Memory cheat(iOS)

Unknown macro: 'html'

List

Memory cheat(iOS)

- 2016 Cheat tool .
  - https://github.com/Lazenca/Lazenca-A-iOS
- Tool Memory cheat .
- Tool , .

Development Tools

- **Cheat tool Theos .**

Theos/Setup

- http://iphonedevwiki.net/index.php/Theos/Setup

Structure of Memory Cheat

- **Memory cheats tool .**

The default behavior of the memory cheat tool.

1	**Process attach**	task_for_pid() .
2	**Check the process memory area**	Memory Memory (Memory map) .
3	**Memory access**	Memory vm_read_overwrite, vm_write .

Process list

SYNOPSIS - sysctl()

- **sysctl() .**

sysctl SYNOPSIS

```
#include <sys/types.h>
#include <sys/sysctl.h>

int sysctl(int *name, u_int namelen, void *oldp, size_t *oldlenp, void *newp, size_t newlen);
```

Management Information Base(MIB)

CTL_KERN	• . ○ ○ v ○ ○ ○ ○
KERN_PROC	• . ○ struct kinfo_proc .

Example

- **sysctl() Process list .**

RunningProcess.h

```
    int mib[4] = {CTL_KERN, KERN_PROC, KERN_PROC_ALL, 0};
size_t miblen = 4;
size_t size = 0;

int st = sysctl(mib, miblen, NULL, &size, NULL, 0);

struct kinfo_proc *process = NULL;
struct kinfo_proc *newprocess = NULL;

do {
    size += size / 10;
    newprocess = (kinfo_proc * )realloc(process, size);
    if (!newprocess){
        if (process){
            free(process);
        }
        return nil;
    }
    process = newprocess;
    st = sysctl(mib, miblen, process, &size, NULL, 0);
} while (st == -1 && errno == ENOMEM);


    for (int i = nprocess - 1; i >= 0; i--){
        NSString *processID = [[NSString alloc] initWithFormat:@"%d", process[i].kp_proc.p_pid];
        NSString *processName = [[NSString alloc] initWithFormat:@"%s", process[i].kp_proc.p_comm];
            [processID release];
            [processName release];
    }

    free(process);
```

BSD Library Functions Manual - SYSCTL(3)

- https://developer.apple.com/legacy/library/documentation/Darwin/Reference/ManPages/man3/sysctl.3.html

Process attach

SYNOPSIS - task_for_pid()

- **task_for_pid()** .
  - ID .
  - ID .

SYNOPSIS - task_for_pid()

```
#include <mach/mach.h>

kern_return_t task_for_pid(struct task_for_pid_args *args);
```

kern_return_t task_for_pid(struct task_for_pid_args *args)

- https://opensource.apple.com/source/xnu/xnu-4570.1.46/bsd/vm/vm_unix.c.auto.html

Example

Process attach

```
int attach(){
    kern_return_t kret;
    tmp_target_task = 0;
    kret = task_for_pid(mach_task_self(),pid,&tmp_target_task);
    if (kret) {
        printf("task_for_pid() failed with message %s!\n",mach_error_string(kret));
    }else{
        printf("attach - target_task : %d, tmp_target_task : %d\n",target_task, tmp_target_task);
        kret = task_suspend(target_task);
        if (kret != KERN_SUCCESS) {
            printf("task_suspend() failed with message %s!\n",mach_error_string(kret));
        }else{
            printf("task_suspend - Success\n");
            return 1;
        }
    }
    return 0;
}
```

Check the process memory area

SYNOPSIS - vm_region_recurse()

- **vm_region_recurse() .**
  - .
  - 1 task_for_pid() .
  - 2 .
  - 3 .
  - 4 .
  - 5 .
    - protection, max_protection .
      - protection :
      - max_protection :
  - 6 "VM_REGION_SUBMAP_INFO_COUNT" .

SYNOPSIS - vm_region_recurse()

```
#include <mach/mach.h>
kern_return_t vm_region_recurse(
        vm_map_t map,
        vm_offset_t *address,
        vm_size_t *size,
        natural_t *depth,
        vm_region_recurse_info_t info32,
        mach_msg_type_number_t *count)
```

- 64bit .

SYNOPSIS - vm_map_region_recurse_64()

```
kern_return_t
vm_map_region_recurse_64(
        vm_map_t                 map,
        vm_map_offset_t        *address,
        vm_map_size_t               *size,
        natural_t                 *nesting_depth,
        vm_region_submap_info_64_t        submap_info,
        mach_msg_type_number_t        *count)
```

vm_region_recurse() and vm_map_region_recurse_64()

- https://opensource.apple.com/source/xnu/xnu-344.49/osfmk/vm/vm_map.c.auto.html
- https://opensource.apple.com/source/xnu/xnu-4570.1.46/osfmk/vm/vm_map.c.auto.html

struct vm_region_submap_info

struct vm_region_submap_info

```
struct vm_region_submap_info {
        vm_prot_t                       protection;     /* present access protection */
        vm_prot_t                       max_protection; /* max avail through vm_prot */
        vm_inherit_t                inheritance;/* behavior of map/obj on fork */
        uint32_t                        offset;                 /* offset into object/map */
    unsigned int            user_tag;        /* user tag on map entry */
    unsigned int            pages_resident;         /* only valid for objects */
    unsigned int            pages_shared_now_private; /* only for objects */
    unsigned int            pages_swapped_out; /* only for objects */
    unsigned int            pages_dirtied;   /* only for objects */
    unsigned int            ref_count;          /* obj/map mappers, etc */
    unsigned short          shadow_depth;          /* only for obj */
    unsigned char           external_pager;  /* only for obj */
    unsigned char           share_mode;         /* see enumeration */
        boolean_t                       is_submap;         /* submap vs obj */
        vm_behavior_t                behavior;        /* access behavior hint */
        vm32_object_id_t        object_id;        /* obj/map name, not a handle */
        unsigned short                user_wired_count;
};
```

struct vm_region_submap_info

- https://opensource.apple.com/source/xnu/xnu-4570.1.46/osfmk/mach/vm_region.h.auto.html

