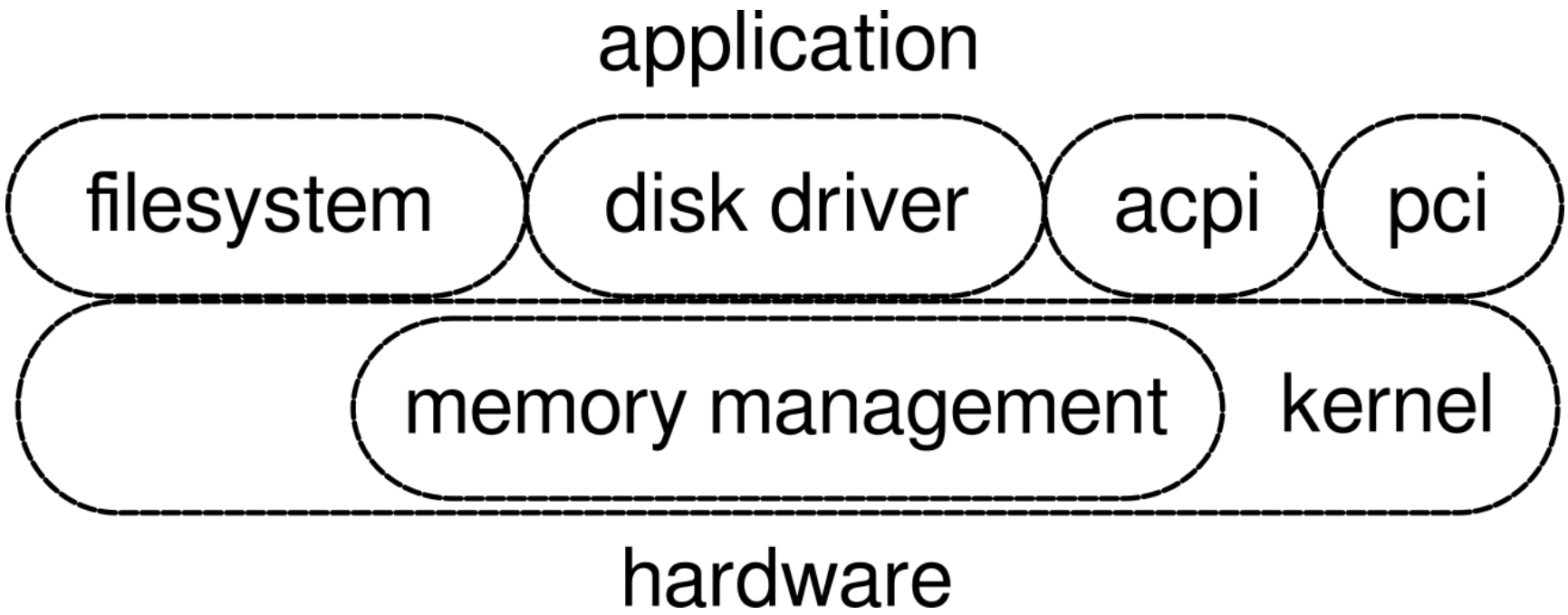


Figure: eg, gnumach kernel

How do processes communicate in a microkernel model?

- gnumach provides:
  - a message bus to grant rights to make Remote Procedure Calls (RPCs)
  - thus enabling inter-process communication (IPC)

