Porting LeakSanitizer:

A beginner's guide

Francis Ricci

Software Engineer, Facebook

Background

Heap Checker vs LeakSanitizer

Advantages of LeakSanitizer

- Performance
- Direct vs Indirect leak reports
- Thread-local data handling
- Suppressions

Heap Checker vs LeakSanitizer

Heap Checker supported platforms

	Linux	FreeBSD	Darwin	Android	Windows	Fuchsia
x86_64						
x86						
aarch64						
arm						
powerpc						
mips						

Heap Checker vs LeakSanitizer

LeakSanitizer supported platforms

	Linux	FreeBSD	Darwin	Android	Windows	Fuchsia
x86_64						
x86						
aarch64						
arm						
powerpc						
mips						

Heap Checker vs LeakSanitizer

AddressSanitizer supported platforms

	Linux	FreeBSD	Darwin	Android	Windows	Fuchsia
x86_64						
x86						
aarch64						
arm						
powerpc						
mips						

Implementation

AddressSanitizer -> LeakSanitizer

Additional functionality required for leak checking

- Thread suspension
- Memory map
  - Thread-local storage
  - Static and global variables
- Platform-specific data sections

Thread suspension

Scan registers and stacks for heap pointers

- Suspend threads
  - Linux - `ptrace()`
  - Darwin - `thread_suspend()`
- Thread state parsing
  - Linux - `ptrace()`
  - Darwin – `thread_get_state()`

Memory map

Scan global data regions for pointers

- Generate map
  - Linux - `/proc/maps`
  - Darwin - `vm_region_recurse()/_dyld_get_image_header()`
- Data
  - Linux - `dl_phdr_info`
  - Darwin – `segment_command`
- Thread-local storage
  - `%fs/%gs`

Platform-specific data

Allocations requiring special handling

- Linux
  - Linker allocations (dynamic TLS blocks)
- Darwin
  - Kernel alloc once page
  - mmap’d regions

Testing

Testing LeakSanitizer

- compiler-rt
  - LSan test suite
  - ASan test suite with `DETECT_LEAKS=1`
- llvm+clang
  - `DETECT_LEAKS=1` for bootstrapped ASan builds
- Very large internal project
  - asm, c, c++, objective-c, swift
  - Compare behavior with LSan on Linux
  - Compare behavior with gperftools HeapChecker