Yandex

Yandex

Fighting against Flash 0-day

Andrey Kovalev, Konstantin Otrashkevich, Evgeny Sidorov

Agenda

- Short Intro
- Flash internals & use-after-free in general
- Popular Flash exploits
- Detection
- Google Mitigations
- Life after G’s mitigations etc.

Short Intro

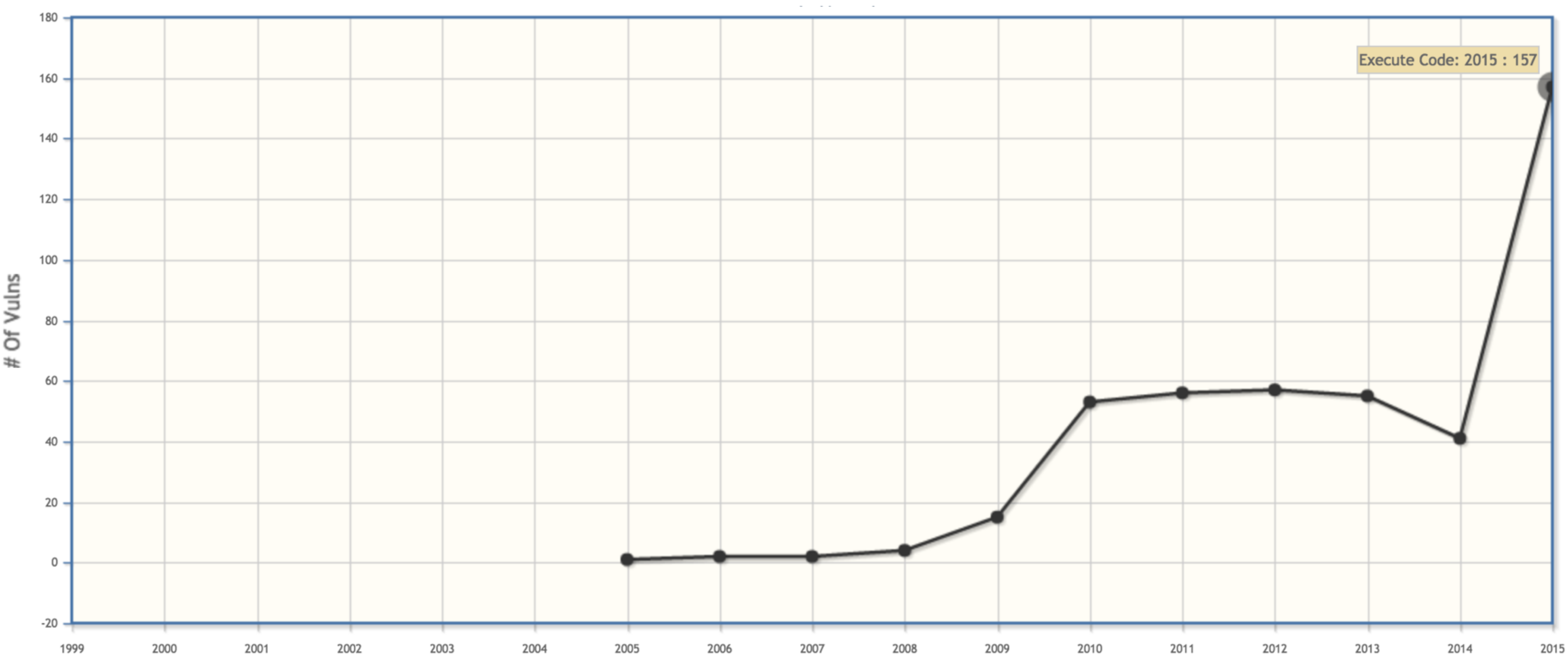
Why Flash exploitation is so popular?

- 2013: Java exploits were the main ‘workhorse’
- January 2014: Oracle blocked the execution of unsigned applets
- June 2014: isolated heap and delayed frees in MS IE
- Exploit developers focused on Adobe Flash

Why Flash exploitation is so popular?

- Cross platform
- Cross browser
- Can be embedded in other documents and formats
- Powerful programming opportunities
- Very popular in WEB
- Flash has less security mitigations than IE