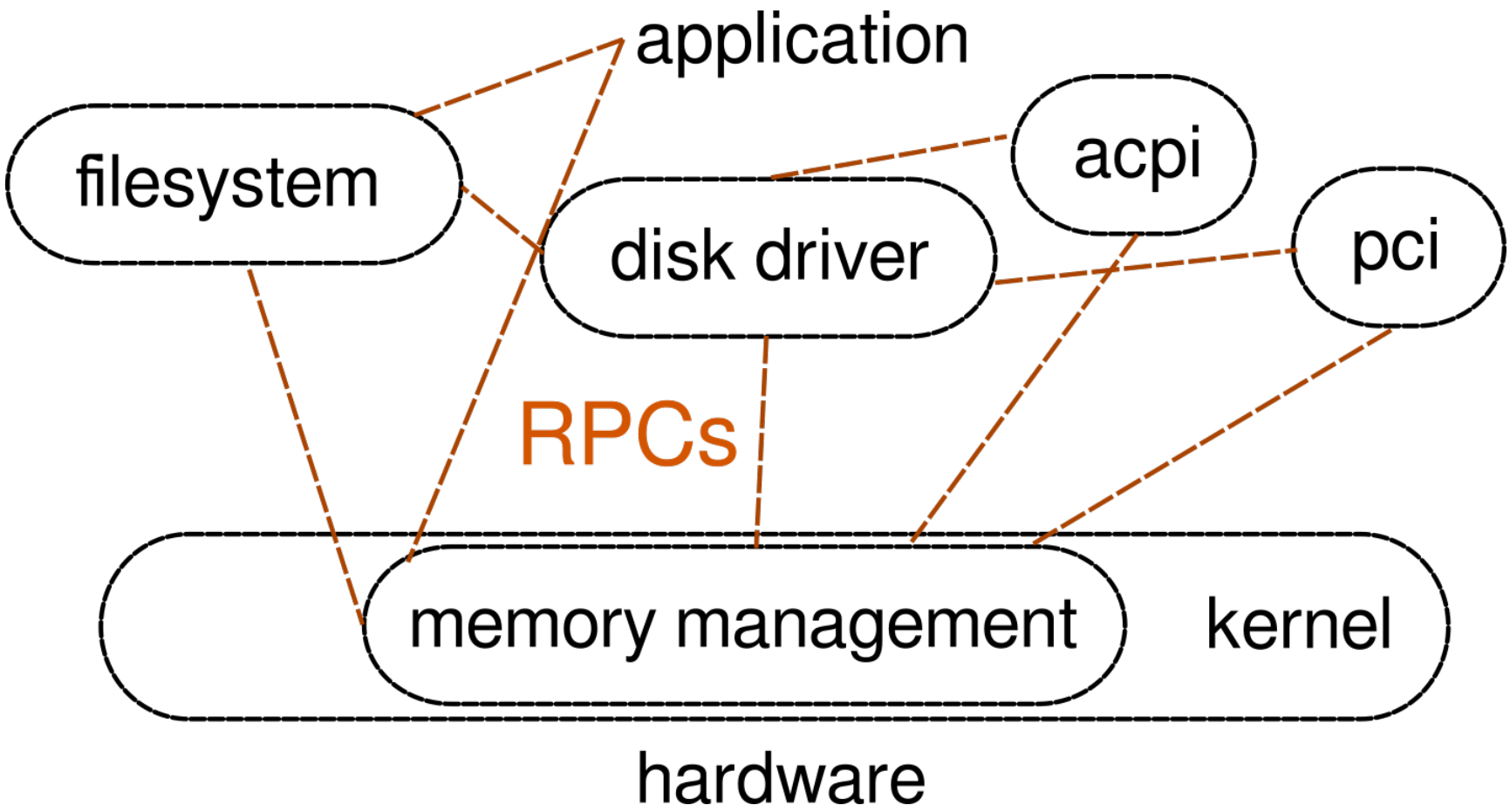


Figure: RPCs are like function calls

What is rump?

- A userspace implementation of NetBSD's kernel drivers as unikernel libraries.

anykernel

- runs only on NetBSD
- has two modes:
  - compiled modules running as **unikernels** in userspace
  - compiled modules running entirely in monolithic kernel

unikernel

- runs on any OS
- compiled libraries
- runs as process in userspace

History

- Rump was implemented as a PhD project of Antti Kantee.
- 2014:
  - possible to run a rump kernel
  - backed by real pci hardware
  - due to pci-userspace implementation.
- 2015: Robert Millan
  - packaged rump for debian-hurd
  - implemented mplayer to work with rump audio driver in Hurd
- 2019-2021: Damien Zammit
  - made small improvements to pci-userspace
  - introduced ahcisata driver into rump
  - added ahcisata driver to Hurd for disk support

Current rump status

- NetBSD 9.99.x rump works with GNU/Hurd
- upstream rumpkernel on github is out of date
- newest rump is part of NetBSD src tree
- better to work with NetBSD's upstream tree not old fork of src
- work remains for modern cross-builds of rump

How to integrate rump with another OS

- Port pci-userspace to the desired OS target
- Compile rump libraries for the target
- Write a simple test C program linked with rump libraries, such as

```
#include <rump/rump.h>
int main() {
  int fd;

  rump_init();
  fd = rump_sys_open("/dev/rwd0d", RUMP_O_RDONLY);
  /* Do something with the wd0 block device */

  rump_sys_close(fd);
  rump_sys_reboot(0, NULL);
  return 0;
}
```

How does GNU/Hurd use rump?

