TECHNISCHE UNIVERSITÄT DRESDEN

Dept. of Computer Science, Institute of System Architecture, Operating Systems Group

Capability Wrangling Made Easy: Debugging on a Microkernel with Valgrind

Aaron Pohle, Björn Döbel, Michael Roitzsch, Hermann Härtig

Technische Universität Dresden, Germany

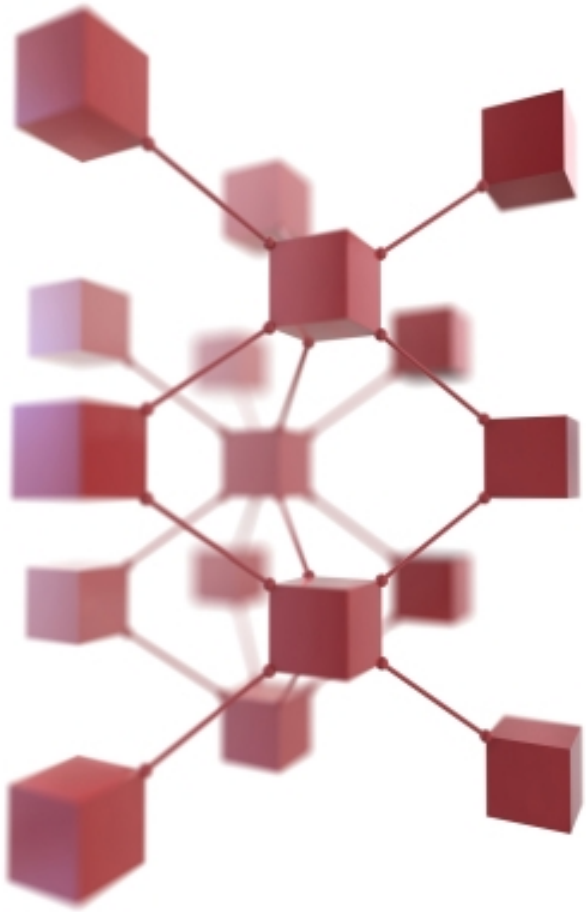
Pittsburgh, 2010-03-17

Programmers Make Mistakes

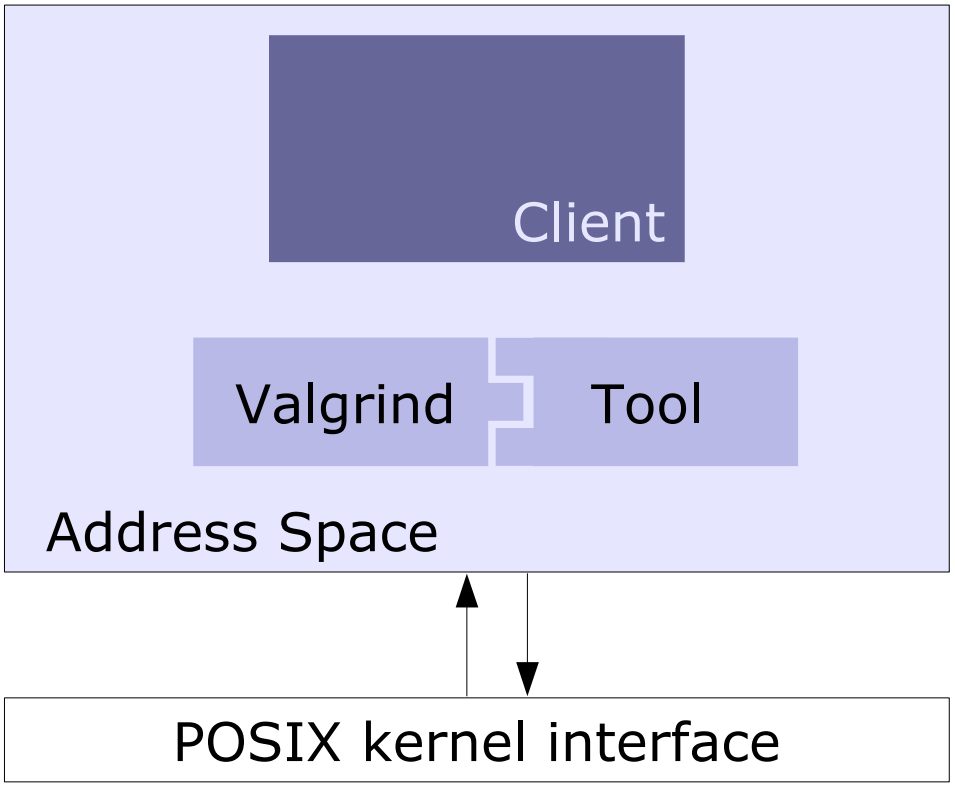
```
void *grow_heap(unsigned size)
{
    int idx       = alloc_capability();
    mem_area *mem = mem_alloc(size, idx);

    return mem->addr;
}


void shrink_heap(void *addr)
{
    mem_free(addr);
}
```