Adobe Flash Code Excitation Vulnerabilities



Source: https://cvedetails.com

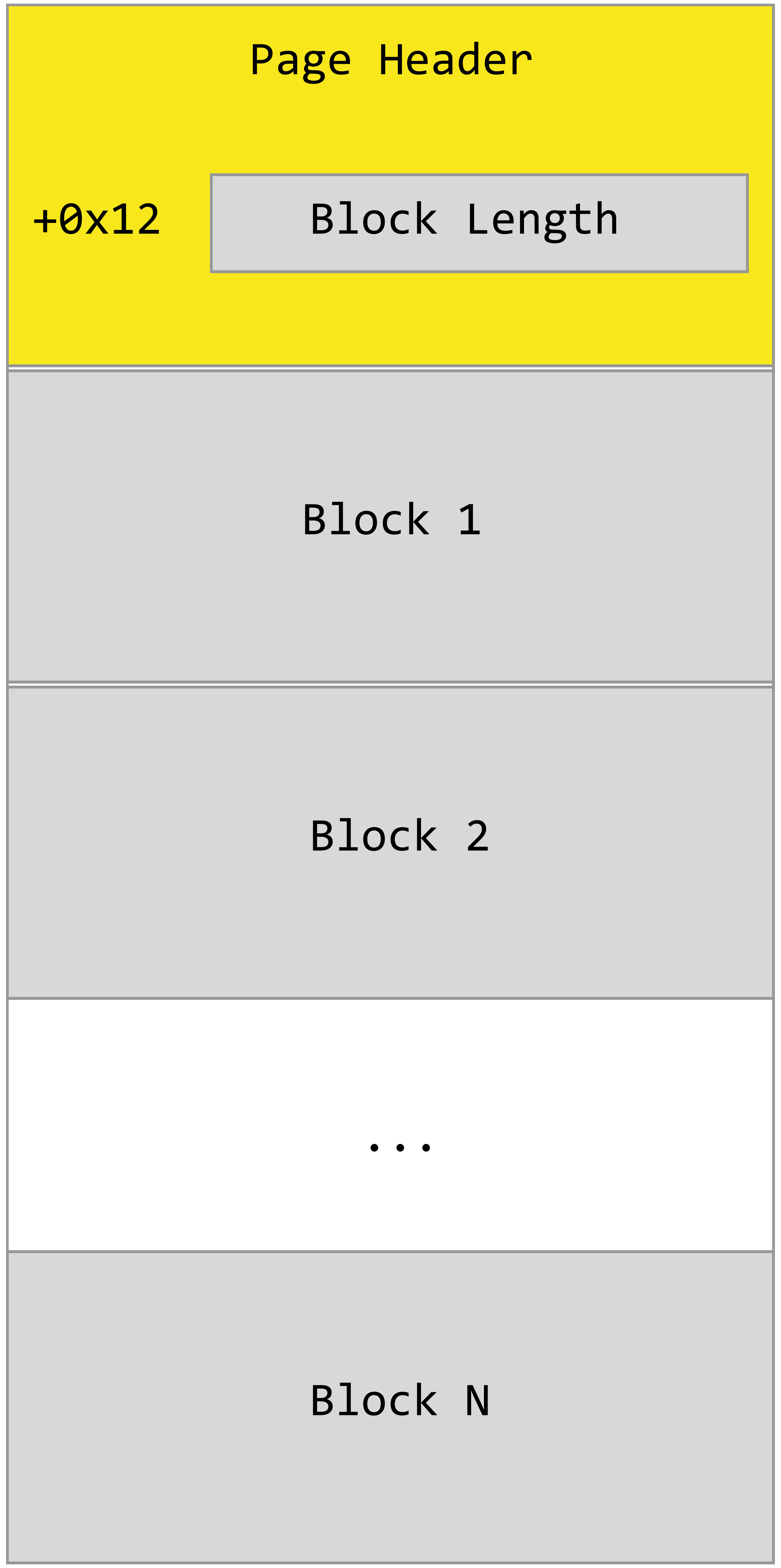
Flash internals

Flash memory allocation

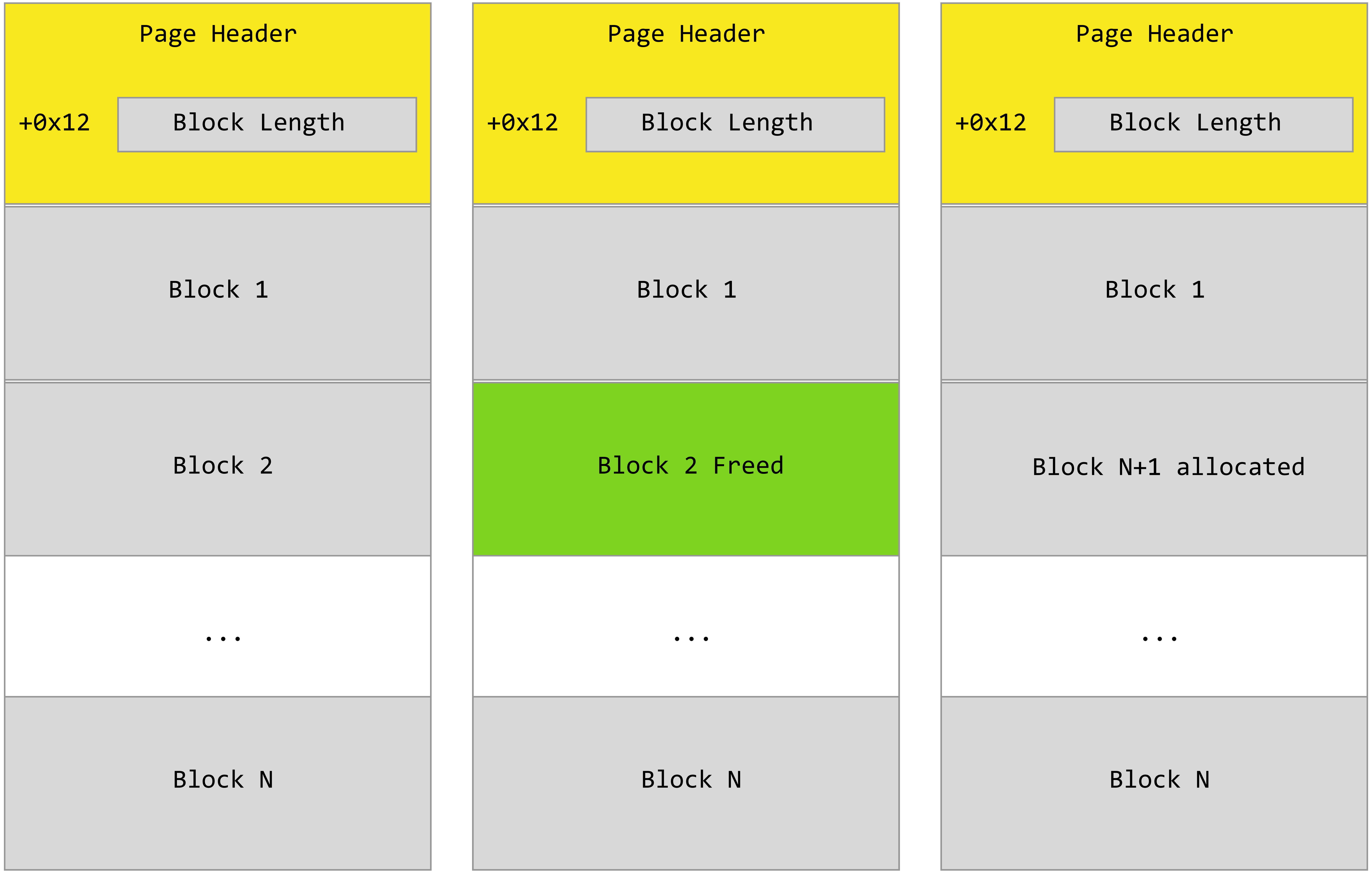
- Blocks > 0x7F0 are allocated by system
- Need small block, there is an appropriate «freed» block - return the «freed» block
- There is no appropriate «freed» block - allocate large memory page, divide it to small equal blocks, return one block

Small blocks allocation

- Memory page contains a number of blocks
- All blocks have the same length
- Page Header contains Block Length field at offset 0x12
- Block size is aligned with 8 in blocks less than 0x80
- In other clocks size is aligned with 16



Flash memory allocation



Page Header

+0x12 Block Length

Block 1

Block 2

...

Block N

Page Header

+0x12 Block Length

Block 1

Block 2 Freed

...

