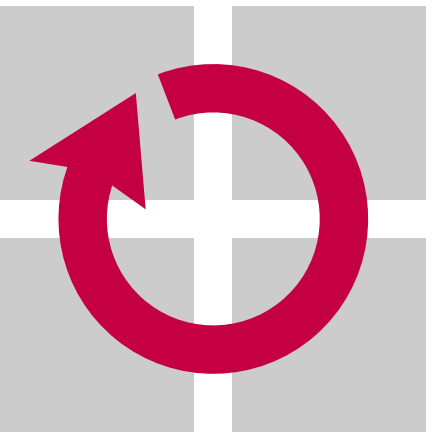
Beyond Address Spaces -

Flexibility, Performance, Protection, and Resource Management in the Type-Safe

JX Operating System

Michael Golm



University of Erlangen-Nuremberg
Computer Science Department 4
(Operating Systems and Distributed Systems)
Martensstraße 1
91058 Erlangen, Germany

Overview

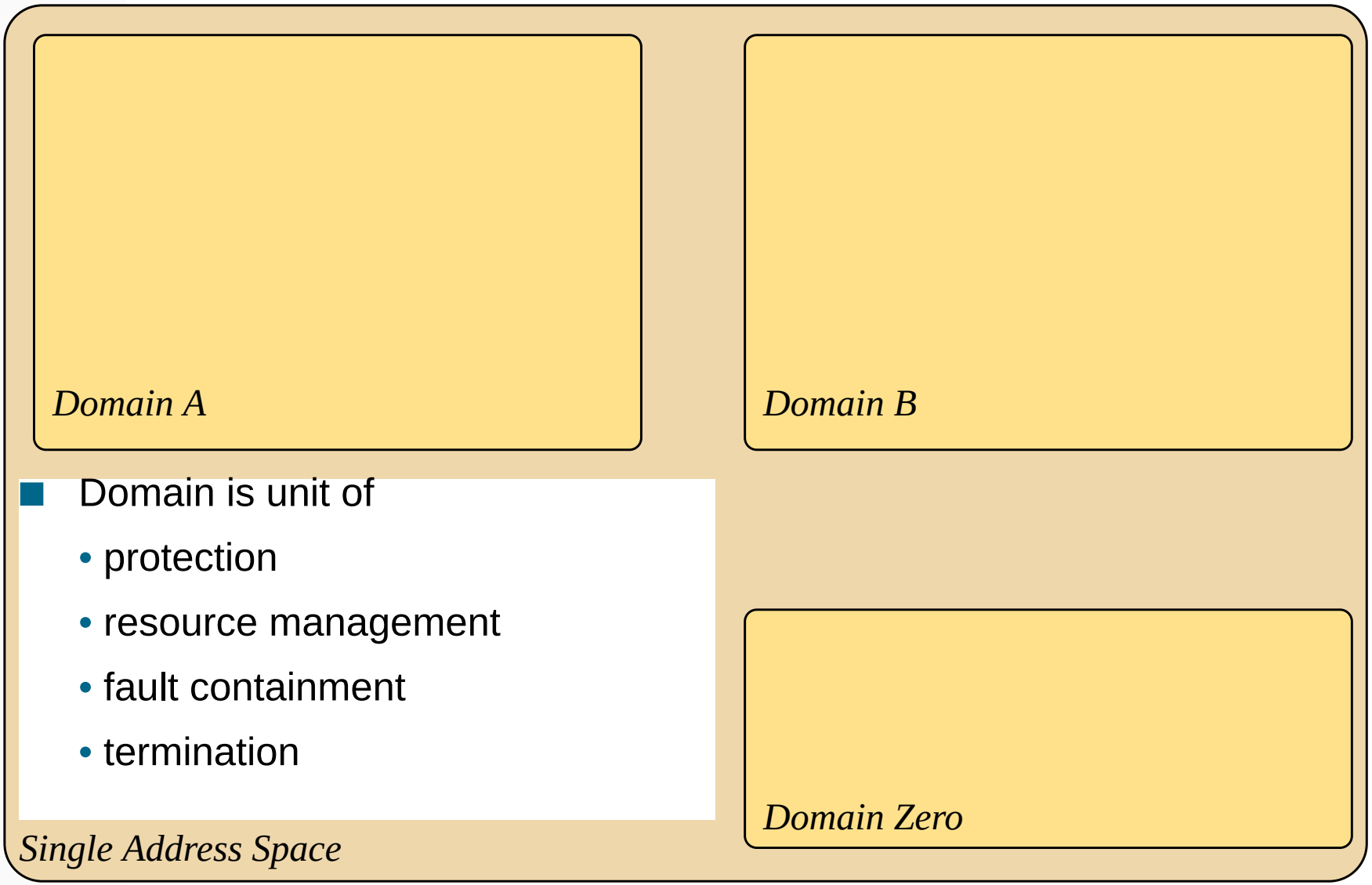
- Objectives
- Architecture
- Protection
- Flexibility
- Performance

Objectives of the JX system

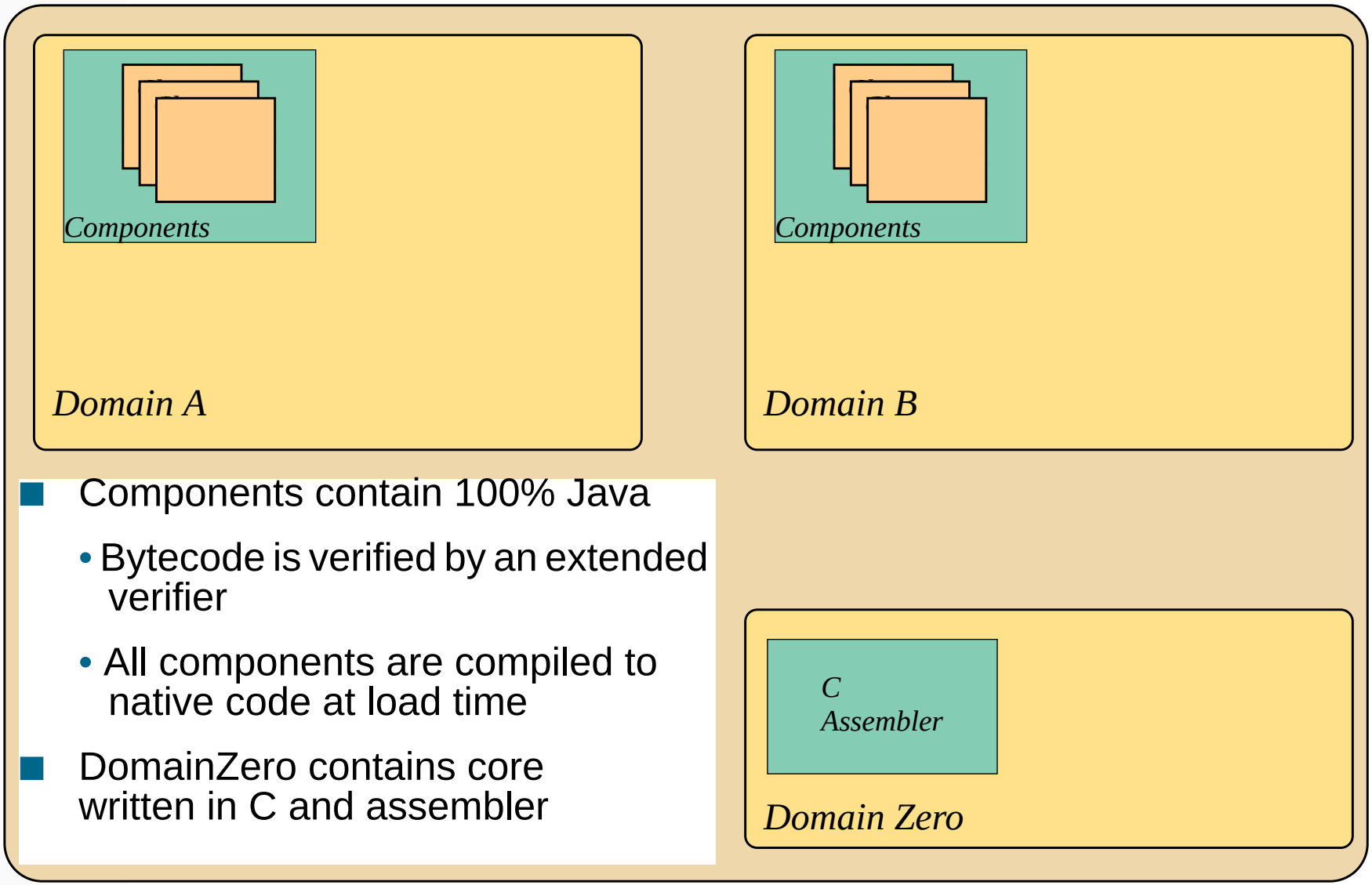
- Dynamic OS extension with untrusted components
- Code reuse
  - tailored OS configurations
- Exact resource accounting and customizable management
- Protection
  - flexibility
  - performance
  - robustness