Examples of Hurd servers:

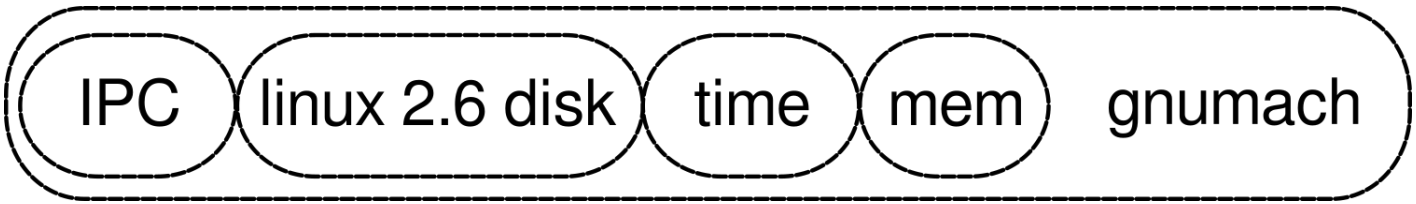
- **pci-arbiter** enumerates PCI
- **acpi** enumerates ACPI
- **rumpdisk** provides block device

Components that remain in gnumach:

- **irq** device: configure and deliver interrupts
- **time** device: provide access to system timer
- **mem** device: provide access to system memory

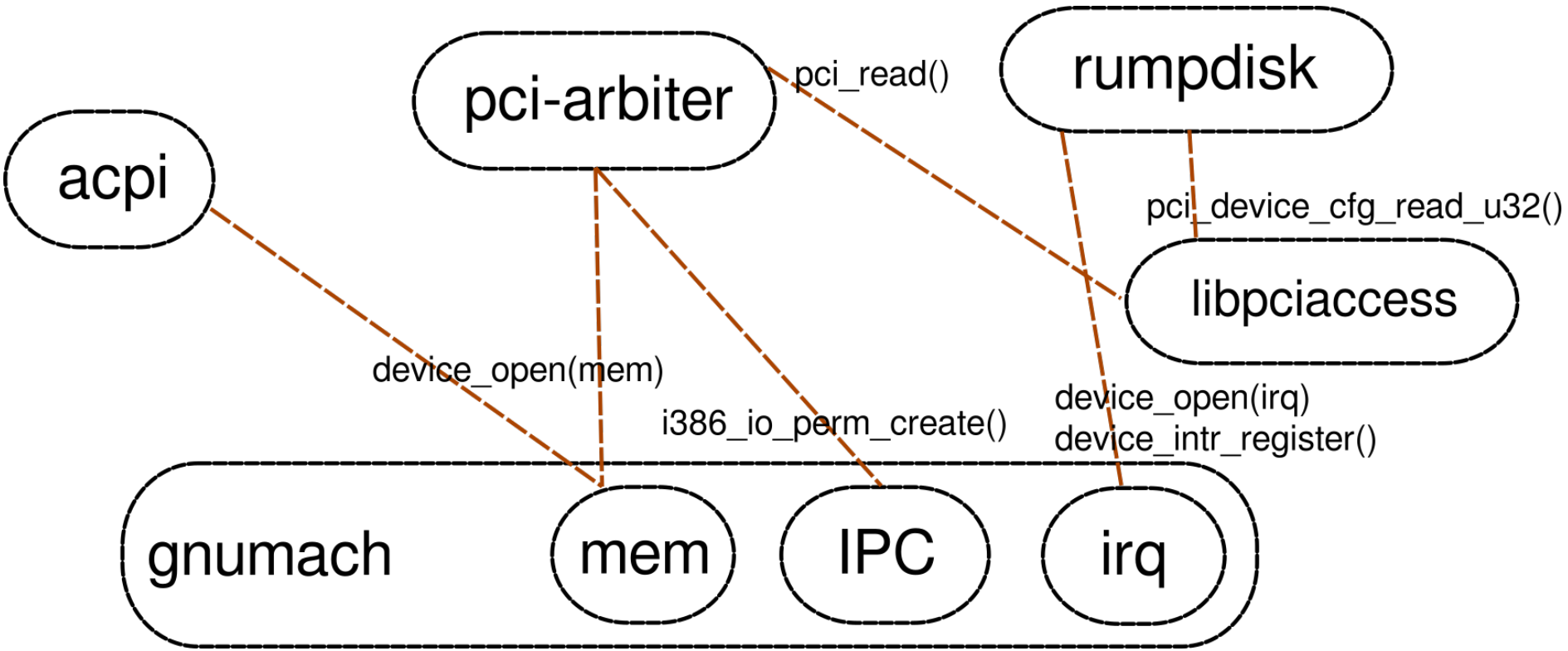
GNU/Hurd before and after integrating rump