Block N

Page Header

+0x12 Block Length

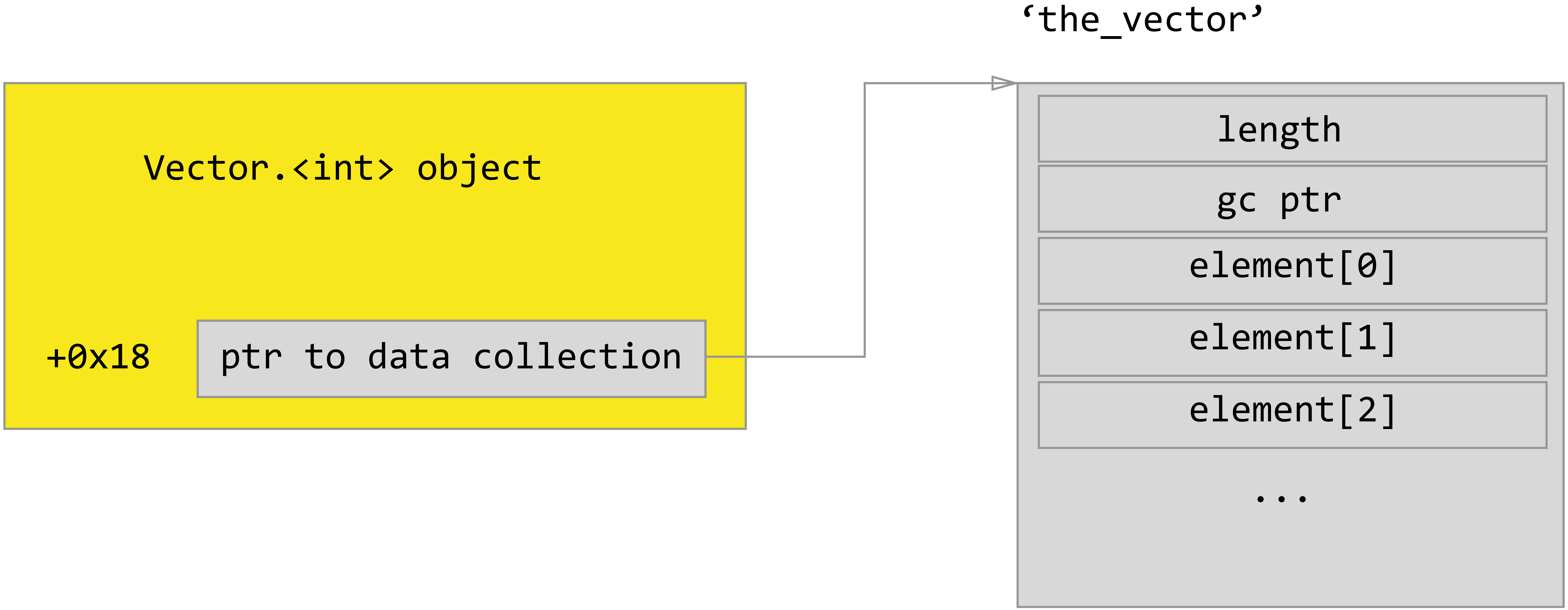
Block 1

Block N+1 allocated

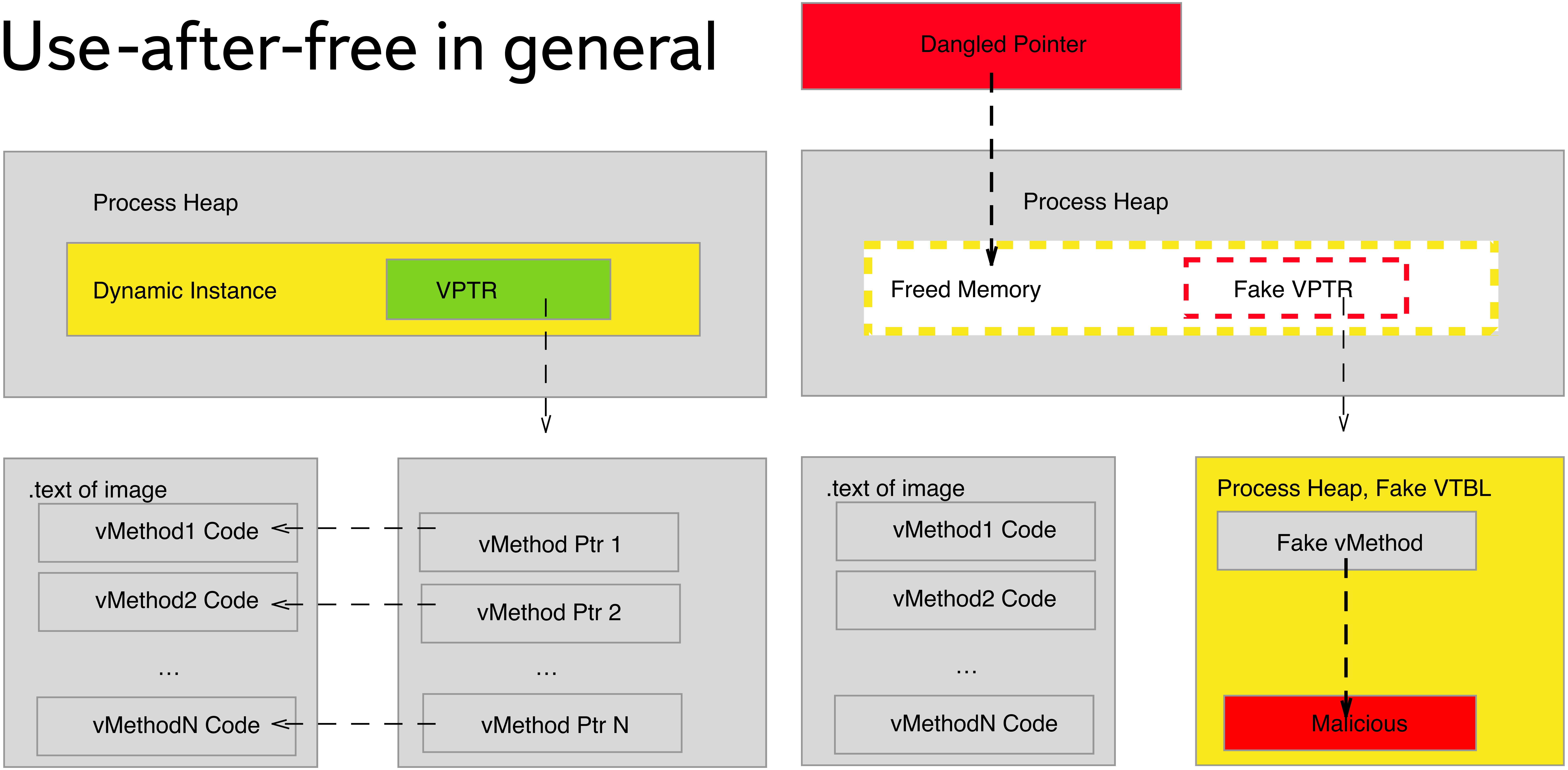
...