Single Address Space

Domains

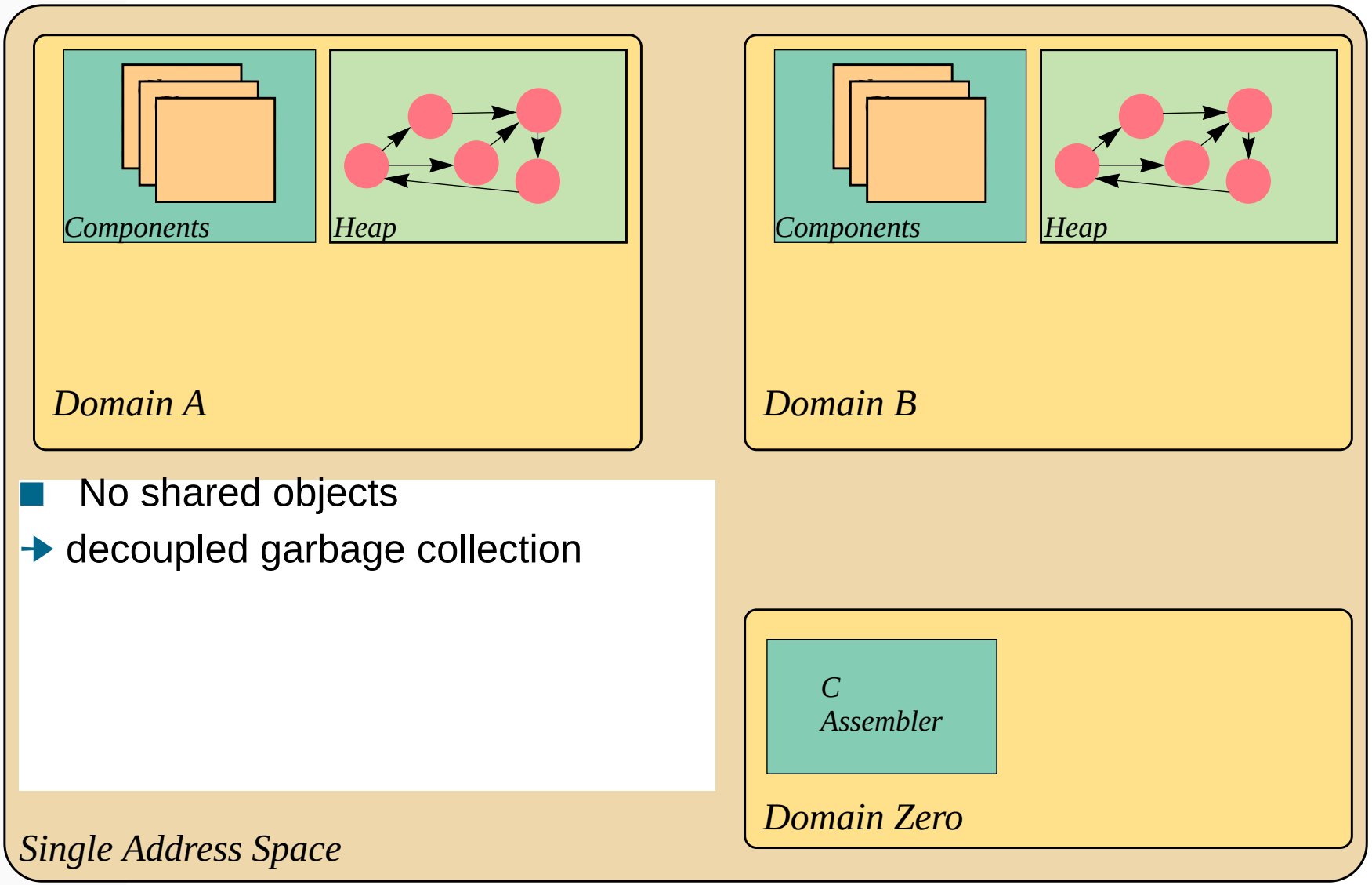


Components



- Components contain 100% Java
  - Bytecode is verified by an extended verifier
  - All components are compiled to native code at load time
- DomainZero contains core written in C and assembler

