NAME

MemGuard - memory allocator for debugging purposes

SYNOPSIS

options DEBUG_MEMGUARD

DESCRIPTION

MemGuard is a simple and small replacement memory allocator designed to help detect tamper-after-free scenarios. These problems are more and more common and likely with multithreaded kernels where race conditions are more prevalent.

MemGuard can take over **malloc**(), **realloc**() and **free**() for a single malloc type. Alternatively **MemGuard** can take over **uma_zalloc**(), **uma_zalloc_arg**() and **uma_free**() for a single uma(9) zone. Also **MemGuard** can guard all allocations larger than PAGE_SIZE, and can guard a random fraction of all allocations. There is also a knob to prevent allocations smaller than a specified size from being guarded, to limit memory waste.

EXAMPLES

To use **MemGuard** for a memory type, either add an entry to */boot/loader.conf*:

vm.memguard.desc=<memory_type>

Or set the *vm.memguard.desc* sysctl(8) variable at run-time:

sysctl vm.memguard.desc=<memory_type>

Where *memory_type* can be either a short description of the memory type to monitor, either name of uma(9) zone. Only allocations from that *memory_type* made after *vm.memguard.desc* is set will potentially be guarded. If *vm.memguard.desc* is modified at run-time then only allocations of the new *memory_type* will potentially be guarded once the sysctl(8) is set. Existing guarded allocations will still be properly released by either free(9) or uma_zfree(9), depending on what kind of allocation was taken over.

To determine short description of a malloc(9) type one can either take it from the first column of vmstat(8) **-m** output, or to find it in the kernel source. It is the second argument to MALLOC_DEFINE(9) macro. To determine name of uma(9) zone one can either take it from the first column of vmstat(8) **-z** output, or to find it in the kernel source. It is the first argument to the uma_zcreate(9) function.

The *vm.memguard.divisor* boot-time tunable is used to scale how much of the system's physical

memory **MemGuard** is allowed to consume. The default is 10, so up to *vm_cnt.v_page_count*/10 pages can be used. **MemGuard** will reserve *vm_kmem_max* / *vm.memguard.divisor* bytes of virtual address space, limited by twice the physical memory size. The physical limit is reported as *vm.memguard.phys_limit* and the virtual space reserved for **MemGuard** is reported as *vm.memguard.mapsize*.

MemGuard will not do page promotions for any allocation smaller than *vm.memguard.minsize* bytes. The default is 0, meaning all allocations can potentially be guarded. **MemGuard** can guard sufficiently large allocations randomly, with average frequency of every one in 100000 / *vm.memguard.frequency* allocations. The default is 0, meaning no allocations are randomly guarded.

MemGuard can optionally add unmapped guard pages around each allocation to detect overflow and underflow, if *vm.memguard.options* has the 1 bit set. This option is enabled by default. **MemGuard** will optionally guard all allocations of PAGE_SIZE or larger if *vm.memguard.options* has the 2 bit set. This option is off by default. By default **MemGuard** does not guard uma(9) zones that have been initialized with the UMA_ZONE_NOFREE flag set, since it can produce false positives on them. However, this safety measure can be turned off by setting bit 3 of the *vm.memguard.options* tunable.

SEE ALSO

sysctl(8), vmstat(8), contigmalloc(9), malloc(9), redzone(9), uma(9)

HISTORY

MemGuard first appeared in FreeBSD 6.0.

AUTHORS

MemGuard was originally written by Bosko Milekic <*bmilekic@FreeBSD.org*>. This manual page was originally written by Christian Brueffer <*brueffer@FreeBSD.org*>. Additions have been made by Matthew Fleming <*mdf@FreeBSD.org*> and Gleb Smirnoff <*glebius@FreeBSD.org*> to both the implementation and the documentation.