

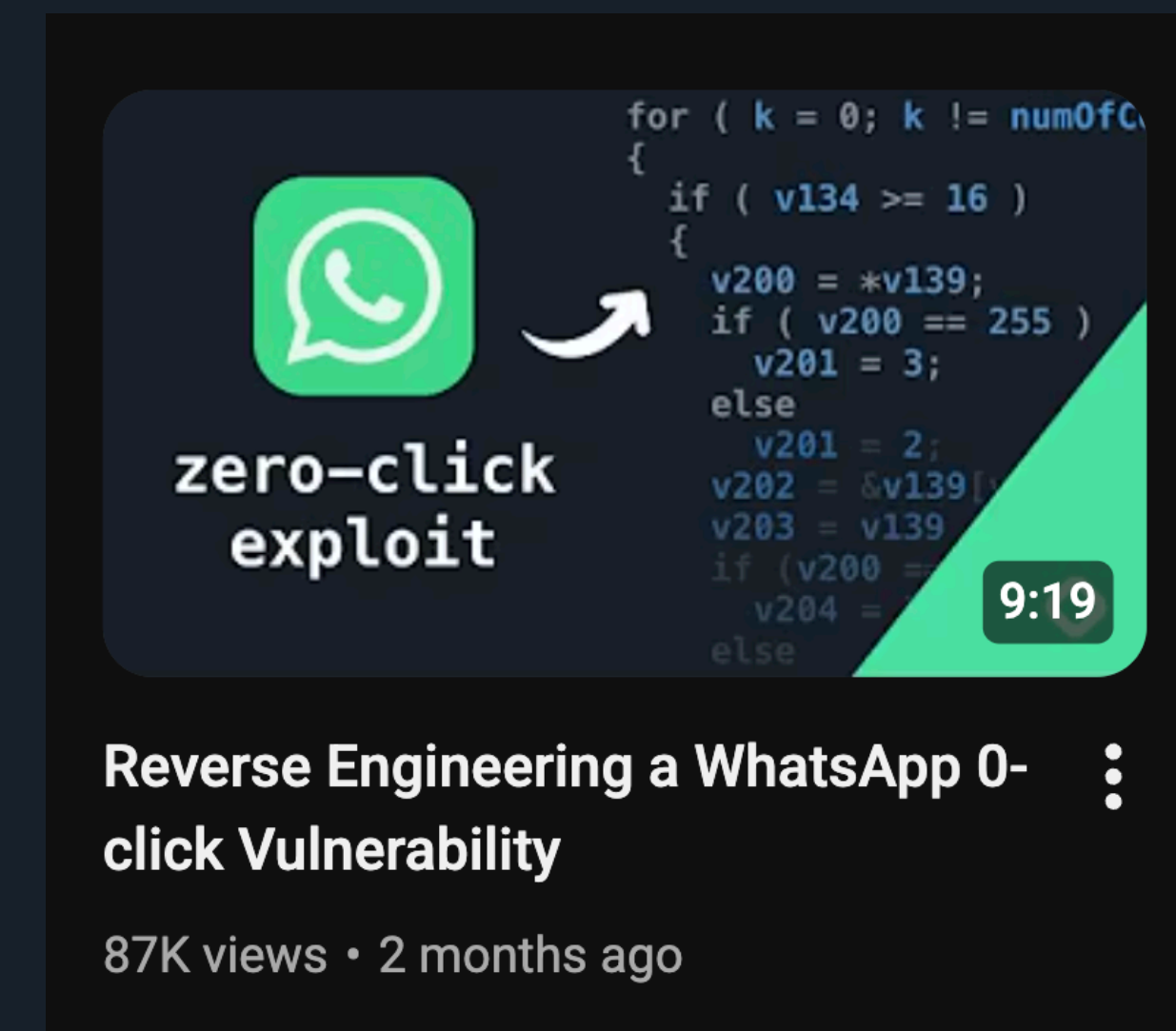
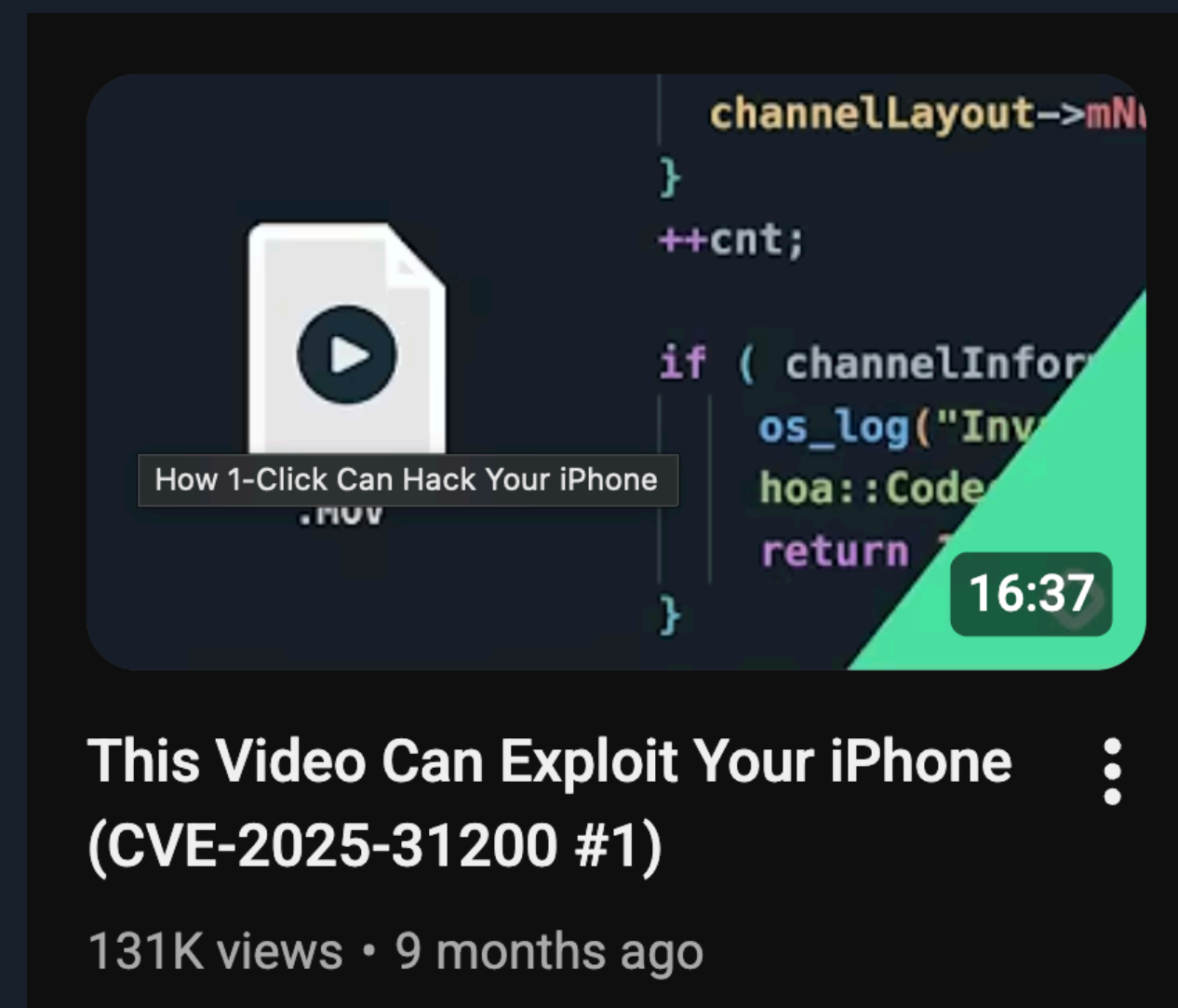
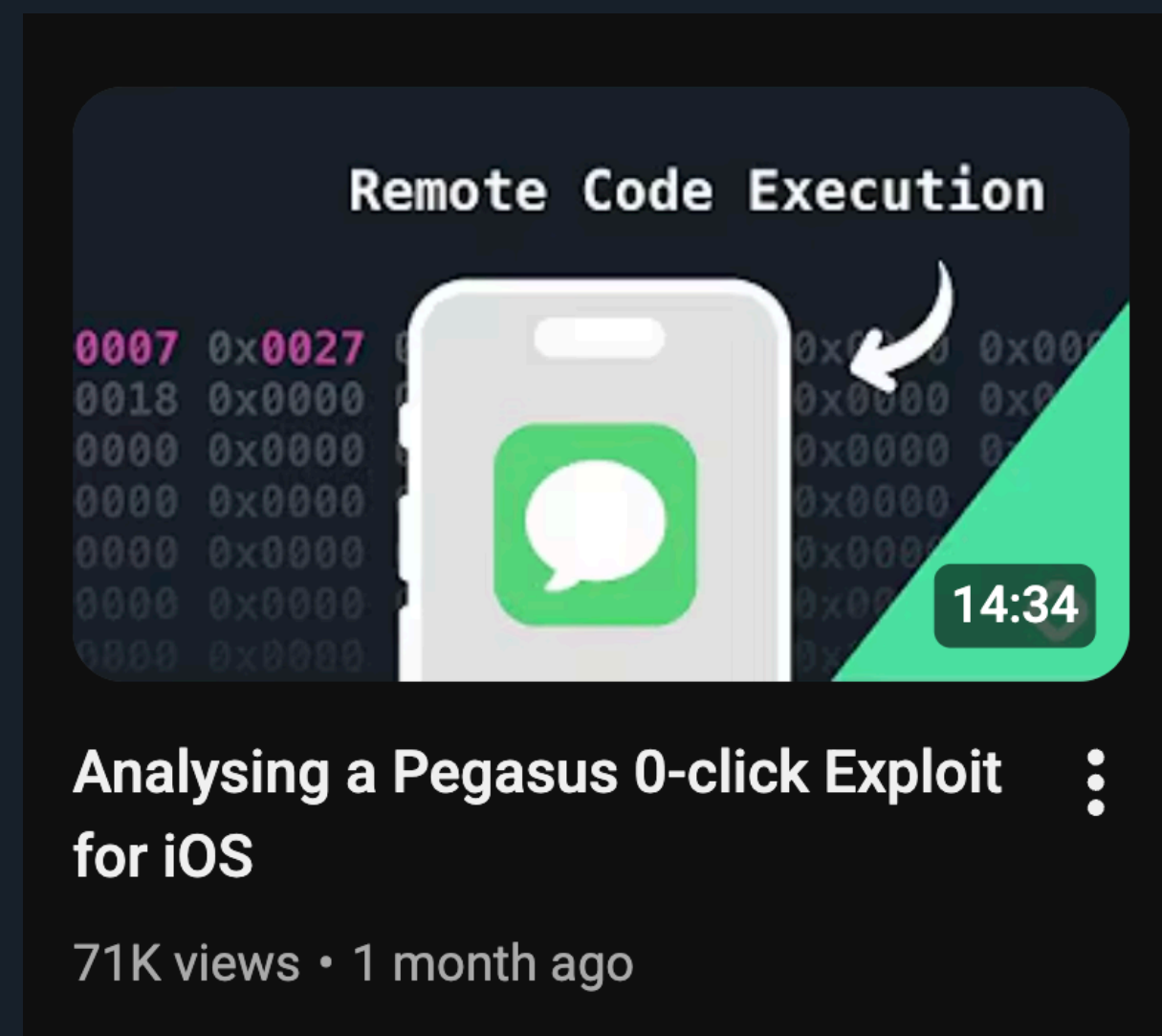
Reading between the patches: Getting started with **iOS security**

~ Billy Ellis



About me

- iOS researcher from London, UK
- Focused on vulnerability research & exploit development
- Provide iOS technical breakdowns on YouTube



Agenda

- iOS security overview
- Research setup in 2026
- Patch-diffing your first bug

Where do we start?

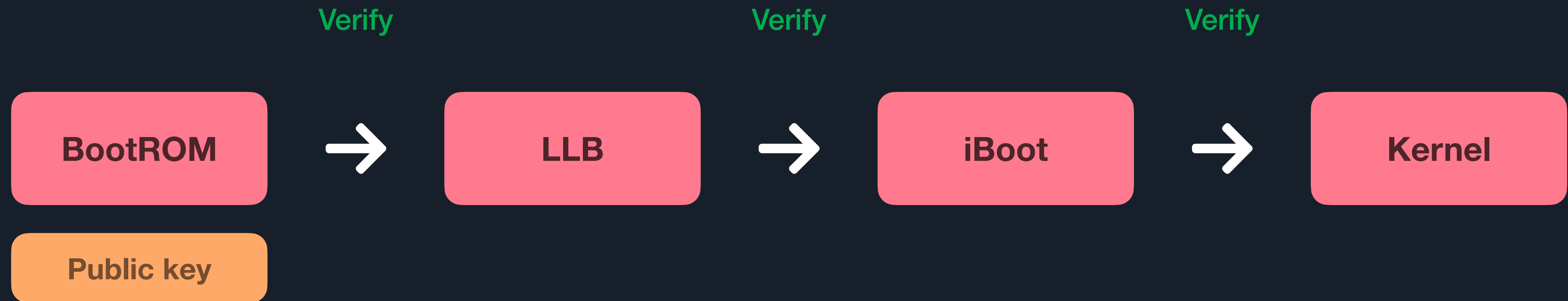


iOS security model

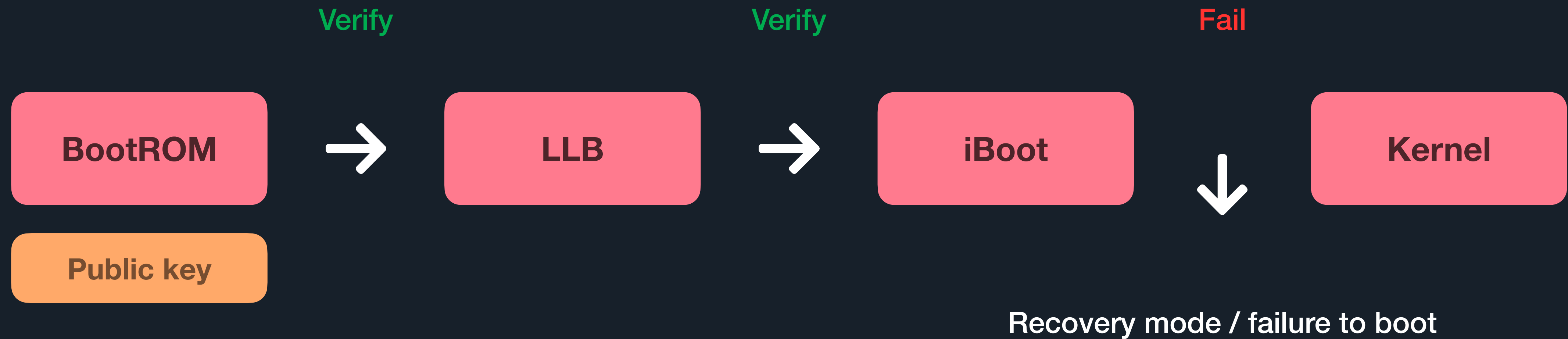
iOS security model

- Secure boot chain
- Codesigning
- Sandboxing

Secure boot chain



Secure boot chain



Codesigning

- All executable code is signed by trusted party
- Signature verification enforced at kernel level



Codesigning

- Strict implementation in modern iOS
- Hardware-backed enforcements (TXM, SPTM)



Codesigning

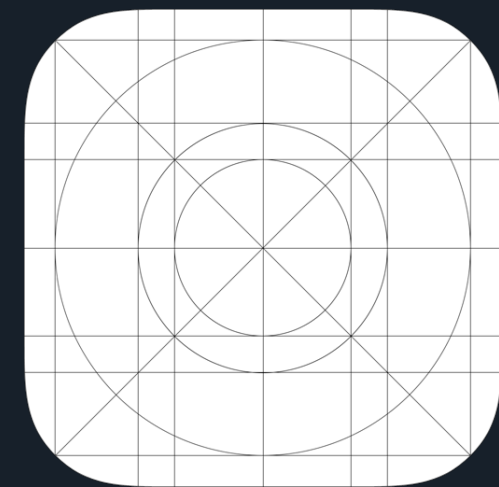
- Prevents code tampering
- Facilitates entitlements



Sandboxing

- Restrict process access to attack surface
- 'Risky' functionality is proxied through daemon (i.e. mediaplaybackd)

Sandboxing



My app



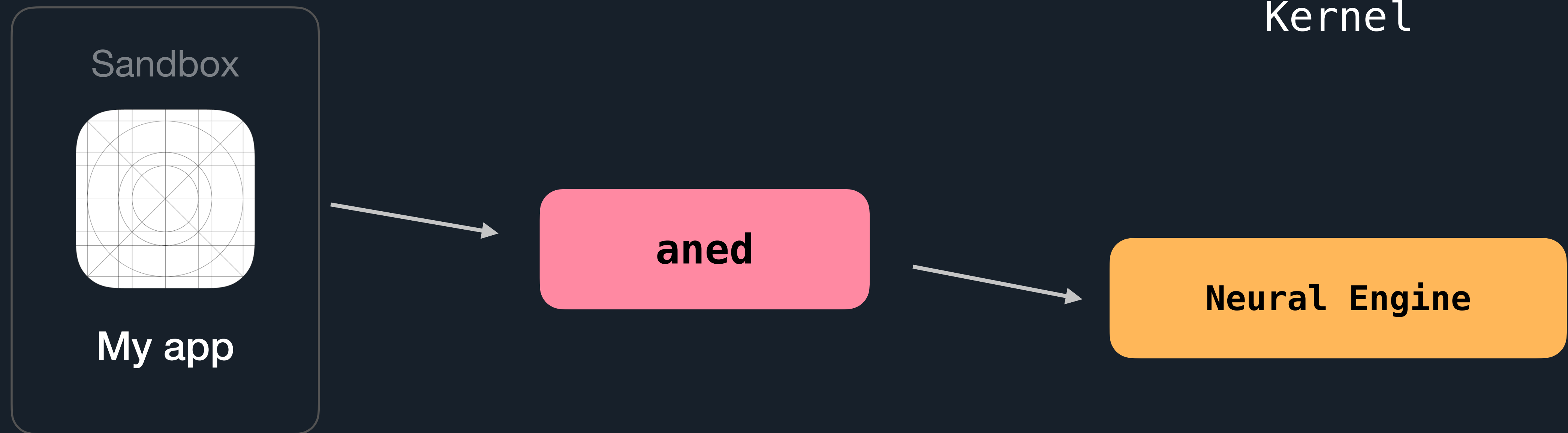
Kernel

Video Decoder

Neural Engine

Graphics Processing

Sandboxing



Sandboxing

- Implemented in the kernel via MACF hooks
- Check against sandbox profile on each operation

```
625      (allow iokit-open
626          (iokit-user-client-class "IOAccelContext")
627          (iokit-user-client-class "IOAccelContext2")
628          (iokit-user-client-class "IOAccelDevice"))
```


Modern mitigations

- Hardware-based Apple mitigations
- Making use of custom registers / CPU features

The attacker's perspective

Aim: exploit bugs, break security features, run
privileged code on device

Anatomy of an iOS exploit chain



Anatomy of an iOS exploit chain (simplified)



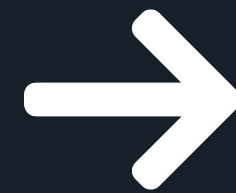
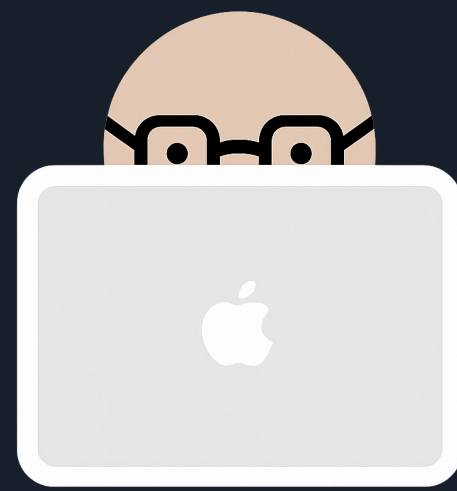
Anatomy of an iOS exploit chain (simplified)



Remote Code Execution (RCE)

- Stage1 exploit
- Get initial code running on target device

Remote Code Execution (RCE)



Remote Code Execution (RCE)

- WebKit memory corruption
- Image/video/audio corruption (browser & messages)
- Bluetooth, AirDrop, close-proximity corruption?