Block N

Flash Vector.<int> structure



Use-after-free in general



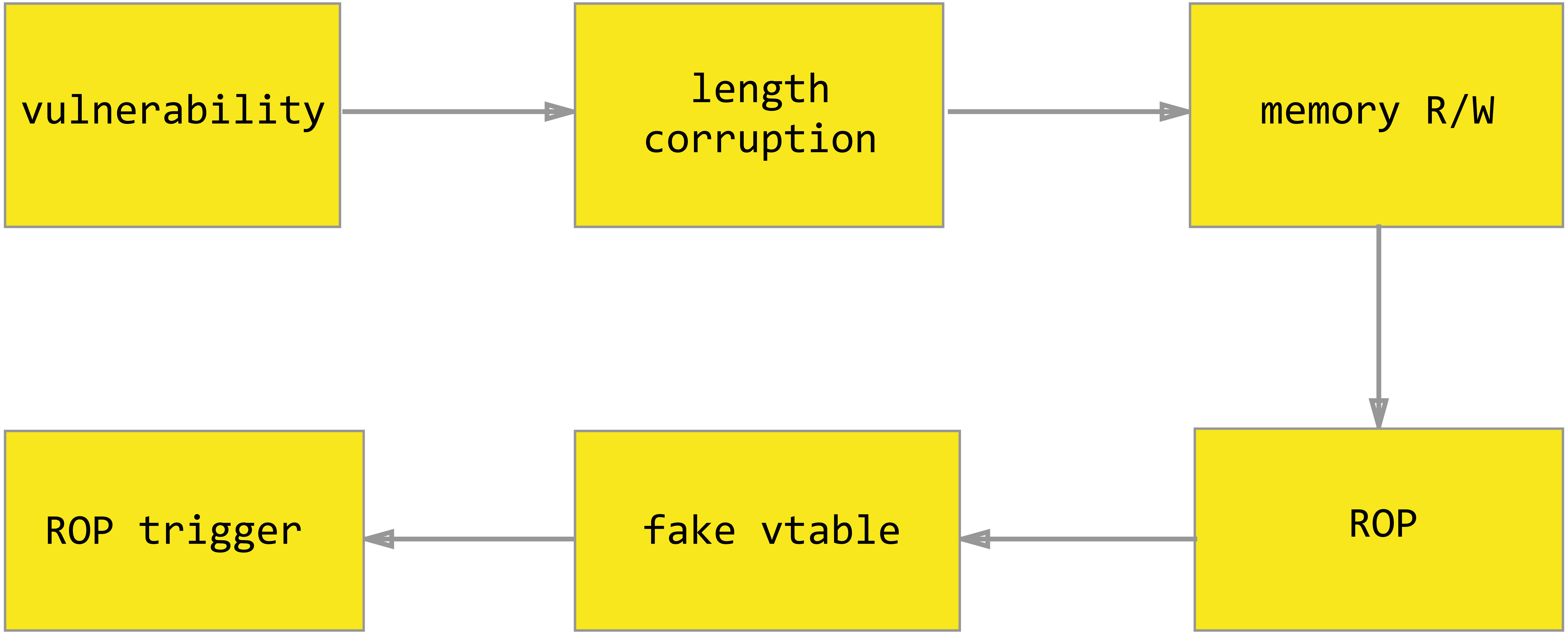
Vector<uint>.length corruption

- Developed in 2013 (CVE-2013-0634 Lady Boyle)
- Became a basis for exploits in 2014 and 2015

Length corruption exploitation

1. Corrupt Vector<uint>.length field (override it to be 0xffffffff)
2. Read / write arbitrary values in memory to create ROP chain
3. Create fake vtable with an entry that points to ‘pivot’ gadget
4. Overwrite vtable of any object with the fake vtable
5. Call virtual method (execute ROP chain)

