I Don’t talk about Darwin, no, no, no..

But… for 0x41con…

To follow along: http://NewOSXBook.com/bonus/0x41-16.pdf

download iPhone14/15 iOS16 KC (I know, you all have it already…)
and http://NewOSXBook.com/tools/jtool2 (or jtool2.x64)

Caveat

- I really did quit Darwin (D20, Feb 10th 2020!)
  - Coincided with a bit of a Twitter flame war
  - Really driven by a much more intriguing set of events (get me drunk ;-))
  - Remember last 0x41Con (2019)? That's when I notified y'all in advance
- Short version: I got tired of repeatedly hitting the glass ceiling..
- But.. 0x41con is a damn special occasion*!
- Observations here may be incomplete/inaccurate

- And you made me use KeyNote! I actually MISS PowerPoint now!

LC_FILESET_ENTRY

- Biggest noticeable change in kernelcache

```
morpheus@Bifröst (~/Documents/OSXBook/2nd/src/jtool2) %JCOLOR=1 jtool2 -l ~/Downloads/kernelcache.release.iphone15
This is a BVX kernelcache - We support that now
LC 00: LC_UUID                  UUID: 18FED9BC-0674-CDC5-AF29-AD3214034273
LC 01: LC_BUILD_VERSION         Build Version:           Platform: unknown?? 0.0.0 SDK: 0
LC 02: LC_UNIXTHREAD            Entry Point:             0xfffffff007d1c540
LC 03: LC_DYLD_CHAINED_FIXUPS   Offset:  70565888, Size:    942 (0x434c000-0x434c3ae)
LC 04: LC_SEGMENT_64             Mem: 0xfffffff007004000-0xfffffff00700c000     __TEXT
LC 05: LC_SEGMENT_64             Mem: 0xfffffff00700c000-0xfffffff00779c000     __PRELINK_TEXT
       Mem: 0xfffffff00700c000-0xfffffff00779c000                __PRELINK_TEXT.__text   (Normal)
LC 06: LC_SEGMENT_64             Mem: 0xfffffff00779c000-0xfffffff007c5c000     __DATA_CONST
LC 07: LC_SEGMENT_64             Mem: 0xfffffff007c5c000-0xfffffff00a0f4000     __TEXT_EXEC
LC 08: LC_SEGMENT_64             Mem: 0xfffffff00a0f4000-0xfffffff00a298000     __PRELINK_INFO
       Mem: 0xfffffff00a0f4000-0xfffffff00a298000                __PRELINK_INFO.__info
LC 09: LC_SEGMENT_64             Mem: 0xfffffff00a298000-0xfffffff00a454000     __DATA
LC 10: LC_SEGMENT_64             Mem: 0xfffffff00a454000-0xfffffff00b354000     __LINKEDIT
LC 11: LC_FILESET_ENTRY         vmaddr: 0xfffffff007c5c000 offset: 0xc58000 com.apple.kernel
LC 12: LC_FILESET_ENTRY         vmaddr: 0xfffffff00700c000 offset: 0x8000 com.apple.AGXFirmwareKextG15P_A0RTBuddy
LC 13: LC_FILESET_ENTRY         vmaddr: 0xfffffff00700cbe0 offset: 0x8be0 com.apple.AGXFirmwareKextRTBuddy64
LC 14: LC_FILESET_ENTRY         vmaddr: 0xfffffff007010be0 offset: 0xcbe0 com.apple.AGXG15P_A0
LC 15: LC_FILESET_ENTRY         vmaddr: 0xfffffff00701e110 offset: 0x1a110 com.apple.driver.AOPTouchKext
LC 16: LC_FILESET_ENTRY         vmaddr: 0xfffffff00701ea60 offset: 0x1aa60 com.apple.driver.ASIOKit
LC 17: LC_FILESET_ENTRY         vmaddr: 0xfffffff007026e30 offset: 0x22e30 com.apple.AUC

LC 249: LC_FILESET_ENTRY        vmaddr: 0xfffffff007798800 offset: 0x794800 com.apple.driver.mDNSOffloadUserClient-Embedded
LC 250: LC_FILESET_ENTRY        vmaddr: 0xfffffff007799be0 offset: 0x795be0 com.apple.kec.pthread
LC 251: LC_FILESET_ENTRY        vmaddr: 0xfffffff00779aa20 offset: 0x796a20 com.apple.filesystems.tmpfs
```