Remote Code Execution (RCE)

- WebKit memory corruption
- Image/video/audio corruption (browser & messages)
- Bluetooth, AirDrop, close-proximity corruption?
- A bug in **anything** that can be reached from the outside world

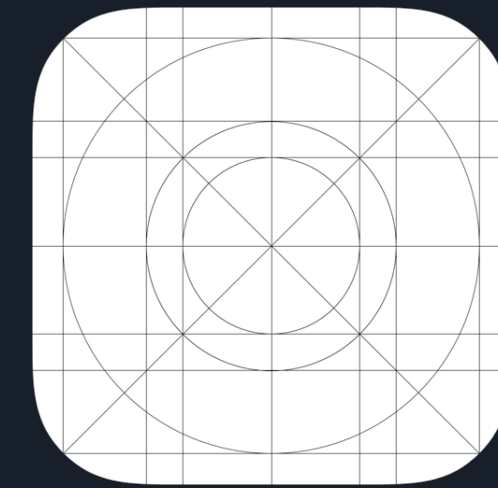
Remote Code Execution (RCE)



Browser exploit



Messaging app
exploit



3rd party app
trojan (jailbreak
app)

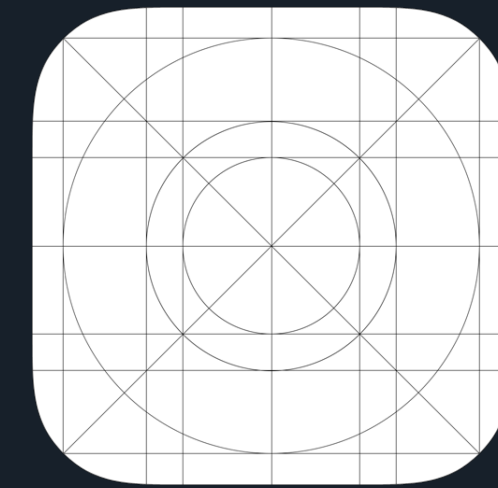
Remote Code Execution (RCE)



Browser exploit



Messaging app
exploit



3rd party app
trojan (jailbreak
app)



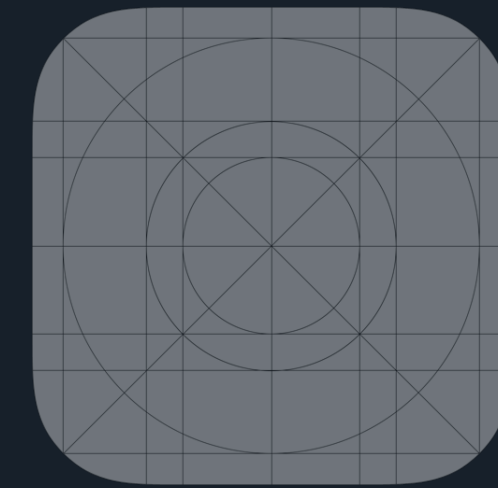
Remote Code Execution (RCE)



Browser exploit



Messaging app
exploit



3rd party app
trojan (jailbreak
app)

WebKit bugs

- Memory corruption in browser engine
- Get r/w in renderer process





JavaScript environment

WebKit engine



```
// this is the bug trigger: CVE-2020-9802
function hax(arr, n) {
  n &= 0xffffffff;

  if (n < -1) {

    let v = (-n)&0xffffffff;
    let i = Math.abs(n);

    if (i < arr.length) {
      if (i & 0x80000000) {
        i += -0x7fffffff9;
      }
      if (i > 0) {
        arr[i] = 1.04380972981885e-310;
      }
    }
  }
}
```

JavaScript environment

WebKit engine




Old JIT bug

```
// this is the bug trigger: CVE-2020-9802
function hax(arr, n) {
  n &= 0xffffffff;

  if (n < -1) {

    let v = (-n)&0xffffffff;
    let i = Math.abs(n);

    if (i < arr.length) {
      if (i & 0x80000000) {
        i += -0x7fffffff9;
      }
      if (i > 0) {
        arr[i] = 1.04380972981885e-310;
      }
    }
  }
}
```

JavaScript environment

WebKit engine



```
// this is the bug trigger: CVE-2020-9802
function hax(arr, n) {
  n &= 0xffffffff;

  if (n < -1) {

    let v = (-n)&0xffffffff;
    let i = Math.abs(n);

    if (i < arr.length) {
      if (i & 0x80000000) {
        i += -0x7fffffff9;
      }
      if (i > 0) {
        arr[i] = 1.04380972981885e-310;
      }
    }
  }
}
```

JavaScript environment

WebKit engine

Confuse
state of the
engine





```
// this is the bug trigger: CVE-2020-9802
function hax(arr, n) {
  n &= 0xffffffff;

  if (n < -1) {

    let v = (-n)&0xffffffff;
    let i = Math.abs(n);

    if (i < arr.length) {
      if (i & 0x80000000) {
        i += -0x7fffffff9;
      }
      if (i > 0) {
        arr[i] = 1.04380972981885e-310;
      }
    }
  }
}
```

JavaScript environment

Corrupt
JSArray

WebKit engine

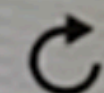
Confuse
state of the
engine



iPod

19:26

192.168.1.191



de exploit

Copy

Look Up

Share...

```
[+] float_arr JSArray len is 0x1337
[+] addrof target is 0x000000010dd80920
[+] addrof float_arr is 0x000000010dd80930
[+] addrof obj_arr is 0x000000010dd80940
[+] addrof hax is 0x0000000102771050
[+] addrof target1 is 0x000000010dd80980
[+] addrof float_arr1 is 0x000000010dd80990
```


WebKit bugs

- Memory corruption in browser engine
- Get r/w in renderer process



WebKit bugs



- Memory corruption in browser engine
- Get r/w in renderer process
- Jump to shellcode in JIT memory!



rwX

Load shellcode on JIT page



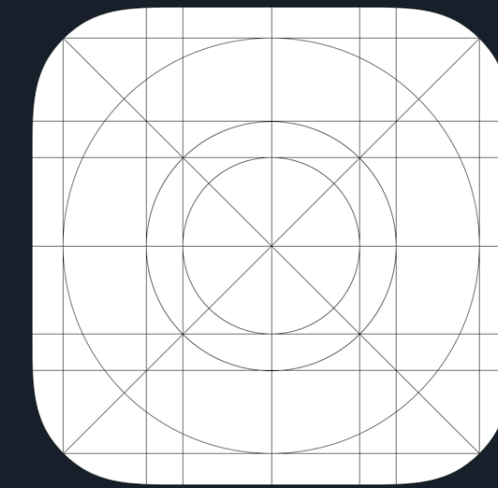
Remote Code Execution (RCE)



Browser exploit



Messaging app
exploit



3rd party app
trojan (jailbreak
app)

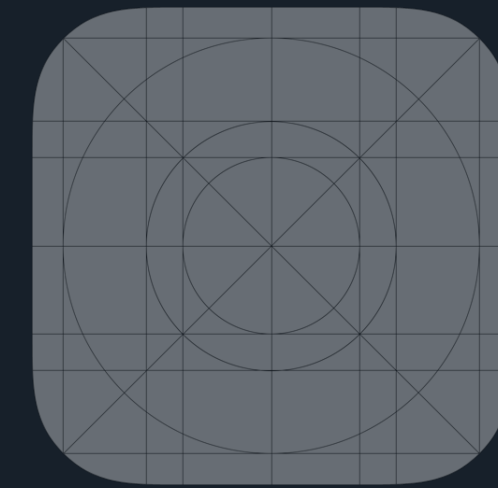
Remote Code Execution (RCE)



Browser exploit



Messaging app
exploit

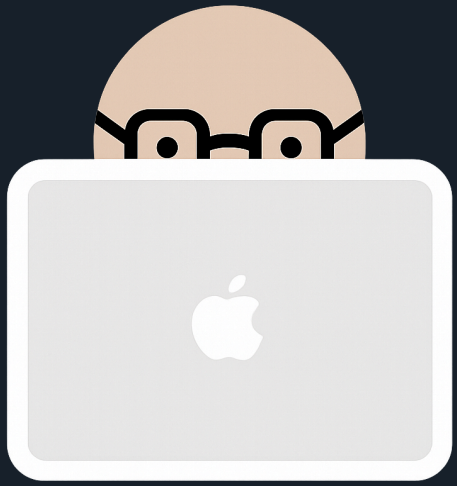


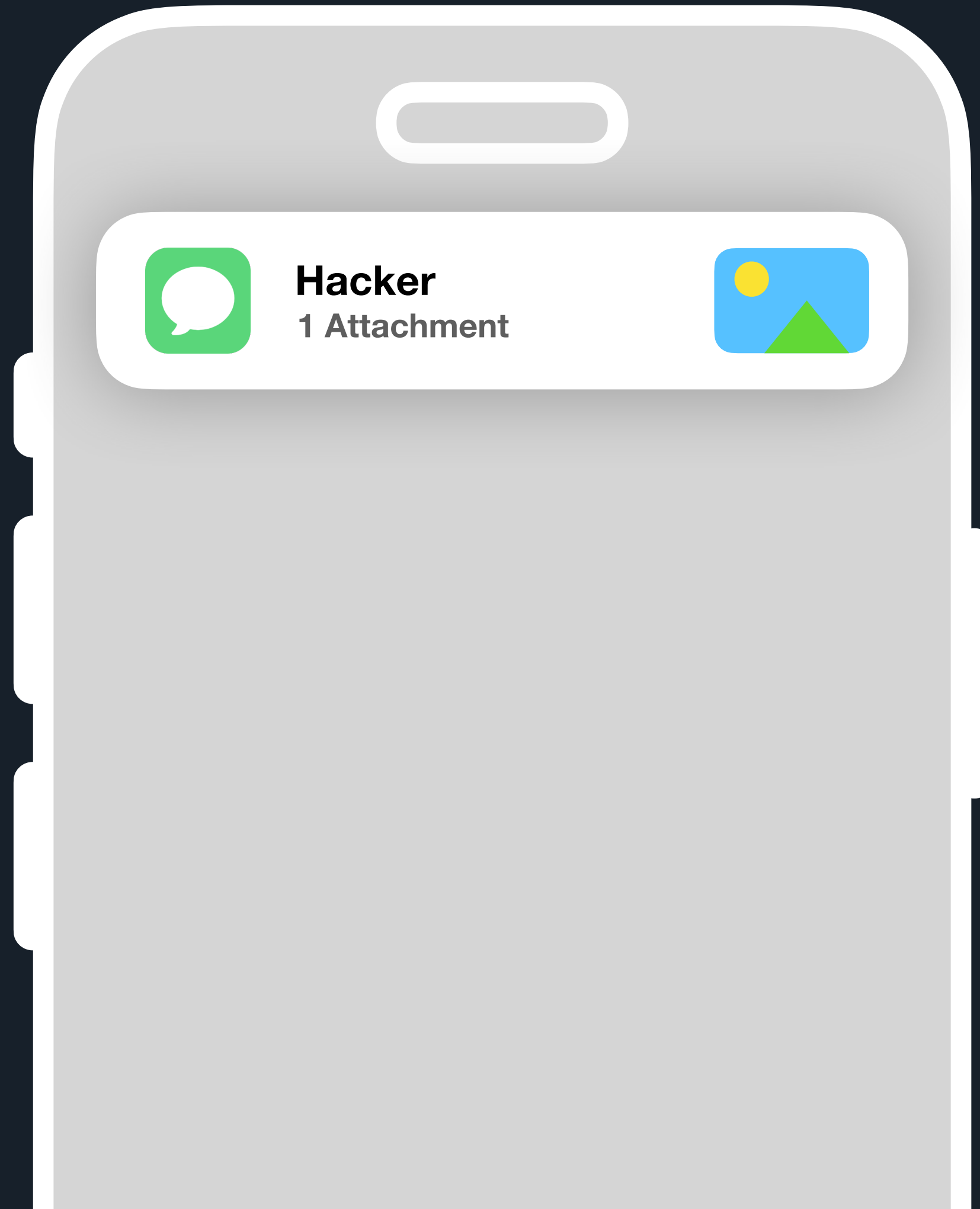
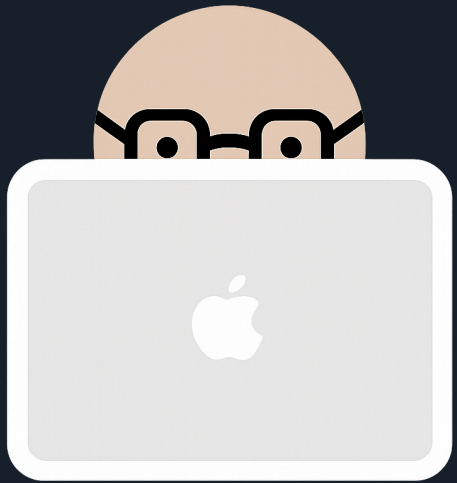
3rd party app
trojan (jailbreak
app)

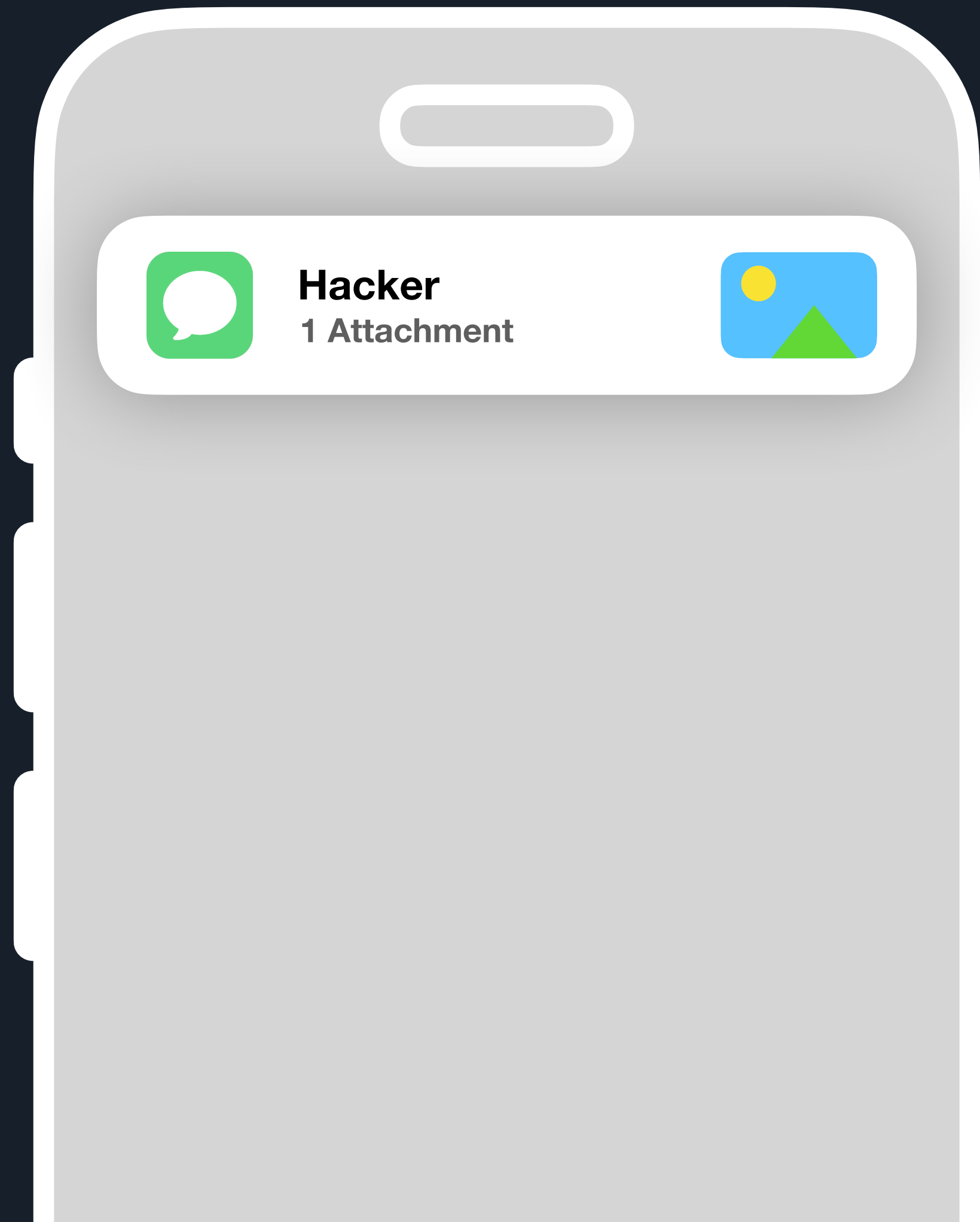
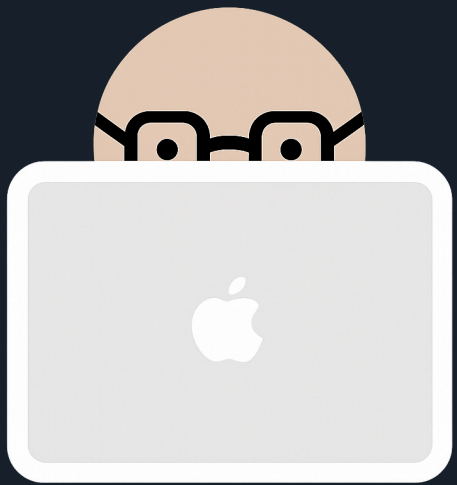
Message app bugs

- 0click or 1click
- Parsing issues for media formats
- (Images, video, audio, PDF, fonts, etc)







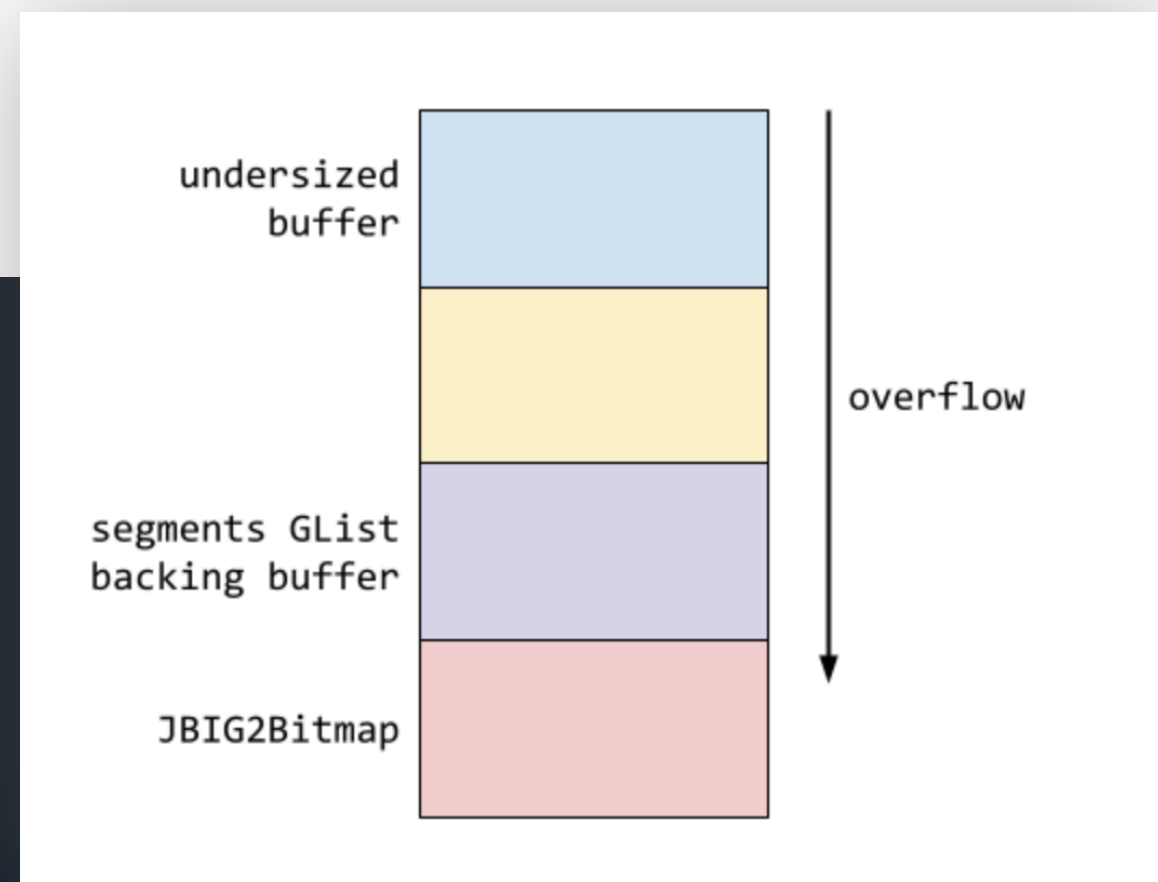


Thumbnail renders
with 0 interaction

Message app bugs

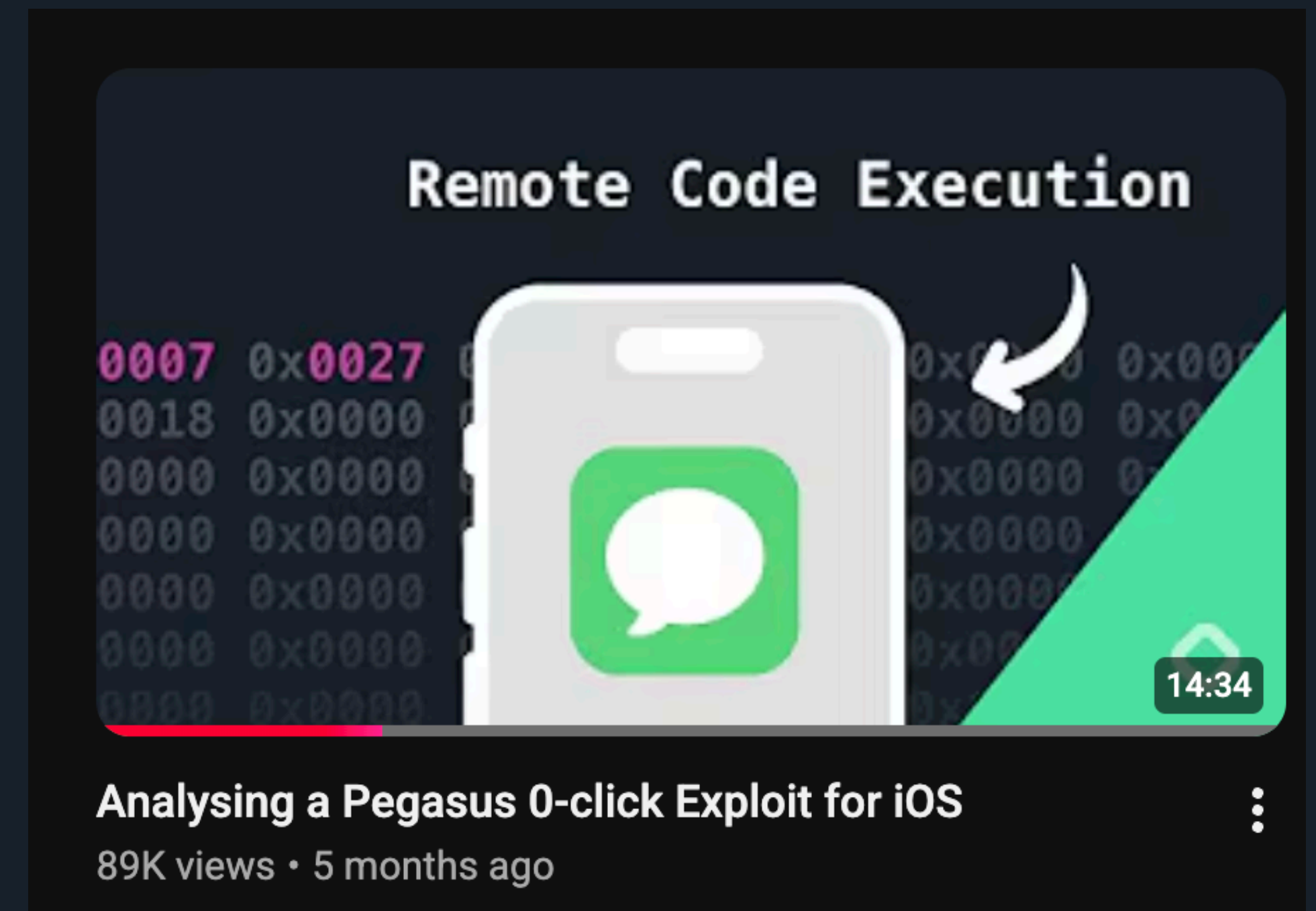
A deep dive into an NSO zero-click iMessage exploit: Remote Code Execution

2021-DEC-15 Ian Beer



JBIG2
parsing bug

Webp
parsing bug



Message app bugs



21:45 DNGerousLINK: A Deep Dive into WhatsApp 0- 22:45 Click Exploits on iOS and Samsung Devices

Tag 1

📍 Fuse 👤 ZHONGRUI LI 👤 YIZHE ZHUANG 👤 KIRA CHEN 🗣️ Englisch

📺 AUFGEZEICHNET ★ OFFIZIELL SECURITY

The spyware attack targeting WhatsApp, disclosed in August as an in-the-wild exploit, garnered significant attention. By simply knowing a victim's phone number, an attacker could launch a remote, zero-interaction attack against the WhatsApp application on Apple devices, including iPhones, iPads, and Macs.