Outline

- Valgrind and Fiasco.OC
- Porting challenges
- CapCheck leak detector

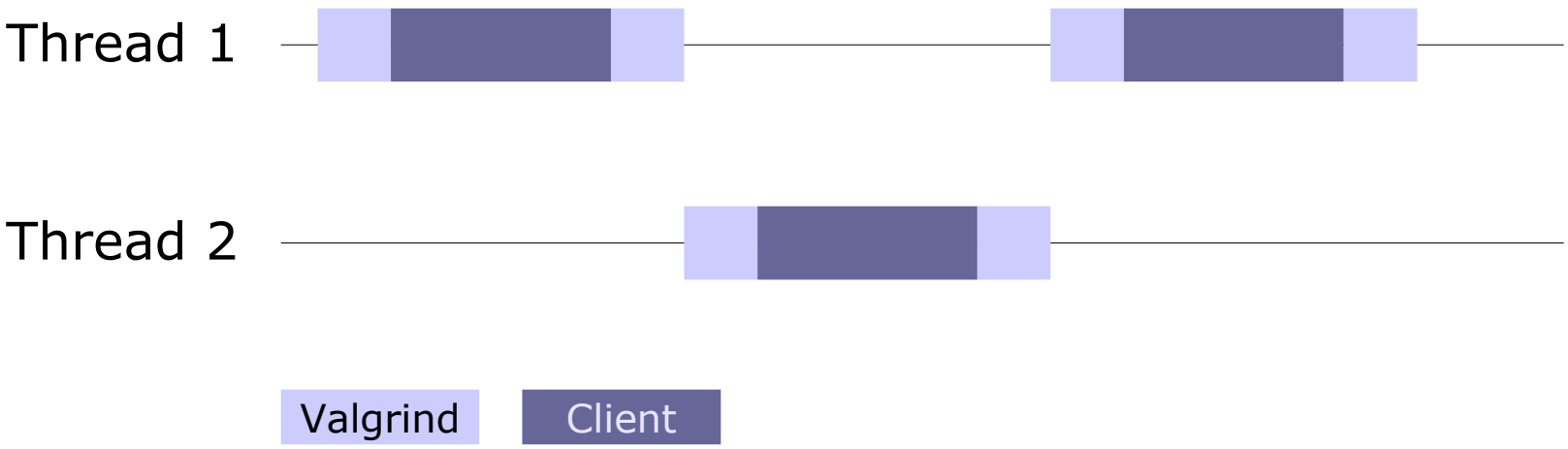


Valgrind: Binary Instrumentation

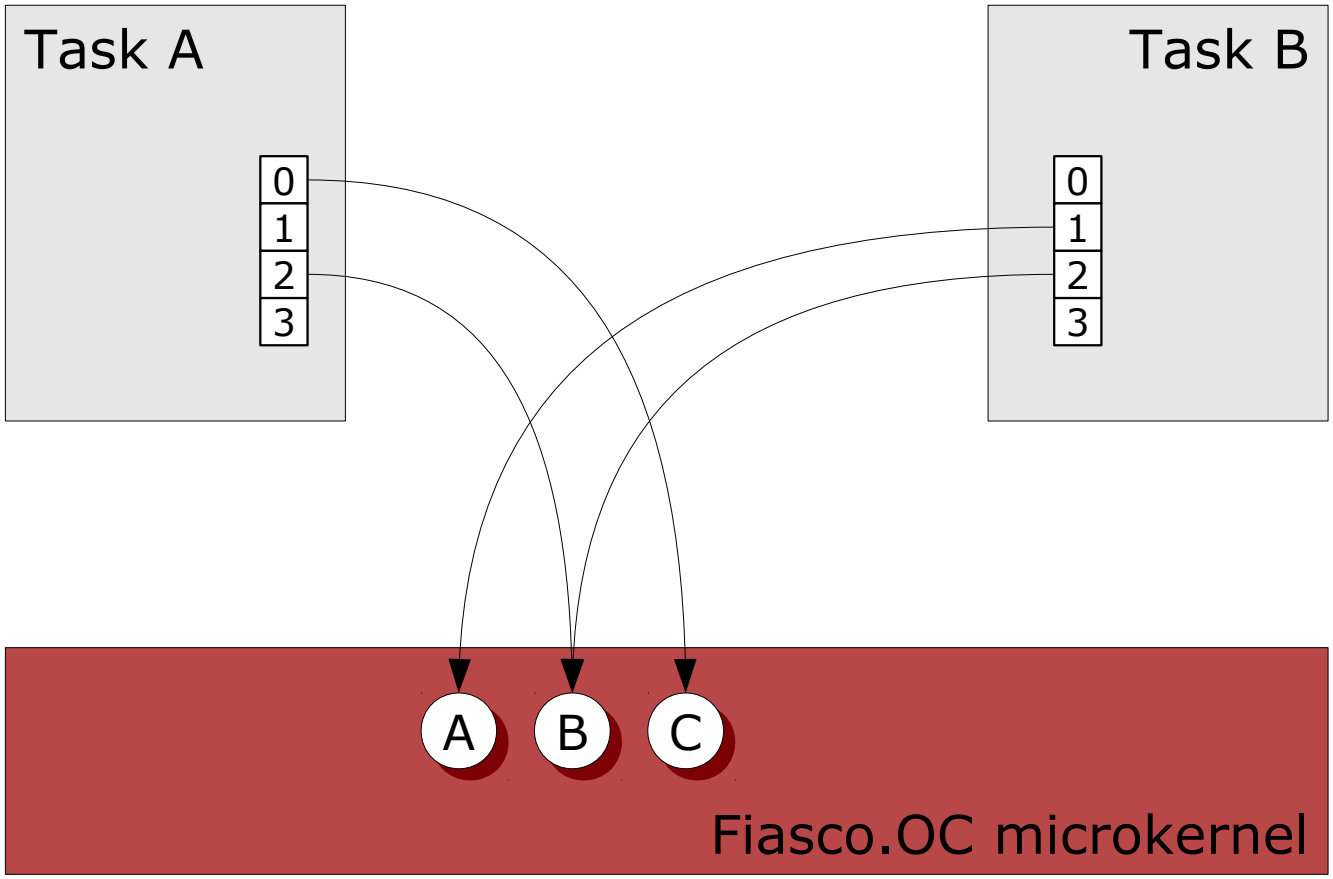


Valgrind: Complex Tools

- Shadow values
- Consistency requirement:
  Basic blocks must be atomic.