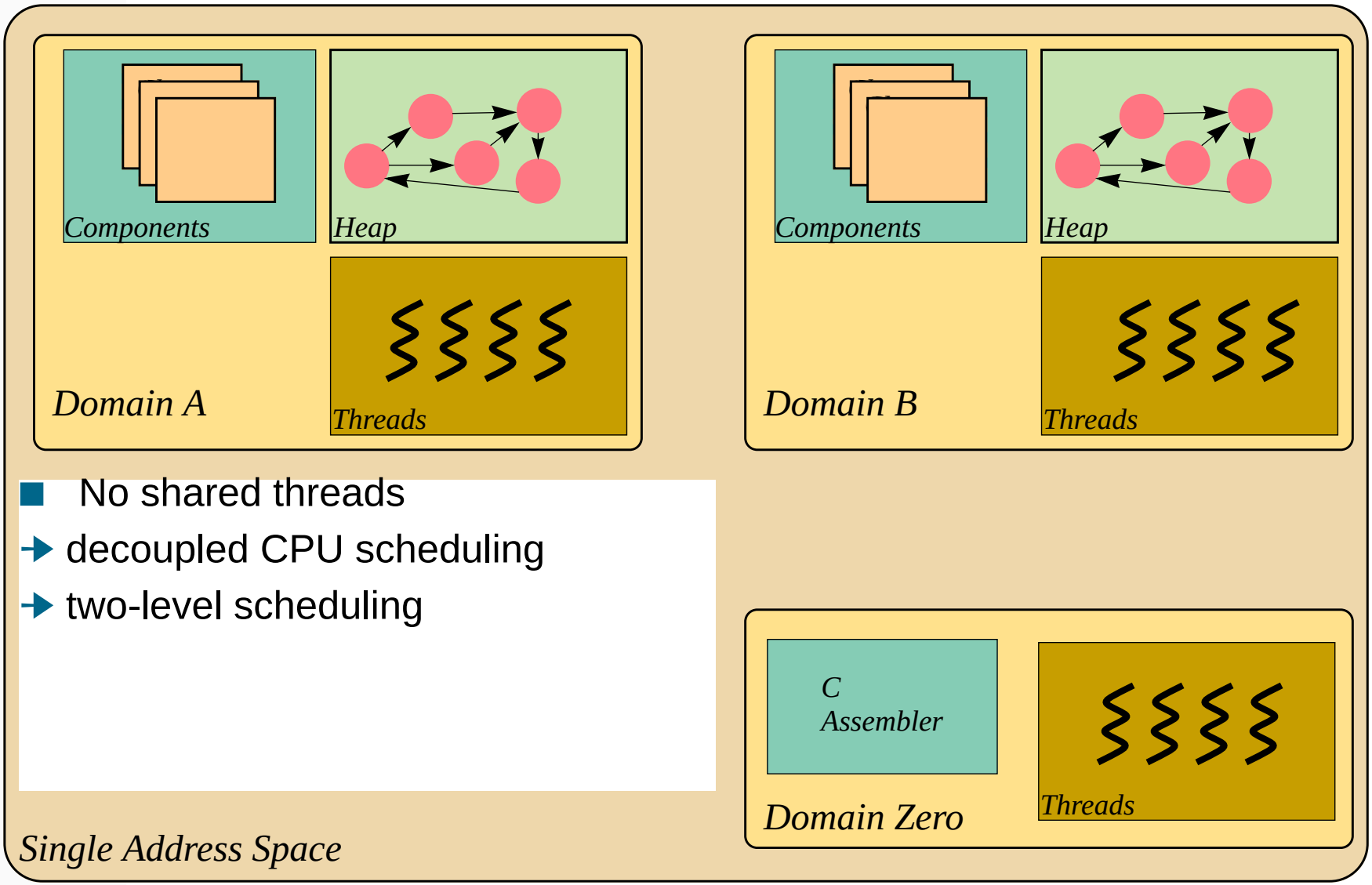
Objects & Heap



- No shared objects

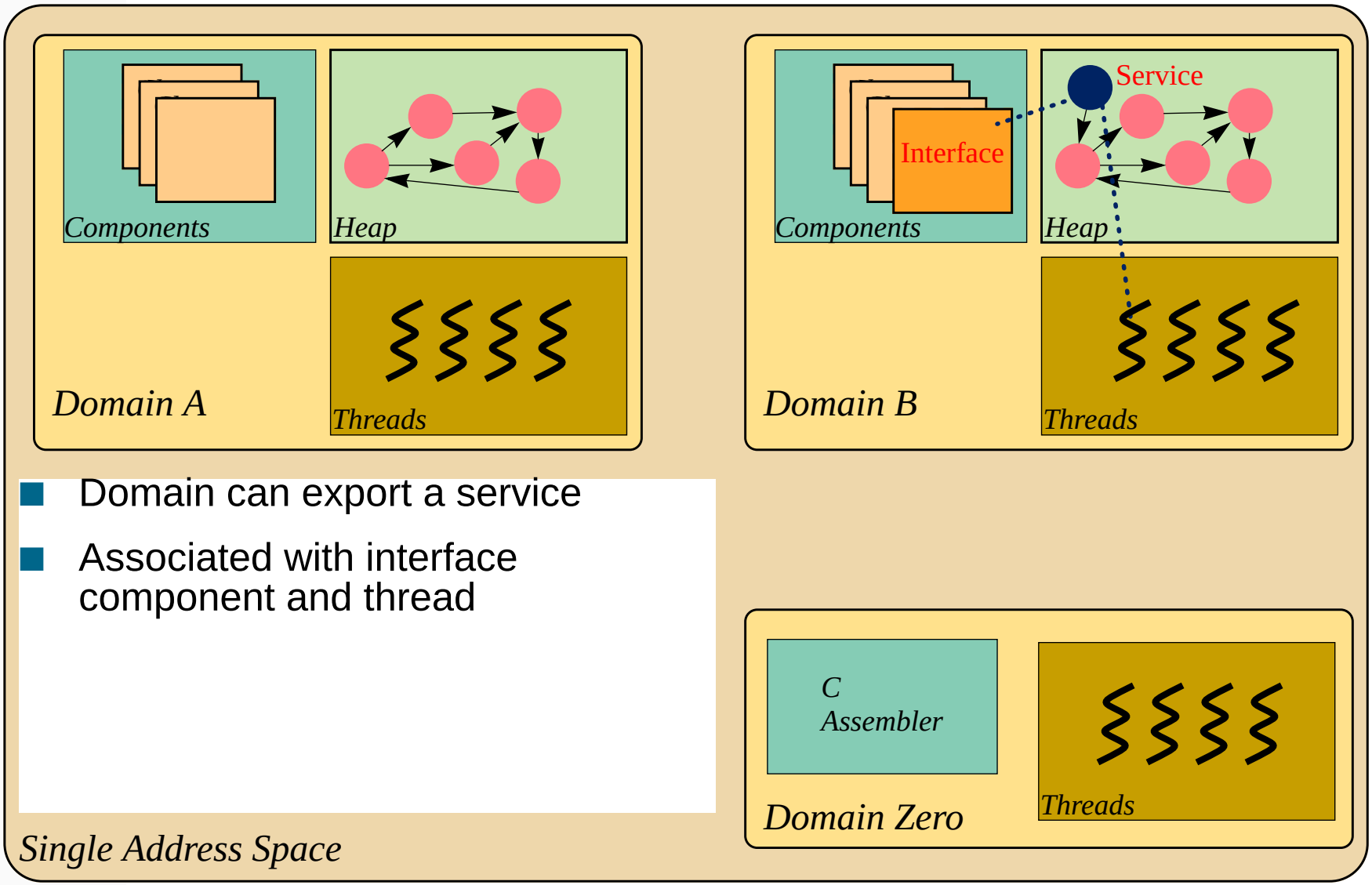
→ decoupled garbage collection

Threads



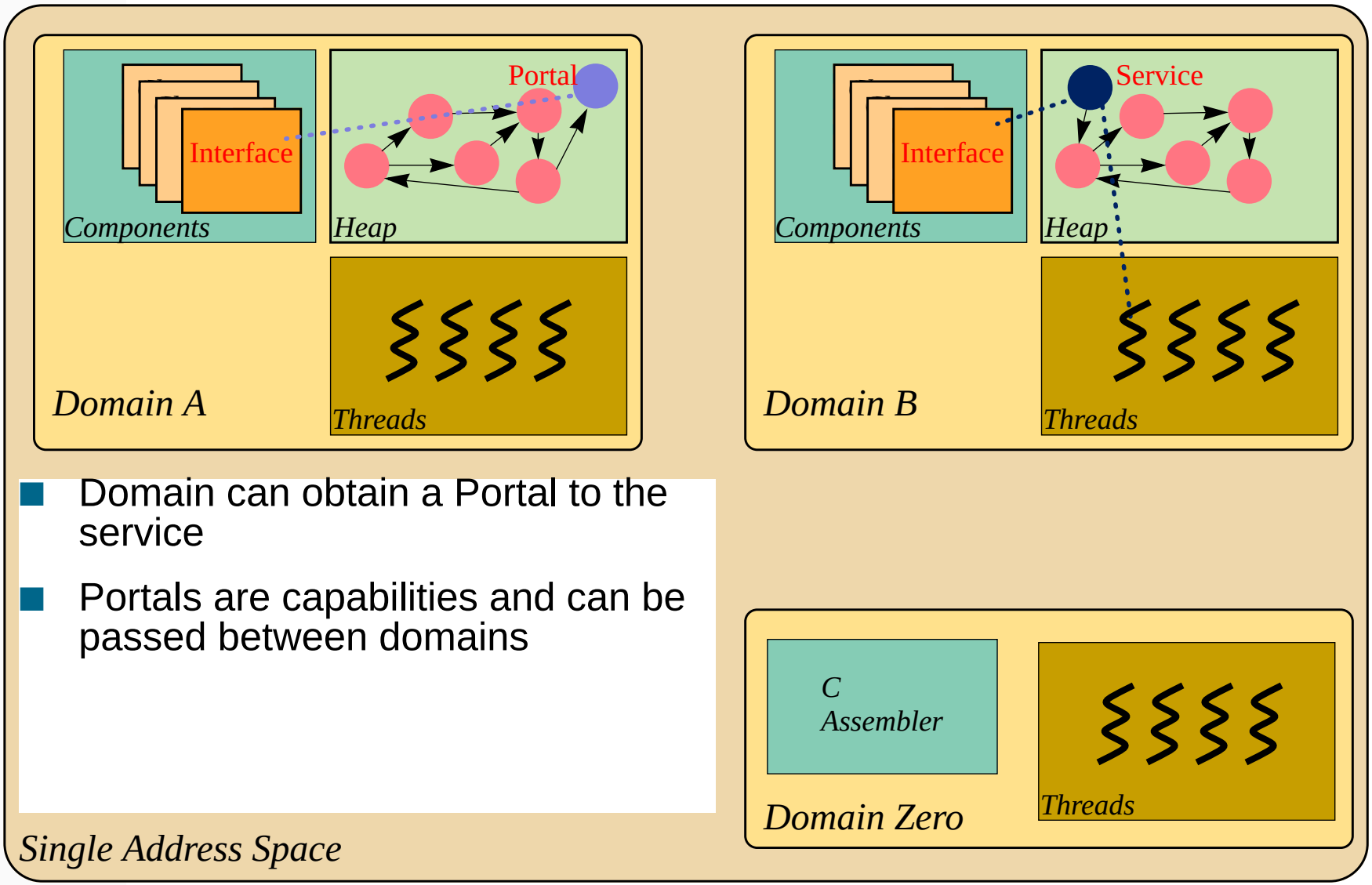
- No shared threads
  - → decoupled CPU scheduling
  - → two-level scheduling

