



Fast Fault Injections with Virtual Machines

Martin Süßkraut, Stephan Creutz, Christof Fetzer

*martin.suesskraut@tu-dresden.de, stephan.creutz@inf.tu-dresden.de,
christof.fetzer@tu-dresden.de*

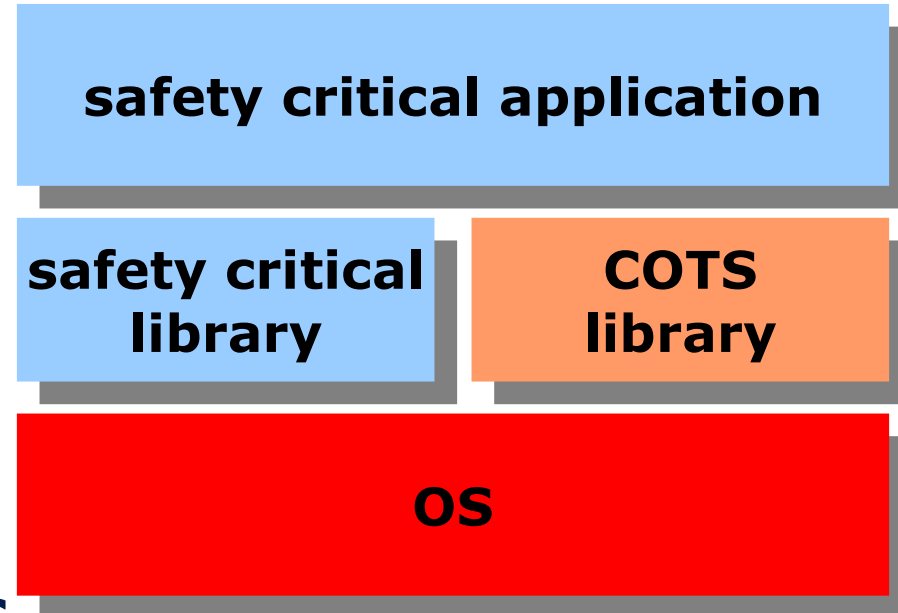
**Department of Computer Science
Technische Universität Dresden, Germany**

37th DSN, 06/26/2007

Motivation: Issues in Safety Critical Systems

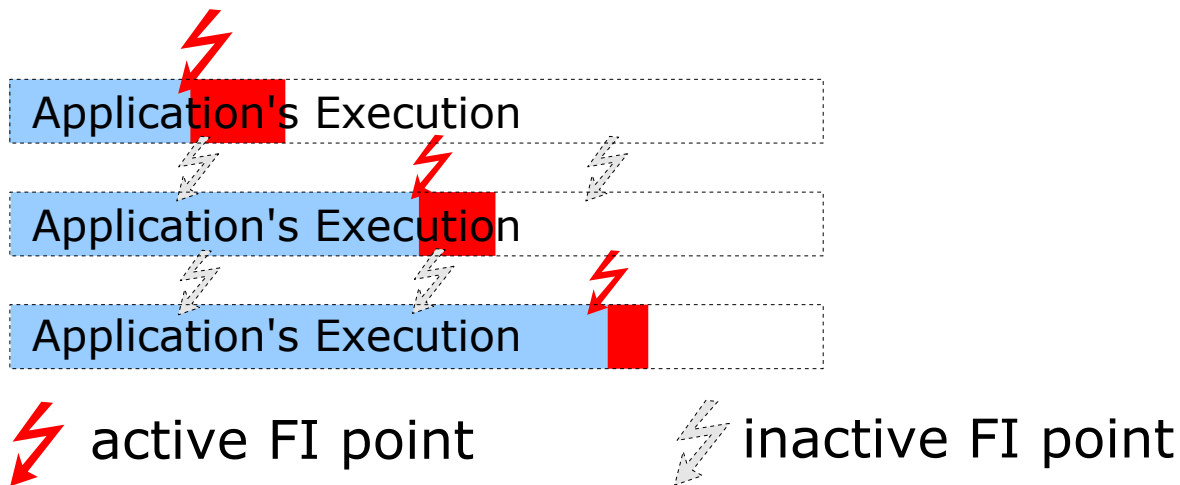
- build up on unreliable components
- missing isolation (e.g., in embedded systems)
- large number of components