Exploitation



vulnerability

length corruption

memory R/W

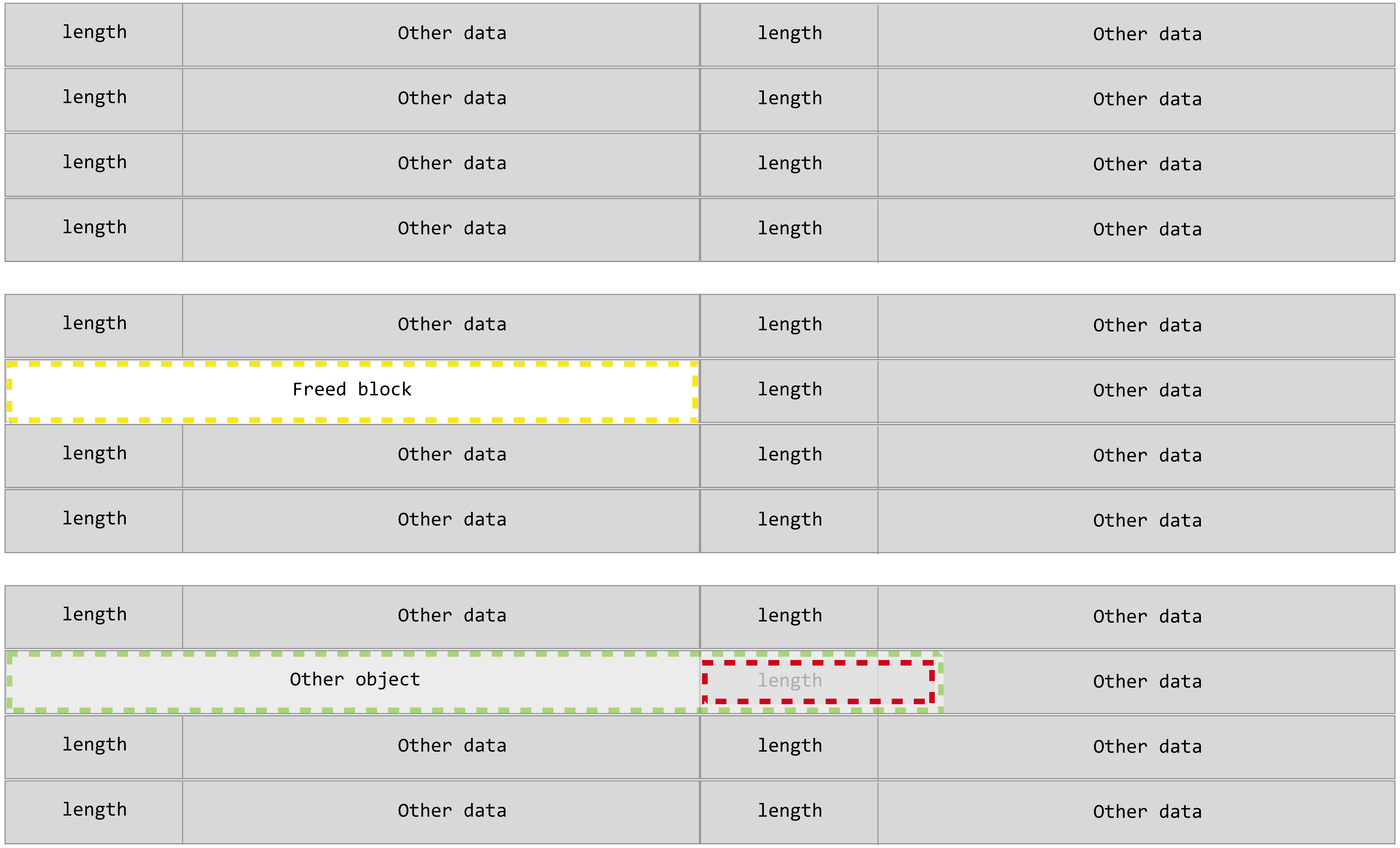
ROP

fake vtable

ROP trigger

Examples

CVE-2013-0634 (Lady Boyle)



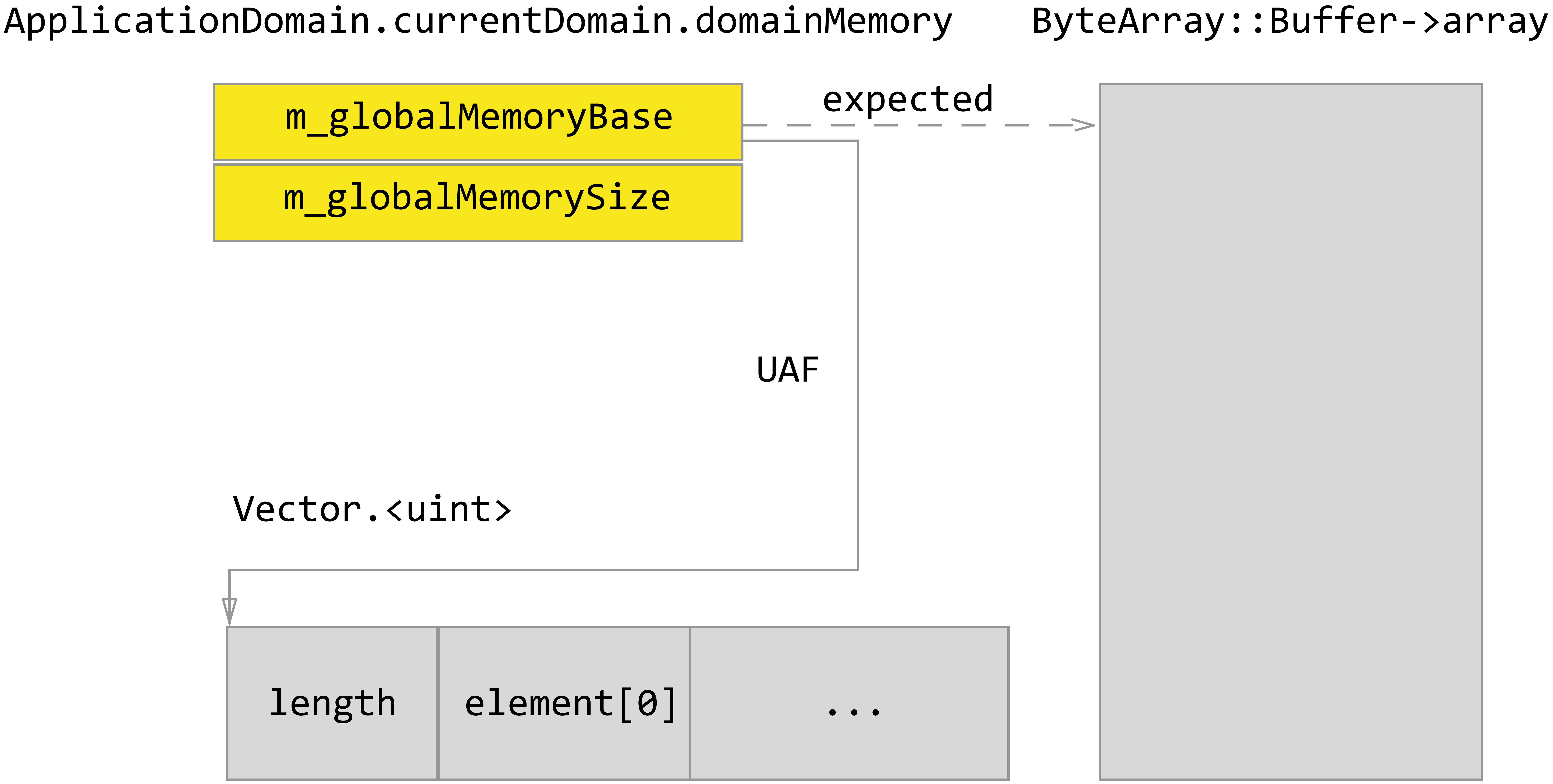
length	Other data	length	Other data
length	Other data	length	Other data
length	Other data	length	Other data
length	Other data	length	Other data

length	Other data	length	Other data
Freed block		length	Other data
length	Other data	length	Other data
length	Other data	length	Other data

length	Other data	length	Other data
Other object		length	Other data
length	Other data	length	Other data
length	Other data	length	Other data

```
_loc_2 = "(?i)()()(?-i)||||||||||||||||||||||";
var _loc_20:* = new RegExp(_loc_2, "");
```