Communication: Portals



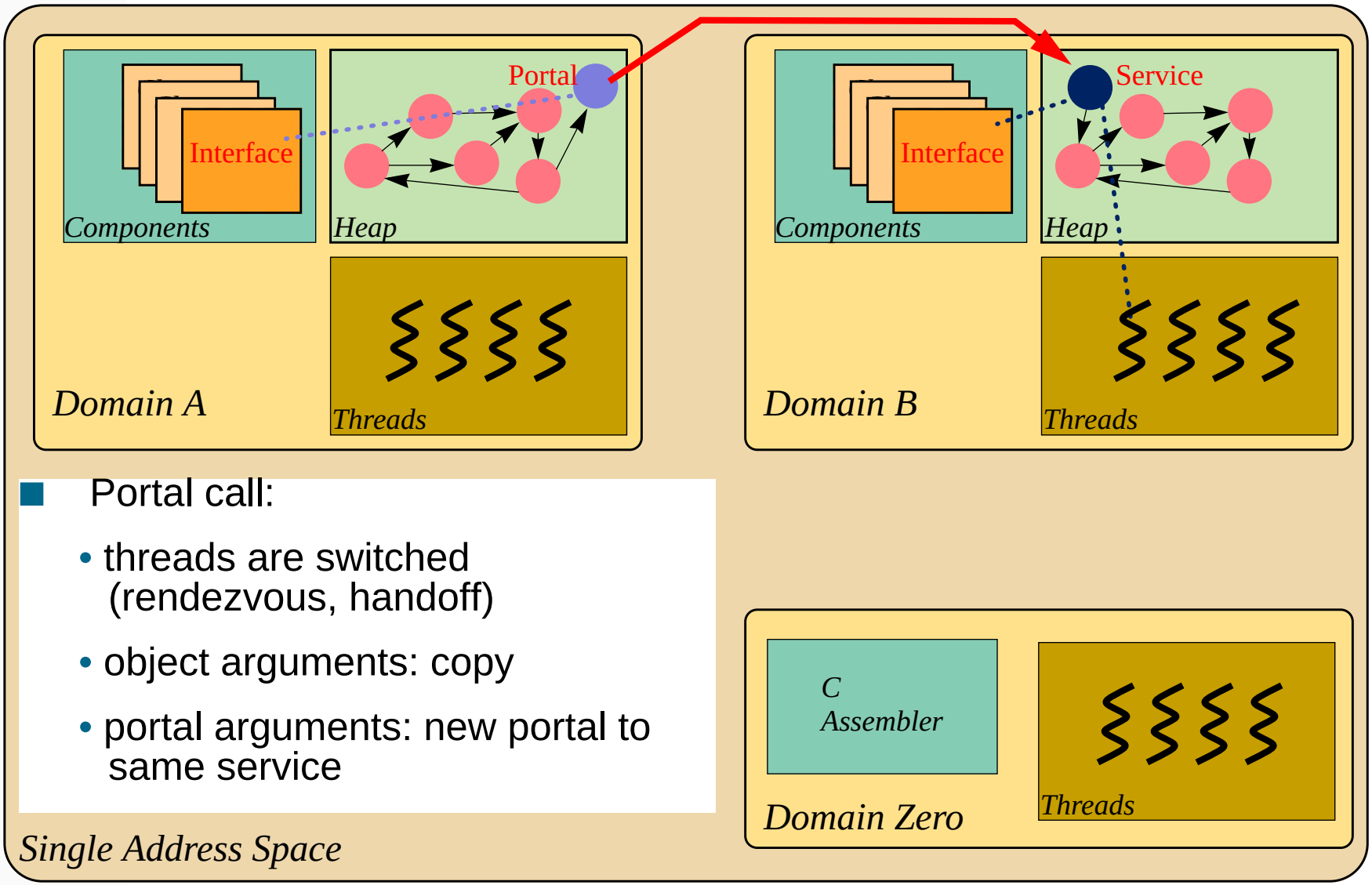
- Domain can export a service
- Associated with interface component and thread

Communication: Portals



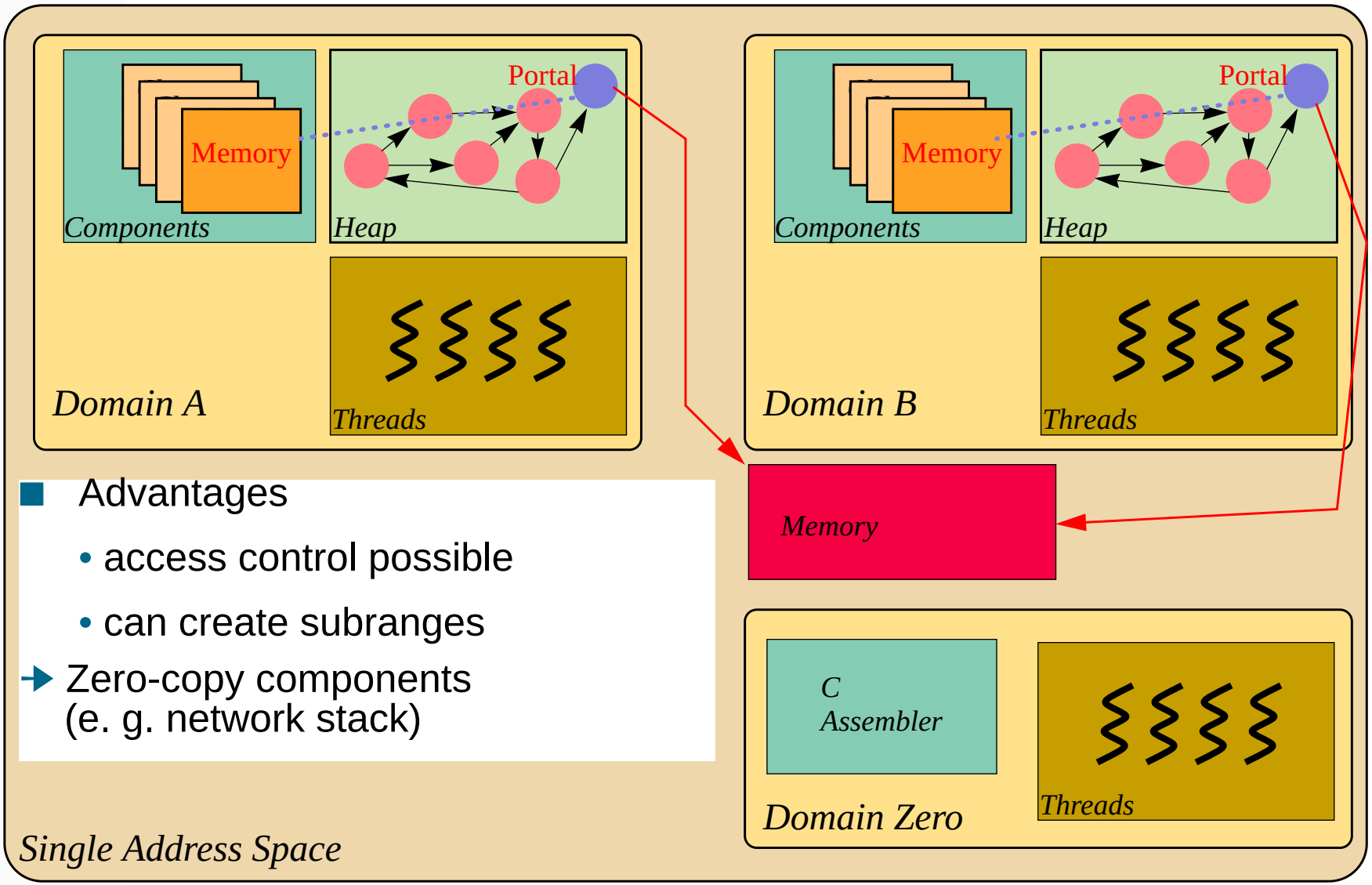
- Domain can obtain a Portal to the service
- Portals are capabilities and can be passed between domains