→ **robustness tests of unreliable components**



Fault Injection (FI) Performance

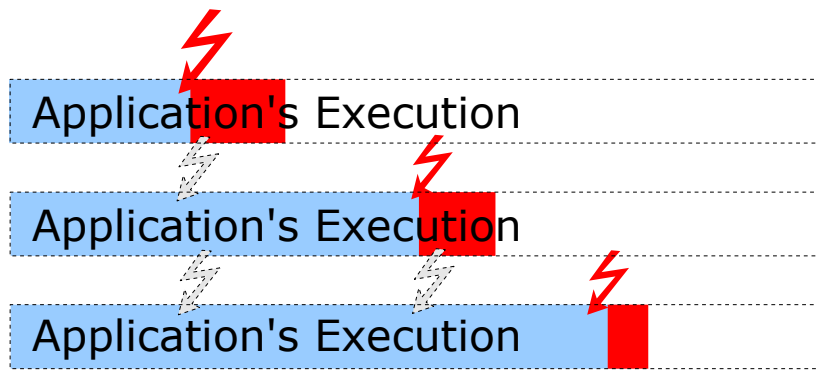
- one run per injected fault



- n number of fault injection points
- Performance: $O(n^2)$

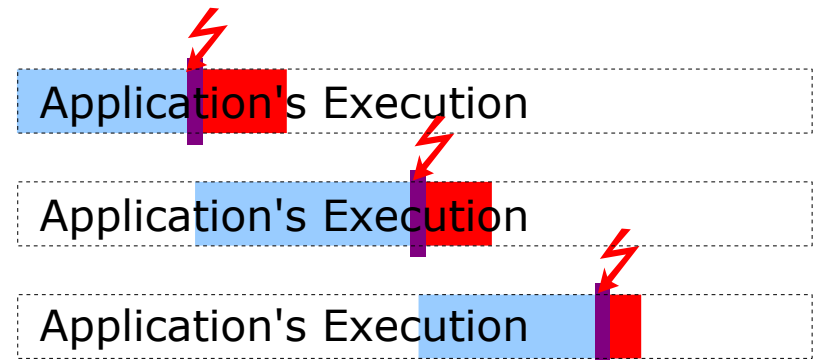
Fault Injection (FI) Performance

- one run per injected fault
- checkpoint before FI



 active FI point

 inactive FI point



 checkpoint

- n number of fault injection points
- Performance: $O(n^2)$
- Performance: $O(n)$

Scalability

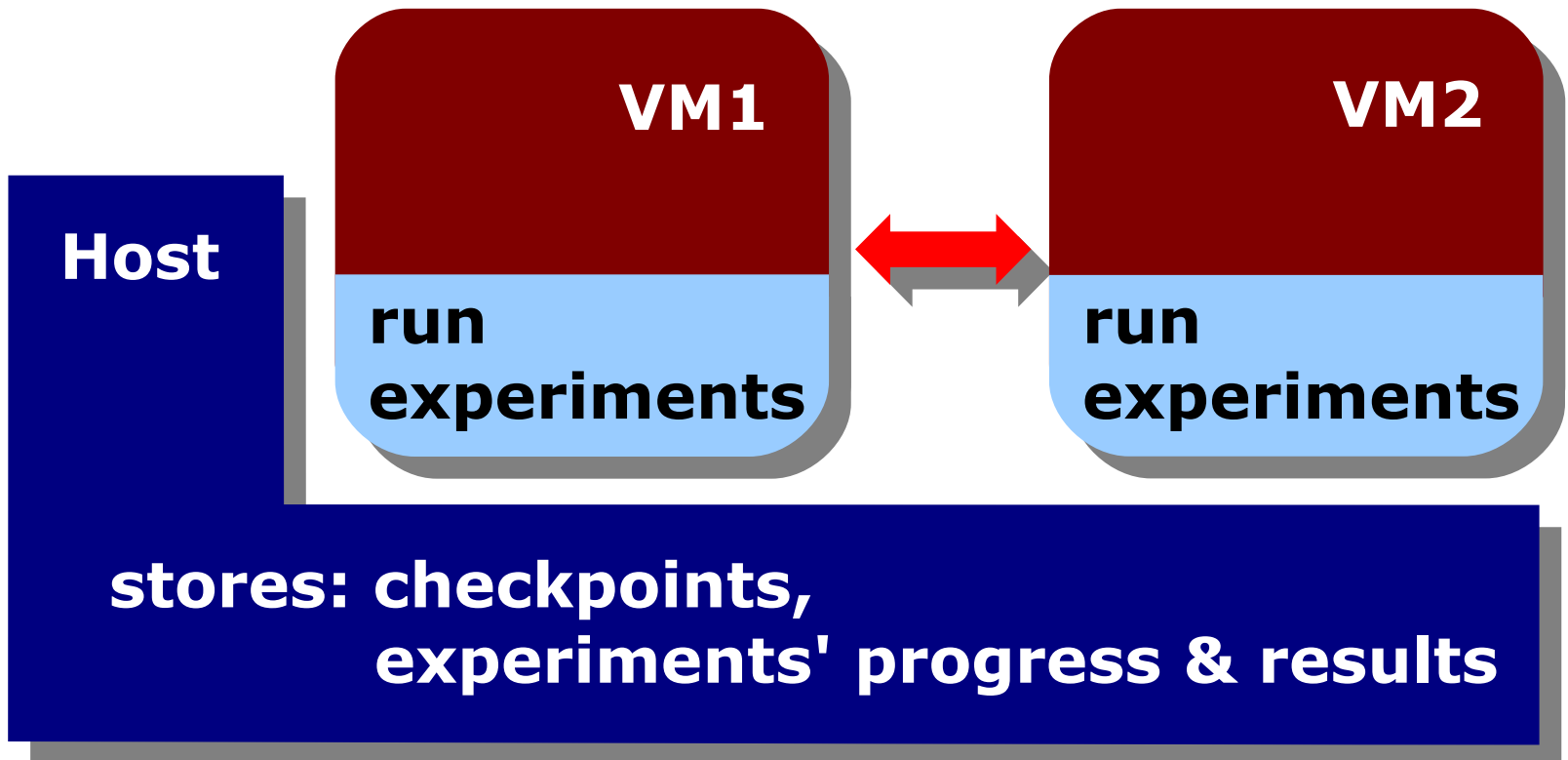
- scalability is important – inject faults in every:
 - system call
 - IO operation
 - call to a certain library

Isolation Problems

- common checkpointing mechanism focus on process state
 - **fork**
- but application might have external state:
 - file system
 - other applications
 - OS
 - network

VM Approach

- checkpoint complete environment with virtual machines



Details

- running prototype
- pseudo code for fault injection:

```
set_state (ERROR_INJECT)
do_checkpoint ()
if get_state () == ERROR_INJECT
    do_fault_injection ()
    log_results ()
    set_state (NEXT_ERROR)
    rollback ()
end
```

Conclusion

- run fault injection within VMs
- ✗ checkpointing gets more expensive
 - copy-on-write
 - HDD
 - memory
- ✓ no isolation problems
- ✓ distributed fault injection experiments