XPC vs uintr

- 什么是 ipc
  - XPC
- uintr 实现 ipc
- 分析与讨论

什么是 IPC ?

- micro-kernel: fs service / database service
- android binder
    - binder: `onTransact()` - flags: `ONE_WAY`
- IPC = RPC(remote procedure call) = 指定处理函数 + 参数传递 = 同步机制 + 数据传递

IPC 步骤

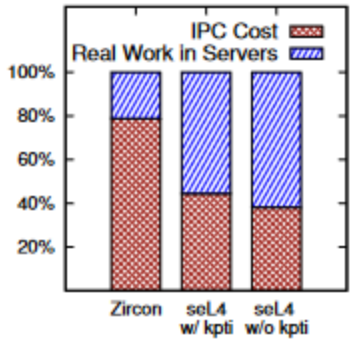
- 建立链接
  - receiver 注册，sender 连接.... 无法 bypass kernel
- 发出请求
  - 传统：调用某一个系统调用，讲请求 pend 在 server 的请求队列（内存），通知 server (信号量等)
- 响应请求
  - 传统: server 进程被唤醒，调用某一个系统调用 （查看某一块内存 buffer），响应请求。等待或者休眠。
- 断开链接 (类似建立链接)

IPC 开销

Table 1: One-way IPC latency of seL4. seL4 (4K) will use shared memory. The evaluation is done on a RISC-V U500 FPGA board.

Phases (cycles)	seL4(0B) fast path	seL4(4KB) fast path
Trap	107	110
IPC Logic	212	216
Process Switch	146	211
Restore	199	257
Message Transfer	0	4010
Sum	**664**	**4804**

IPC 开销



(a)

Parts (cycles)	w/o KPTI	w/ KPTI
Privilege Switch	158	690
Process Switch	295	Included above
Others	277	320
Total	**730 ***	**1010**

*The result conforms to the officially reported data (722∼736) [13].

(b)

XPC

目标：一种同步/高效/安全/易用的 IPC 机制

XPC = 用户态进程切换 + 特殊映射传递数据 = 高效 RPC

- 同步：使用新增指令，硬件完成进程切换
- 高效：bypass kernel + no memory copy
- 安全：传递内存，非共享（hand over）
- 易用：兼容同步接口

其他：underbridge，shijuku

uintr

- 同步机制，6bit 信息（用来分辨来源）
- uintr + shmem (DSA?) == RPC ?
    - RPC = 数据传递 + 同步机制

简单设计

- server 注册时，默认注册 64 个内存文件(作为信道)
  - 默认大小？如何调整?
- client 建立链接(uitt) 时:
  - 获取内存文件 fd，mmap 打开
  - 特殊报文可以进行文件属性修改(如 size)。
- 基于报文传递消息
  - 报文格式
  - 有状态？无状态?
  - 半双工？全双工?

问题 / 分析

- locality
  - uintr : no domain switch, 很不错
- latency
  - os switch : sel4 = 600 - 1000 cycles, linux = 2000-3000 cycles
  - XPC xcall: 150 ==> 21 cycles
  - uint : 高效 IPI，linux domain switch 的 1/17
- 支持全双工 / 同步 vs 异步
  - 建立双向连接：异步单工 + 异步单工 = 异步全双工
  - 发送端轮询：半双工 ==》 异步单工 + 同步单工

问题 / 分析

- 链接数量问题：每个线程 64
  - 支持连接数量 = 64 * 线程数？ 64 * 核数？
  - 连接在线程之间的迁移
  - 内核介入?
- 线程 offline:
  - XPC: 同步接口不存在这个问题
  - uintr: 无法很好解决，对应 uintr_wait，依赖于 os 本身的调度。
  - 调整内核调度，实现 uintr_wait 的快速响应?
- shmem 带来的安全问题
  - TOCTTOU，代码注入等

问题 / 分析

- 易用性 / 兼容性 / 软件成本
  - XPC: 兼容同步接口，需要大量内核修改和少量用户库修改。
  - uintr: 较少的内核修改，用户库移植。
- 硬件成本: 无
- 嵌套 RPC
  - XPC：同步嵌套
  - uintr: 等同于多次顺序的 RPC

其他讨论

- 利用 dsa 完成数据传输?
- 异步内核与异步 syscall