CVE-2015-0311 (domainMemory UAF)



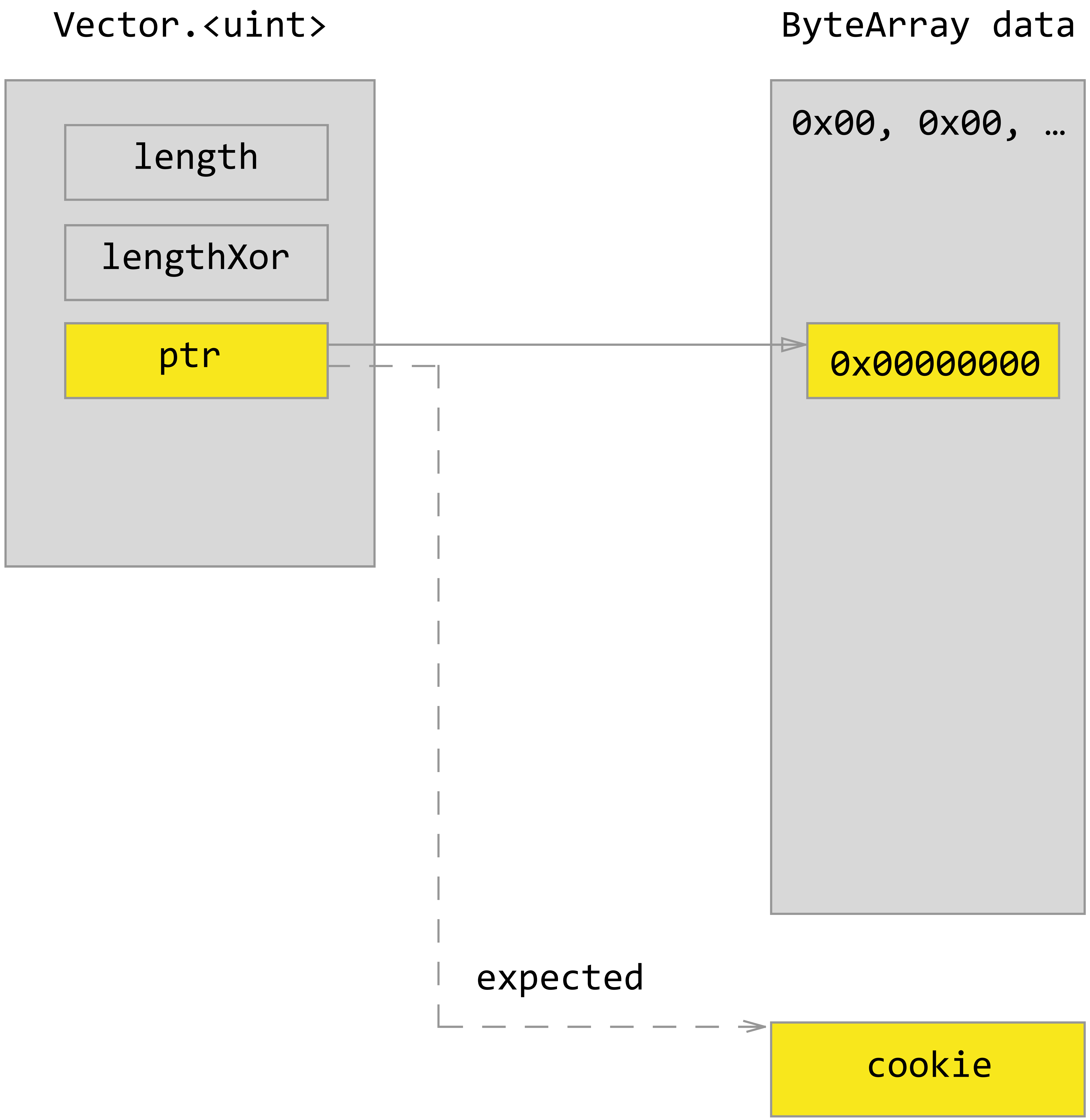
Length corruption mitigations

Google Mitigation: buffer heap partitioning

- Vector.<int> buffers are allocated in the separate heap
- Not powerful on 32-bit systems and 32-bit software due to address space limitations

Google Mitigation: Vector.<*> length validation

- An additional cookie and additional check
- assert(len ^ lenXor == ptr->cookie);