LC_FILESET_ENTRY

- Filesets are mini-Mach-Os at `fileoff` to be loaded into `vmaddr`
- Surprisingly, no filesize/vmsize specified (all contiguous anyway)

<mach-o/loader.h>

```
#define LC_FILESET_ENTRY (0x35 | LC_REQ_DYLD) /* used with fileset_entry_command */

/*
 * LC_FILESET_ENTRY commands describe constituent Mach-O files that are part
 * of a fileset. In one implementation, entries are dylibs with individual
 * mach headers and repositionable text and data segments. Each entry is
 * further described by its own mach header.
 */
struct fileset_entry_command {
    uint32_t     cmd;        /* LC_FILESET_ENTRY */
    uint32_t     cmdsize;    /* includes entry_id string */
    uint64_t     vmaddr;     /* memory address of the entry */
    uint64_t     fileoff;    /* file offset of the entry */
    union lc_str entry_id;   /* contained entry id */
    uint32_t     reserved;   /* reserved */
};
```

LC_FILESET_ENTRY

- Makes loader's life easy, but a little less so for kextraction:
  - `__TEXT` segment can be easily patched
  - `__TEXT_EXEC`, `__DATA`, `__DATA_CONST` point back to KC
  - `__LC_[DY]SYMTAB` Symbol/String tables point back to KC
  - Extracting filesets thus requires some tinkering..
  - Handled by `jtool2 -filesets` (use `JDEBUG=1` to see how)
    - Note that inter KEXT references will remain dangling
    - Taken into consideration by `--analyze` (or use `fixkextsyms.sh`)

LC_DYLD_CHAINED_FIXUPS

- LC_DYLD_INFO is so.. 20th Darwin (and still on macOS/Intel)
- LC_DYLD_CHAINED_FIXUPS used instead:
  - Handles imports + symbol references
  - Defines pointers in __DATA[_CONST] for ASLR/signing

LC_DYLD_CHAINED_FIXUPS

<mach-o/loader.h>

```
#define LC_DYLD_CHAINED_FIXUPS (0x34 | LC_REQ_DYLD) /* used with linkedit_data_command */

/*
 * The linkedit_data_command contains the offsets and sizes of a blob
 * of data in the __LINKEDIT segment.
 */
struct linkedit_data_command {
    uint32_t    cmd;            /* LC_CODE_SIGNATURE, LC_SEGMENT_SPLIT_INFO,
                                   LC_FUNCTION_STARTS, LC_DATA_IN_CODE,
                                   LC_DYLIB_CODE_SIGN_DRS,
                                   LC_LINKER_OPTIMIZATION_HINT,
                                   LC_DYLD_EXPORTS_TRIE, or
                                   LC_DYLD_CHAINED_FIXUPS. */
    uint32_t    cmdsize;        /* sizeof(struct linkedit_data_command) */
    uint32_t    dataoff;        /* file offset of data in __LINKEDIT segment */
    uint32_t    datasize;       /* file size of data in __LINKEDIT segment  */
};
```