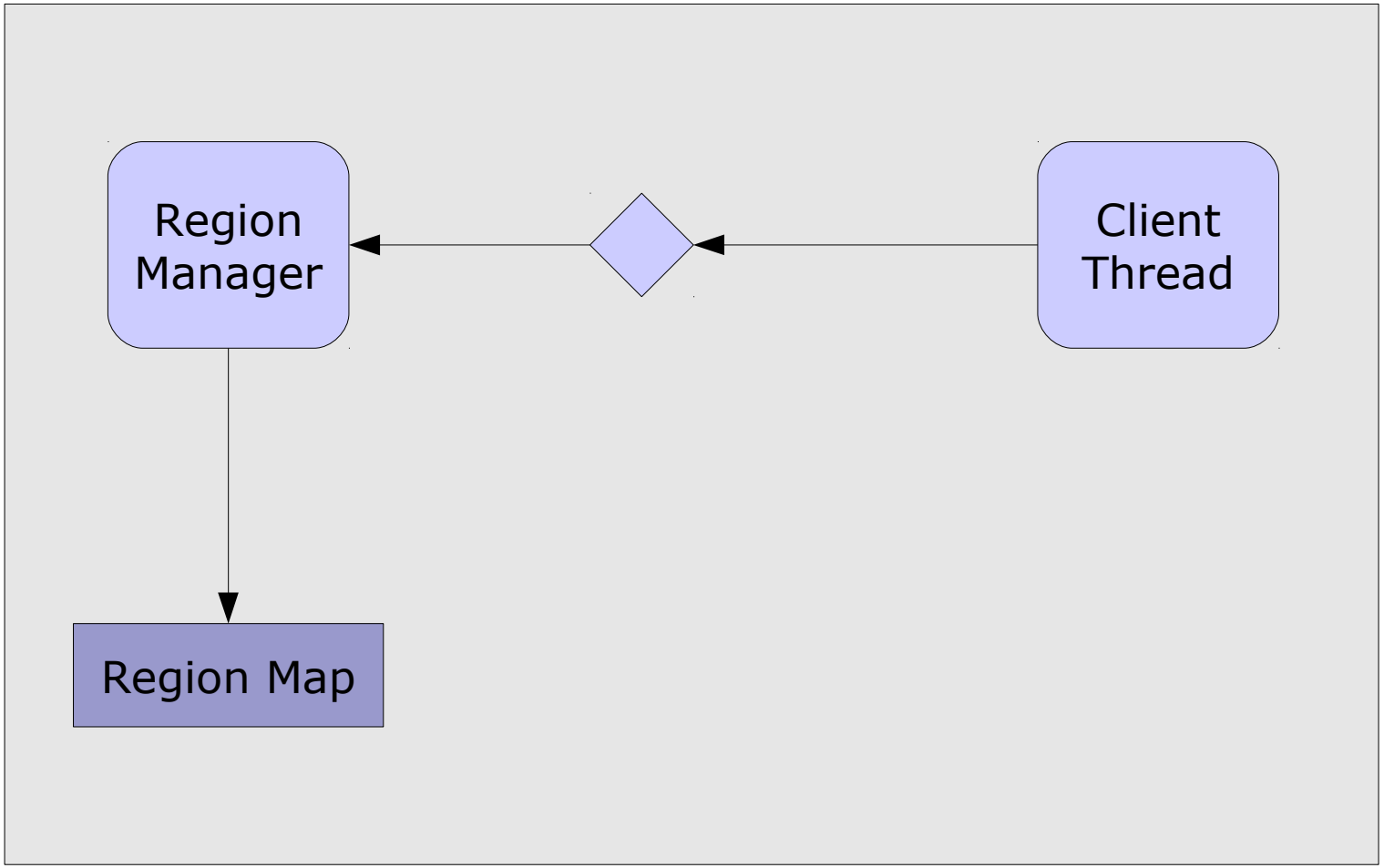
Fiasco.OC – Capabilities



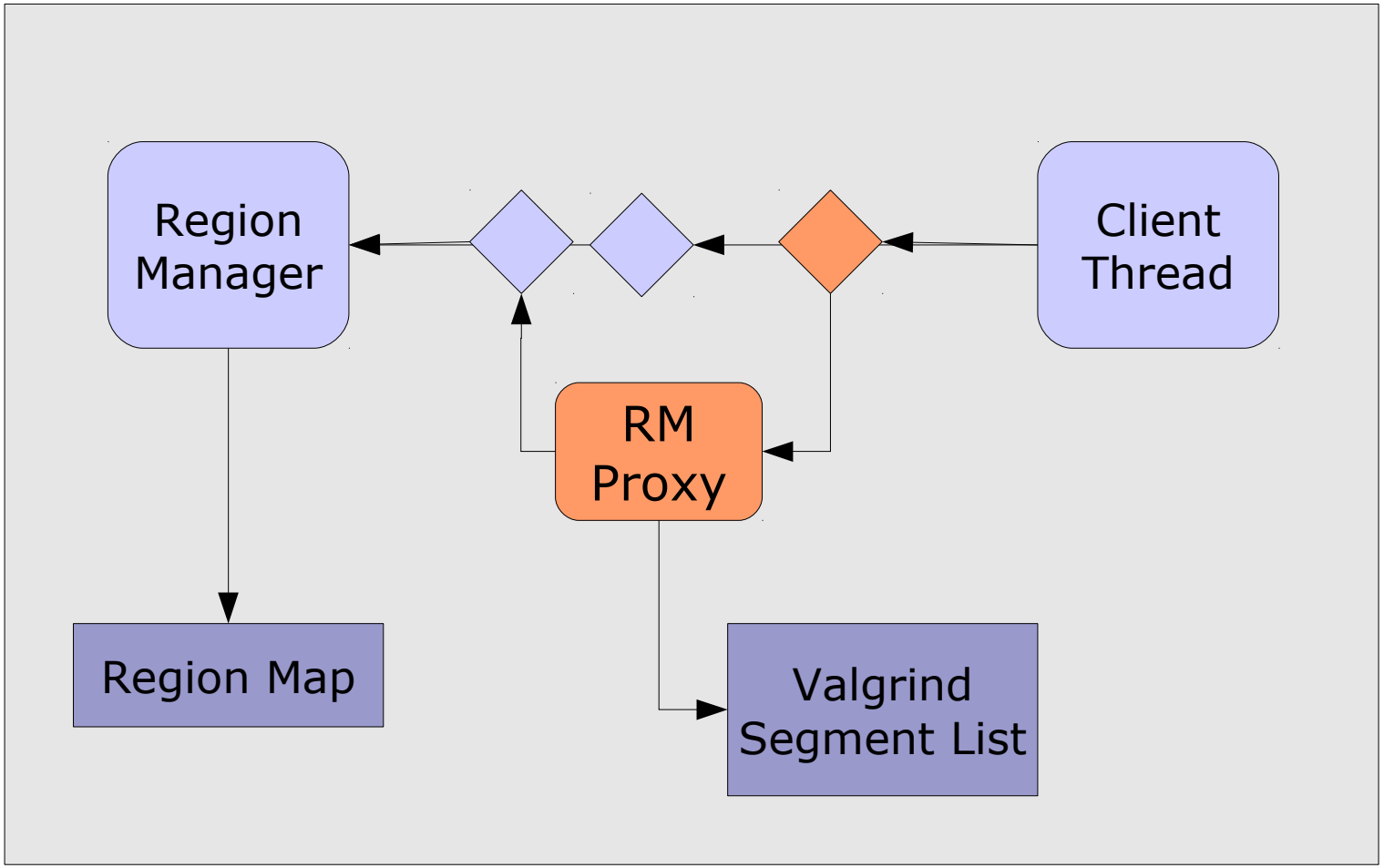
Porting Valgrind to Fiasco.OC

- POSIX environment
- Threads
- User-level thread control block (UTCB)
  - Carries system call payload
  - Need one for each thread role
- User-level memory management