DNG WhatsApp bug

Anatomy of an iOS exploit chain (simplified)



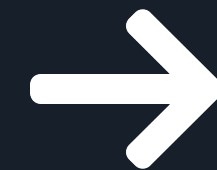
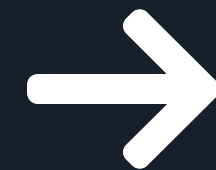
Anatomy of an iOS exploit chain (simplified)



Sandbox escape (SBX)

- Probably stuck in a limited sandbox
- Likely need wider access to reach the goal (kernel compromise)
- Not always required however

Pegasus iOS 9.x (~2015/2016)



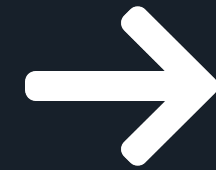
WebKit RCE
(CVE-2016-4657)

Sandbox
escape

Kernel info leak
(CVE-2016-4655)

Kernel code
execution
(CVE-2016-4656)

Pegasus iOS 9.x (~2015/2016)



WebKit RCE
(CVE-2016-4657)

Kernel info leak
(CVE-2016-4655)

Kernel code
execution
(CVE-2016-4656)

Pegasus iOS 9.x (~2015/2016)



WebKit RCE
(CVE-2016-4657)

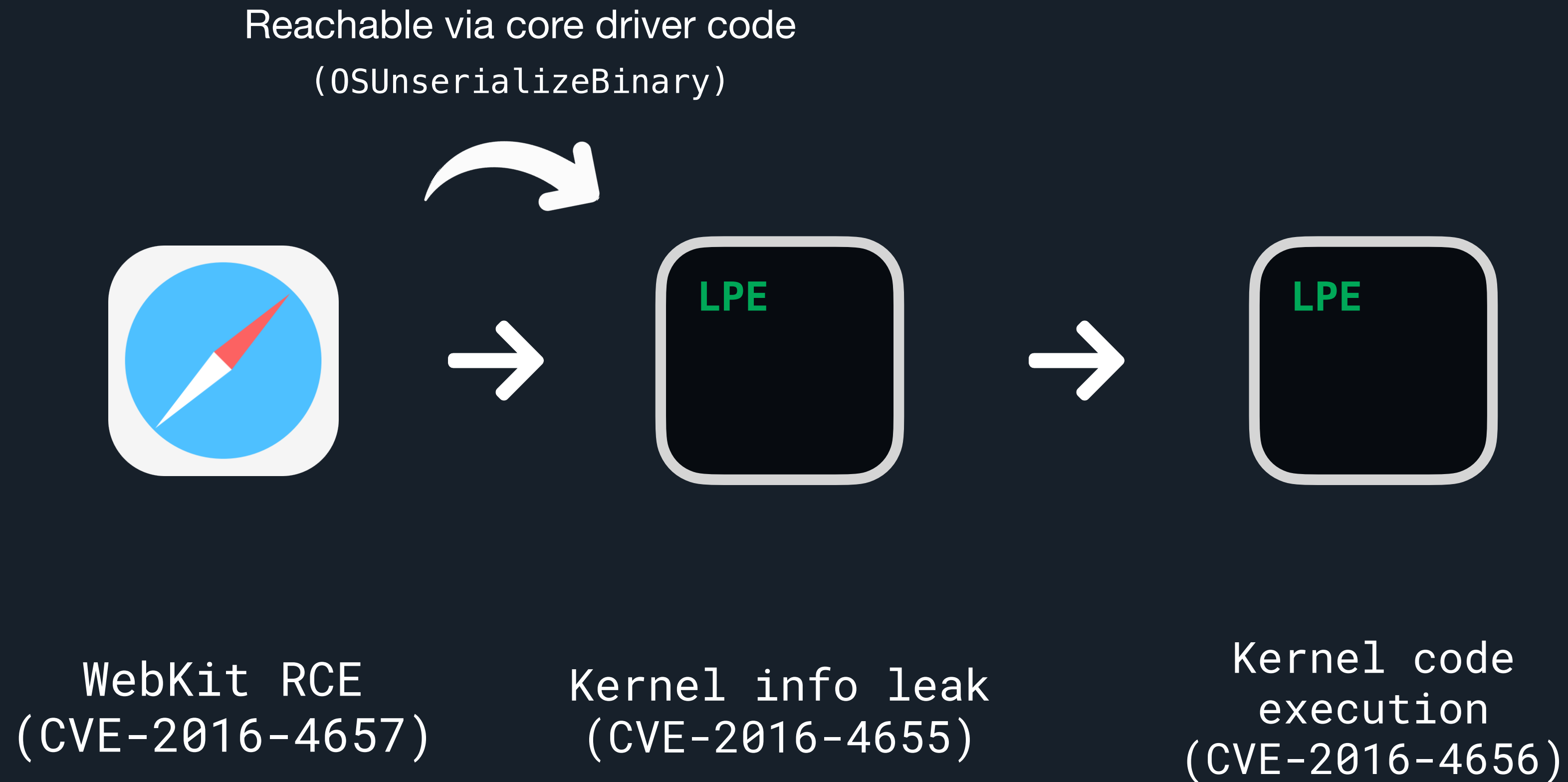


Kernel info leak
(CVE-2016-4655)



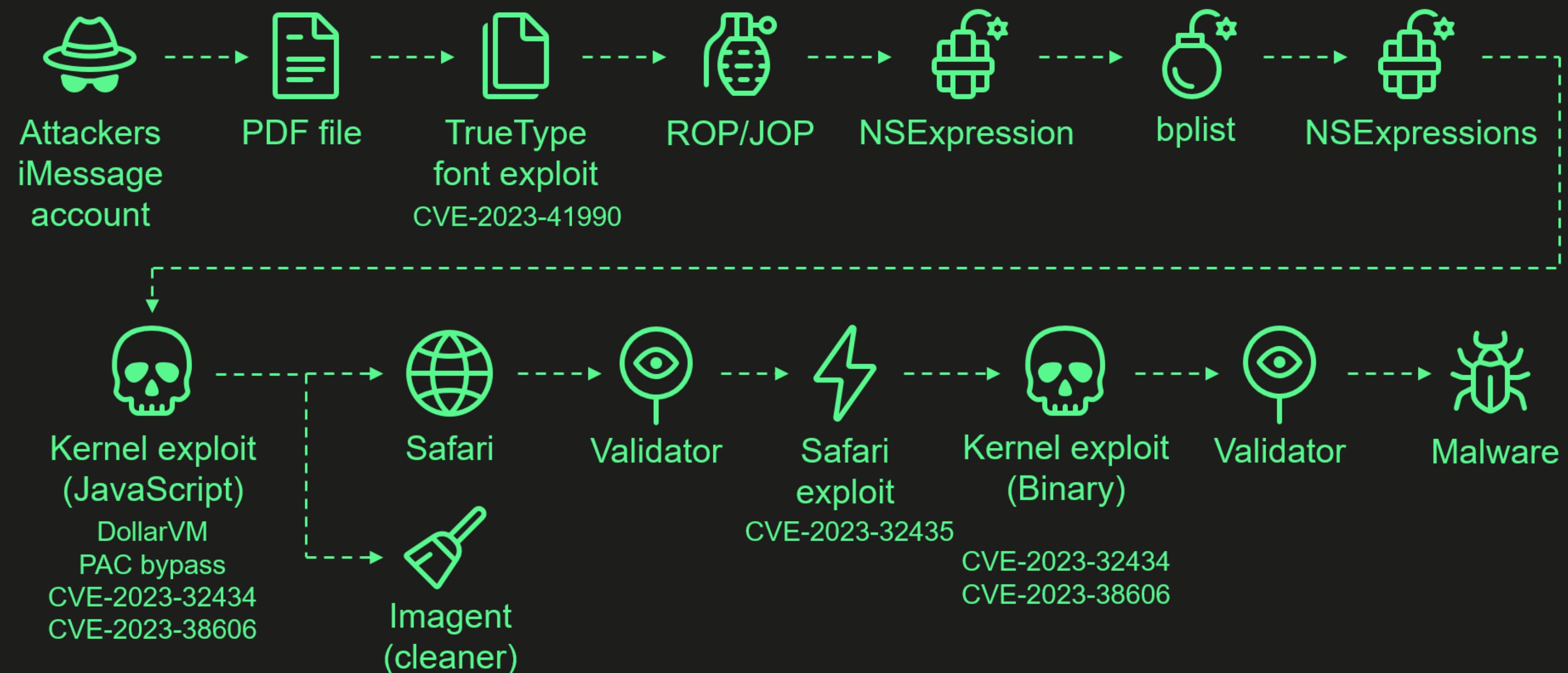
Kernel code
execution
(CVE-2016-4656)

Pegasus iOS 9.x (~2015/2016)



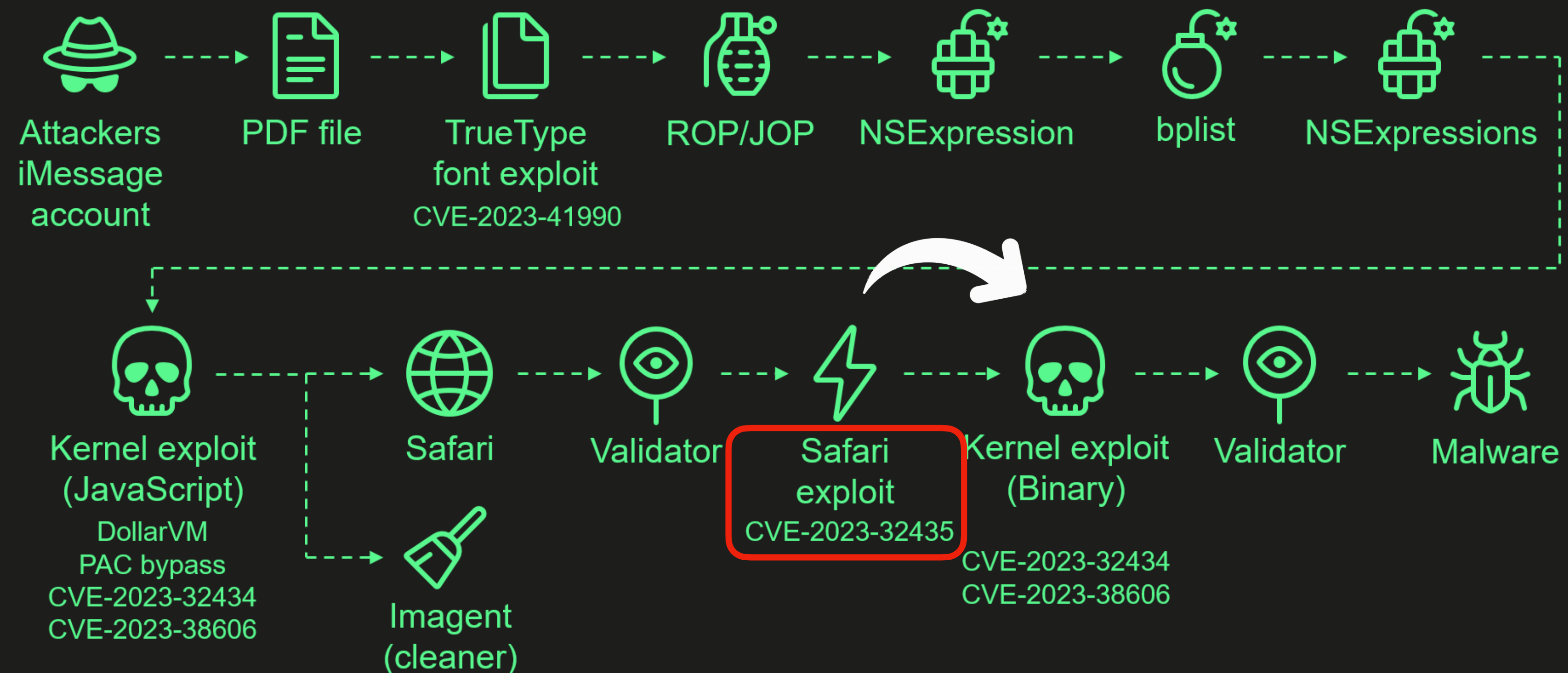
Operation Triangulation (~2023)

Attack chain



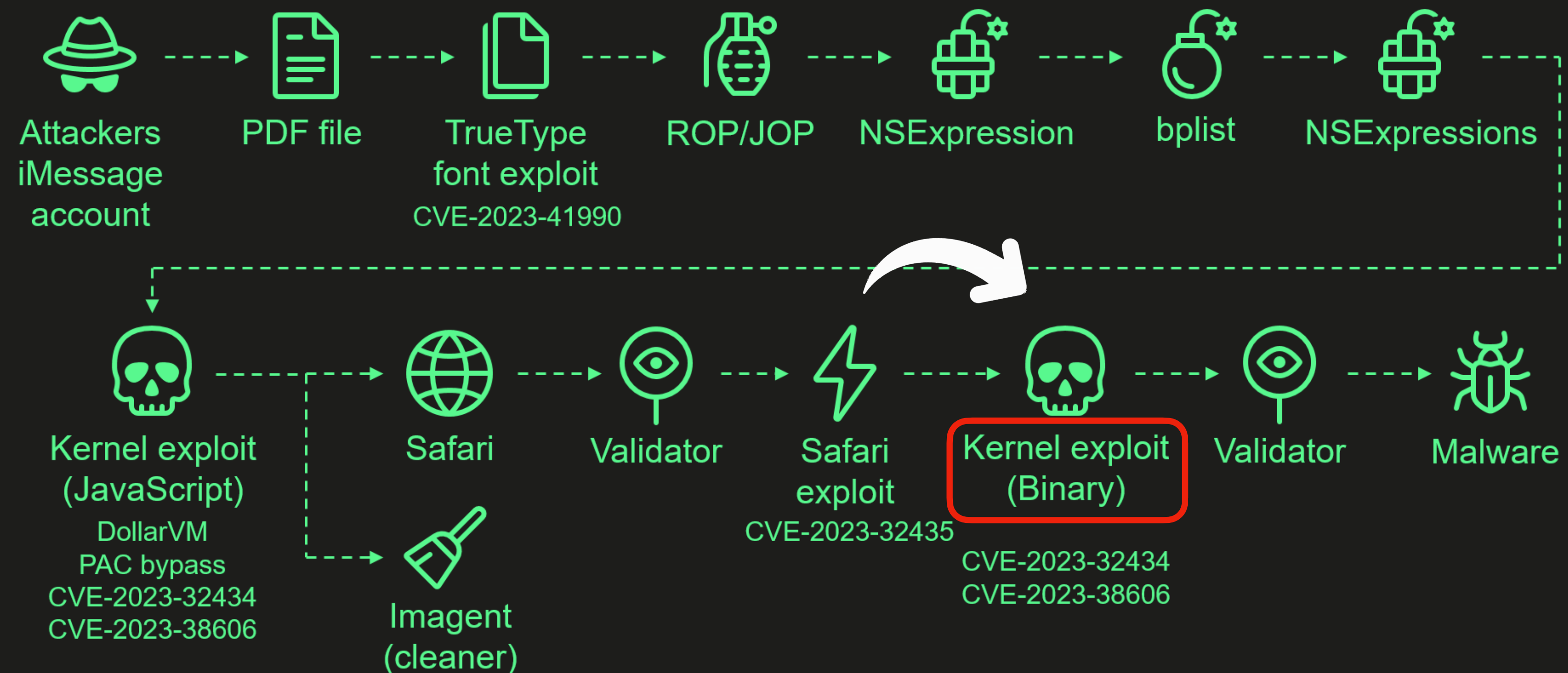
Operation Triangulation (~2023)

Attack chain

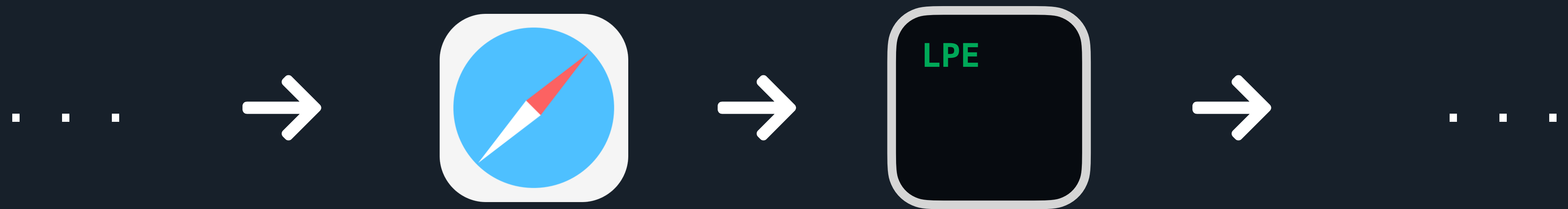


Operation Triangulation (~2023)

Attack chain



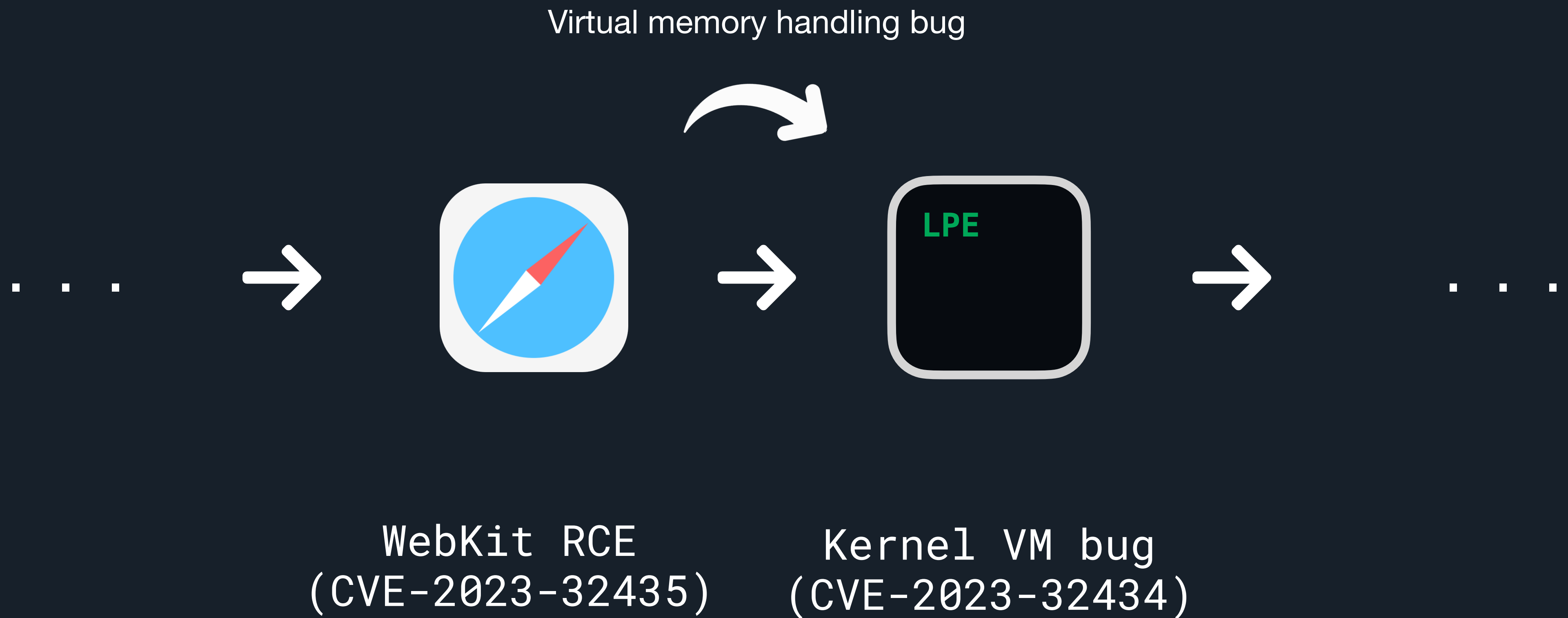
Operation Triangulation (~2023)



WebKit RCE
(CVE-2023-32435)

Kernel VM bug
(CVE-2023-32434)

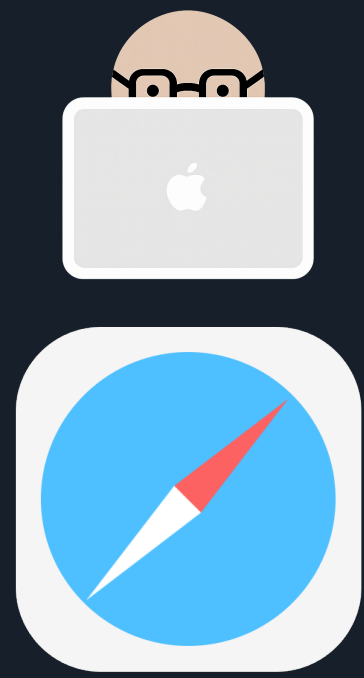
Operation Triangulation (~2023)



Sandbox escape (SBX)

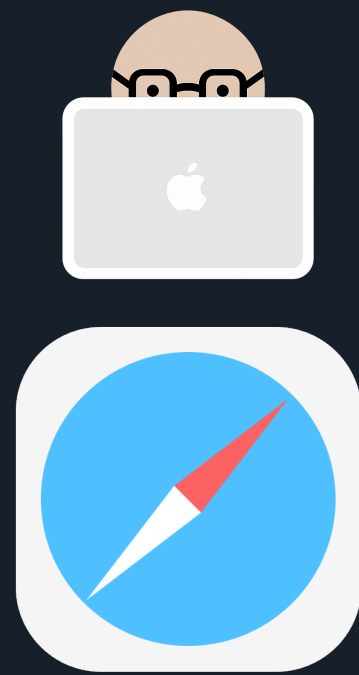
- Look for another target in user space
- One that has interesting kernel surface

Sandbox escape (SBX)



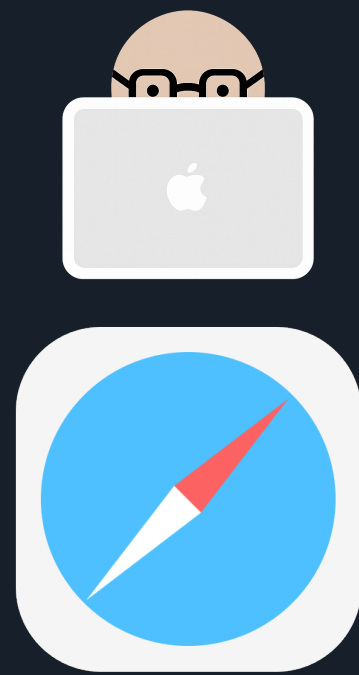
Sandbox escape (SBX)

How do I reach the kernel?



Sandbox escape (SBX)

How do I reach the kernel?

