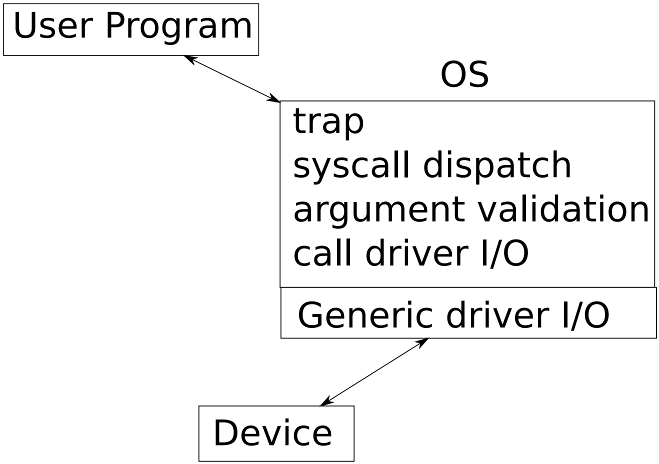
OS bypass performance without OS bypass

Usenix 2010

Outline

1. Currying and process-private system calls

Standard path



User Program

OS

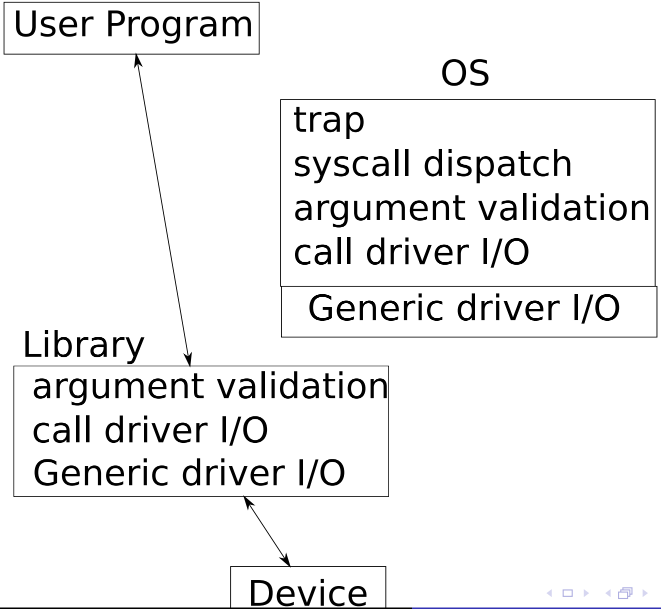
trap
syscall dispatch
argument validation
call driver I/O

Generic driver I/O

Device

- OS overhead known to be primary limit even with very efficient devies

OS Bypass



User Program

OS

trap
syscall dispatch
argument validation
call driver I/O

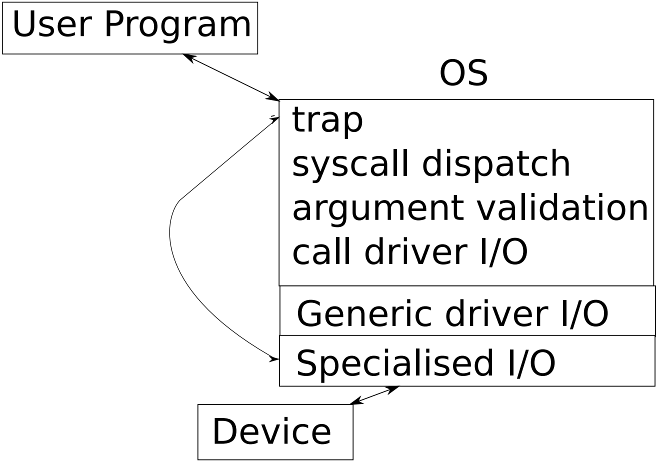
Generic driver I/O

Library

argument validation
call driver I/O
Generic driver I/O

Device

Pushing IO back into the kernel via currying and process-private system calls



How it is used

- fd = open(“/dev/torusctl”)
- newsyscall = write(fd, “fastpath pointer size”);
- paramters and operation “curried” into new system all
- newsyscall attached to this process and its children only
- newsyscall(); – no parameters, fast path
- no parameters and bypasses all system call checking
- 700 ns measured time on barrier
- Now any program, not just mpi programs, can use the network
- Being integrated into new mpi library now