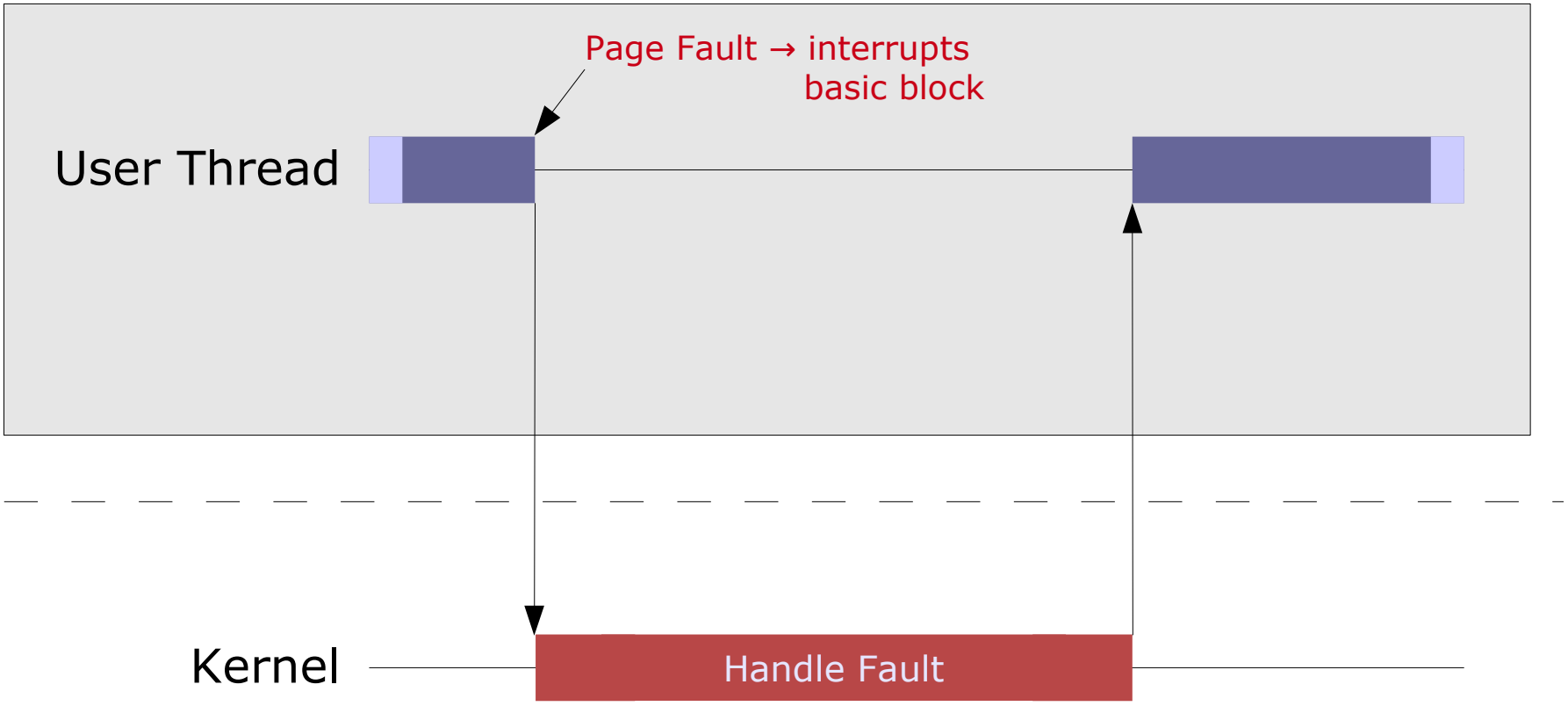
Fiasco.OC – Memory Management



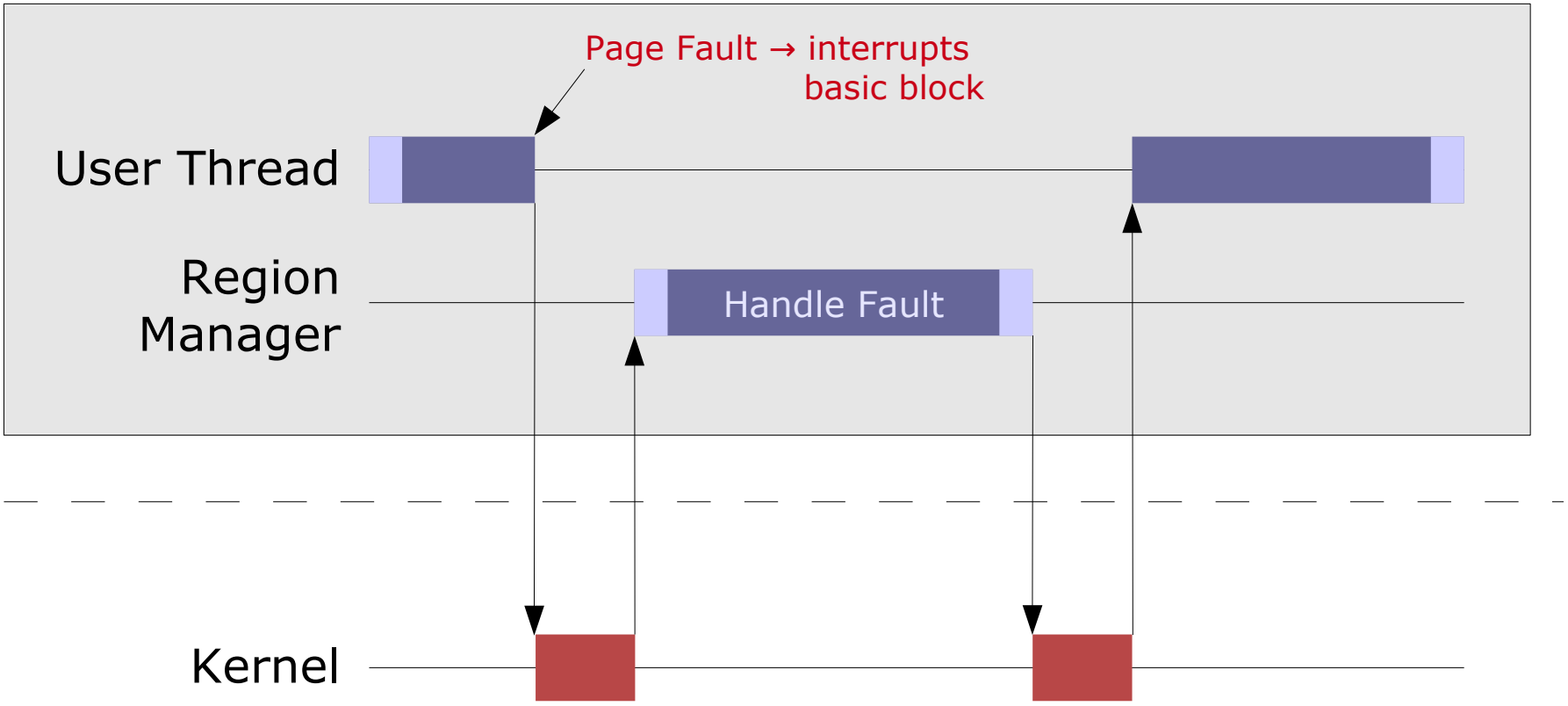
Fiasco.OC – Memory Management



Page Fault Handling (Linux)



Page Fault Handling (Fiasco.OC)