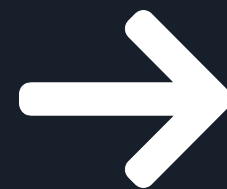
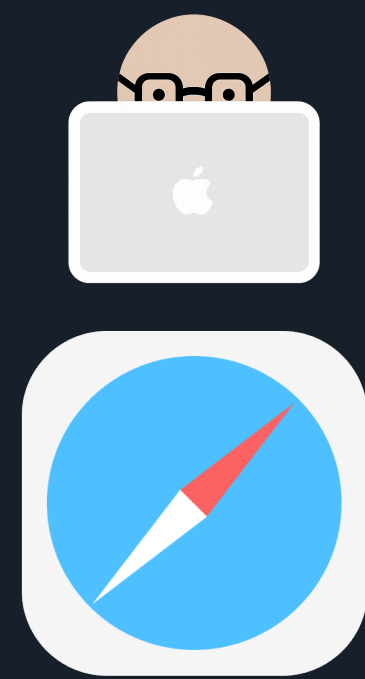


```
(deny iokit-open)  
(allow mach-lookup  
  "processA")
```



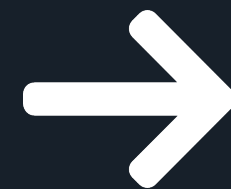
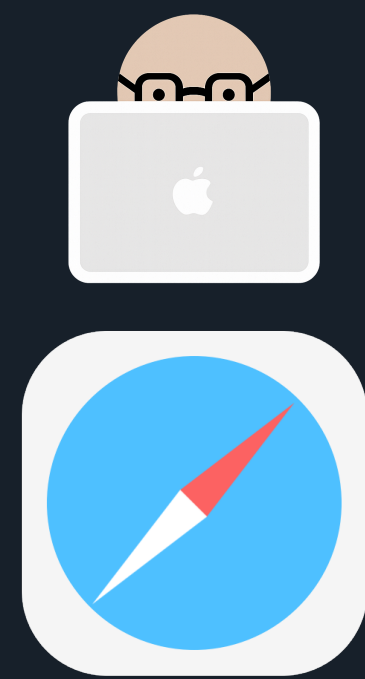
Sandbox escape (SBX)

How do I reach the kernel?



Sandbox escape (SBX)

How do I reach the kernel?

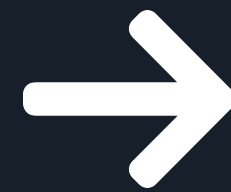
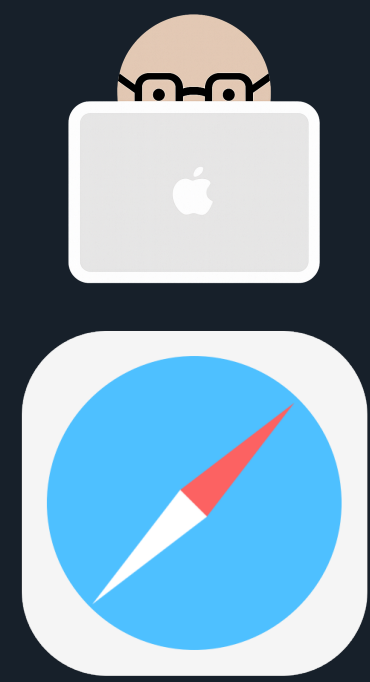


(allow iokit-open
iokit-user-client-class "B")

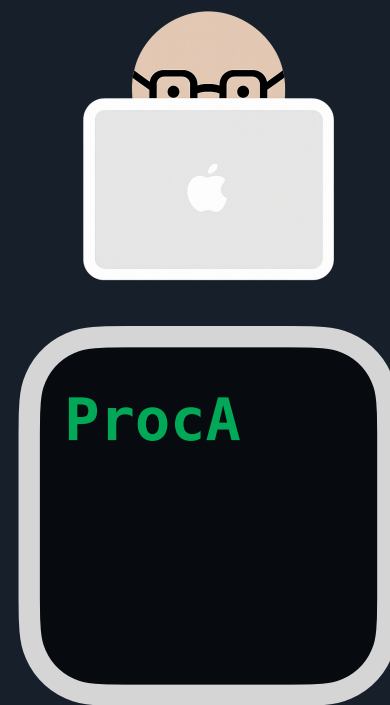
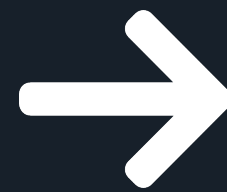
Sandbox escape (SBX)



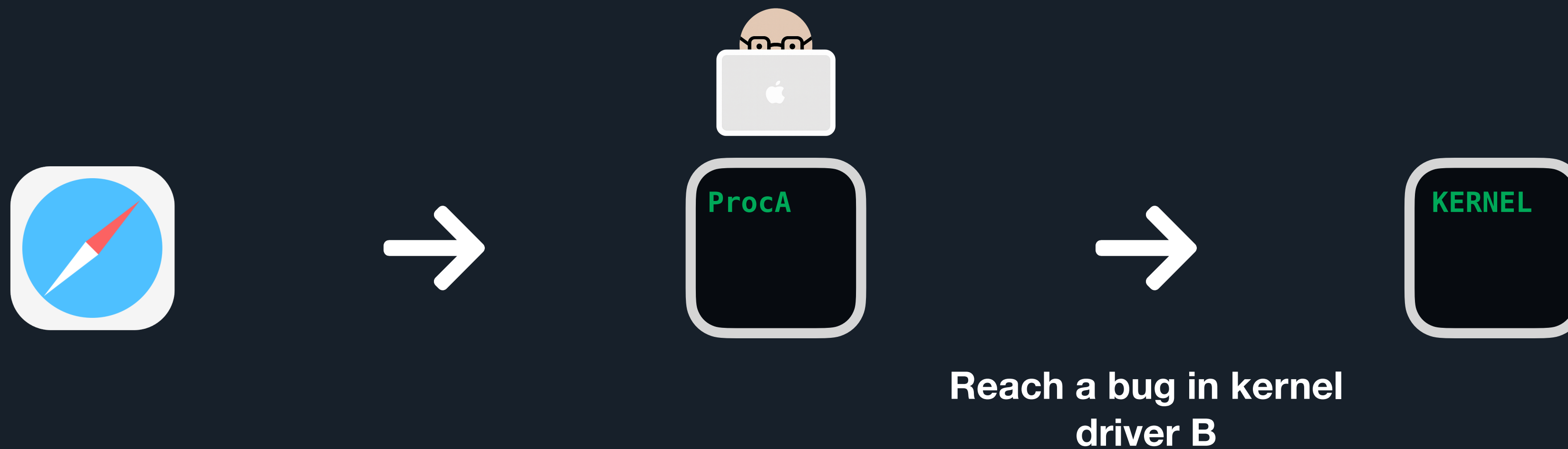
Sandbox escape (SBX)



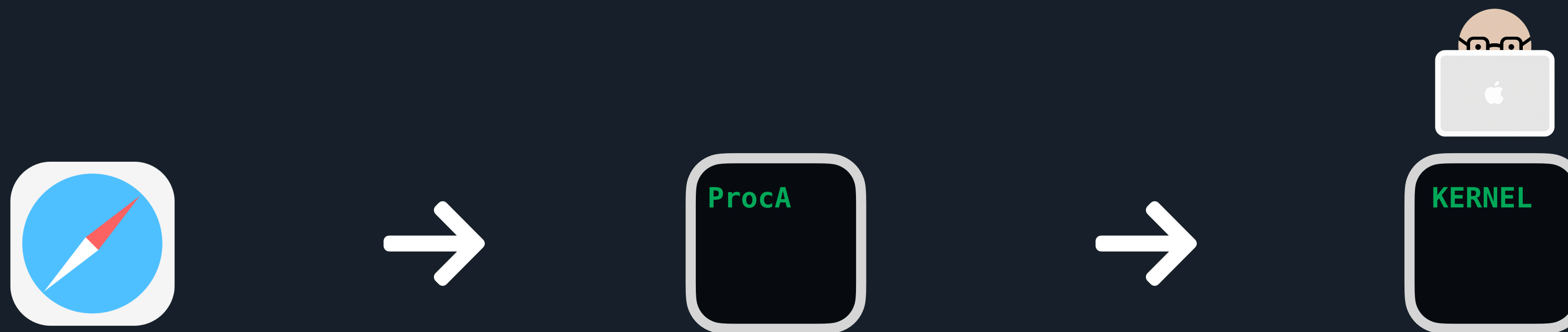
Sandbox escape (SBX)



Sandbox escape (SBX)



Sandbox escape (SBX)



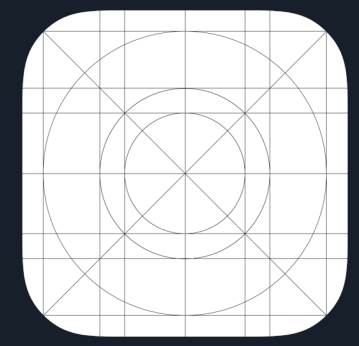
Reach a bug in kernel
driver B

Anatomy of an iOS exploit chain (simplified)



Kernel exploit (LPE)

- Aim: Kernel memory R/W
- 'Full' control over the device (aside from SPTM, TXM)



My process

Kernel

BSD syscalls

Mach traps

IOGPUPFamily

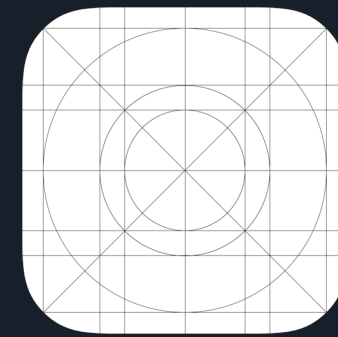
NeuralEngine

IOHIDDevice

JPEGDriver

AppleKeyStore

AppleVideoDecoder



My process



setsockopt()

Kernel

BSD syscalls

Mach traps

IOGPUFamily

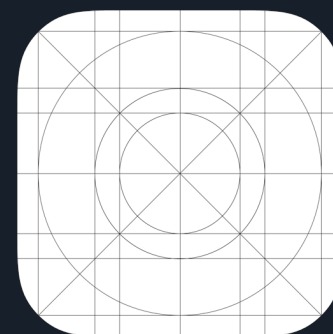
NeuralEngine

IOHIDDevice

JPEGDriver

AppleKeyStore

AppleVideoDecoder



My process



`setsockopt()`

`vm_allocate()`

Kernel

BSD syscalls

Mach traps

IOGPUPFamily

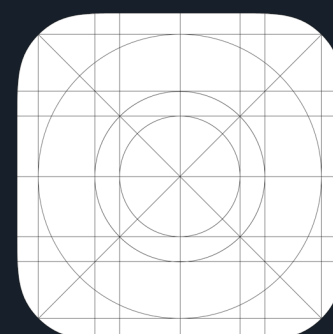
NeuralEngine

IOHIDDevice

JPEGDriver

AppleKeyStore

AppleVideoDecoder



My process



`setsockopt()`

`vm_allocate()`

`IOConnectCallMethod()`

Kernel

BSD syscalls

Mach traps

IOGPUFamily

NeuralEngine

IOHIDDevice

JPEGDriver

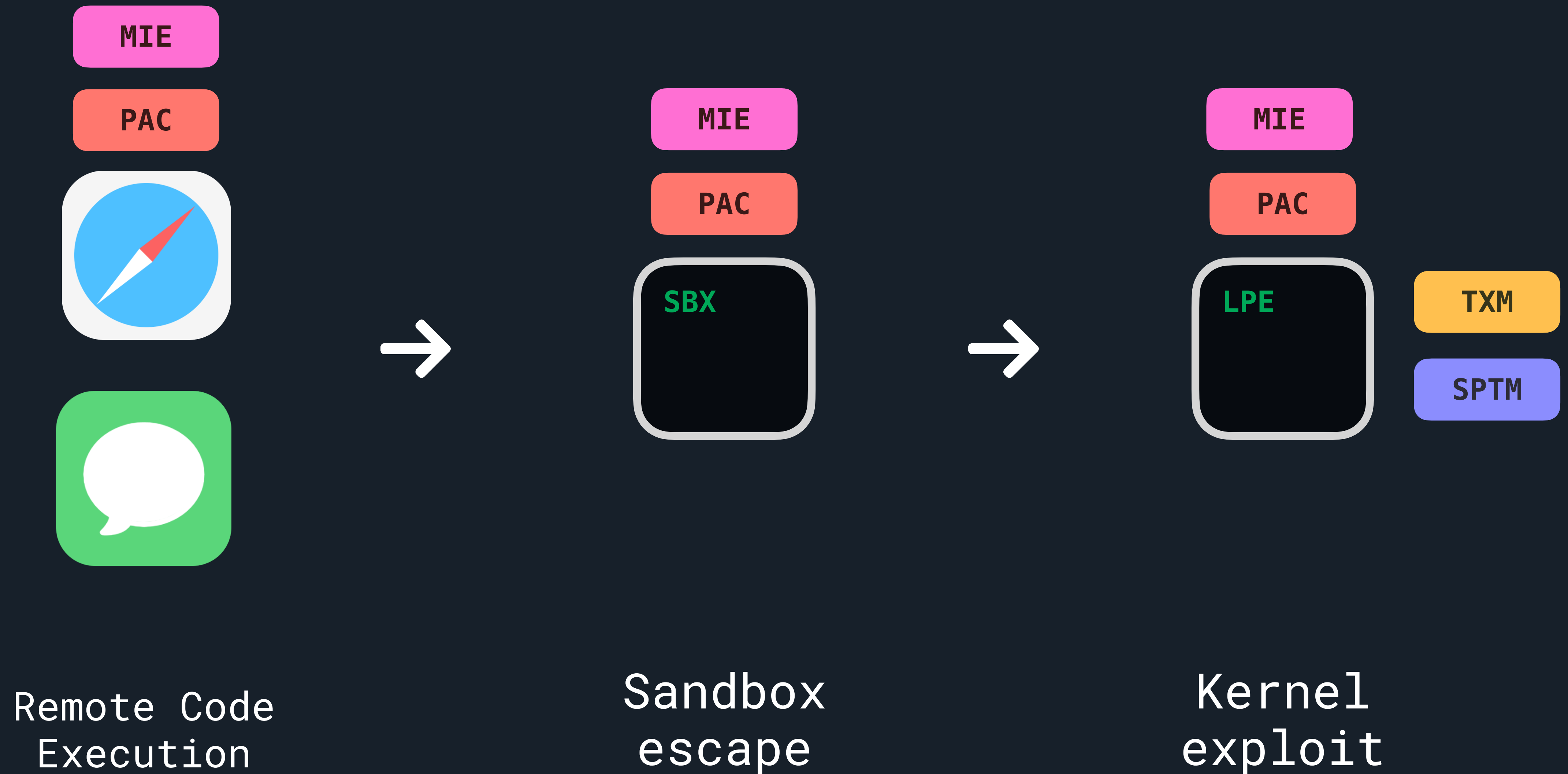
AppleKeyStore

AppleVideoDecoder

Anatomy of an iOS exploit chain



Anatomy of an iOS exploit chain



Where do *you* begin?

Where do *you* begin?

- iOS research setup

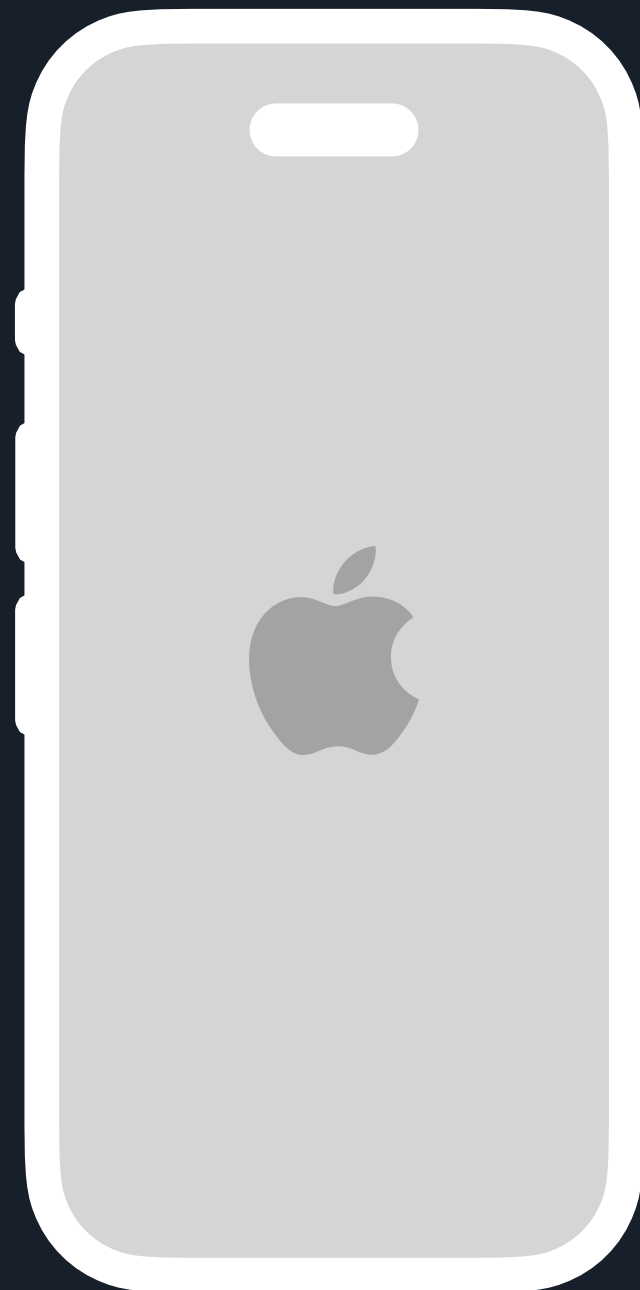
Where do *you* begin?

- iOS research setup
 - Hardware: testing devices
 - Software: tools

Test devices



Real iPhone



Virtual iPhone

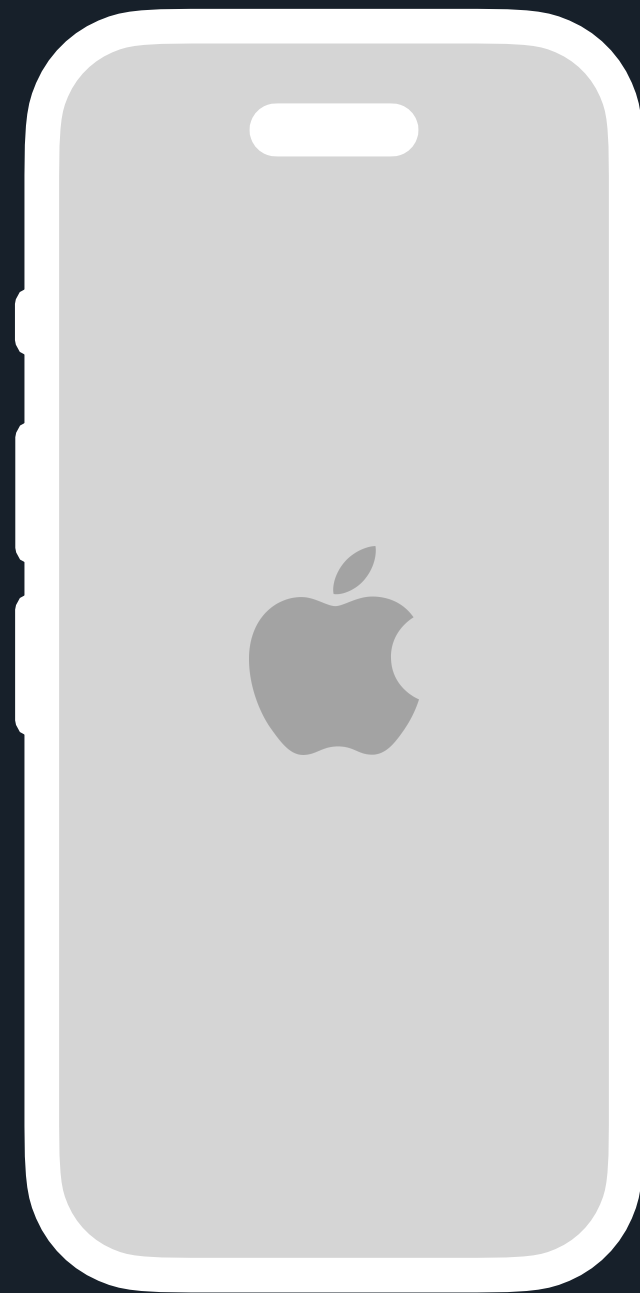


M-chip Mac

Test devices



Real iPhone



Virtual iPhone



M-chip Mac



Real iPhone

Pros

- 1:1 mapping of research : end goal

Cons

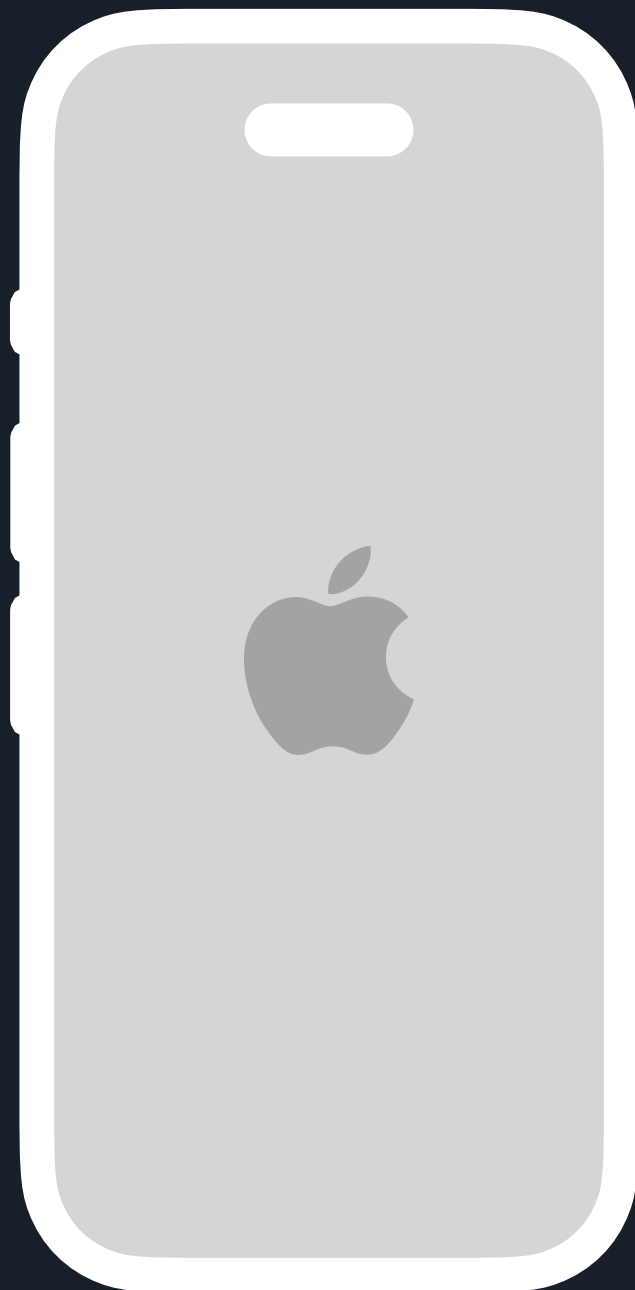
- Need a jailbreak for most debugging tasks
- No public jailbreak since iOS 16



Test devices



Real iPhone



Virtual iPhone



M-chip Mac

Virtual iPhone



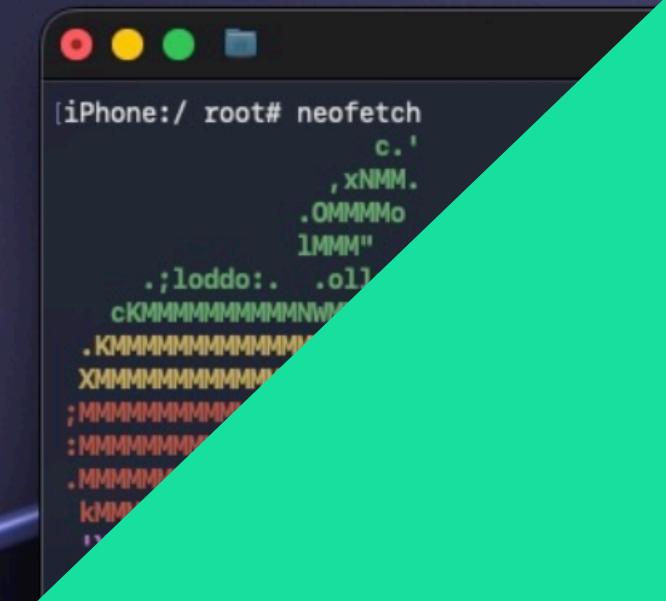
Virtual iPhone



vphone-cli

Boot a virtual iPhone via Apple's Virtualization.framework using PCC research VM infrastructure.

