Mirage/kFreeBSD

Gábor Páli
pgj@FreeBSD.org

FreeBSD
Developer Summit 2012,10
Warsaw, Poland

October 20, 2012

Mirage

What is Mirage...?

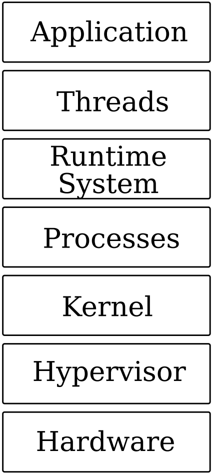
- *Mirage* is an exokernel which can be used for building secure, high-performance network application on top of various cloud and mobile platforms.
- Source code can be developed on arbitrary operating system, e.g. FreeBSD or GNU/Linux that could be then translated to a specialized standalone microkernel to be run on top of Xen.
- It is developed in OCaml, supplemented with some additional syntax constructs and libraries that is going to be mapped directly to the corresponding operating system primitives.
- Completely event-driven architecture, without preemptive thread scheduling... Yeah! :-)

UNIVERSITY OF
CAMBRIDGE

Powered by
Caml

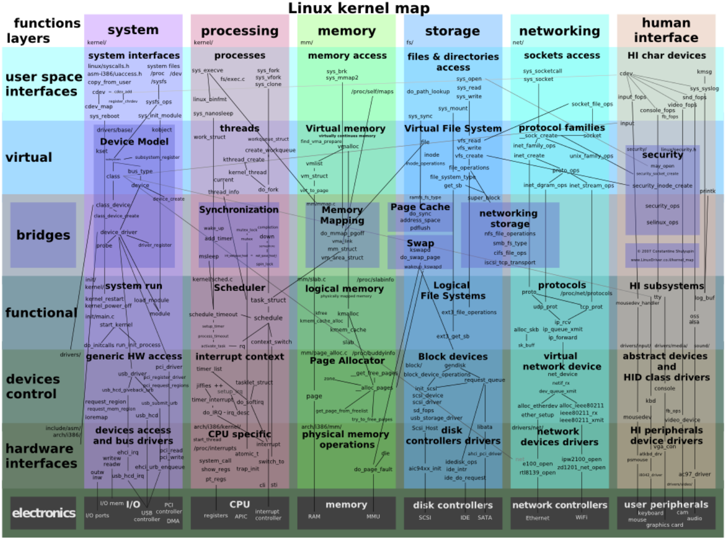
Why Mirage...?

There are way too many layers in today's systems...



Why Mirage...?

Complex and complicated (sometimes even bloated) source code...



How Mirage Can Help With This...?

- Static typing guarantees provided by, and different kind of domain-specific languages expressed in OCaml help to reduce the risk of potential (security) bugs.

- There are more possibilities to deploy compile-time optimizations, *„whole OS optimization”* – less layers are actually present in the resulting binary.

- Each of the systems are simple, easy-to-tackle – multiple cores are handled by the hypervisor.

(Hopefully) A Simple Example

```
let echo () =
  lwt mgr, mgr_t = Manager.create () in
  let src = None, 8081 in
  Flow.listen mgr (‘TCPv4 (src,
    (fun (addr, port) t ->
       Console.log "From %s:%d" (ipv4_addr_to_string addr) port;
       let rec echo () =
         lwt res = Flow.read t in
         match res with
         | None ->
            Console.log "Connection closed";
            return ()
         | Some data ->
            Flow.write t data >>= echo
       in
       echo ()
    )
  ))
```

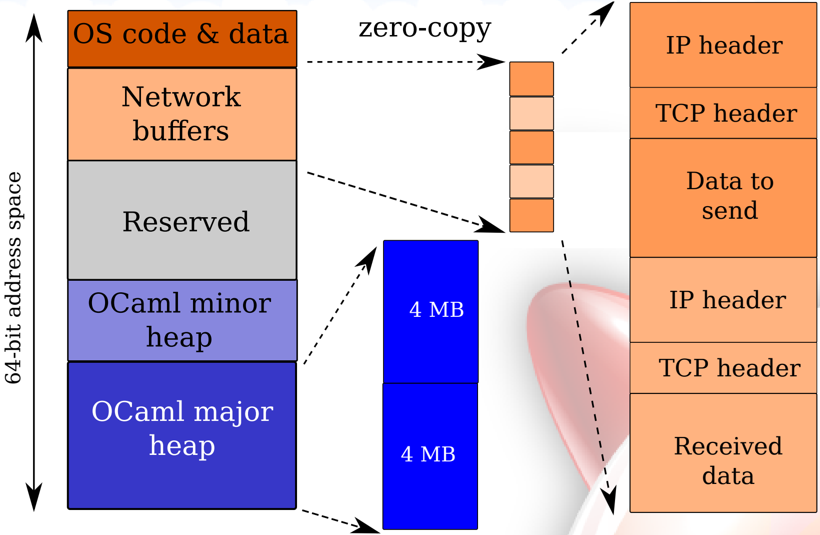
Mirage/Xen

```
$ ocamlopt -output-obj -o app.o echo.ml
```