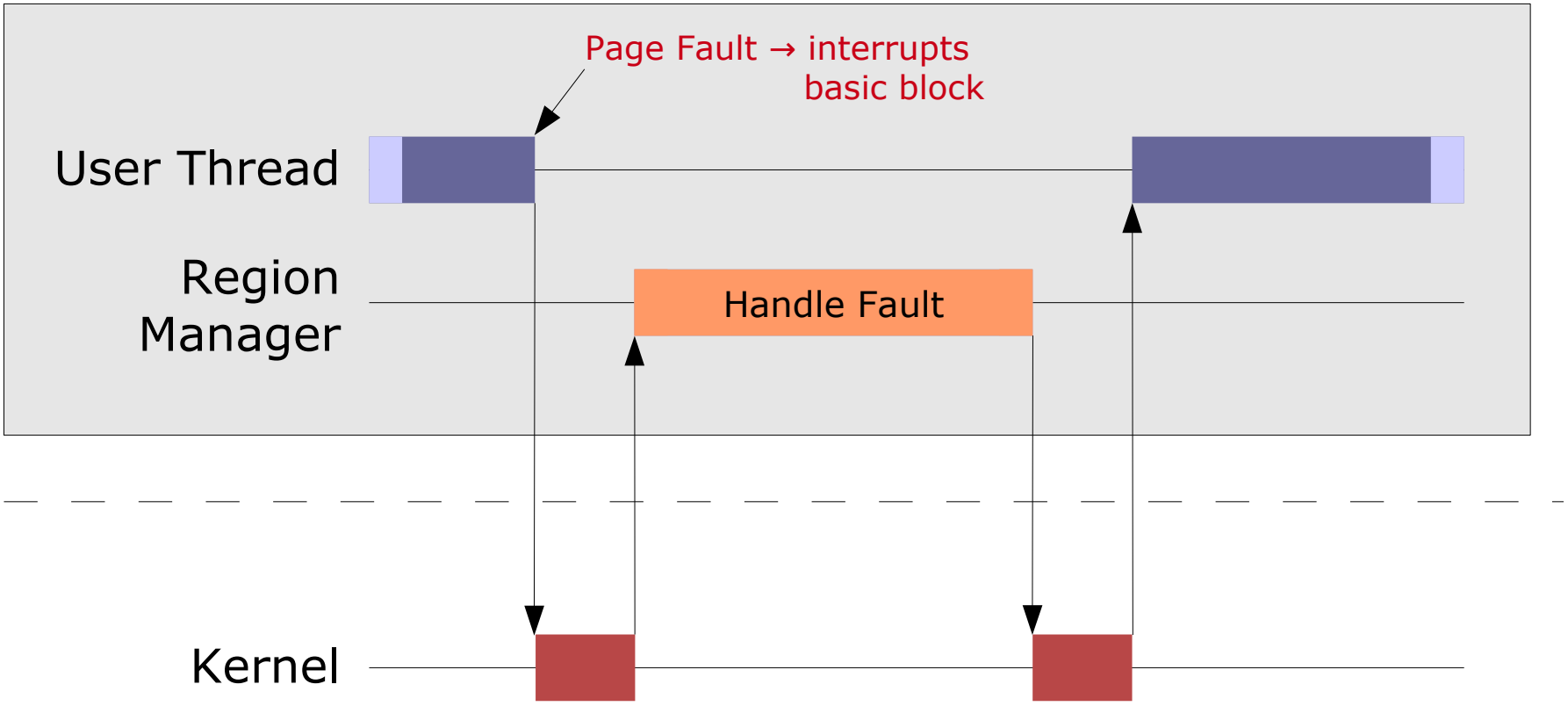
Solving the Parallelism Problem

Two basic blocks