Everything runs through the single `vphone-cli` binary — create, patch, restore, install, boot, and `make` needed after building.



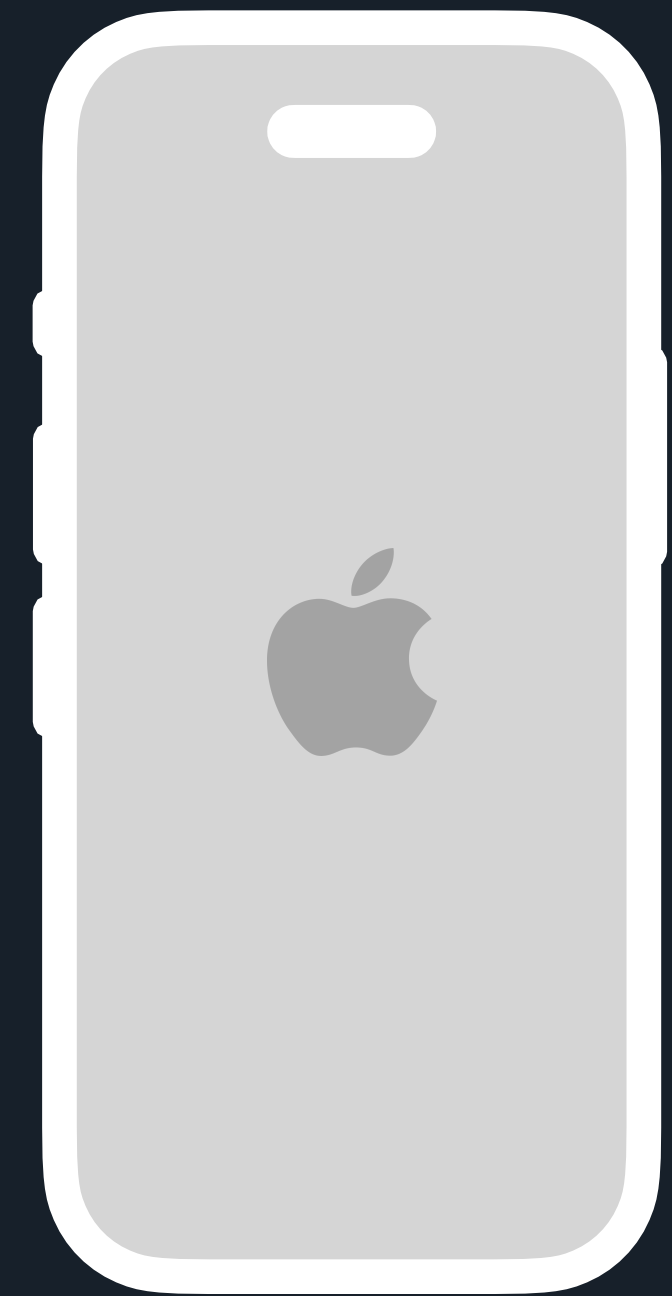
Virtual iPhone

Pros

- Unrestricted debugging capabilities
- Version flexibility
- Scalable (e.g. for fuzzing)

Cons

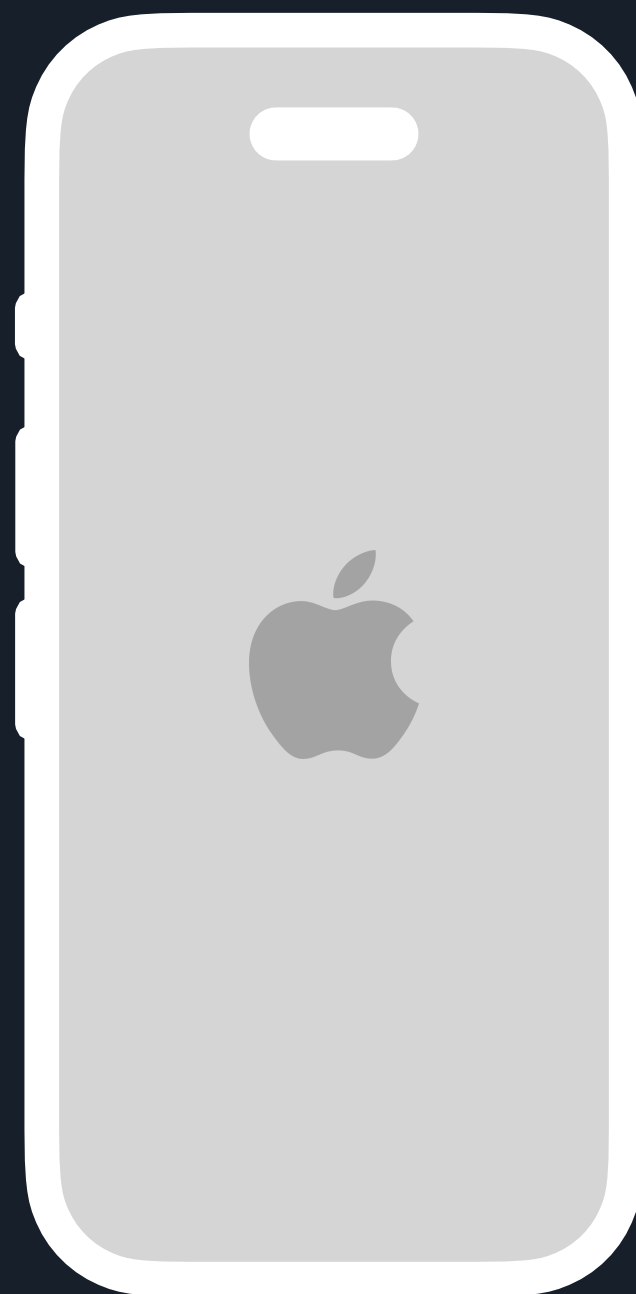
- Not *exactly* the same as a physical iPhone (maybe lacking some kernel extensions)



Test devices



Real iPhone



Virtual iPhone



M-chip Mac

M-chip Mac

Pros

- Easy debugging ability
- Run code as root & with entitlements
- Debug real system processes

Cons

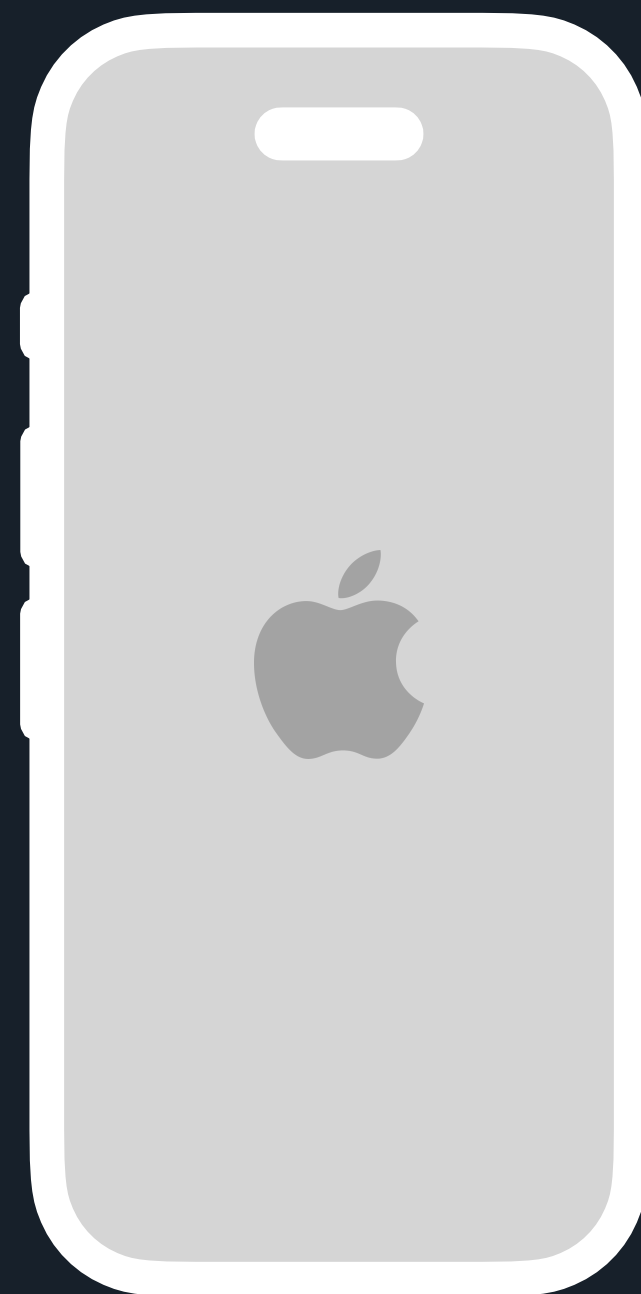
- Not all code is shared with iOS
- Lacks some mobile-specific kernel extensions



Test devices



Real iPhone



Virtual iPhone



M-chip Mac

Tools

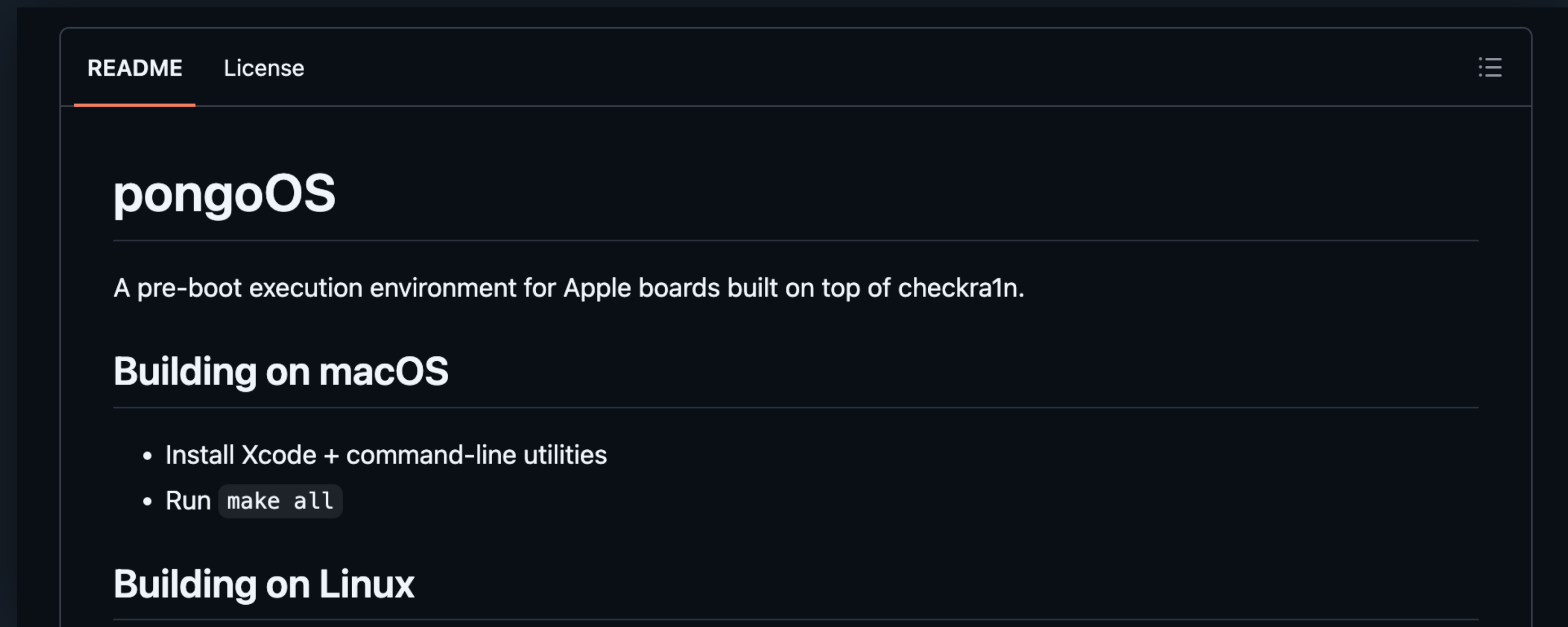
Tools

- Decompiler/Disassembler



Tools

- Jailbreak/debug setup



Some useful tools

- <https://github.com/jsherman212/xnuspy>
- <https://blacktop.github.io/ipsw/>
- <https://frida.re/>
- <https://github.com/checkra1n/pongo0S>
- <https://github.com/siguza/ios-kern-utils>

Getting hands-on

Patch diffing

Patch diffing

- Compare two iOS versions
- Identify a change/patch
- Analyse & trigger the vulnerability on the older version

Patch diffing

- Compare two iOS versions
- Identify a change/patch
- Analyse & trigger the vulnerability on the older version
- Great hands-on beginner task!

What to diff?

- Scan Apple security notes page
- Find interesting-sounding patches
- Investigate

Model I/O

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: Processing a maliciously crafted image may corrupt process memory

Description: The issue was addressed with improved memory handling.

CVE-2026-28940: Michael DePlante (@izobashi) of TrendAI Zero Day Initiative

ImageIO

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: Processing a maliciously crafted image may corrupt process memory

Description: The issue was addressed with improved memory handling.

CVE-2026-28990: Jiri Ha, Arni Hardarson

Kernel

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: An app may be able to leak sensitive kernel state

Description: A logging issue was addressed with improved data redaction.

CVE-2026-28987: Dhiyanesh Selvaraj (@redroot97)

Kernel

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: An app may be able to disclose kernel memory

Description: The issue was addressed with improved memory handling.

CVE-2026-43654: Vaagn Vardanian, Nathaniel Oh (@calysteon)

IOSurfaceAccelerator

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: An app may be able to cause unexpected system termination or read kernel memory

Description: An out-of-bounds read was addressed with improved bounds checking.

CVE-2026-43655: Somair Ansar and an anonymous researcher

IOHIDFamily

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: An app may be able to determine kernel memory layout

Description: A logging issue was addressed with improved data redaction.

CVE-2026-28943: Google Threat Analysis Group

Model I/O

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: Processing a maliciously crafted image may corrupt process memory

Description: The issue was addressed with improved memory handling.

CVE-2026-28940: Michael DePlante (@izobashi) of TrendAI Zero Day Initiative



ImageIO

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: Processing a maliciously crafted image may corrupt process memory

Description: The issue was addressed with improved memory handling.

CVE-2026-28990: Jiri Ha, Arni Hardarson

Kernel

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: An app may be able to leak sensitive kernel state

Description: A logging issue was addressed with improved data redaction.

CVE-2026-28987: Dhiyanesh Selvaraj (@redroot97)

Kernel

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: An app may be able to disclose kernel memory

Description: The issue was addressed with improved memory handling.

CVE-2026-43654: Vaagn Vardanian, Nathaniel Oh (@calysteon)

IOSurfaceAccelerator

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: An app may be able to cause unexpected system termination or read kernel memory

Description: An out-of-bounds read was addressed with improved bounds checking.

CVE-2026-43655: Somair Ansar and an anonymous researcher

IOHIDFamily

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: An app may be able to determine kernel memory layout

Description: A logging issue was addressed with improved data redaction.

CVE-2026-28943: Google Threat Analysis Group

Example

ImageIO

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: Processing a maliciously crafted image may corrupt process memory

Description: The issue was addressed with improved memory handling.

CVE-2026-28990: Jiri Ha, Arni Hardarson

Example

ImageIO

Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: Processing a maliciously crafted image may corrupt process memory

Description: The issue was addressed with improved memory handling.

CVE-2026-28990: Jiri Ha, Arni Hardarson

Example

ImageIO

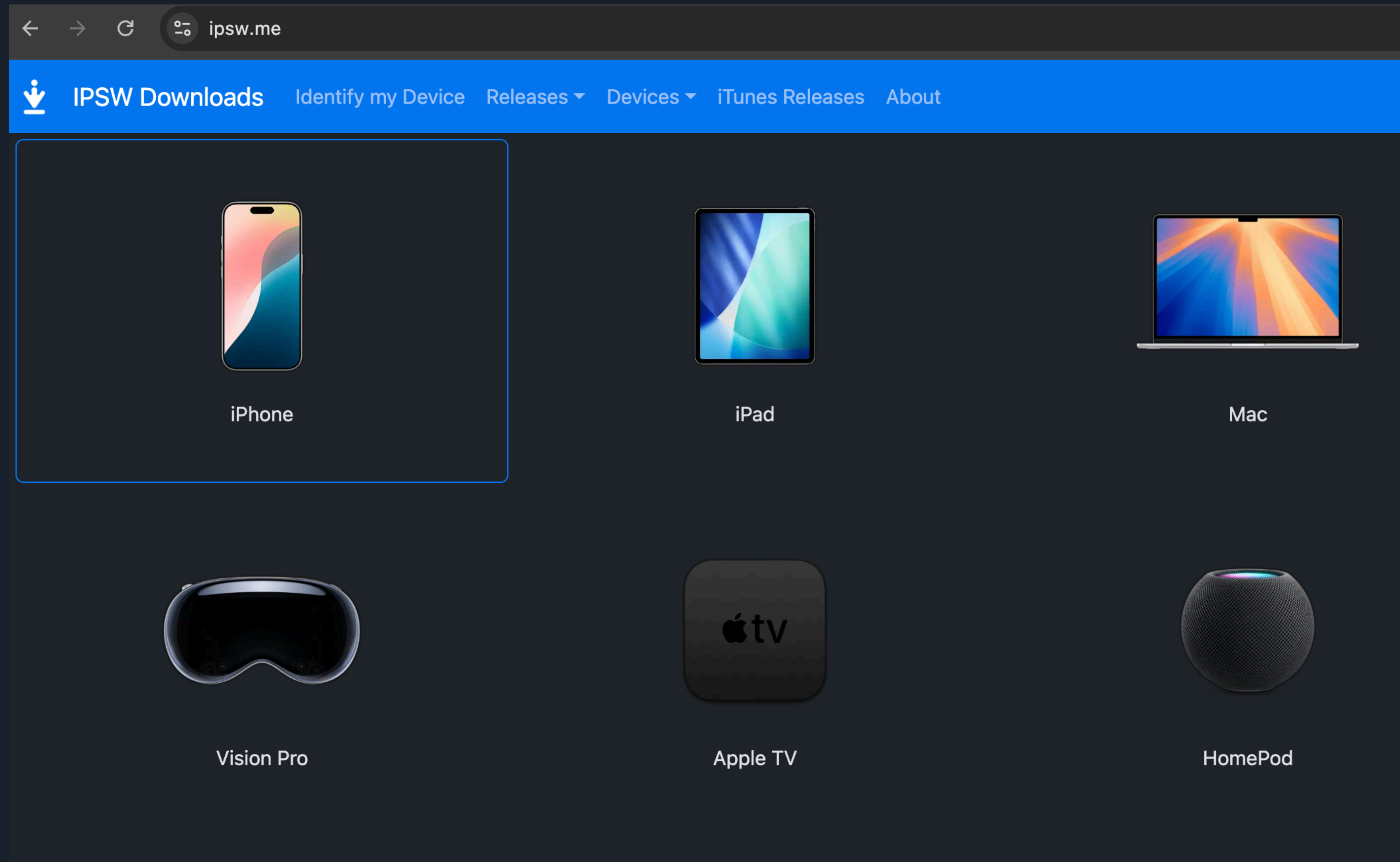
Available for: iPhone 11 and later, iPad Pro 12.9-inch 3rd generation and later, iPad Pro 11-inch 1st generation and later, iPad Air 3rd generation and later, iPad 8th generation and later, and iPad mini 5th generation and later

Impact: Processing a maliciously crafted image may corrupt process memory

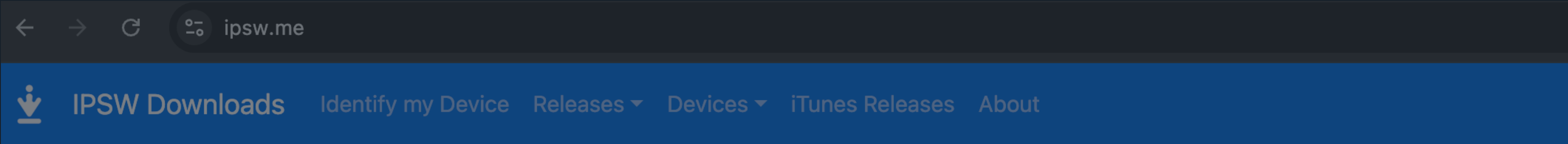
Description: The issue was addressed with improved memory handling.