Disk access without rump

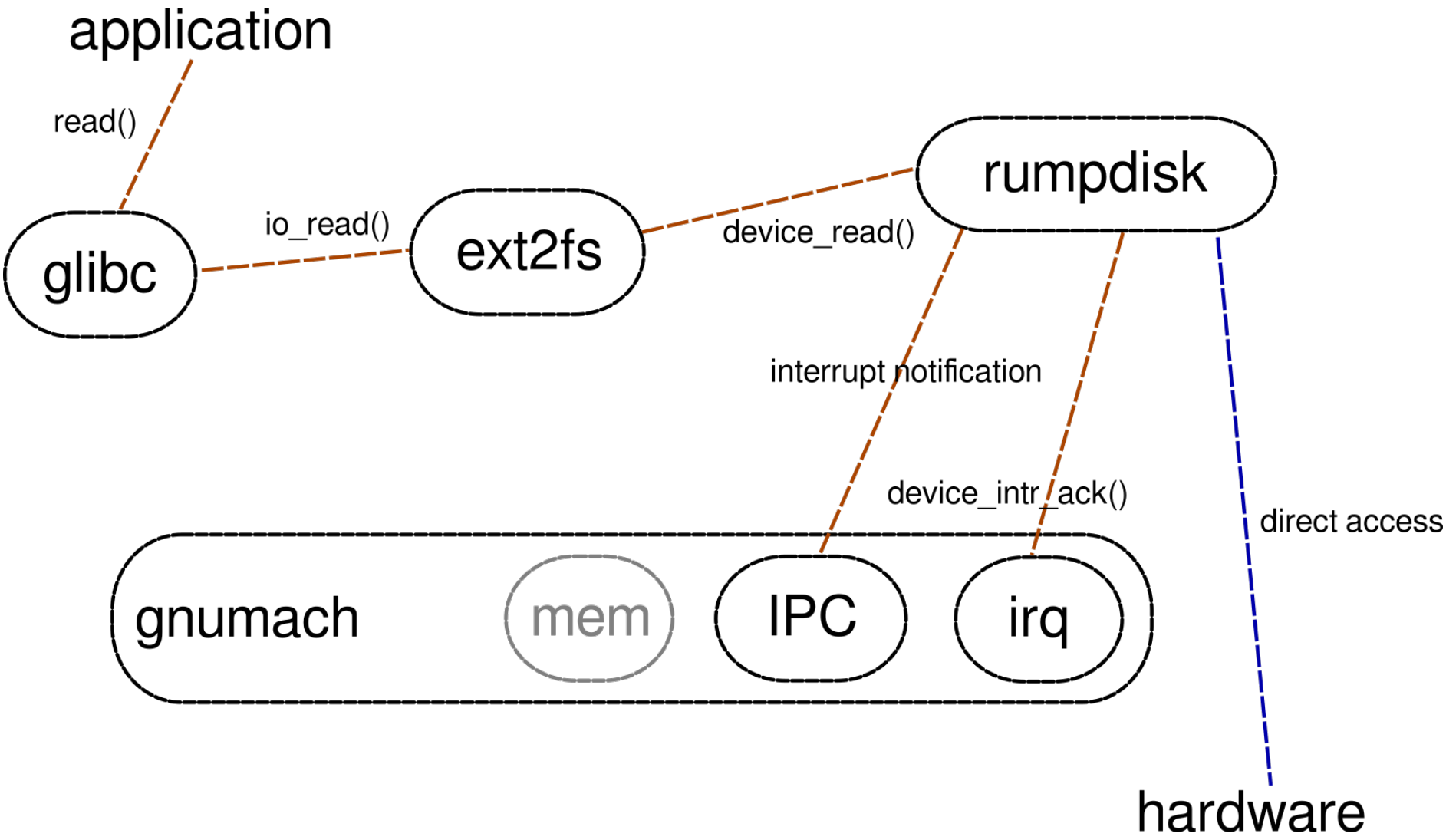


Disk access with rump

Interaction between servers at startup



Interaction between servers during use



Advantages of using rump in GNU/Hurd

- More drivers with a stable interface
- No need to reinvent the wheel
- Debug any process (including the actual driver)
- Licensing concerns are separated

Demo

LIVE DEMO!

References

- http://www.fixup.fi/misc/rumpkernel-book/
- https://github.com/rumpkernel
- https://hurd.gnu.org
- https://netbsd.org

Q&A

Questions?