may execute in parallel.

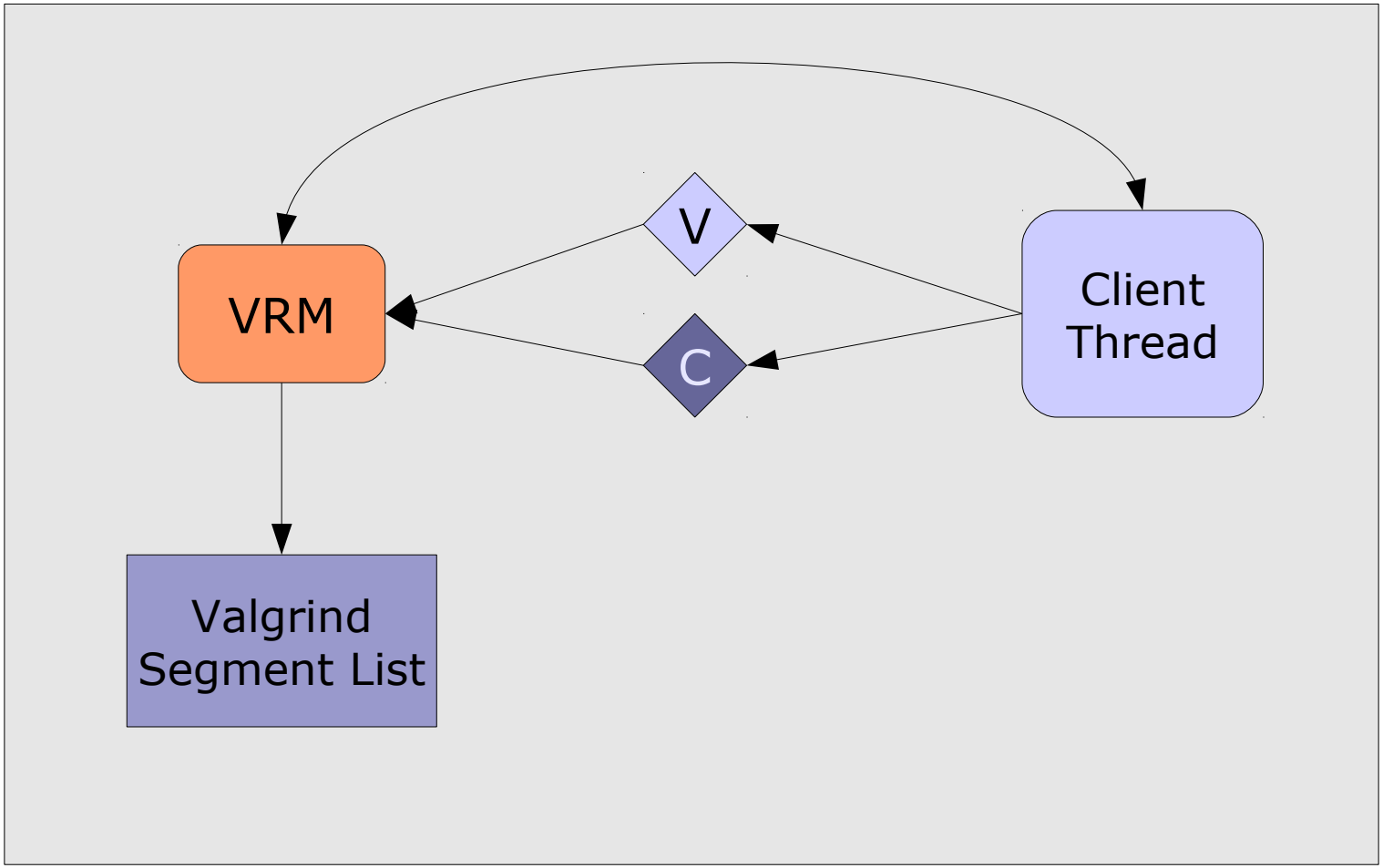
Potential solutions:

- Eliminate atomicity assumption
- Checkpoint & restart for basic blocks
- Eliminate special case

Eliminate special case



Virtual Region Manager



Capability bugs

- User-level slot management
  - *Capability leakage*

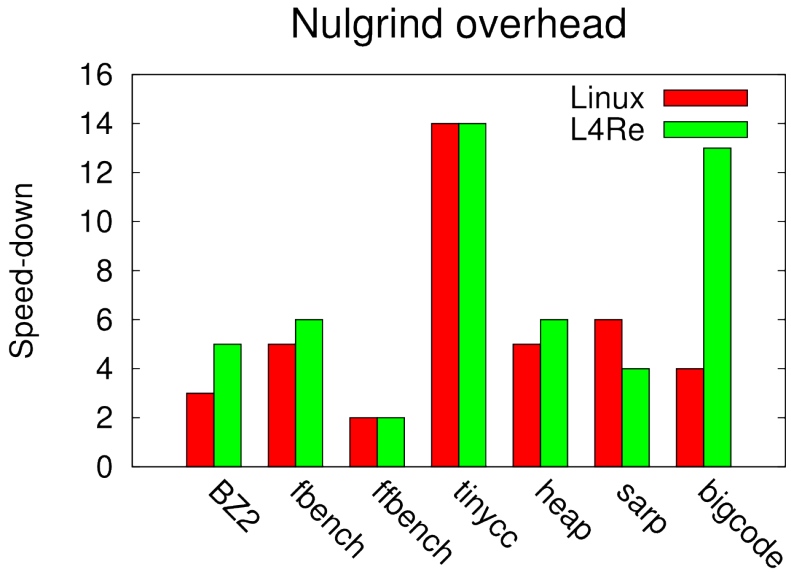
- Advanced feature: capability overmap
  - Optimization
  - Error

CapCheck

- Track CAP_ALLOC / CAP_FREE events
  - Cap alloc stack trace
- Track capability mappings
  - Map stack trace
- Track capability invocations
  - Protocol ID
  - Detect mismatches

Evaluation

LibC wrappers	~ 400 LoC
Binary translator	13 LoC
System call handling	~ 200 LoC
Virtual Region Manager	~ 400 LoC
CapCheck tool	~ 200 LoC



Summary

- Valgrind (and tools) running on Fiasco.OC
- Memory management issues
  - Virtual region manager
- CapCheck tool for
  - Detecting capability leakage
  - Detecting capability overmap

Lessons Learned

- Moving POSIX kernel features to user space
- Capabilities aid flexibility.

Valgrind Tools

- MemCheck
  - Memory leak detector
- Helgrind
  - Thread checker / race detector
- CacheGrind
  - Cache profiler
- Massif
  - Heap profiler

- Chronicle-Recorder
  - Memory tracer (in the works)

Non-POSIX Difficulties: Files

- Common in Valgrind core:

```
NSegment *s = VG_(lookup_nsegment)(addr);

int fd = open(filename, ...)

/* use segment s */
```

- Problem: only works, if nsegment array stays constant
  - L4Re's open() may establish a new memory mapping → modifies nsegment array

Valgrind vs. Fiasco.OC Assumptions

(1) There is exactly *one pager per thread*.

(2) There is exactly *one region manager per task*.

(3) Basic blocks are executed *atomically*.