Communication: Portals



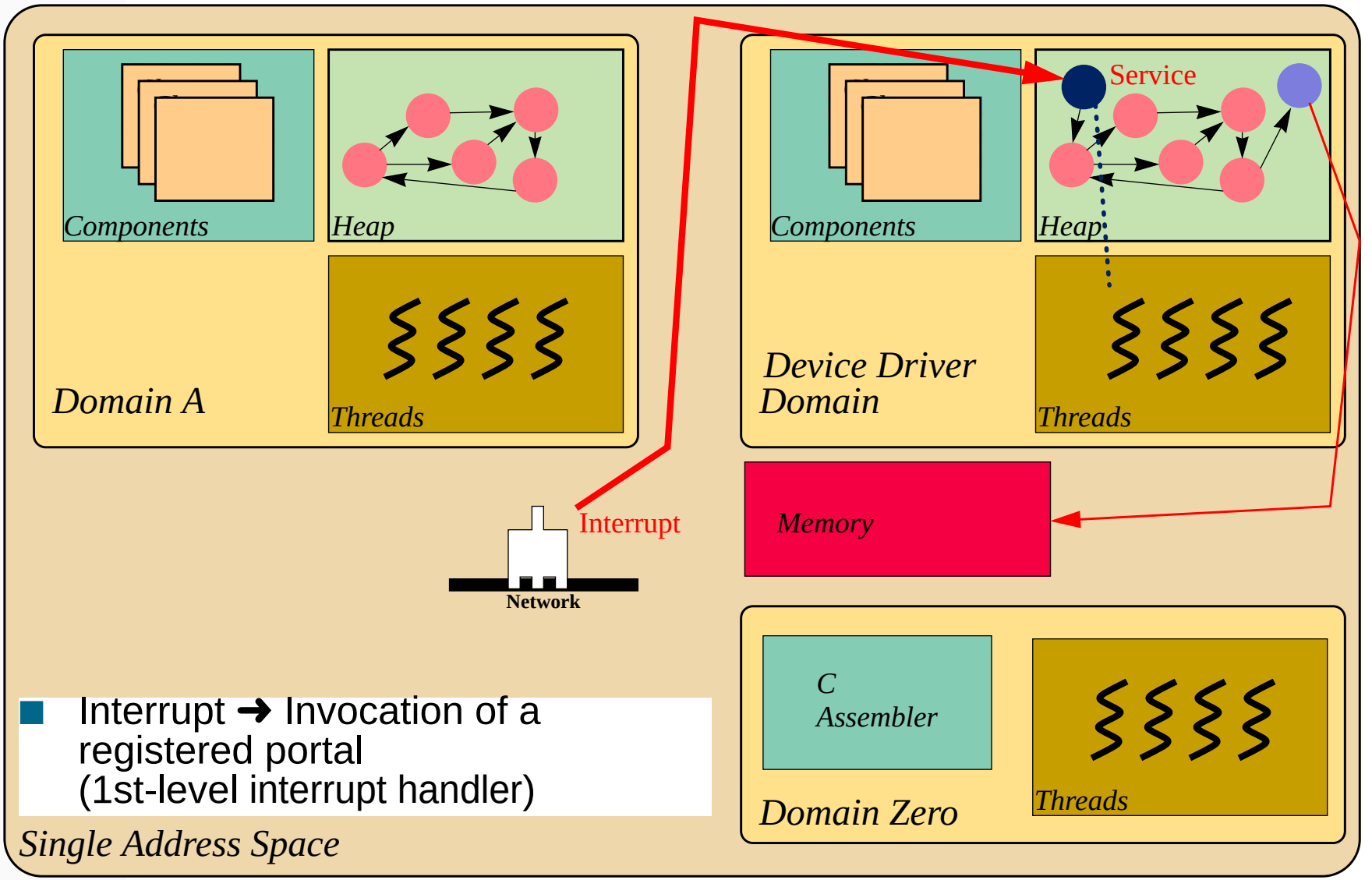
- Portal call:
  - threads are switched (rendezvous, handoff)
  - object arguments: copy
  - portal arguments: new portal to same service

Communication: Memory



- Advantages
  - access control possible
  - can create subranges
- → Zero-copy components (e. g. network stack)

Device Driver



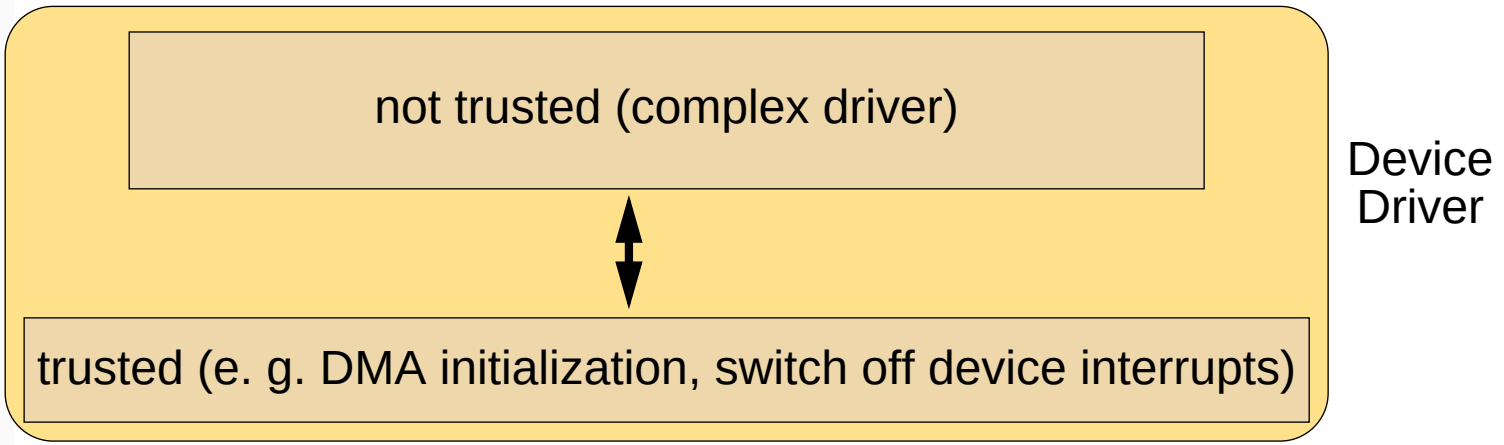
- Interrupt ➜ Invocation of a registered portal (1st-level interrupt handler)