Vector.<uint>

length

lengthXor

ptr

ByteArray data

0x00, 0x00, …

0x00000000

expected

cookie

Vector.<*> length validation bypass

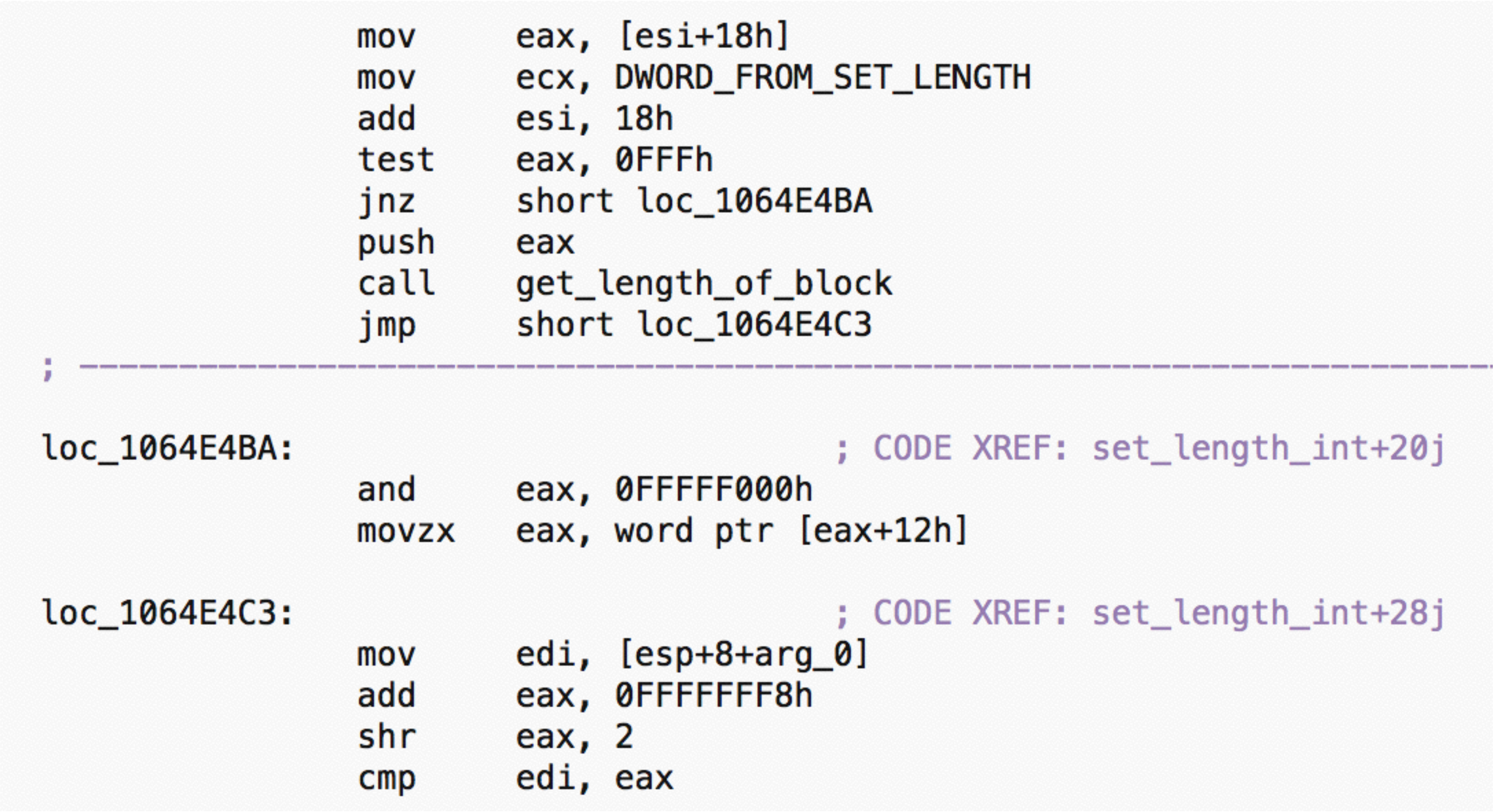
- ptr->cookie was near length field
- Allocate big ByteArray (1Gb) with null values
- Corrupt cookie ptr to point inside the ByteArray (cookie value becomes 0)
- Corrupt len and lenXor with same values
- Fixed by moving cookie from the ptr (to one of the image sections)

Length corruption detection

Length corruption detection

- Behavioral
- Stateless
- Reliable
- With small overhead

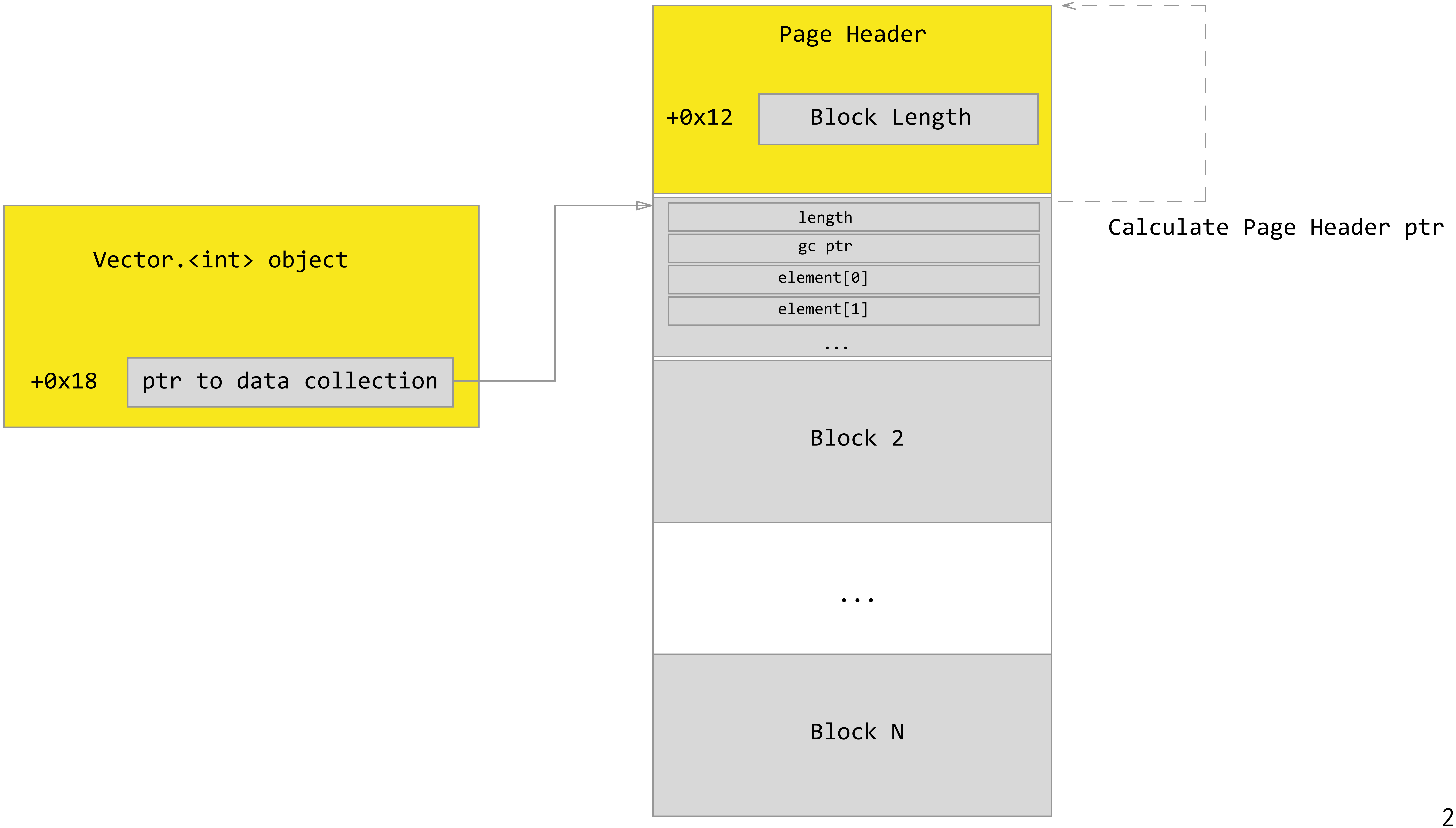
Vector.<int> set_length



```
                mov     eax, [esi+18h]
                mov     ecx, DWORD_FROM_SET_LENGTH
                add     esi, 18h
                test    eax, 0FFFh
                jnz     short loc_1064E4BA
                push    eax
                call    get_length_of_block
                jmp     short loc_1064E4C3
; -----------------------------------------------------------------------

loc_1064E4BA:                           ; CODE XREF: set_length_int+20j
                and     eax, 0FFFFF000h
                movzx   eax, word ptr [eax+12h]

loc_1064E4C3:                           ; CODE XREF: set_length_int+28j
                mov     edi, [esp+8+arg_0]
                add     eax, 0FFFFFFF8h
                shr     eax, 2
                cmp     edi, eax
```



Page Header

+0x12 Block Length

Vector.<int> object

+0x18 ptr to data collection

length

gc ptr

element[0]

element[1]

...