CVE-2026-28990: Jiri Ha, Arni Hardarson

Example



Example



Unsigned IPSWs

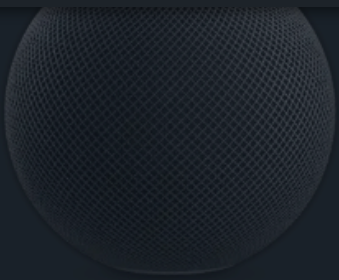
✗	iOS 26.5.1 (23F81)	June 1, 2026	11 GiB	iPhone18,3_26.5.1_23F81_Restore.ipsw
✗	iOS 26.5 (23F77)	May 11, 2026	11 GiB	iPhone18,3_26.5_23F77_Restore.ipsw
✗	iOS 26.4.2 (23E261)	April 22, 2026	10 GiB	iPhone18,3_26.4.2_23E261_Restore.ipsw
✗	iOS 26.4.1 (23E254)	April 8, 2026	10 GiB	iPhone18,3_26.4.1_23E254_Restore.ipsw



Vision Pro

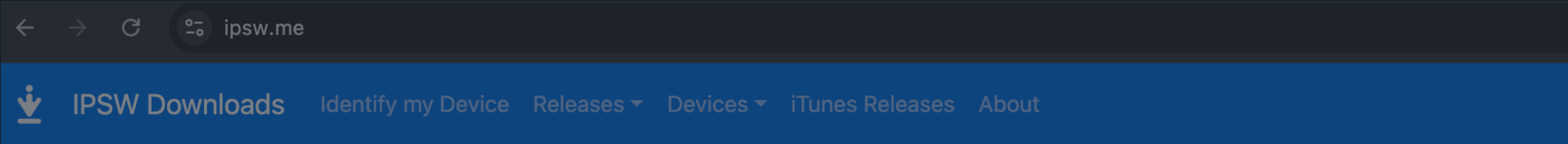


Apple TV



HomePod

Example



Unsigned IPSWs

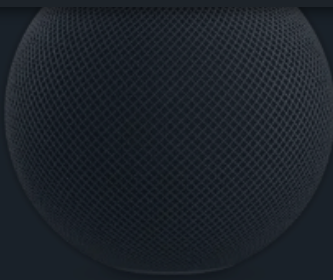
✗	iOS 26.5.1 (23F81)	June 1, 2026	11 GiB	iPhone18,3_26.5.1_23F81_Restore.ipsw
✗	iOS 26.5 (23F77)	May 11, 2026	11 GiB	iPhone18,3_26.5_23F77_Restore.ipsw
✗	iOS 26.4.2 (23E261)	April 22, 2026	10 GiB	iPhone18,3_26.4.2_23E261_Restore.ipsw
✗	iOS 26.4.1 (23E254)	April 8, 2026	10 GiB	iPhone18,3_26.4.1_23E254_Restore.ipsw



Vision Pro



Apple TV



HomePod

Regular function ■ Instruction ■ Data ■ Unexplored ■ External symbol ■ Lumina function

[2] IDA ... [3] Pseudoc... [4] Pseudoc... [5] Hex ... [6] Local... [7] DSC Index

```
1 __int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(EXRReadPlugin *this, char *a2,
2 {
3     int dword118; // w24
4     __int64 v7; // x20
5     __int64 Size; // x0
6     void *v9; // x20
7     __int64 logical_image_list; // x0
8     __n128 v11; // q0
9     void *v12; // x21
10    __int64 v13; // x24
11    float double1E0; // s0
12    void *decoder; // x22
13    unsigned __int64 v16; // x27
14    unsigned __int64 v17; // x8
15    char *v18; // x0
16    char *v19; // x28
17    char *v20; // x25
18    __int64 pixels; // x0
19    __int64 v22; // x3
20    __int64 v23; // x28
21    __int64 v24; // x27
22    __int64 Width; // x26
23    __int64 BytesPerRow; // x25
24    __int64 BaseAddress; // x0
25    char *v28; // x28
26    unsigned int v29; // w21
27    size_t v30; // x26
28    __int64 v32; // [xsp+8h] [xbp-C8h]
29    void *object; // [xsp+10h] [xbp-C0h]
30    __int64 v34; // [xsp+18h] [xbp-B8h] BYREF
31    __int64 v35; // [xsp+20h] [xbp-B0h]
32    __int64 v36; // [xsp+28h] [xbp-A8h]
33    __int64 v37; // [xsp+30h] [xbp-A0h] BYREF
34    __int64 v38; // [xsp+38h] [xbp-98h]
35    __int64 v39; // [xsp+40h] [xbp-90h] BYREF
36    __int64 v40; // [xsp+48h] [xbp-88h]
37    __QWORD v41[4]; // [xsp+50h] [xbp-80h] BYREF
38    __int64 v42; // [xsp+70h] [xbp-60h] BYREF
39    void *v43; // [xsp+78h] [xbp-58h] BYREF
40
41    if ( ! (unsigned int)AppleEXREnabled() )
```

[7] DSC Index

Image

- > usr
- ▼ System
 - ▼ Library
 - ▼ Frameworks
 - > Foundation.framework
 - > CoreServices.framework
 - > Network.framework
 - > CoreTelephony.framework
 - > CoreFoundation.framework
 - > QuartzCore.framework
 - > CoreGraphics.framework
 - > CoreText.framework
 - > ExtensionFoundation.framework
 - > CoreData.framework
 - > Metal.framework
 - ▼ ImageIO.framework
 - ImageIO
 - > CloudKit.framework
 - > Security.framework
 - > AVKit.framework
 - > SwiftUI.framework
 - > SwiftUICore.framework
 - > Intents.framework
 - > AppIntents.framework
 - > VideoToolbox.framework
 - > HealthKit.framework
 - > WidgetKit.framework
 - > CoreML.framework
 - > Contacts.framework
 - > MediaToolbox.framework
 - > CoreMedia.framework
 - > IOKit.framework
 - > SwiftData.framework
 - > CoreSpotlight.framework
 - > CFNetwork.framework
 - > JavaScriptCore.framework
 - > CoreText.framework

Regular function ■ Instruction ■ Data ■ Unexplored ■ External symbol ■ Lumina function

[2] IDA ... [3] Pseudoc... [4] Pseudoc... [5] Hex ... [6] Local... [7] DSC Index

```
1 __int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(EXRReadPlugin *this, char *a2,
2 {
3     int dword118; // w24
4     __int64 v7; // x20
5     __int64 Size; // x0
6     void *v9; // x20
7     __int64 logical_image_list; // x0
8     __n128 v11; // q0
9     void *v12; // x21
10    __int64 v13; // x24
11    float double1E0; // s0
12    void *decoder; // x22
13    unsigned __int64 v16; // x27
14    unsigned __int64 v17; // x8
15    char *v18; // x0
16    char *v19; // x28
17    char *v20; // x25
18    __int64 pixels; // x0
19    __int64 v22; // x3
20    __int64 v23; // x28
21    __int64 v24; // x27
22    __int64 Width; // x26
23    __int64 BytesPerRow; // x25
24    __int64 BaseAddress; // x0
25    char *v28; // x28
26    unsigned int v29; // w21
27    size_t v30; // x26
28    __int64 v32; // [xsp+8h] [xbp-C8h]
29    void *object; // [xsp+10h] [xbp-C0h]
30    __int64 v34; // [xsp+18h] [xbp-B8h] BYREF
31    __int64 v35; // [xsp+20h] [xbp-B0h]
32    __int64 v36; // [xsp+28h] [xbp-A8h]
33    __int64 v37; // [xsp+30h] [xbp-A0h] BYREF
34    __int64 v38; // [xsp+38h] [xbp-98h]
35    __int64 v39; // [xsp+40h] [xbp-90h] BYREF
36    __int64 v40; // [xsp+48h] [xbp-88h]
37    _QWORD v41[4]; // [xsp+50h] [xbp-80h] BYREF
38    __int64 v42; // [xsp+70h] [xbp-60h] BYREF
39    void *v43; // [xsp+78h] [xbp-58h] BYREF
40
41    if ( ! (unsigned int)AppleEXREnabled() )
```

[7] DSC Index

Image

- > usr
- ▼ System
 - ▼ Library
 - ▼ Frameworks
 - > Foundation.framework
 - > CoreServices.framework
 - > Network.framework
 - > CoreTelephony.framework
 - > CoreFoundation.framework
 - > QuartzCore.framework
 - > CoreGraphics.framework
 - > CoreText.framework
 - > ExtensionFoundation.framework
 - > CoreData.framework
 - > Metal.framework
 - ▼ ImageIO.framework
 - ImageIO
 - > CloudKit.framework
 - > Security.framework
 - > AVKit.framework
 - > SwiftUI.framework
 - > SwiftUICore.framework
 - > Intents.framework
 - > AppIntents.framework
 - > VideoToolbox.framework
 - > HealthKit.framework
 - > WidgetKit.framework
 - > CoreML.framework
 - > Contacts.framework
 - > MediaToolbox.framework
 - > CoreMedia.framework
 - > IOKit.framework
 - > SwiftData.framework
 - > CoreSpotlight.framework
 - > CFNetwork.framework
 - > JavaScriptCore.framework
 - > CoreText.framework

Regular function ■ Instruction ■ Data ■ Unexplored ■ External symbol ■ Lumina function

[2] IDA ... [3] Pseudoc... [4] Pseudoc... [5] Hex ... [6] Local...

[7] DSC Index

Image

> usr
v System
v Library

```
1 __int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(EXRReadPlugin *this, char *a2, int a3)
2 {
3     int dword118; // w24
4     __int64 v7; // x20
5     __int64 Size; // x0
6     void *v9; //
7     __int64 logic
8     __n128 v11; //
9     void *v12; //
10    __int64 v13;
11    float double1
12    void *decoder
13    unsigned __in
14    unsigned __in
15    char *v18; //
16    char *v19; //
17    char *v20; //
18    __int64 pixel
19    __int64 v22;
20    __int64 v23;
21    __int64 v24;
22    __int64 Width
23    __int64 Bytes
24    __int64 BaseA
25    char *v28; //
26    unsigned int
27    size_t v30; //
28    __int64 v32;
29    void *object;
30    __int64 v34;
31    __int64 v35;
32    __int64 v36;
33    __int64 v37; // [xsp+30h] [xbp-A0h] BYREF
34    __int64 v38; // [xsp+38h] [xbp-98h]
35    __int64 v39; // [xsp+40h] [xbp-90h] BYREF
36    __int64 v40; // [xsp+48h] [xbp-88h]
37    _QWORD v41[4]; // [xsp+50h] [xbp-80h] BYREF
38    __int64 v42; // [xsp+70h] [xbp-60h] BYREF
39    void *v43; // [xsp+78h] [xbp-58h] BYREF
40
41    if ( ! (unsigned int)AppleEXREnabled() )
```



IDA Pro

File

Edit

Jump

Search

View

Teams



Library func



Functions

Function name

New instance

Open...

Load file

Produce file

BinDiff...

Script file...

Script command...

Similarity ^	Confidence	Change	EA Primary	Name Primary	EA Secondary	Name Secondary	Correlation	Algorithm
0.99	0.99	-I-----	00000001865A...	ETCReadPlugin::initialize(IIODictionary *)	00000001865B...	ETCReadPlugin::initialize(IIODictionary *)		Name Hash
0.99	0.99	-I--E--	0000000186610...	sub_1866104E4	000000018661...	sub_18661CD80		Edges Flow Graph MD Index
0.99	0.99	-I-----	00000001865B...	_JPEGWriter::processInput(void)	00000001865C...	_JPEGWriter::processInput(void)		Name Hash
0.99	0.99	-I-----	00000001864D...	KTX2ReadPlugin::initialize(IIODictionary *)	00000001864E...	KTX2ReadPlugin::initialize(IIODictionary *)		Name Hash
0.98	0.98	-I--E--	000000018652...	loc_1865239E0	000000018652F...	sub_18652FC38		MD Index (Flow Graph MD Index, Top
0.98	0.98	-I-----	000000018652...	sub_1865239F0	000000018652F...	sub_18652FC50		Address Sequence
0.98	0.98	-I-----	000000018652...	sub_1865239F4	000000018652F...	sub_18652FC54		Address Sequence
0.98	0.99	GI-J---	000000018653...	TIFFReadPlugin::initialize(IIODictionary *)	0000000186541...	TIFFReadPlugin::initialize(IIODictionary *)		Name Hash
0.97	0.97	-I-----	000000018652...	sub_1865239F8	000000018652F...	sub_18652FC58		Instruction Count
0.96	0.99	GI-----	000000018637F...	IIO_ReaderHandler::readerForBytesImp(uchar const...	000000018638...	IIO_ReaderHandler::readerForBytesImp(uchar const...		Name Hash
0.96	0.99	GI-J---	000000018656...	nv::decompress_etc_eac(void const*,nv::Vector4 *)	0000000186575...	nv::decompress_etc_eac(void const*,nv::Vector4 *)		Name Hash
0.95	0.99	GI-JE--	000000018656...	nv::DecompressETC_EAC(nvtt::Format,uint,uint,void...	0000000186575...	nv::DecompressETC_EAC(nvtt::Format,uint,uint,void...		Name Hash
0.94	0.95	-I--E--	0000000186575...	sub_186575F74	000000018660...	sub_18660D4FC		Address Sequence
0.94	0.95	-I--E--	00000001865A...	sub_1865A4CD4	000000018658...	sub_1865826F8		Address Sequence
0.94	0.95	-I--E--	00000001865D...	sub_1865D71E4	00000001865B1...	sub_1865B1458		Address Sequence
0.94	0.95	-I--E--	000000018660...	sub_186600CD4	00000001865E...	sub_1865E39A8		Address Sequence
0.94	0.99	GI-----	000000018660...	EXRReadPlugin::decodeBlockAppleEXR(void *,ulong)	000000018660...	EXRReadPlugin::decodeBlockAppleEXR(void *,ulong)		Name Hash
0.94	0.98	GI-JE--	000000018639...	IIOReadPlugin::callDecodeImage(IIODecodeParamet...	000000018639...	IIOReadPlugin::callDecodeImage(IIODecodeParamet...		Name Hash
0.93	0.99	GI-JE--	000000018660...	PSDReadPlugin::decodeBlockSubsampled(IIODecod...	000000018661...	PSDReadPlugin::decodeBlockSubsampled(IIODecod...		Name Hash
0.93	0.99	GI-J---	000000018660...	sub_18660FC20	000000018661...	sub_18661C45C		Loop Count
0.84	0.97	GI-JE--	000000018637F...	___ZN14IIO_Reader_ATX13getImageCountEP19IIOI...	000000018638...	___ZN14IIO_Reader_ATX13getImageCountEP19IIOI...		Name Hash
0.67	0.97	-I--E--	00000001863D...	_OUTLINED_FUNCTION_0_63	00000001863D...	_OUTLINED_FUNCTION_0_63		Name Hash
0.67	0.97	-I--E--	00000001863D...	_OUTLINED_FUNCTION_1_42	00000001863D...	_OUTLINED_FUNCTION_1_42		Name Hash
0.67	0.97	-I--E--	00000001865FF...	___Block_byref_object_copy__1	00000001865E...	___Block_byref_object_copy__1		Name Hash
0.66	0.97	-I--E--	00000001865F...	___Block_byref_object_copy__1	00000001865F...	___Block_byref_object_copy__1		Name Hash

Similarit ^	Confidt	Change	EA Primary	Name Primary	EA Secondary	Name Secondary	Cor	Algorithm
0.99	0.99	-I-----	00000001865A...	ETCReadPlugin::initialize(IIODictionary *)	00000001865B...	ETCReadPlugin::initialize(IIODictionary *)		Name Hash
0.99	0.99	-I--E--	0000000186610...	sub_1866104E4	000000018661...	sub_18661CD80		Edges Flow Graph MD Index
0.99	0.99	-I-----	00000001865B...	_JPEGWriter::processInput(void)	00000001865C...	_JPEGWriter::processInput(void)		Name Hash
0.99	0.99	-I-----	00000001864D...	KTX2ReadPlugin::initialize(IIODictionary *)	00000001864E...	KTX2ReadPlugin::initialize(IIODictionary *)		Name Hash
0.98	0.98	-I--E--	000000018652...	loc_1865239E0	000000018652F...	sub_18652FC38		MD Index (Flow Graph MD Index, Top
0.98	0.98	-I-----	000000018652...	sub_1865239F0	000000018652F...	sub_18652FC50		Address Sequence
0.98	0.98	-I-----	000000018652...	sub_1865239F4	000000018652F...	sub_18652FC54		Address Sequence
0.98	0.99	GI-J---	000000018653...	TIFFReadPlugin::initialize(IIODictionary *)	0000000186541...	TIFFReadPlugin::initialize(IIODictionary *)		Name Hash
0.97	0.97	-I-----	000000018652...	sub_1865239F8	000000018652F...	sub_18652FC58		Instruction Count
0.96	0.99	GI-----	000000018637F...	IIO_ReaderHandler::readerForBytesImp(uchar const...	000000018638...	IIO_ReaderHandler::readerForBytesImp(uchar const...		Name Hash
0.96	0.99	GI-J---	000000018656...	nv::decompress_etc_eac(void const*,nv::Vector4 *)	0000000186575...	nv::decompress_etc_eac(void const*,nv::Vector4 *)		Name Hash
0.95	0.99	GI-JE--	000000018656...	nv::DecompressETC_EAC(nvtt::Format,uint,uint,void...	0000000186575...	nv::DecompressETC_EAC(nvtt::Format,uint,uint,void...		Name Hash
0.94	0.95	-I--E--	0000000186575...	sub_186575F74	000000018660...	sub_18660D4FC		Address Sequence
0.94	0.95	-I--E--	00000001865A...	sub_1865A4CD4	000000018658...	sub_1865826F8		Address Sequence
0.94	0.95	-I--E--	00000001865D...	sub_1865D71E4	00000001865B1...	sub_1865B1458		Address Sequence
0.94	0.95	-I--E--	000000018660...	sub_186600CD4	00000001865E...	sub_1865E39A8		Address Sequence
0.94	0.99	GI-----	000000018660...	EXRReadPlugin::decodeBlockAppleEXR(void *,ulong)	000000018660...	EXRReadPlugin::decodeBlockAppleEXR(void *,ulong)		Name Hash
0.94	0.98	GI-JE--	000000018639...	IIOReadPlugin::callDecodeImage(IIODecodeParamet...	000000018639...	IIOReadPlugin::callDecodeImage(IIODecodeParamet...		Name Hash
0.93	0.99	GI-JE--	000000018660...	PSDReadPlugin::decodeBlockSubsampled(IIODecod...	000000018661...	PSDReadPlugin::decodeBlockSubsampled(IIODecod...		Name Hash
0.93	0.99	GI-J---	000000018660...	sub_18660FC20	000000018661...	sub_18661C45C		Loop Count
0.84	0.97	GI-JE--	000000018637F...	___ZN14IIO_Reader_ATX13getImageCountEP19IIOI...	000000018638...	___ZN14IIO_Reader_ATX13getImageCountEP19IIOI...		Name Hash
0.67	0.97	-I--E--	00000001863D...	_OUTLINED_FUNCTION_0_63	00000001863D...	_OUTLINED_FUNCTION_0_63		Name Hash
0.67	0.97	-I--E--	00000001863D...	_OUTLINED_FUNCTION_1_42	00000001863D...	_OUTLINED_FUNCTION_1_42		Name Hash
0.67	0.97	-I--E--	00000001865FF...	___Block_byref_object_copy__1	00000001865E...	___Block_byref_object_copy__1		Name Hash
0.66	0.97	-I--E--	00000001865F...	___Block_byref_object_copy__1	00000001865E...	___Block_byref_object_copy__1		Name Hash

EXRReadPlugin::decodeBlockAppleEXR()

EXRReadPlugin::decodeBlockAppleEXR()

EXRReadPlugin::decodeBlockAppleEXR()

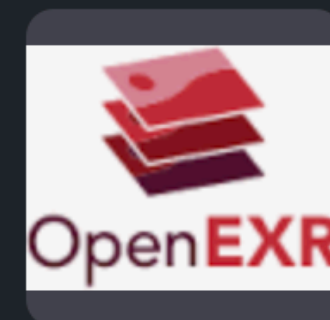


Wikipedia

[https://en.wikipedia.org › wiki › OpenEXR](https://en.wikipedia.org/wiki/OpenEXR)

OpenEXR

OpenEXR is a high-dynamic range, **multi-channel raster file format**, released as an open standard along with a set of software tools created by Industrial Light ... [Read more](#)



Decoding part of an EXR?



