CS510 Languages and Low Level Programming: Portfolio submission, Topic 6

Due on May 13, 2016 at 11:59pm

Mark P. Jones Spring 2016

Konstantin Macarenco

Topic 6.

Describe the motivation, implementation, and application of microkernel abstractions for managing address spaces, threads, and interprocess communication (IPC).

Motivation

Idea behind microkernel is to provide "policy free" environment, with no assumptions about services it will be running. All functionality of a traditional kernel is moved out, and implemented as a set of independent services. Microkernel provides only minimum set of abstraction over hardware, such as

- Address Spacing
- Inter-process Communication (IPC)
- Thread Management
- Unique Identifiers

Systems that use this approach will implement only the functionality needs to operate, have smaller footprint. All nonessential for the kernel functions, like network support, are implemented as services. Each service has it's own address space, communication with other services allowed only through Only kernel functionality is in privileged mode (ring 0).

Advantages

- Robustness, a service crash doesn't affect any other part of the system.
- Easier maintenance.
- Enforces modular structure.
- Small size of Trusted Computing Base.

Disadvantages

- Performance loss due to high overhead of services interaction.
- Larger memory footprint.
- Complicated process management.

Earlier versions of microkernel implementation suffered from high IPC overhead and did not look promising. L3/L4 (Jochen Liedtke) implementation proved that microkernel can be successful. Liedtke IPC design was 20 times faster than Mach's (fist generation of microkernel).

Implementation

History

Mach

L3

L4

Implementation details

1. **Thread Management**
   Kernel Information Page (kernel version, host system info, address space layout info, system call entry points)
   Predefined KIP adddress
   Slow system call LOCK NOP → illegal opcode exception → kernel → load KIP address to context registers → return to user mode

and User Thread COntorl Block (One UTCB for eac thread) in the address space Message registers (MRs), and thread control registers (TCRs)
all UTCBs are in UTCB area
64 message registers $MR_0 - MR_63$
Miscellaneous fields: error code, exceptin handler pager acceptor
UTCB address points to the middle of the UTCB

User processes can read/write data to any UTCB available
thread priority is stored in protected structure, any data read form UTCB cannot be trusted and must be validated by the kernel before use.
UTCB mappings created by kernel

2. Address Spacing
3. Inter-process Communication (IPC)

Application