Block 2

...

Block N

Calculate Page Header ptr

‘Big’ vectors

- ‘Big’ vectors can use more than one page
- Use ‘get_length_of_block’ function

Vector<*>.length corruption detection

- Set hook to set_length, [..] operator etc.
- In the hook functions:

    1. calculate ptr to Page Header
    2. get Page Header block size
    3. compare it with the value from Vector<..>.length
       detect if vector<..>.length != (Block Size - 8) / sizeof(element)

Results

- The mitigation works only for Vector<uint>.length etc.
- Made vector<..>.length corruption technique almost useless

Length corruption technique

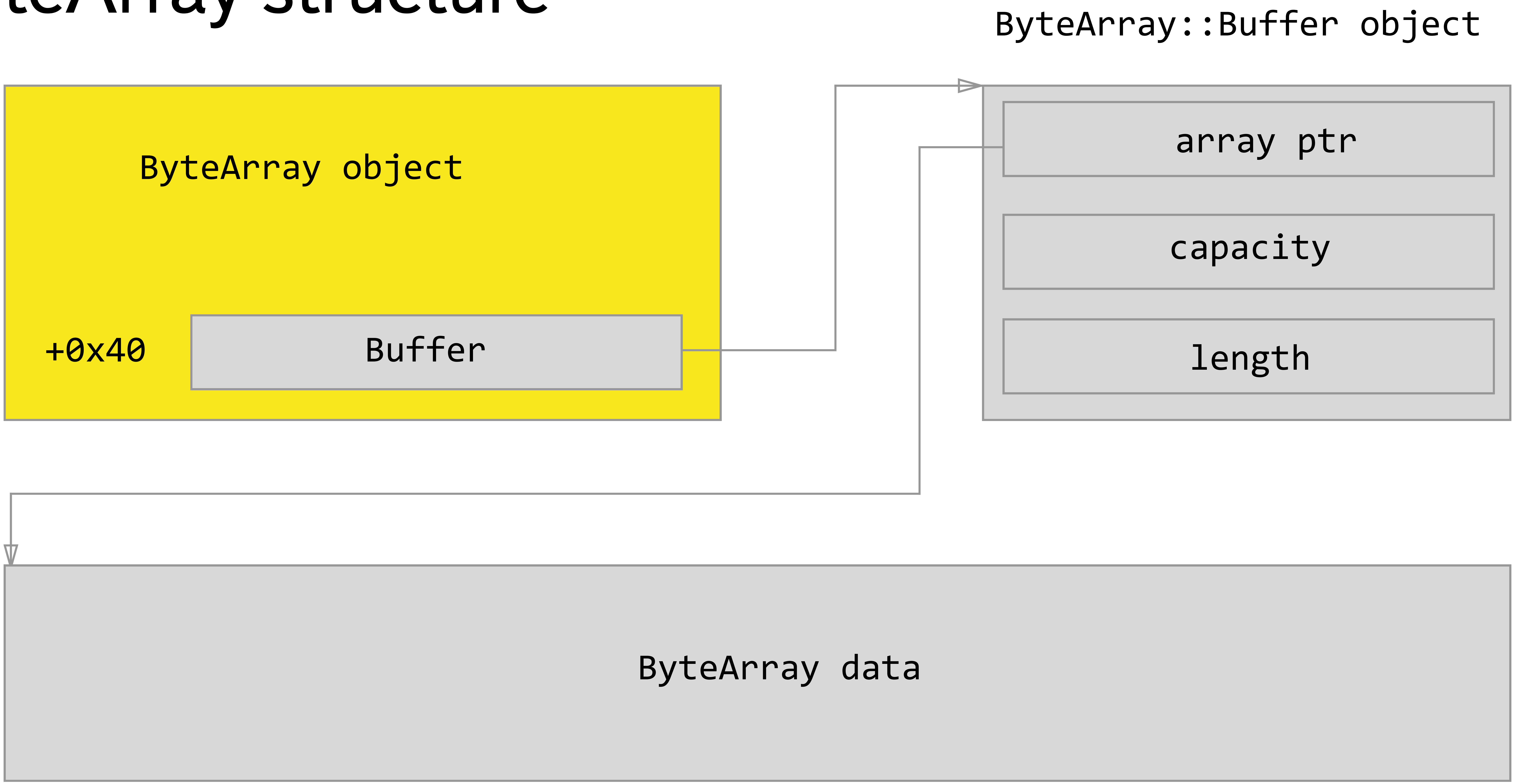
Image from http://raunds.townvoice.co.uk/wp-content/uploads/sites/20/2014/02/Rip_cat-269x300.jpg

Length corruption strikes back

Length corruption strikes back

- The mitigation works only for Vector<..>.length
- Are there other similar objects?
- ByteArray is similar to Vector
- ByteArray length corruption is used in CVE-2015-7645

ByteArray structure



ByteArray::Buffer object

ByteArray object

+0x40

Buffer

array ptr

capacity

length

ByteArray data

Conclusions

- We developed a reliable detection approach for Length corruption technique
- A reliable mitigation for other Vector-like objects is needed

To read

- Haifei Li ‘Smashing the Heap with Vector’ http://bit.ly/1X61uZD
- F. Falcon ‘Exploiting Adobe Flash Player in the era of CGF’ http://ubm.io/1Ynqk4g
- Ga1lois & Bo Qu ‘Inside Flash: Flash Exploit Detection Uncovered’ http://bit.ly/1QBGxlc
- Project Zero Blog http://googleprojectzero.blogspot.ru

Contacts

✉ Andrew Kovalev <avkov@yandex-team.ru>

✉ Konstantin Otrashkevich <freeoks@yandex-team.ru>

✉ Evgeny Sidorov <e-sidorov@yandex-team.ru>