Protection: Interrupt Handler

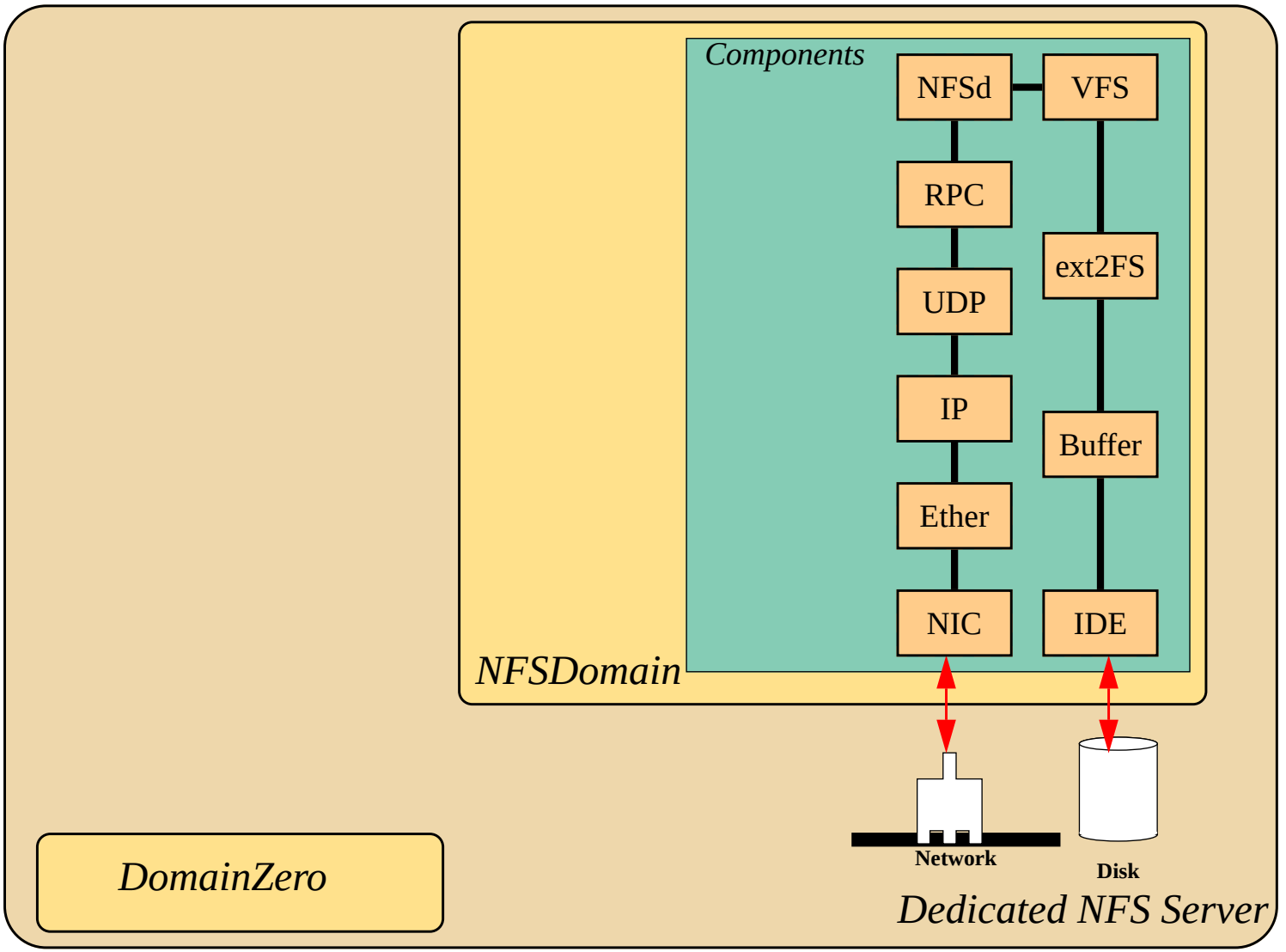
- Interrupts on the interrupted CPU are blocked during execution of the interrupt handler
- Verifier checks interrupt handler for upper limit of execution time
  - can insert runtime checks to ensure timely termination
  - runtime check can terminate interrupt handler and initiate counter measure (e. g., switch off device interrupts)

Protection: Device Driver

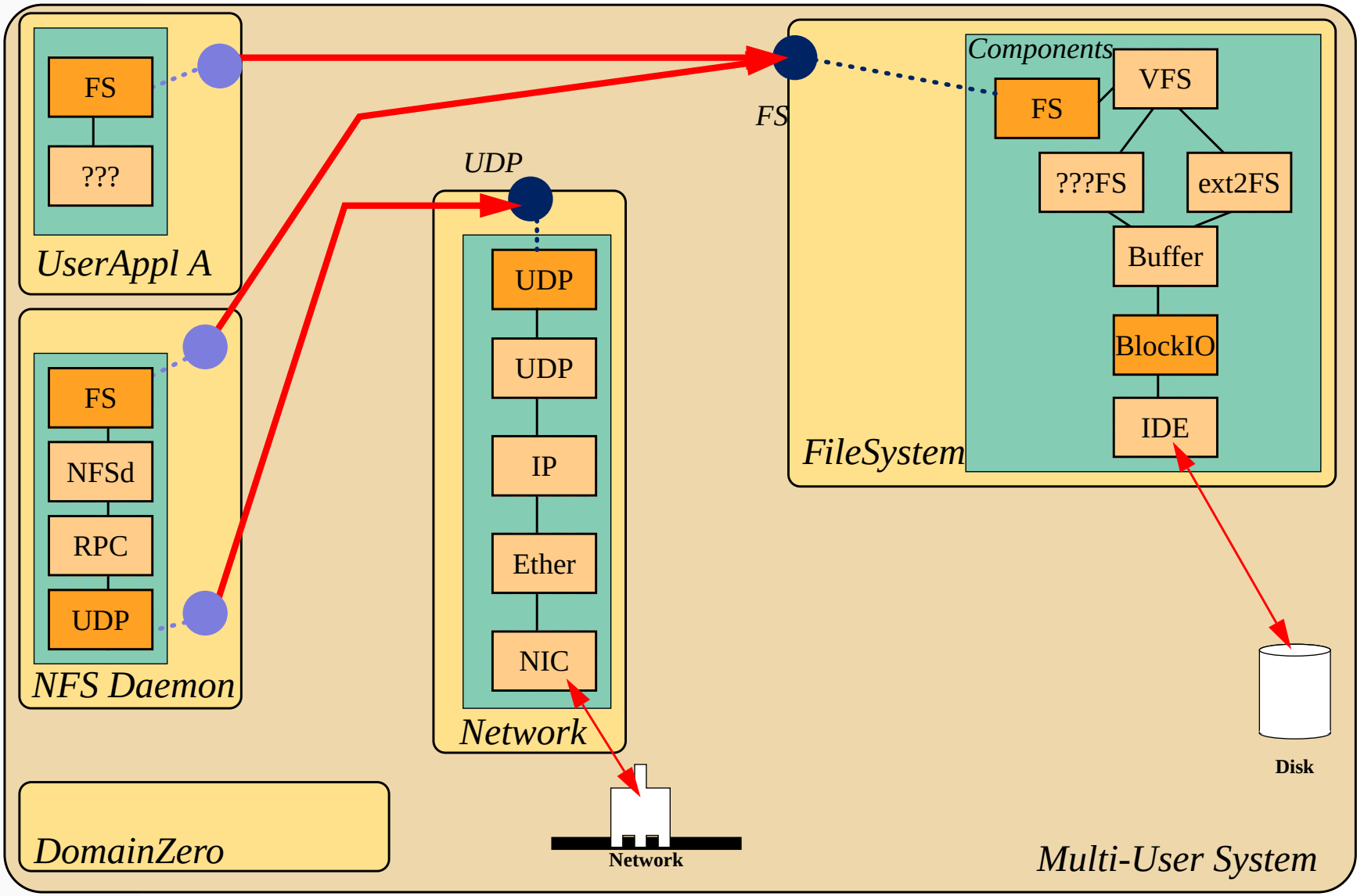
- JX protection is based on type safety and portals
- Some domains can circumvent these mechanisms
  - DomainZero, Translator, Verifier ➜ Trust
  - (some) device drivers