… Points to …

LC_DYLD_CHAINED_FIXUPS

<mach-o/fixup-chains.h>

```
// header of the LC_DYLD_CHAINED_FIXUPS payload
struct dyld_chained_fixups_header {
    uint32_t    fixups_version;    // 0
    uint32_t    starts_offset;     // offset of dyld_chained_starts_in_image in chain_data
    uint32_t    imports_offset;    // offset of imports table in chain_data
    uint32_t    symbols_offset;    // offset of symbol strings in chain_data
    uint32_t    imports_count;     // number of imported symbol names
    uint32_t    imports_format;    // DYLD_CHAINED_IMPORT*
    uint32_t    symbols_format;    // 0 => uncompressed, 1 => zlib compressed
};

// This struct is embedded in LC_DYLD_CHAINED_FIXUPS payload
struct dyld_chained_starts_in_image {
    uint32_t    seg_count;
    uint32_t    seg_info_offset[1];  // each entry is offset into this struct for that segment
    // followed by pool of dyld_chain_starts_in_segment data
};

// This struct is embedded in dyld_chain_starts_in_image
// and passed down to the kernel for page-in linking
struct dyld_chained_starts_in_segment {
    uint32_t    size;               // size of this (amount kernel needs to copy)
    uint16_t    page_size;          // 0x1000 or 0x4000
    uint16_t    pointer_format;     // DYLD_CHAINED_PTR_*
    uint64_t    segment_offset;     // offset in memory to start of segment
    uint32_t    max_valid_pointer;  // for 32-bit OS, any value beyond this is not a pointer
    uint16_t    page_count;         // how many pages are in array
    uint16_t    page_start[1];      // each entry is offset in each page of first element in chain
                                    // or DYLD_CHAINED_PTR_START_NONE if no fixups on page
 // uint16_t    chain_starts[1];    // some 32-bit formats may require multiple starts per page.
                                    // for those, if high bit is set in page_starts[], then it
                                    // is index into chain_starts[] which is a list of starts
                                    // the last of which has the high bit set
};
```

LC_DYLD_CHAINED_FIXUPS

- Several possible pointer_formats:

<mach-o/fixup-chains.h>

```
// values for dyld_chained_starts_in_segment.pointer_format
enum {
    DYLD_CHAINED_PTR_ARM64E                 =  1,    // stride 8, unauth target is vmaddr
    DYLD_CHAINED_PTR_64                     =  2,    // target is vmaddr
    DYLD_CHAINED_PTR_32                     =  3,
    DYLD_CHAINED_PTR_32_CACHE               =  4,
    DYLD_CHAINED_PTR_32_FIRMWARE            =  5,
    DYLD_CHAINED_PTR_64_OFFSET              =  6,    // target is vm offset
    DYLD_CHAINED_PTR_ARM64E_OFFSET          =  7,    // old name
    DYLD_CHAINED_PTR_ARM64E_KERNEL          =  7,    // stride 4, unauth target is vm offset
    DYLD_CHAINED_PTR_64_KERNEL_CACHE        =  8,
    DYLD_CHAINED_PTR_ARM64E_USERLAND        =  9,    // stride 8, unauth target is vm offset
    DYLD_CHAINED_PTR_ARM64E_FIRMWARE        = 10,    // stride 4, unauth target is vmaddr
    DYLD_CHAINED_PTR_X86_64_KERNEL_CACHE    = 11,    // stride 1, x86_64 kernel caches
    DYLD_CHAINED_PTR_ARM64E_USERLAND24      = 12,    // stride 8, unauth target is vm offset, 24-bit bind
};
```

LC_DYLD_CHAINED_FIXUPS

```
morpheus@Bifröst (~/Documents/OSXBook/2nd/src/jtool2) %JFIXUP=NO jtool2 --filesets ~/Downloads/kernelcache.release.iphone1
This is a BVX kernelcache - We support that now
Warning: NOT FIXING UP! Not taking responsibility for kextraction!
Got PRELINK_INFO
241 LC_FILESET_ENTRY commands
rebuilding - 57638912 bytes (from 0xc58000)
All files are in /tmp/extracted. Kernel is in /tmp/extracted/kernel.rebuilt
```

(or use dd if=... bs=0x... skip= ... instead)

```
morpheus@Bifröst (~/Documents/OSXBook/2nd/src/jtool2) %jtool2 -d __DATA_CONST /tmp/extracted/kernel.rebuil
Dumping 994576 bytes from 0xffffff00779c000 (Offset 0x798000, __DATA_CONST.__auth_ptr):
0xffffff00779c000: E4 E3 44 01 00 00 10 80    ..D.....
0xffffff00779c008: C0 47 2B 01 00 00 10 00    .G+.....
0xffffff00779c010: 64 4D 2B 01 00 00 10 00    dM+.....
0xffffff00779c018: 20 86 2B 01 00 00 10 00     .+.....
0xffffff00779c020: A4 A3 2B 01 00 00 10 00    ..+.....
0xffffff00779c028: 64 B0 2B 01 00 00 10 00    d.+.....
0xffffff00779c030: EC B9 2B 01 00 00 10 00    ..+.....
0xffffff00779c038: FC CA 2B 01 00 00 10 00    ..+.....
0xffffff00779c040: 70 E2 2B 01 00 00 10 00    p.+.....
0xffffff00779c048: 3C 13 2C 01 00 00 10 00    <.,.....
```