By using the "Xen MiniOS", we can easily get a bootable kernel.

- It starts up in 64-bit mode, all the memory is available.
- Relatively small size, ab. 50 – 100 KB
- The implementation uses the `lwt` OCaml library, which implements cooperative threading and enables to create monadic blocks.
- Real concurrency is available through Xen (vCPUs).

Memory Management, Processing Network Buffers



Portability

Thanks to OCaml, Mirage is extremely portable. Some of the existing backends:

- **POSIX / TUNTAP**: Standard OCaml runtime + Ethernet tap
- **POSIX**: Standard socket interface via a TCP/UDP socket
- **Javascript**: `ocamljs` – WebSockets
- **Google AppEngine**: `ocamljava` – HTTP
- **Android, iMotes**: `ocamlopt` ARM backend

Further backends are in the works...!

Mirage/kFreeBSD

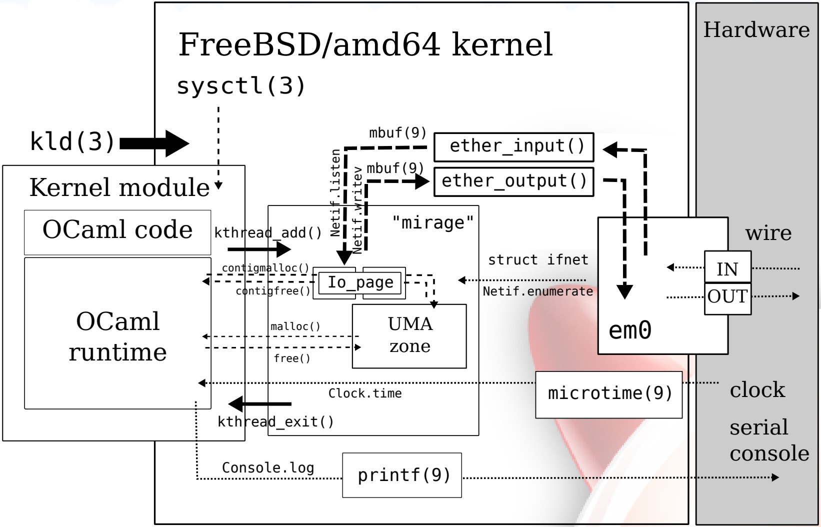
Mirage/kFreeBSD

A summer project at Cambridge University Computer Laboratory: porting Mirage to the FreeBSD kernel.

Objectives:

- Run Mirage-based applications *in* the FreeBSD kernel.
- Implement all the required system-level primivites:
  - Accept and send frames
  - Handle events, interaction with the kernel
  - Preserve „zero-copy” properties
- Study the performance of the generated OCaml code by DTrace.
- Detailed comparison of performance and maintainability of the Mirage network stack and the original FreeBSD network stack.

Current Architecture



Experiences, Challenges

- The OCaml standard library (of version 3.12.1) builds upon floating-point numbers – while the FreeBSD kernel does not really this.

- OCaml ports in the FreeBSD Ports Collection are getting stale. They may possibly need a new maintainer.

- The C compiler is more strict when building kernel modules, uses a different memory model, and it does not support PIC.

- Because of the hybrid C – OCaml solution, it is hard to debug sometimes – logging to serial console works though :-)

- FreeBSD DTrace port has some limitations (when working with kernel modules).

Further Reading

Some recommended links to study:

http://openmirage.org/
http://github.com/avsm/mirage/
http://github.com/mirage/
http://github.com/pgj/mirage-kfreebsd/
http://ocsigen.org/lwt/