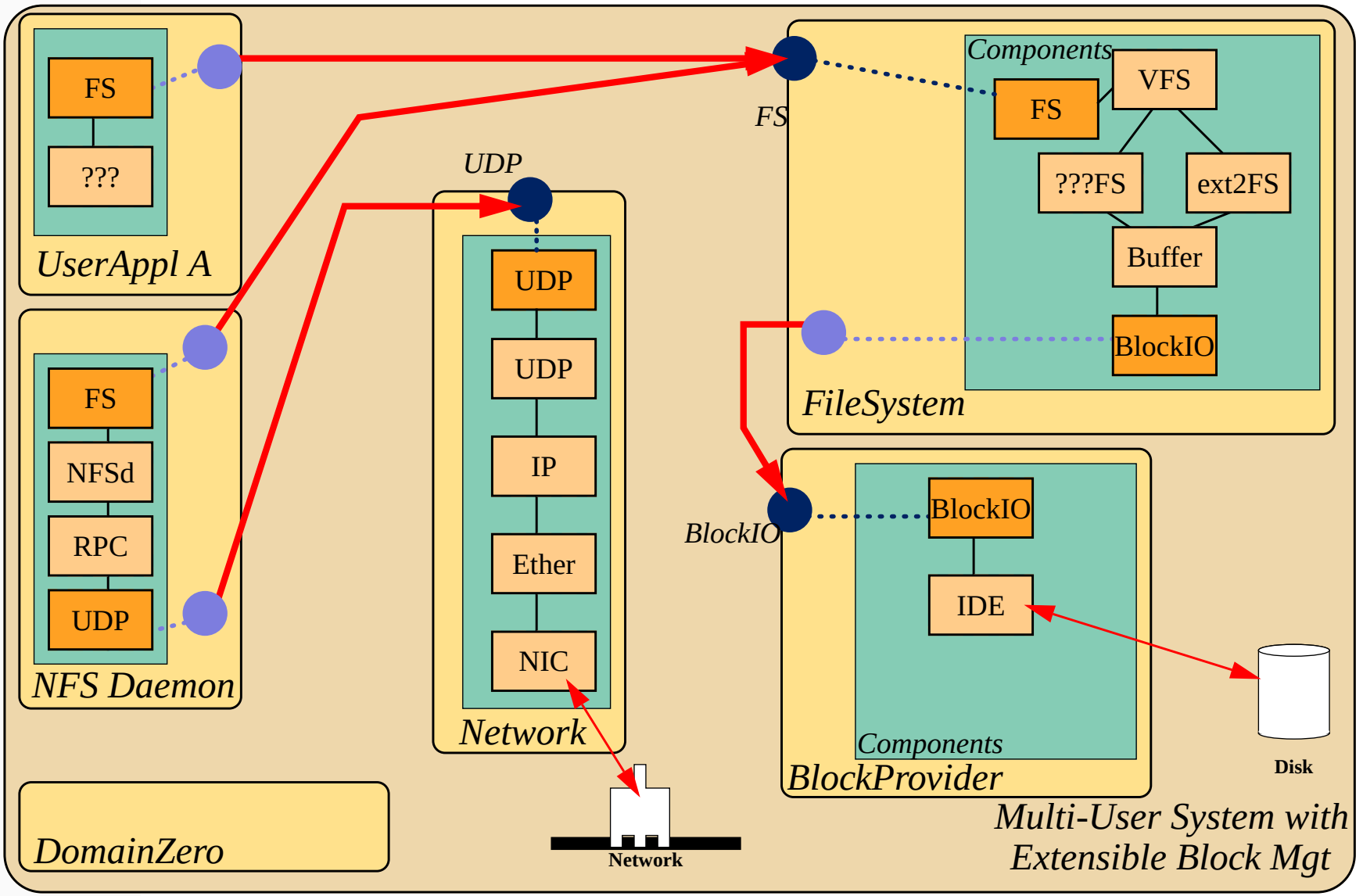
Building an OS: A Dedicated System



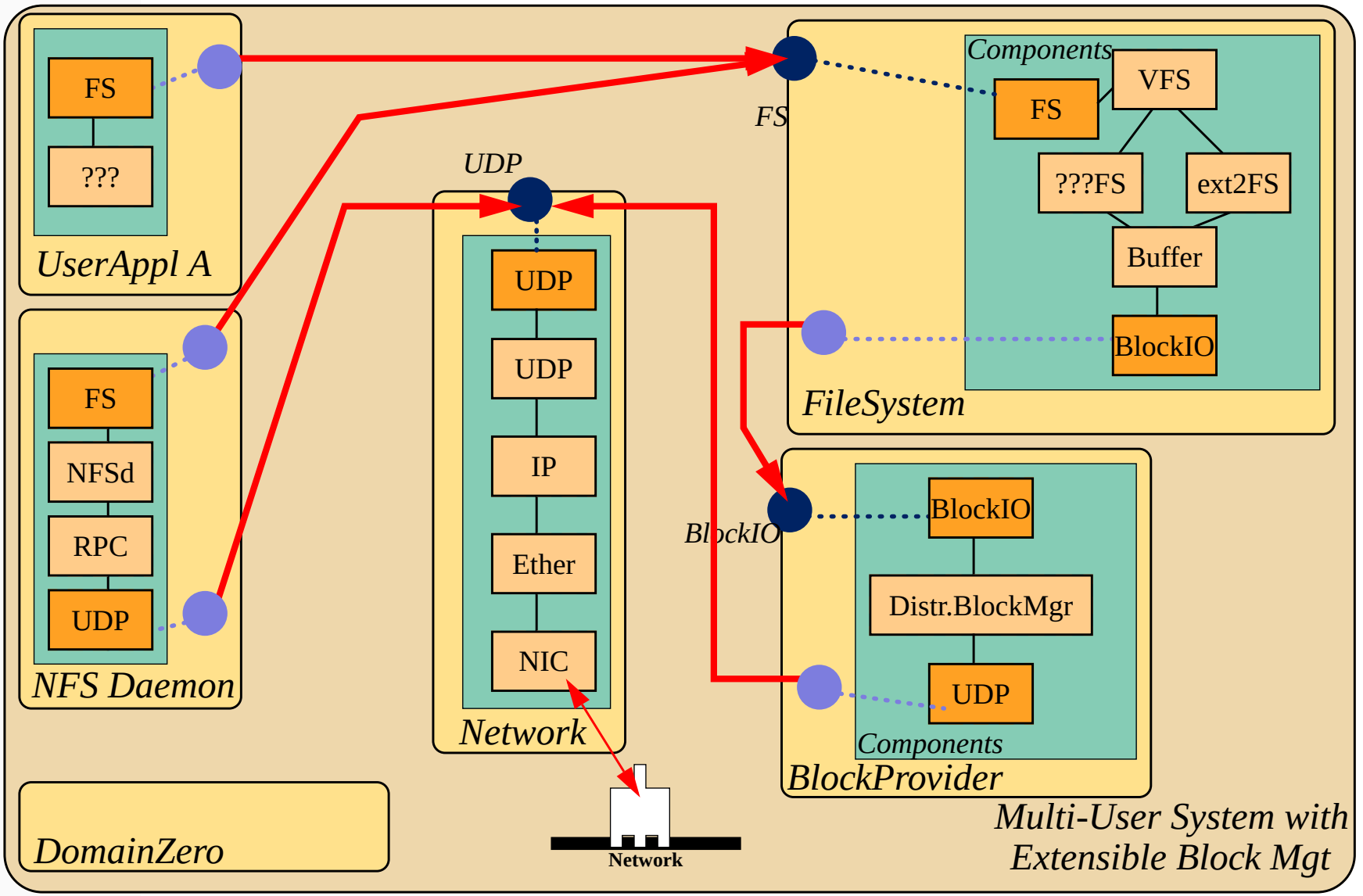
Building an OS: A Multiuser System



Building an OS: Extensibility



Building an OS: Extensibility

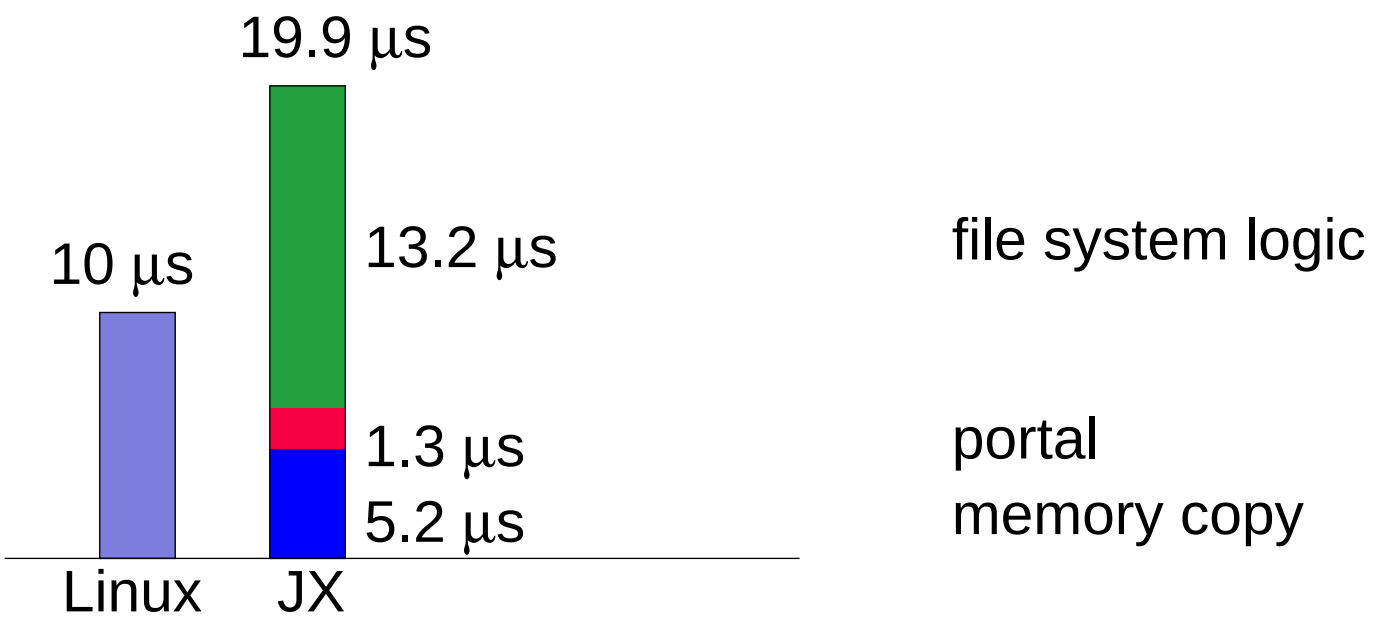


Performance

- IPC:
  - Portal round trip 650 cycles
  - L4 (including RPC stubs): 800 cycles
  - KaffeOS: 27270 cycles