EXRReadPlugin::decodeBlockAppleEXR()

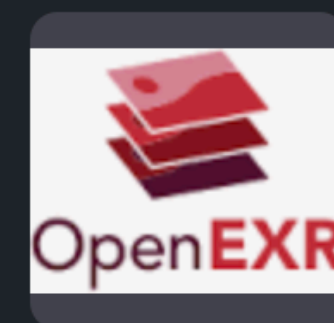


Wikipedia

[https://en.wikipedia.org › wiki › OpenEXR](https://en.wikipedia.org/wiki/OpenEXR)

OpenEXR

OpenEXR is a high-dynamic range, **multi-channel raster file format**, released as an open standard along with a set of software tools created by Industrial Light ... [Read more](#)



iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```


iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

iOS 26.5

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR != 1 || this->word1D0 != 0x20 ) {  
        ...  
    } else {  
        if ( is_mul_ok(v16, v17) )  
        {  
            a3 = v16 * v17;  
            buf = (char *)malloc_type_malloc(size: v16 * v17, type_id: 0xFD4C0FE5);  
            if ( buf )  
            {  
                ...  
            }  
            ...  
        }  
        ...  
    }  
    ...  
}
```

iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

iOS 26.5

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR != 1 || this->word1D0 != 0x20 ) {  
        ...  
    } else {  
        Integer overflow check  
        if ( is_mul_ok(v16, v17) )  
        {  
            a3 = v16 * v17;  
            buf = (char *)malloc_type_malloc(size: v16 * v17, type_id: 0xFD4C0FE5);  
            if ( buf )  
            {  
                ...  
            }  
            ...  
        }  
        ...  
    }  
    ...  
}
```

iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    Unchecked multiplication (width * height * 4)  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

iOS 26.5

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR != 1 || this->word1D0 != 0x20 ) {  
        ...  
        Integer overflow check  
        if ( is_mul_ok(v16, v17) )  
        {  
            a3 = v16 * v17;  
            buf = (char *)malloc_type_malloc(size: v16 * v17, type_id: 0xFD4C0FE5);  
            if ( buf )  
            {  
                ...  
            }  
            ...  
        }  
        ...  
    }  
    ...  
}
```


iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

Result used as size for malloc()



iOS 26.5

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR != 1 || this->word1D0 != 0x20 ) {  
        ...  
    } else {  
        Integer overflow check  
        if ( is_mul_ok(v16, v17) )  
        {  
            a3 = v16 * v17;  
            buf = (char *)malloc_type_malloc(size: v16 * v17, type_id: 0xFD4C0FE5);  
            if ( buf )  
            {  
                ...  
            }  
            ...  
        }  
        ...  
    }  
    ...  
}
```

iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

Result used as size for malloc()



iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        Width      Height  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

Result used as size for malloc()



iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        Width      Height  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

Result used as size for malloc()



iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        Width      Height  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ..Channel count  
    }  
    ...  
}
```

Result used as size for malloc()



width * height * channelCount * 4

width * height * 4 * 4

16384 * height * 4 * 4

$$16384 * 65536 * 4 * 4$$

16384 * 65536 * 4 * 4

= 0x4_00000000

16384 * 65536 * 4 * 4

= 0x4_00000000

32-bits

$$16384 * 65536 * 4 * 4$$

$$= 0x4\text{ }00000000$$

Beyond range 32-bits

16384 * 65536 * 4 * 4

= 0x0

Wraps to zero!

iOS 26.4.2

```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

size = 0



```
malloc(0);
```



```
malloc(0);
```

→ 0x102b983f0

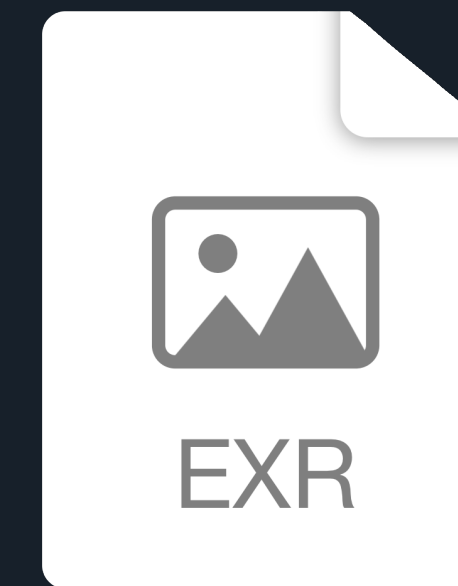
```
malloc(0);
```

→ 0x102b983f0



```
malloc(0);
```

→ 0x102b983f0

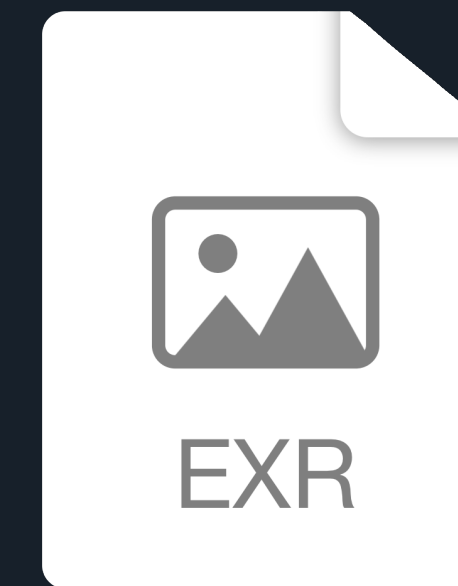


poc.exr

`malloc(0);`

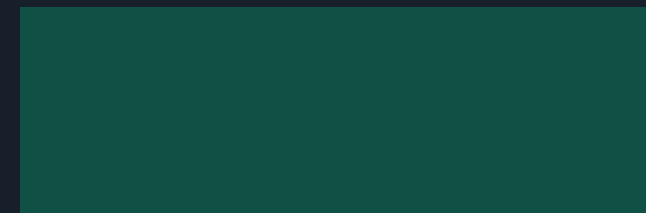
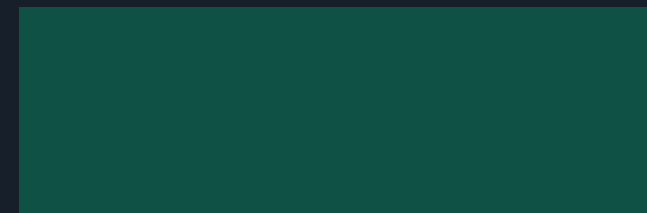
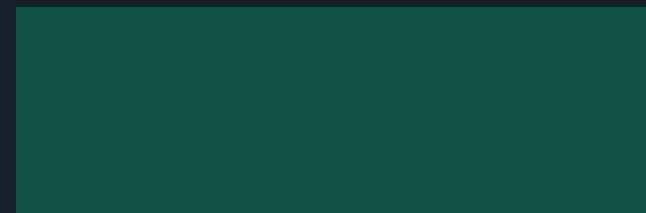
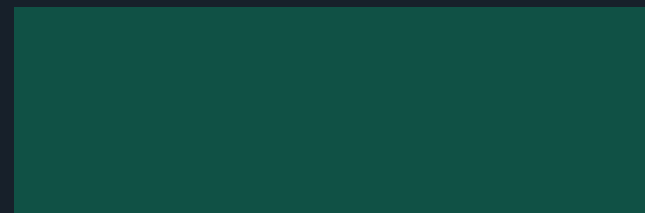
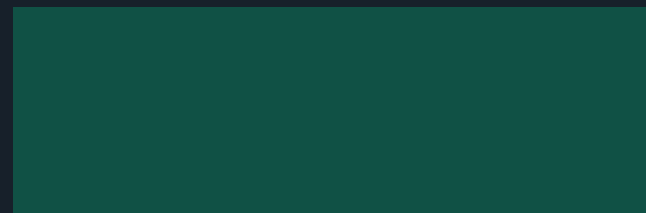
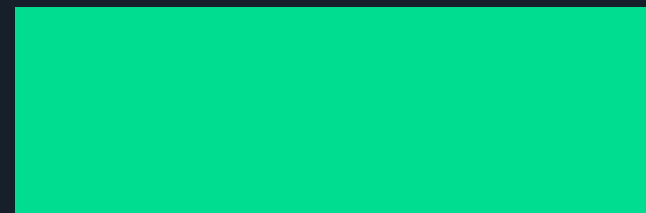
`→ 0x102b983f0`

`memcpy()`



`poc.exr`

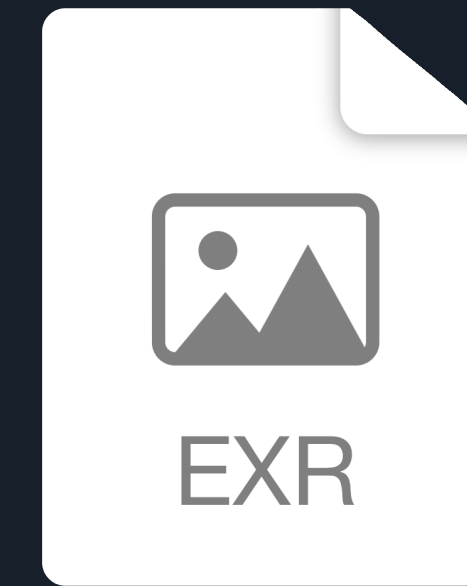
8 bytes



`malloc(0);`

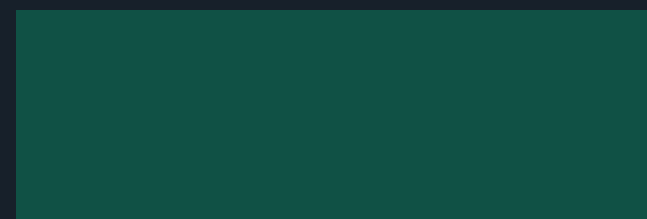
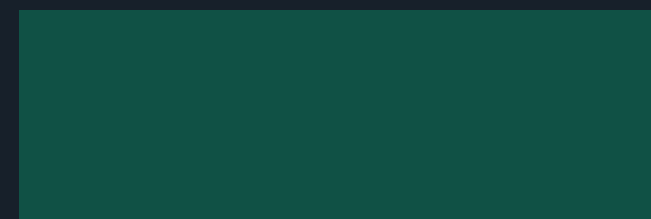
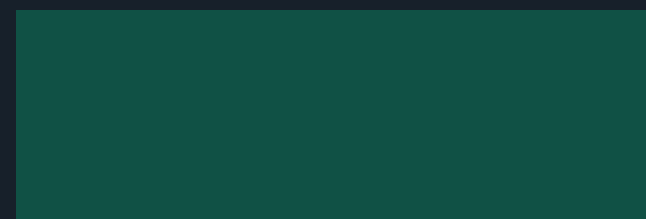
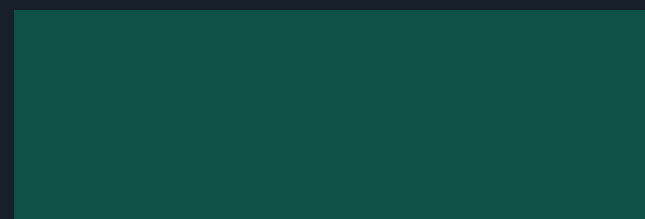
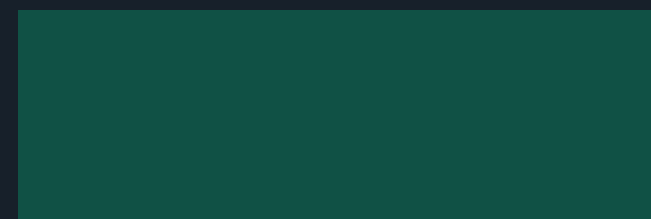
`→ 0x102b983f0`

`memcpy()`



`poc.exr`

8 bytes



Memory corruption

How do we reach this?

iOS 26.4.2



```
__int64 __fastcall EXRReadPlugin::decodeBlockAppleEXR(  
    EXRReadPlugin *this,  
    char *a2,  
    __int64 a3)  
{  
    ...  
    if ( this->DecodeToSDR == 1 && this->word1D0 == 0x20 )  
    {  
        v16 = 4 * this->dword260 * this->dwordF4;  
        size = v16 * this->unsigned_int138;  
        buf = (char *)malloc_type_malloc(size, type_id: 0x407CBF5u);  
        ...  
    }  
    ...  
}
```

size = 0



How do we reach this?

- ImageIO is the main parsing library on iOS/macOS
- Most apps call into it
- Can also build a harness around it

```
// read the sample either from file or
// shared memory
sample_size = *(uint32_t*)(shm_data);
if(sample_size > MAX_SAMPLE_SIZE) sample_size = MAX_SAMPLE_SIZE;
sample_bytes = (char *)malloc(sample_size);
memcpy(sample_bytes, shm_data + sizeof(uint32_t), sample_size);
NSData* content = [NSData dataWithBytes:sample_bytes length:sample_size];
free(sample_bytes)

UIImage* img = [[UIImage alloc] initWithData:content];

// UIImage *img = [[UIImage alloc] initWithContentsOfFile:objcstring];
if (img == nil) {
    NSLog(@"image nil");
}

CGImageRef cgImg = [img CGImageForProposedRect:nil context:nil hints:nil];
if (cgImg) {
    size_t width = CGImageGetWidth(cgImg);
    size_t height = CGImageGetHeight(cgImg);
    CGColorSpaceRef colorspace = CGColorSpaceCreateDeviceRGB();
    CGContextRef ctx = CGContextCreate(0, width, height, 8, 0, colorspace, 1);
    void* renderingState = CGContextGetRenderingState(ctx);
    CGContextSetAllowsAcceleration(renderingState, false);
    CGRect rect = CGRectMake(0, 0, width, height);
    CGContextDrawImage(ctx, rect, cgImg);
    CGColorSpaceRelease(colorspace);
    CGContextRelease(ctx);
    CGImageRelease(cgImg);
}
[img release];
}
```

```
// read the sample either from file or
// shared memory
sample_size = *(uint32_t*)(shm_data);
if(sample_size > MAX_SAMPLE_SIZE) sample_size = MAX_SAMPLE_SIZE;
sample_bytes = (char *)malloc(sample_size);
memcpy(sample_bytes, shm_data + sizeof(uint32_t), sample_size);
NSData* content = [NSData dataWithBytes:sample_bytes length:sample_size];
free(sample_bytes)

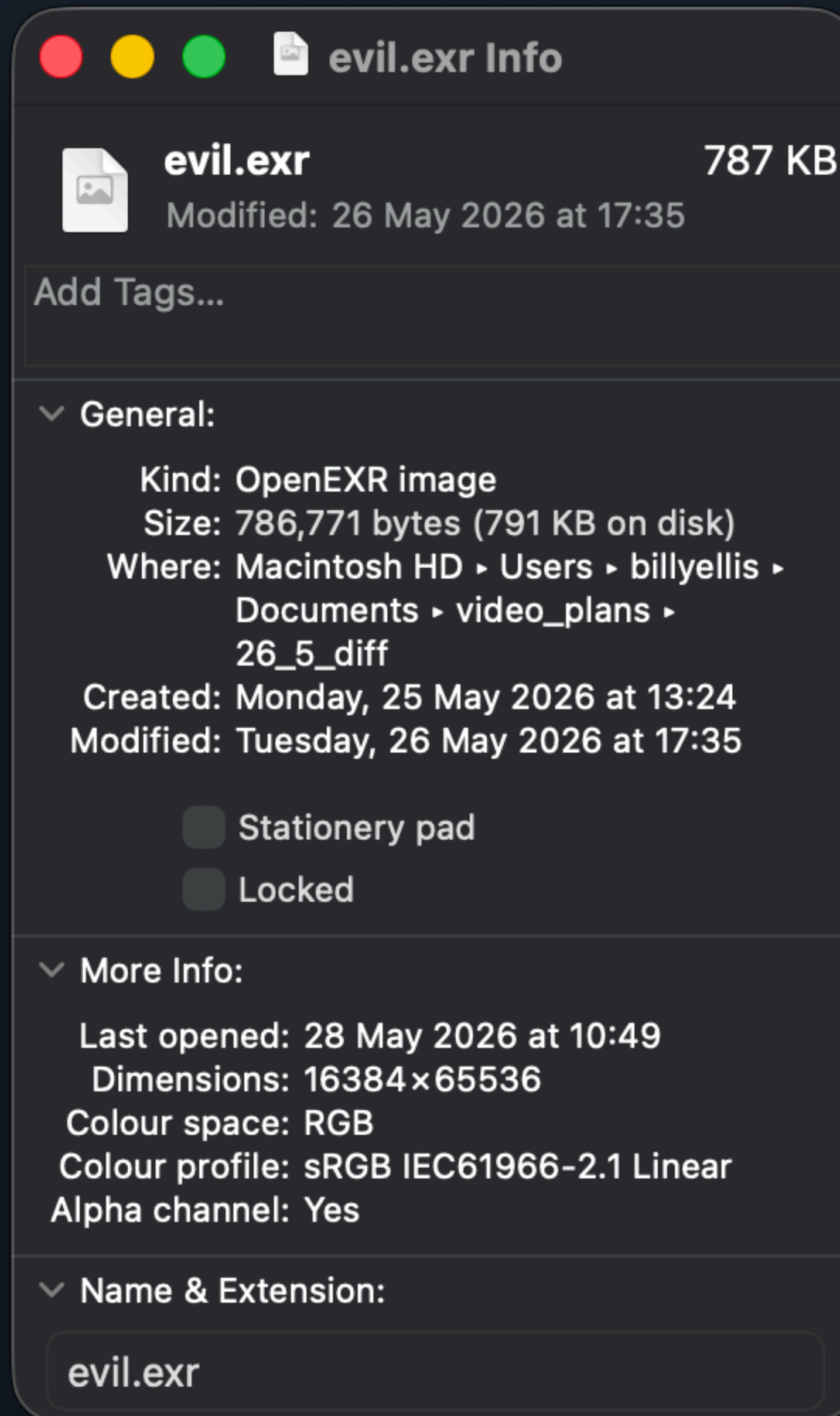
UIImage* img = [[UIImage alloc] initWithData:content];

// UIImage *img = [[UIImage alloc] initWithContentsOfFile:objcstring];
if (img == nil) {
    NSLog(@"image nil");
}

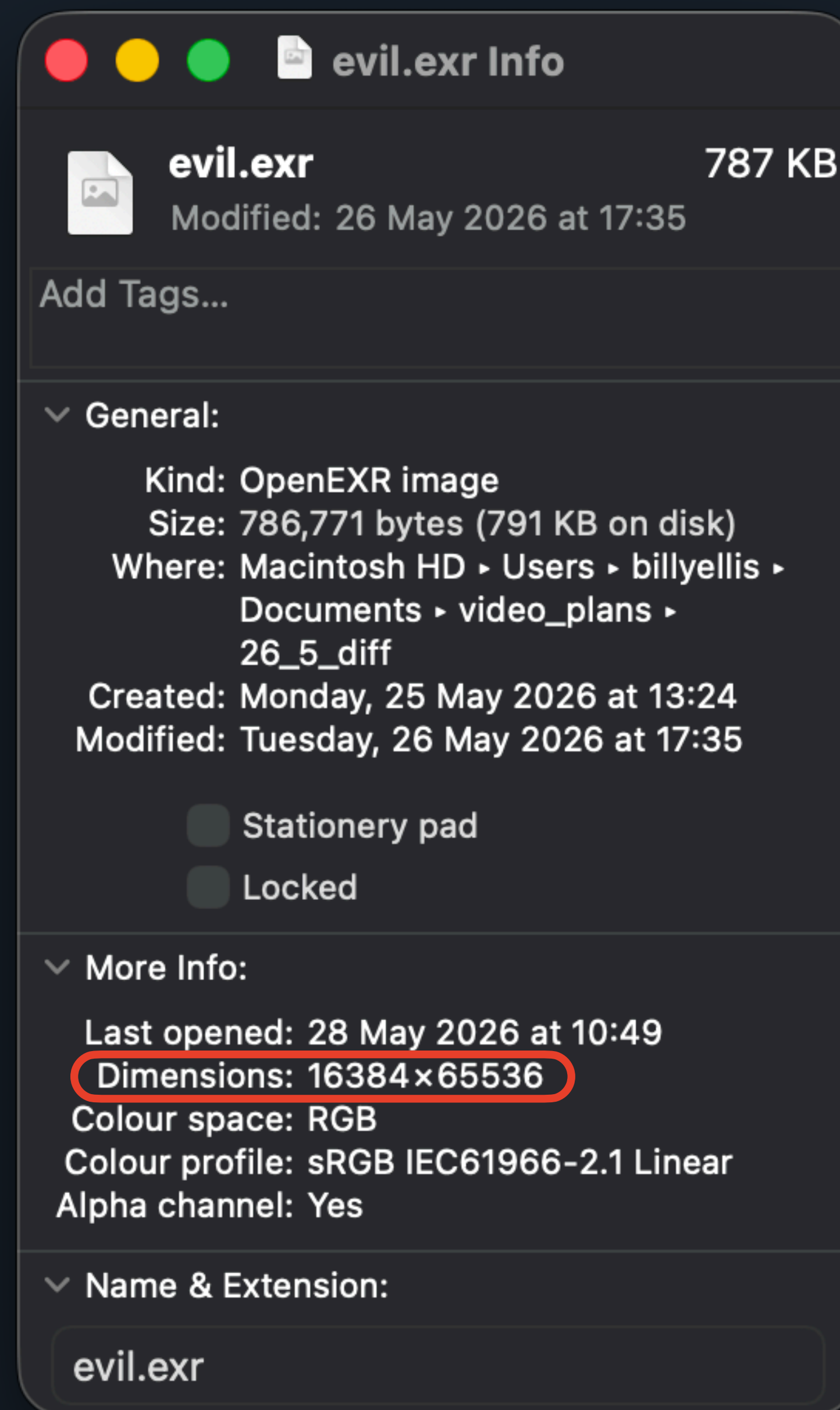
CGImageRef cgImg = [img CGImageForProposedRect:nil context:nil hints:nil];
if (cgImg) {
    size_t width = CGImageGetWidth(cgImg);
    size_t height = CGImageGetHeight(cgImg);
    CGColorSpaceRef colorspace = CGColorSpaceCreateDeviceRGB();
    CGContextRef ctx = CGContextCreate(0, width, height, 8, 0, colorspace, 1);
    void* renderingState = CGContextGetRenderingState(ctx);
    CGContextSetAllowsAcceleration(renderingState, false);
    CGRect rect = CGRectMake(0, 0, width, height);
    CGContextDrawImage(ctx, rect, cgImg);
    CGContextRelease(ctx);
    CGColorSpaceRelease(colorspace);
    CGContextRelease(ctx);
    CGImageRelease(cgImg);
}
[img release];
}
```

