Linux Kernel Training

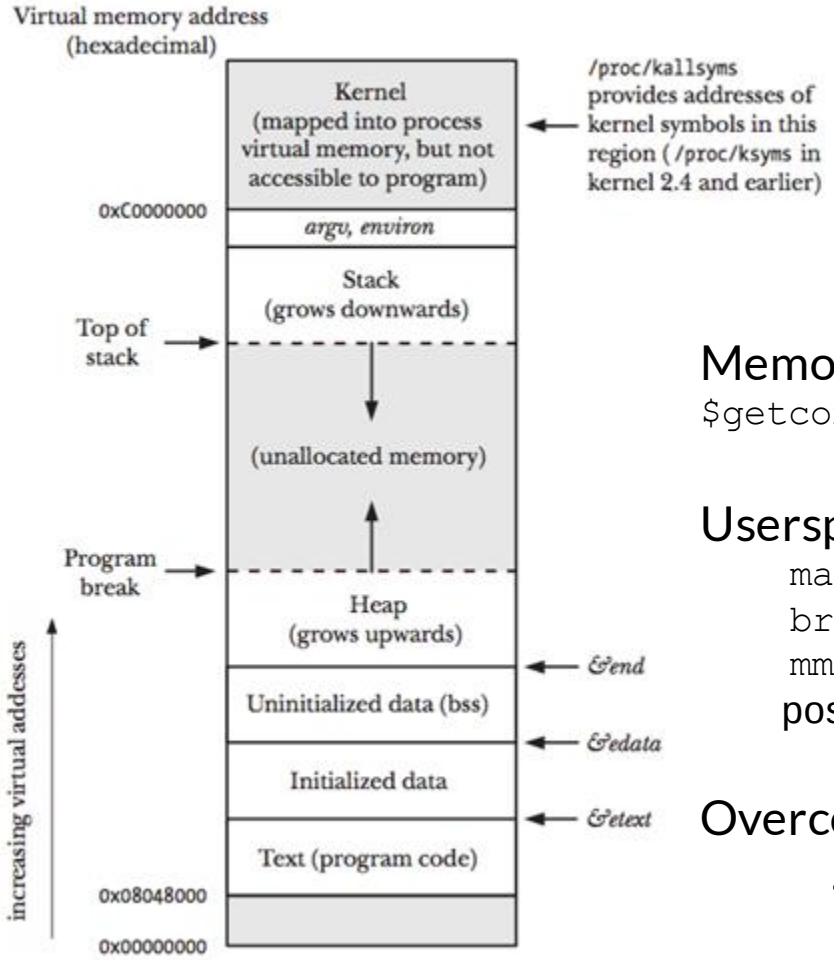
Internal Kernel API

Memory Management and Allocators

Agenda

Memory Management and Allocators

Memory Layout



Memory page frame size on x86-32 is 4096 bytes

```
$getconf PAGESIZE
```

Userspace functions

```
malloc()/calloc()/realoc()/free()
brk()/sbrk()
mmap()
```

posix_memalign()

Overcommiting and OOM

Userspace Memory Allocation

Internal and external fragmentation are the common issues

Glibc allocates memory via
- buddy memory allocation scheme
- anonymous memory mapping

MMAP_THRESHOLD is 128 kB (malopt() to change)

Memory Management

#include <include/linux/gfp.h>
or/and
#include <include/linux/types.h>

void *kmalloc(size_t size, gfp_t flags);

gfp_t: is a bitmask. The most popular predefined values are:
GFP_ATOMIC, GFP_KERNEL, GFP_USER

void kfree(const void *);

include/linux/vmalloc.h

```
void *vmalloc(unsigned long size);
void vfree(const void *addr);
```

Memory Management (Low level)

include/linux/gfp.h

```
struct_page *alloc_pages(gfp_t gfp_mask, unsigned int order);
struct page * alloc_page(gfp_t gfp_mask);

void *page_address( struct_page * page );
unsigned long get_zeroed_page(unsigned int gfp_mask);
```

```
void free_pages(unsigned long addr, unsigned int order);
void free_page(unsigned long addr);
```

Interface SLAB

File system point: /proc/slabinfo

include/linux/slab.h
struct kmem_cache *kmem_cache_create(const char *name, size_t size, size_t align, unsigned long flags, void (*ctor)(void *));
void kmem_cache_destroy(struct kmem_cache *s);
void *kmem_cache_alloc(struct kmem_cache *, gfp_t flags);
void kmem_cache_free(struct kmem_cache *, void *);

- SLAB
- SLUB
- SLOB

```
cat /boot/<path_to_config> | grep "CONFIG_SL[AOU]B"
```

MemPool

include/linux/mempool.h

```
typedef void *(mempool_alloc_t)(int gfp_mask, void
*pool_data);
typedef void (mempool_free_t)(void *element, void
*pool_data);
```

```
mempool_t *mempool_create(int min_nr, mempool_alloc_t
*alloc_fn, mempool_free_t *free_fn, void *pool_data);

void *mempool_alloc(mempool_t *pool, gfp_t gfp_mask);
void mempool_free(void *element, mempool_t *pool);
```

mempool_alloc_slab()
mempool_kmalloc()
mempool_alloc_pages()

Home reading

O. Tsiliurik Linux Kernel Development Book: Ch 6

May the Force be with you