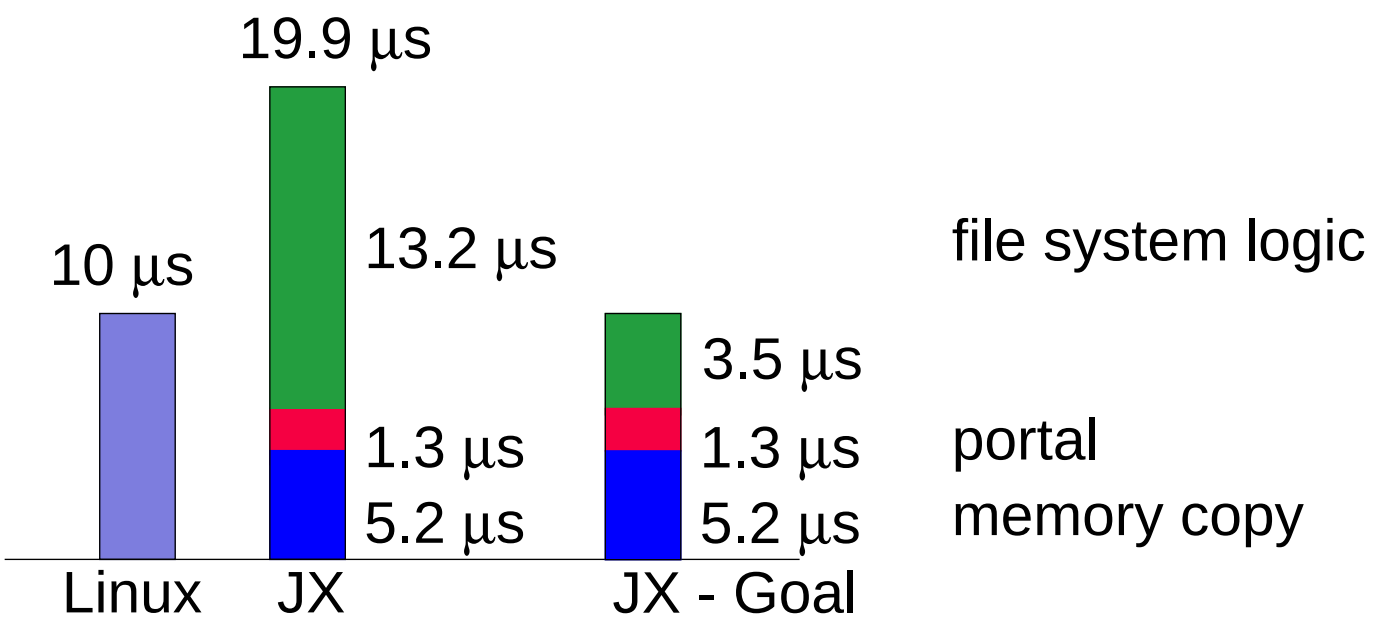
Performance

- Iozone-like benchmark: record=4kB, file=4kB, re-read
  - JX (PIII 500): 201 MB/s
  - Linux (PIII 500): 400 MB/s



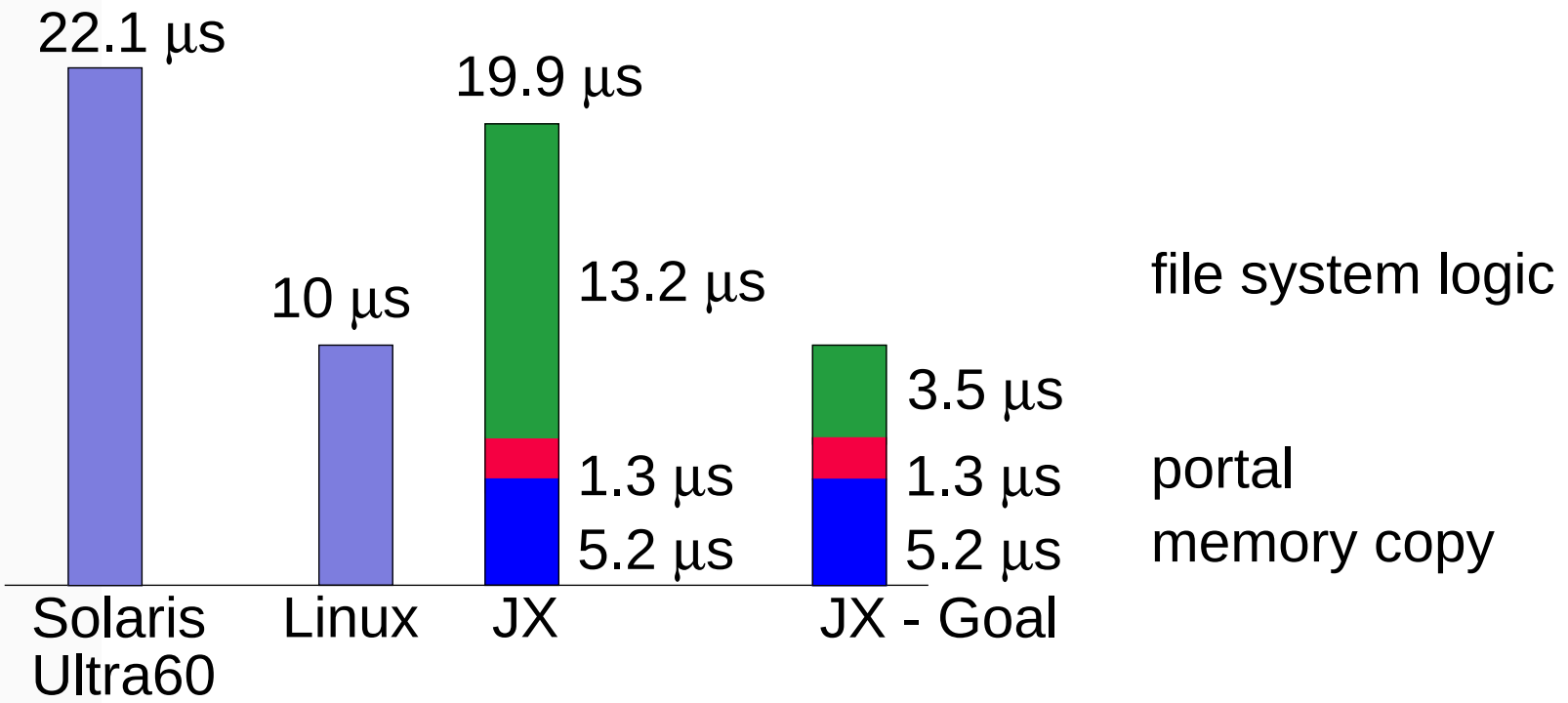
Performance

- Iozone-like benchmark: record=4kB, file=4kB, re-read
  - JX (PIII 500): 201 MB/s
  - Linux (PIII 500): 400 MB/s



Performance

- Iozone-like benchmark: record=4kB, file=4kB, re-read
  - JX (PIII 500): 201 MB/s
  - Linux (PIII 500): 400 MB/s
  - Solaris (Ultra60 - UltraSparcII 360): 181 MB/s



Conclusion

- Single Address Space
- Full Protection
  - completely decoupled domains
  - fast communication using portals or memory objects
- Reusable components
- Dynamic extensibility
- Good performance

➜ **http://www4.cs.fau.de/Projects/JX/**

Status

- JX runs on off-the-shelf PCs
- Drivers for: IDE, Matrox G200, 3COM 3C905B, BT848
- Ext2-FS
- UDP, TCP, RPC (client), NFS (client)
- SMP support