Calls into imageIO to render image file





Large width and height values



```
(lldb) bt
* thread #1, queue = 'com.apple.root.user-interactive-qos', stop reason = EXC_GUARD (code=1, subcode=0x4141414141414149)
  * frame #0: 0x00000001855bd01c libdispatch.dylib`_dispatch_apply_set_attr_behavior + 16
    frame #1: 0x00000001855bd718 libdispatch.dylib`_dispatch_apply_invoke_and_wait + 64
    frame #2: 0x00000001855bc944 libdispatch.dylib`_dispatch_apply_with_attr_f + 1312
    frame #3: 0x00000001d8bd33b4 libAppleEXR.dylib`axr_error_t LaunchBlocks<ReadPixelsArgs>(void (*)(void*, unsigned long),
*, unsigned long, axr_flags_t) + 472
    frame #4: 0x00000001d8bd2ff4 libAppleEXR.dylib`TileDecoder::ReadPixels(unsigned int, unsigned int, void*, unsigned long
    frame #5: 0x00000001d8bc9d70 libAppleEXR.dylib`Part::ReadRGBAPixels(axr_decoder*, void*, unsigned long, double, axr fla
    frame #6: 0x00000001929a1724 ImageIO`EXRReadPlugin::decodeBlockAppleEXR(void*, unsigned long) + 500
    frame #7: 0x00000001929a1400 ImageIO`EXRReadPlugin::decodeImageImp(II0DecodeParameter*, II0ImageType, __IOSurface**, __
ockSet**) + 612
    frame #8: 0x000000019295a8e4 ImageIO`II0ReadPlugin::callDecodeImage(II0DecodeParameter*, II0ImageType, __IOSurface**, _
lockSet**) + 1000
    frame #9: 0x00000001928a6b04 ImageIO`II0_Reader::CopyImageBlockSetProc(void*, CGImageProvider*, CGRect, CGSize, __CFDic
    frame #10: 0x00000001928c2984 ImageIO`II0ImageProviderInfo::copyImageBlockSetWithOptions(CGImageProvider*, CGRect, CGSi
nst*) + 408
    frame #11: 0x00000001928a6670 ImageIO`II0ImageProviderInfo::CopyImageBlockSetWithOptions(void*, CGImageProvider*, CGRec
nary const*) + 908
    frame #12: 0x000000018c53b25c CoreGraphics`imageProvider_retain_data + 96
    frame #13: 0x000000018c53b194 CoreGraphics`CGDataProviderRetainData + 76
    frame #14: 0x000000018c53b1dc CoreGraphics`provider_for_destination_retain_data + 28
    frame #15: 0x000000018c53b194 CoreGraphics`CGDataProviderRetainData + 76
```



```
(lldb) bt
* thread #1, queue = 'com.apple.root.user-interactive-qos', stop reason = EXC_GUARD (code=1, subcode=0x4141414141414149)
  * frame #0: 0x00000001855bd01c libdispatch.dylib`_dispatch_apply_set_attr_behavior + 16
    frame #1: 0x00000001855bd718 libdispatch.dylib`_dispatch_apply_invoke_and_wait + 64
    frame #2: 0x00000001855bc944 libdispatch.dylib`_dispatch_apply_with_attr_f + 1312
    frame #3: 0x00000001d8bd33b4 libAppleEXR.dylib`axr_error_t LaunchBlocks<ReadPixelsArgs>(void (*)(void*, unsigned long),
*, unsigned long, axr_flags_t) + 472
    frame #4: 0x00000001d8bd2ff4 libAppleEXR.dylib`TileDecoder::ReadPixels(unsigned int, unsigned int, void*, unsigned long
    frame #5: 0x00000001d8bc9d70 libAppleEXR.dylib`Part::ReadRGBAPixels(axr_decoder*, void*, unsigned long, double, axr fla
    frame #6: 0x00000001929a1724 ImageIO`EXRReadPlugin::decodeBlockAppleEXR(void*, unsigned long) + 500
    frame #7: 0x00000001929a1400 ImageIO`EXRReadPlugin::decodeImageImp(IIODecodeParameter*, II0ImageType, __IOSurface**, __
lockSet**) + 612
    frame #8: 0x000000019295a8e4 ImageIO`II0ReadPlugin::callDecodeImage(IIODecodeParameter*, II0ImageType, __IOSurface**, _
lockSet**) + 1000
    frame #9: 0x00000001928a6b04 ImageIO`II0_Reader::CopyImageBlockSetProc(void*, CGImageProvider*, CGRect, CGSize, __CFDic
    frame #10: 0x00000001928c2984 ImageIO`II0ImageProviderInfo::copyImageBlockSetWithOptions(CGImageProvider*, CGRect, CGSi
nst*) + 408
    frame #11: 0x00000001928a6670 ImageIO`II0ImageProviderInfo::CopyImageBlockSetWithOptions(void*, CGImageProvider*, CGRec
nary const*) + 908
    frame #12: 0x000000018c53b25c CoreGraphics`imageProvider_retain_data + 96
    frame #13: 0x000000018c53b194 CoreGraphics`CGDataProviderRetainData + 76
    frame #14: 0x000000018c53b1dc CoreGraphics`provider_for_destination_retain_data + 28
    frame #15: 0x000000018c53b194 CoreGraphics`CGDataProviderRetainData + 76
```



```
(lldb) bt
* thread #1, queue = 'com.apple.root.user-interactive-qos', stop reason = EXC_GUARD (code=1, subcode=0x4141414141414149)
  * frame #0: 0x00000001855bd01c libdispatch.dylib`_dispatch_apply_set_attr_behavior + 16
    frame #1: 0x00000001855bd718 libdispatch.dylib`_dispatch_apply_invoke_and_wait + 64
    frame #2: 0x00000001855bc944 libdispatch.dylib`_dispatch_apply_with_attr_f + 1312
    frame #3: 0x00000001d8bd33b4 libAppleEXR.dylib`axr_error_t LaunchBlocks<ReadPixelsArgs>(void (*)(void*, unsigned long),
*, unsigned long, axr_flags_t) + 472
    frame #4: 0x00000001d8bd2ff4 libAppleEXR.dylib`TileDecoder::ReadPixels(unsigned int, unsigned int, void*, unsigned long
    frame #5: 0x00000001d8bc9d70 libAppleEXR.dylib`Part::ReadRGBAPixels(axr_decoder*, void*, unsigned long, double, axr fla
    frame #6: 0x00000001929a1724 ImageIO`EXRReadPlugin::decodeBlockAppleEXR(void*, unsigned long) + 500
    frame #7: 0x00000001929a1400 ImageIO`EXRReadPlugin::decodeImageImp(IIODecodeParameter*, II0ImageType, __IOSurface**, __
ockSet**) + 612
    frame #8: 0x000000019295a8e4 ImageIO`II0ReadPlugin::callDecodeImage(IIODecodeParameter*, II0ImageType, __IOSurface**, _
lockSet**) + 1000
    frame #9: 0x00000001928a6b04 ImageIO`II0_Reader::CopyImageBlockSetProc(void*, CGImageProvider*, CGRect, CGSize, __CFDic
    frame #10: 0x00000001928c2984 ImageIO`II0ImageProviderInfo::copyImageBlockSetWithOptions(CGImageProvider*, CGRect, CGSi
nst*) + 408
    frame #11: 0x00000001928a6670 ImageIO`II0ImageProviderInfo::CopyImageBlockSetWithOptions(void*, CGImageProvider*, CGRec
nary const*) + 908
    frame #12: 0x000000018c53b25c CoreGraphics`imageProvider_retain_data + 96
    frame #13: 0x000000018c53b194 CoreGraphics`CGDataProviderRetainData + 76
    frame #14: 0x000000018c53b1dc CoreGraphics`provider_for_destination_retain_data + 28
    frame #15: 0x000000018c53b194 CoreGraphics`CGDataProviderRetainData + 76
```


13:08

Photos-2026-05-29-130710....

```
{
  "app_name": "Photos",
  "timestamp": "2026-05-29 13:07:10.00 +0100",
  "app_version": "1.0",
  "slice_uuid": "89aa69a2-739b-3fb6-baa9-85c73cfc9418",
  "build_version": "832.0.107",
  "platform": "2",
  "bundleID": "com.apple.mobileslideshow",
  "share_with_app_devs": 0,
  "is_first_party": 1,
  "bug_type": "309",
  "os_version": "iPhone OS 26.3.1 (23D8133)",
  "roots_installed": 0,
  "name": "Photos",
  "incident_id": "86E306E0-B49C-41AB-AE55-E45434E551DD"
}
{
  "uptime": 1600000,
  "procRole": "Foreground",
  "version": 2,
  "userID": 501,
  "deployVersion": 210,
  "modelCode": "iPhone15,2",
  "coalitionID": 21607,
  "osVersion": {
    "isEmbedded": true,
    "train": "iPhone OS 26.3.1",
    "releaseType": "User",
    "build": "23D8133"
  },
  "captureTime": "2026-05-29 13:07:10.1645 +0100",
  "codeSigningMonitor": 2,
  "incident": "86E306E0-B49C-41AB-AE55-E45434E551DD",
  "pid": 7752,
  "translated": false,
  "cpuType": "ARM-64",
  "procLaunch": "2026-05-29 13:07:09.6477 +0100",
  "procStartAbsTime": 38676858238826,
  "procExitAbsTime": 38676870426321,
  "procName": "Photos",
  "procPath": "\private\var\containers\Bundle\Application\FC8A5FA7-2269-4A12-8BBB-C90F8079A38E\Photos.app\Photos",
  "bundleInfo": {
    "CFBundleShortVersionString": "1.0",
    "CFBundleVersion": "832.0.107",
    "CFBundleIdentifier": "com.apple.mobileslideshow",
    "storeInfo": {
      "deviceIdentifierForVendor": "0C7D15F6-B8D6-41A1-8A7E-FA7EE5DA8A2F"
    },
    "parentProc": "launchd",
    "parentPid": 1,
    "coalitionName": "com.apple.mobileslideshow",
    "crashReporterKey": "a6e78dc0fa3e0954f9cfe3d8b1a3c166f96f5745",
    "appleIntelligenceStatus": {
      "reasons": ["deviceNotCapable"],
      "state": "unavailable"
    },
    "developerMode": 1,
    "bootProgressRegister": "0x20800004",
    "wasUnlockedSinceBoot": 1,
    "isLocked": 0,
    "codeSigningID": "com.apple.mobileslideshow",
    "codeSigningTeamID": "",
    "flags": 570434305,
    "validationCategory": 1,
    "trustLevel": 9,
    "auxiliaryInfo": 1516587483788738578,
    "Stream": {
      "beforePC": "\p9AEAkJTyMZfVsS3wtWI7kbQGAPmoHgD59FfBqA==",
      "atPC": "DDxCA8vsAjRC5EJAUC5IAICeugnnoxofAQBxKACAUG==",
      "ID": "2CB39D26-1D58-480E-9C8F-4AC05D05B563",
      "": "1.0.0"
    }
  }
}
```



QuickLookUIService-2026-05-28-150057.ips

Now

Clear

Reload

Reveal

Share

Search

Translated Report (Full Report Below)

Process: QuickLookUIService [45157]

Path: /System/Library/Frameworks/QuickLookUI.framework/Versions/A/XPCServices/QuickLookUIService.xpc/Contents/MacOS/QuickLookUIService

Identifier: com.apple.quicklook.QuickLookUIService

Version: 1.0 (1)

Build Info: QuickLook_executables-1018005005000000~171

Code Type: ARM-64 (Native)

Role: Background

Parent Process: launchd [1]

Coalition: com.apple.finder [761]

Responsible Process: Finder [726]

User ID: 501

Date/Time: 2026-05-28 14:59:17.4976 +0100

Launch Time: 2026-05-28 14:57:15.4103 +0100

Hardware Model: MacBookPro18,3

OS Version: macOS 26.4.1 (25E253)

Release Type: User

Crash Reporter Key: DEBCD430-0B50-EC94-BFE4-B45E2DD070D5

Incident Identifier: DF580219-1AD6-4A6B-AEB1-600C8C228E72

Sleep/Wake UUID: 9E0B1559-2C60-47FF-B4D4-58968DA1EF52

Time Awake Since Boot: 6200 seconds

Time Since Wake: 4488 seconds

System Integrity Protection: disabled

Triggered by Thread: 1, Dispatch Queue: com.apple.root.user-initiated-qos

Exception Type: EXC_BAD_ACCESS (SIGBUS)

Exception Subtype: KERN_PROTECTION_FAILURE at 0x000000fd33c424f

Exception Codes: 0x0000000000000002, 0x000000fd33c424f

Termination Reason: Namespace SIGNAL, Code 10, Bus error: 10

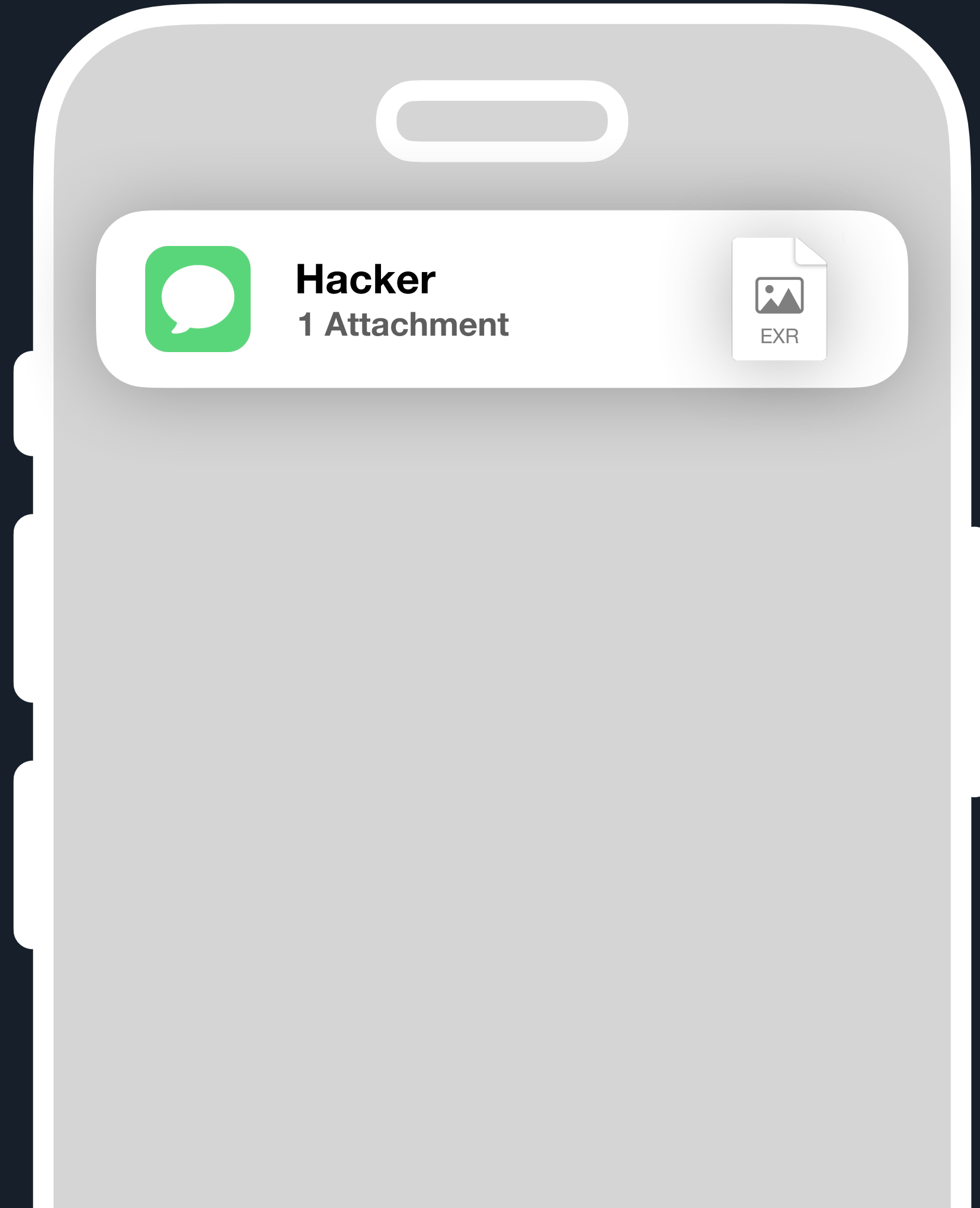
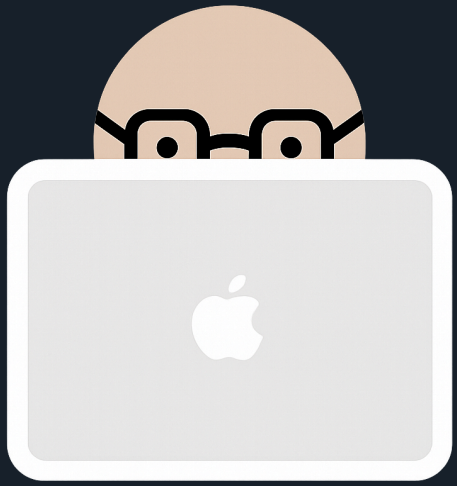
Terminating Process: exc handler [45157]

VM Region Info: 0xfd33c424f is in 0xfc0000000-0x1000000000; bytes after start: 322716239 bytes before end: 751025584

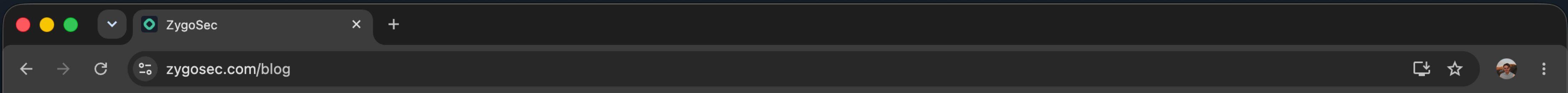
REGION TYPE	START - END	[VSIZE]	PRT/MAX	SHRMOD	REGION DETAIL
Kernel Text (reserved)	bd8800000-ed2000000	[11.9G]	---/---	SM=NUL	reserved VM address space (unallocated)
Kernel Text (reserved)	0xee000000-0x100000000	[1.0G]	---/---	SM=NUL	reserved VM address space (unallocated)
Kernel Text (reserved)	fc0000000-1000000000	[1.0G]	---/---	SM=NUL	reserved VM address space (unallocated)
Kernel Text (reserved)	1000000000-7000000000	[384.0G]	---/---	SM=NUL	reserved VM address space (unallocated)

Dispatch queue: com.apple.main-thread





Maybe 0click reachable?

[Blog](#)[About](#)

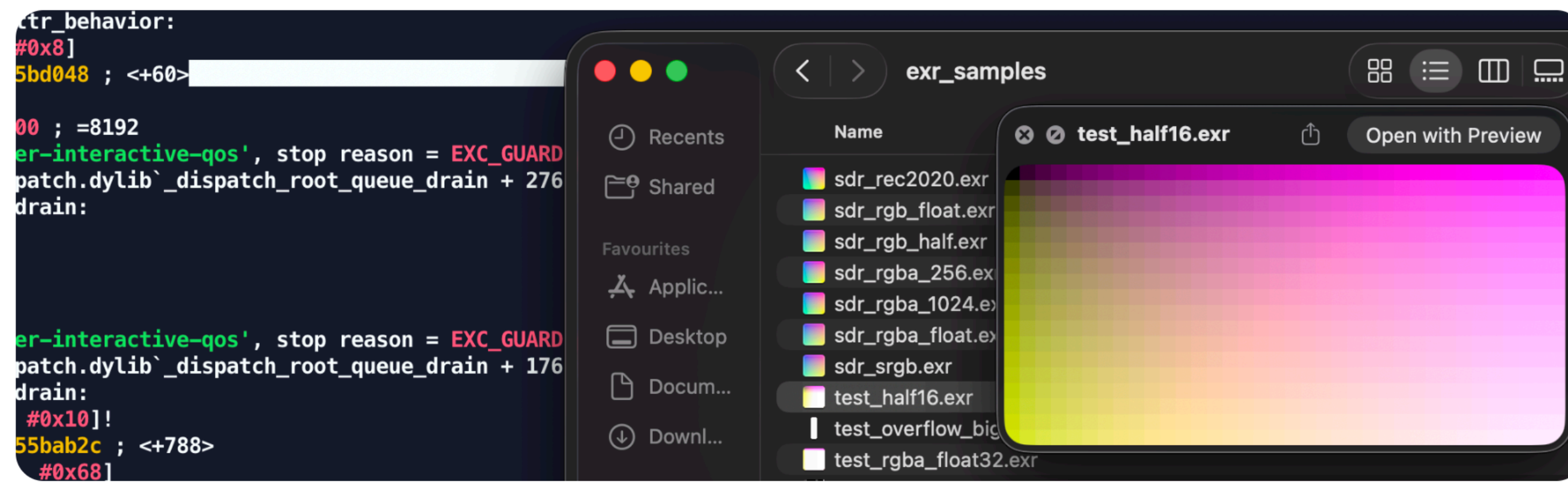
Uncovering an iOS 26 ImageIO Vulnerability (CVE-2026-28990)

In this blog we conduct a root-cause-analysis and build a working PoC for a recent iOS image parsing vulnerability patched in iOS 26.5.



Billy Ellis

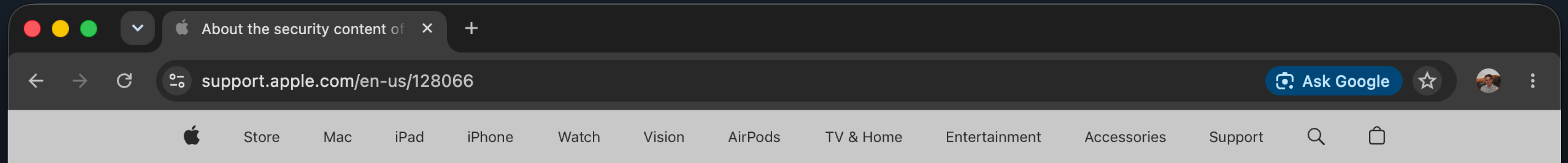
May 25th, 2026 · 8 min read



What next?

Where to go from here?

- Study n-day exploit writeup
- Study in-the-wild samples
- Re-implement as much as possible



About the security content of iOS 26.6 and iPadOS 26.6

This document describes the security content of iOS 26.6 and iPadOS 26.6.

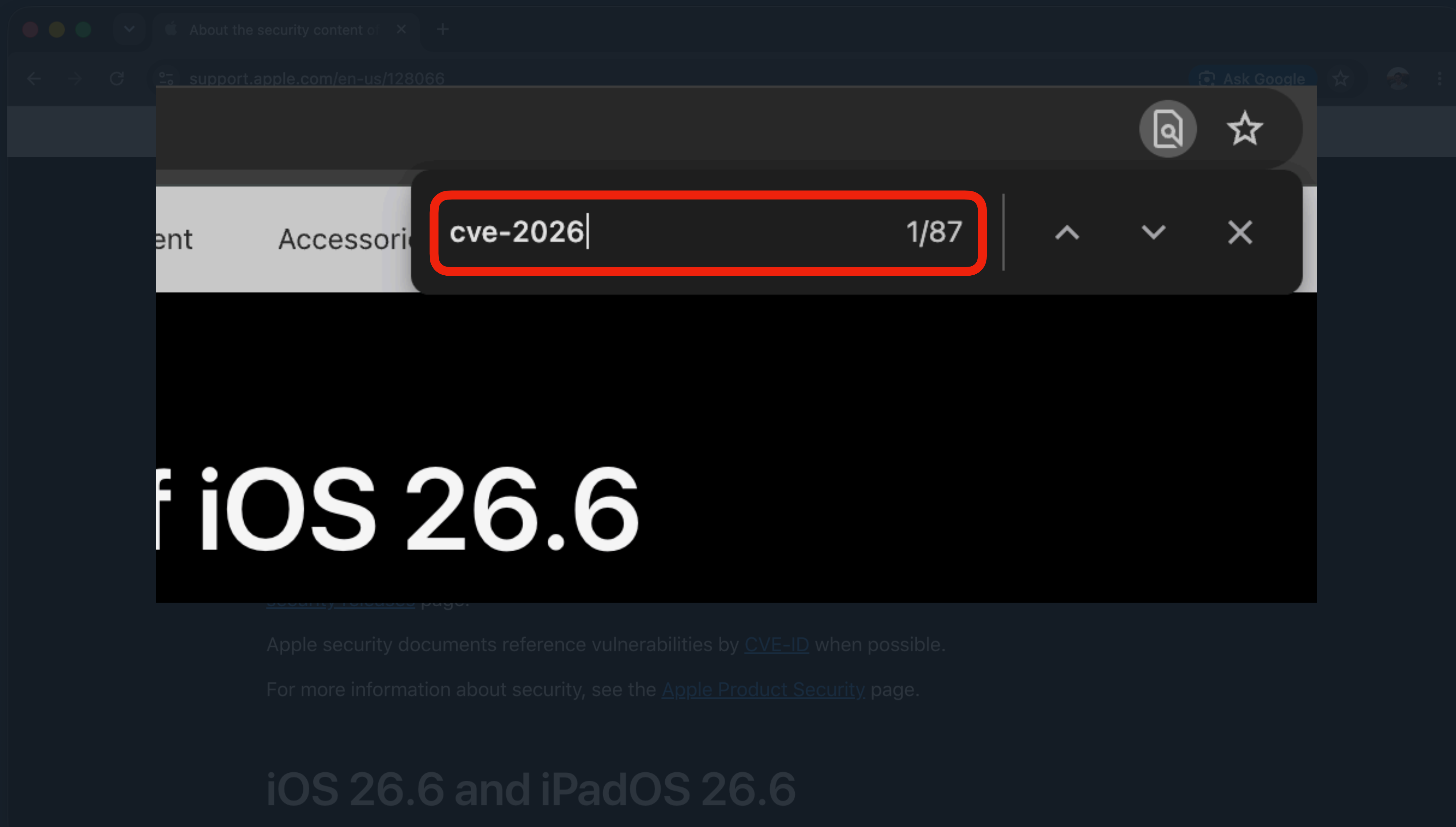
About Apple security updates

For our customers' protection, Apple doesn't disclose, discuss, or confirm security issues until an investigation has occurred and patches or releases are available. Recent releases are listed on the [Apple security releases](#) page.

Apple security documents reference vulnerabilities by [CVE-ID](#) when possible.

For more information about security, see the [Apple Product Security](#) page.

iOS 26.6 and iPadOS 26.6



zygosec.com

youtube.com/@BillyEllis