IPC

Interaction in a Multiserver Operating System: The Importance of a good RPC Framework

Marcus Brinkmann

The GNU Hurd
A legend in the operating system world

26. Feb 2005 / FOSDEM Brussels

Outline

1. Introduction
   - Inter-Process Communication (IPC)
   - History
2. The Hurd on L4
   - L4
   - The Hurd

Outline

1 Introduction
- Inter-Process Communication (IPC)
- History

2 The Hurd on L4
- L4
- The Hurd

What is it?

IPC is live communication between processes.

- Both processes are active at the time of communication.
- Processes reside in different protection domains.

Not IPC:

- Persistent data.
- Command line.
- Process - Kernel communication.

Partners: One-to-Many

Partners: Many-to-One

Partners: Many-to-Many

Partners: One-to-One

Payload

Different types of payloads:

- Small amounts of data. (parameters)
- Large amounts of data. (memory)
- Access to data. (shared memory)
- Other kernel objects. (capabilities)

Frequency

Overhead:

- Before you can do it: Setup.
- Before you send: Marshalling.
- When you send: Transfer, translation, context switch.
- After you send: Unmarshalling.
- Before you process: Authentication.

→ Consider alternatives. (shared memory)

Overload

Relationship

IPC partners are different. But how different?

- Locality.
- Trust.
- Priority.

Relationship: Locality: Single Node

Single node systems.

Relationship: Locality: Distributed

Network of many nodes.

Relationship: Trust: Symmetric

Mutual trust, equal partners.

Relationship: Trust: Asymmetric

Server - Client.

Outline

1. Introduction
   - Inter-Process Communication (IPC)
   - History
2. The Hurd on L4
   - L4
   - The Hurd

Eniac

One protection domain.

→ No IPC (except I/O, network).

Before Unix

Batch-processing.

- Multiple programs in sequence.
- Output of one is input of next.

→ No IPC.

Simple Time-Sharing.

- Multitasking.
- One program starts others.
- Persistent storage.

→ No IPC.

Alone Together

Unix

More than one protection domain.
Live communication facilities.

- Pipes.
- Sockets.
- Descriptor passing.
- `select()`, `poll()`
- Shared memory.
- SysV IPC.

Unix Critique

- Slow, slow, slow.
- Inflexible.
  - Pipes have small in-kernel buffer.
  - Shared memory requires mutual trust.
- Fragmented. (as opposed to integrated)
- Authentication. (ACL vs capabilities)
- Multiple users? (sockets: yes, else: only cooperative)
- Quality of Service? (SYN flood)

→ Still a lot of isolation.
→ Limited IPC possible.
→ Different needs push for incompatible extensions.

Microkernel

Multiple protection domains.

- Isolation of system services encouraged.

One powerful IPC primitive.

- (+) Efficient, low policy primitives.
- (+) Full integration.
- (-) Uncertain end-to-end cost.

(Some) Capability support in the kernel?

- (+) Authentication.
- (+) Quality of Service.
- (+) Efficient and transparent resource sharing.

Without kernel-level cap support: (+) → (?)

Outline

1. Introduction
   - Inter-Process Communication (IPC)
   - History
2. The Hurd on L4
   - L4
   - The Hurd

L4 IPC

Efficient and powerful IPC primitive.

- Synchronous send operation.
- 64 message registers (MR).
- String buffer support with scatter/gather (up to 4MB).
- Recursive map and grant operations.
- ...

Critique:

- (-) Sender thread ID is exposed.
- (-) DoS attacks on open listeners.
- (-) No low-level support to grant or revoke access.

Outline

1. Introduction
   - Inter-Process Communication (IPC)
   - History
2. The Hurd on L4
   - L4
   - The Hurd

The Hurd

Remote procedure call (RPC):

- Synchronous send and receive.
- Client side: Function call.
- Server side: Function implementation.
- Object orientation. (capabilities)
- Cancellation support.
- Mental picture: Thread migration (but watch out!).

Notifications:

- Asynchronous event delivery.
- Client wants to be notified by events in the future.
- Server creates events and needs to notify clients.
- Mental picture: Signals (but watch out!).

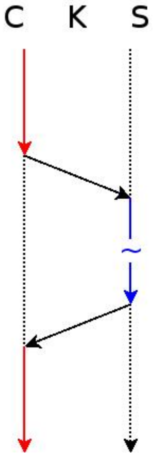
Capabilities

Capabilities give access rights to server-provided objects. RPCs are invoked on capabilities.

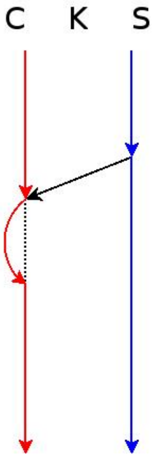
- Managed in server and client.
- → Copying caps expensive. (three-way protocol)
- → Servers must not hold caps on behalf of untrusted clients.

Example: File lookup.

RPC



Notifications



Ex: `select()`

Notifications?

Polling.

- One thread per server to poll (but potentially many objects).
- Block until event occurs.
- Server queues events until they are polled.

Real notifications.

- Client registers notify handler thread.
- Server sends notifications to notify handler thread.
- Server queues events (and retry!) until the client is ready.

System service.

- Trusted system server.
- Client tells that about allowed servers.
- Servers send notifications to service.
- Notification service queues events (in user memory).
- Client polls.

Cap Library

Server part:

- Buckets.
- Objects.
- Classes.
- Capabilities.
- Clients.

- Inhibition.
- Continuations.

Summary

- Even simple things can be hard to get right.
- Outlook
  - We need to write more code.
  - Notifications?
  - Capability support in the kernel?

Legal Stuff

Copyright 2005 by Marcus Brinkmann <marcus@gnu.org>. Verbatim copying and distribution of this entire document is permitted in any medium, provided this notice is preserved.

Sources:

- http://www.archiphoto.com/Images/personal/Ventimiglia
- http://usinfo.state.gov/usa/civilrights/images/mow01-440.jpg
- http://www.afmsteno.com/castenets.jpg
- http://www.ci.minneapolis.mn.us/graffiti/images/index-2.gif
- http://cvnweb.bai.ne.jp/ preston/threedee/gallery/images/overload.jpg
- http://www.ne.jp/asahi/chaleaf/shop/chatomjomhall1.jpg
- http://www.portlandtribune.com/pphoto/pphotogaynew/images/-C.GayMarriage030304LB7.jpg
- http://www.winlockmarbles.com/images/wmkeeper03big.jpg
- http://www.winlockmarbles.com/images/wmkeeper05big.jpg
- http://news.bbc.co.uk/olmedia/1020000/images/_1022809_post300.jpg
- http://www.nordhorncorp.com/images/nostalgia/eniac.jpg