Example

- **vm_region_recurse()** .
  - info.protection, info.max_protection , .(VM_PROT_WRITE | VM_PROT_READ)
  - , .

int findWriteableRegions()

```
int findWriteableRegions(){
    vm_size_t size;
    vm_address_t address;
    natural_t nesting_depth;
    mach_msg_type_number_t infoCnt;

    regionList.clear();

    size = 0;
    address = 0;
    struct vm_region_submap_info info;
    infoCnt = VM_REGION_SUBMAP_INFO_COUNT;

    for (; !vm_region_recurse(target_task,&address,&size,&nesting_depth,(vm_region_recurse_info_t)&info,
&infoCnt);) {
        if (info.is_submap) {
            ++nesting_depth;
        }else{
            if ((info.protection & (VM_PROT_WRITE | VM_PROT_READ)) == 3 && (info.max_protection &
(VM_PROT_WRITE | VM_PROT_READ)) == 3) {
                regionStruct.startAddr = address;
                regionStruct.endAddr = size + address;
                regionStruct.size = size;
                regionList.push_back(regionStruct);
                printf("region: %016x-%016x\n",regionStruct.startAddr,regionStruct.endAddr);
            }
            address += size;
        }
    }
    return 1;
}
```

Memory read - vm_read_overwrite(), vm_read()

- .

SYNOPSIS - vm_read_overwrite()

- **vm_read_overwrite()** .
    - 1 task_for_pid() .
    - 2 .
    - 3 .
    - 4 .
    - 5 .

SYNOPSIS - vm_read_overwrite()

```
#include <mach/mach.h>

kern_return_t vm_read_overwrite(
  vm_map_t            map,
  vm_address_t  address,
  vm_size_t            size,
  vm_address_t  data,
  vm_size_t            *data_size);
```

- vm_read .
- vm_read_overwrite (data_in) .

SYNOPSIS - vm_read()

```
kern_return_t vm_read(
        vm_task_t               target_task,
        vm_address_t        address,
        vm_size_t               size,
        size                       data_out,
        target_task              data_count);
```

vm_read_overwrite and vm_read

- https://opensource.apple.com/source/xnu/xnu-4570.1.46/osfmk/vm/vm_user.c.auto.html
- http://web.mit.edu/darwin/src/modules/xnu/osfmk/man/vm_read.html

Example

- .
  - vm_read_overwrite() .
    - startAddress endAddress .
    - 4096byte buffer .

void getValueArea(vm_address_t startAddress,vm_address_t endAddress, void* buffer,long number)

```
void getValueArea(vm_address_t startAddress,vm_address_t endAddress, void* buffer,long number){
    kern_return_t result;

    long readArea = 0;
    vm_size_t outsize;


    while(endAddress > startAddress){
                if (readArea != (startAddress & 0xFFFFFFFFFFFFF000)) {
                        readArea = startAddress & 0xFFFFFFFFFFFFF000;

                        outsize = 0;
                        result = vm_read_overwrite(target_task, readArea, 4096, (vm_address_t)buffer, &outsize);

                        if(!outsize){
                                printf("stardAddress 64 : %lx, %lx\n",startAddress,endAddress);
                                fprintf(stderr,"vm_read_overwrite failed: %lu\n",startAddress &
0xFFFFFFFFFFFFF000);
                        }
                }

                if (result == KERN_SUCCESS) {
                for (int i=0; i<512; i++) {
                                memInfoStruct.address = startAddress;
                                memInfoStruct.value = *(long*)((char*)buffer + ((startAddress - (startAddress &
0xFFFFFFFFFFFFF000)) & 0xFFFFFFFFFFFFFFF8));
                                memDataList.push_back(memInfoStruct);
                                startAddress += 8;
                        }
                }else{
                        startAddress += 8;
                }
    }
}
```

Memory write - vm_write()

- .

SYNOPSIS - vm_write()

- **vm_write( )** .
  - 1 task_for_pid() .
  - 2 .
  - 3 .
  - 4 .

SYNOPSIS - vm_write()

```
#include <mach/mach.h>
kern_return_t vm_write(
        vm_map_t                          map,
        vm_address_t                   address,
        pointer_t                          data,
        mach_msg_type_number_t  size)
```

vm_write

- https://opensource.apple.com/source/xnu/xnu-4570.1.46/osfmk/vm/vm_user.c.auto.html

Example

void MemoryWrite(vm_address_t address,long value)

```
void MemoryWrite(vm_address_t address,long value){
    vm_size_t outsize;
    vm_address_t startAddress = 0;

    unsigned int data;
    vm_read_overwrite(target_task, startAddress & 0xFFFFFFFFFFFFFFF8, 8, (vm_address_t)&data, &outsize);

    if (!outsize) {
        printf("vm_read_overwrite(%11lx) failed 1.",startAddress & 0xFFFFFFFFFFFFFFF8);
    }

    unsigned int write_data;
    write_data = value;

    kern_return_t kr;
    kr = vm_write(target_task, address, (vm_address_t)&write_data, 8);
    if(kr){
        printf("Fail %x\n", kr);
    }else{
        printf("Sucess!\n");
    }
}
```

Memory fuzzing

- **Memory fuzz** .
  -
  -
  -
  -
- .
  - Fuzz .
  - , .
    - .
    - .

Related site

- https://github.com/Lazenca/Lazenca-A-iOS

Unknown macro: 'html'