LC_DYLD_CHAINED_FIXUPS

```
morpheus@Bifröst (~/Documents/OSXBook/2nd/src/jtool2) %JFIXUP=2 jtool2 --filesets ~/Downloads/kernelcache.release.iphone15
This is a BVX kernelcache - We support that now
FIXING UP
FIXING UP SEGMENT __DATA_CONST
                Size: 0x278, Page Size: 4000, pointer format: 8, seg offset: 0x798000-0xc58000, page count: 304
Fixing page 0 (0x798000) - first pointer @0x0 (0x0):
 0x801000000144e3e4, Type: 8 0x801000000144e3e4 --> 0xfffffff0084523e4 , next: 2
pointer @0x798000:  0x100000012b47c0, Type: 8 0x100000012b47c0 --> 0xfffffff0082b87c0 , next: 2
pointer @0x798008:  0x100000012b4d64, Type: 8 0x100000012b4d64 --> 0xfffffff0082b8d64 , next: 2
pointer @0x798010:  0x100000012b8620, Type: 8 0x100000012b8620 --> 0xfffffff0082bc620 , next: 2
pointer @0x798018:  0x100000012ba3a4, Type: 8 0x100000012ba3a4 --> 0xfffffff0082be3a4 , next: 2
pointer @0x798020:  0x100000012bb064, Type: 8 0x100000012bb064 --> 0xfffffff0082bf064 , next: 2


morpheus@Bifröst (~/Documents/OSXBook/2nd/src/jtool2) %jtool2 -d __DATA_CONST /tmp/extracted/kernel.rebui
Dumping 994576 bytes from 0xfffffff00779c000 (Offset 0x798000, __DATA_CONST.__auth_ptr):
0xfffffff00779c000: 0xfffffff0084523e4          _func_fffffff0084523e4
0xfffffff00779c008: 0xfffffff0082b87c0          _func_fffffff0082b87c0
0xfffffff00779c010: 0xfffffff0082b8d64          _func_fffffff0082b8d64
0xfffffff00779c018: 0xfffffff0082bc620          _func_fffffff0082bc620
0xfffffff00779c020: 0xfffffff0082be3a4          _func_fffffff0082be3a4
0xfffffff00779c028: 0xfffffff0082bf064          _func_fffffff0082bf064
0xfffffff00779c030: 0xfffffff0082bf9ec          _func_fffffff0082bf9ec
0xfffffff00779c038: 0xfffffff0082c0afc          _func_fffffff0082c0afc
0xfffffff00779c040: 0xfffffff0082c2270          _func_fffffff0082c2270
0xfffffff00779c048: 0xfffffff0082c533c          _func_fffffff0082c533c
```

LC_DYLD_CHAINED_FIXUPS

- User mode Fixups encode imports as well
  - (no more DYLD_INFO_ONLY)

```
// values for dyld_chained_fixups_header.imports_format
enum {
    DYLD_CHAINED_IMPORT          = 1,
    DYLD_CHAINED_IMPORT_ADDEND   = 2,
    DYLD_CHAINED_IMPORT_ADDEND64 = 3,
};
```

```
// DYLD_CHAINED_IMPORT
struct dyld_chained_import {
    uint32_t    lib_ordinal :  8,
                weak_import :  1,
                name_offset : 23;
};

// DYLD_CHAINED_IMPORT_ADDEND
struct dyld_chained_import_addend {
    uint32_t    lib_ordinal :  8,
                weak_import :  1,
                name_offset : 23;
    int32_t     addend;
};

// DYLD_CHAINED_IMPORT_ADDEND64
struct dyld_chained_import_addend64 {
    uint64_t    lib_ordinal : 16,
                weak_import :  1,
                reserved    : 15,
                name_offset : 32;
    uint64_t    addend;
};
```

Developer Mode

- Rationale: 99% of Retail builds DO NOT NEED Debugging
- Implementation:
  - Debugging features/DDI etc are disabled by default
  - User needs to explicitly enable Developer Mode
  - