Shared Library Access Pattern Analysis of Android Applications

Xiaowen Dong¹, Sandhya Dwarkadas¹, Alan Cox²
¹University of Rochester, ²Rice University

Impact of Shared Libraries on the Instruction Footprint

- Number of shared libraries (lib*.so) per application:
  - Loaded: 88 to 107
  - Invoked: 20 to 62
- % of instruction pages accessed that:
  - Belonged to shared libraries: 60%
  - Belonged to zygote-preloaded libraries: 35%
- % of instructions fetched that:
  - Belonged to shared libraries: 72%
  - Belonged to zygote-preloaded libraries: 51%

Shared Library Instruction Footprint Intersection Across Applications

There is considerable overlap in the shared library code accessed across different applications

Impact of Shared Libraries on the Instruction Footprint

- Number of shared libraries (lib*.so) per application:
  - Loaded: 88 to 107
  - Invoked: 20 to 62
- % of instruction pages accessed that:
  - Belonged to shared libraries: 60%
  - Belonged to zygote-preloaded libraries: 35%
- % of instructions fetched that:
  - Belonged to shared libraries: 72%
  - Belonged to zygote-preloaded libraries: 51%

Shared Library Instruction Footprint Intersection Across Applications

There is considerable overlap in the shared library code accessed across different applications

% of total instruction pages accessed that are in common for each pair of applications tested:

- Average: 28%
- Max: 64%

Bibliography


Acknowledgement

This work was supported in part by the U.S. National Science Foundation grants CCF-1217920, CNS-1319353, and CCF-137224

Implications

The analysis motivates a redesign of Linux’s memory management subsystem to:

A. Share page table pages
   - Reducing soft page faults
   - Improving fork performance
   - Reducing storage overhead

B. Share TLB entries
   - Reducing the number of TLB misses

Shared memory management infrastructure has the potential to improve Android’s application launch, steady-state, and context switch efficiency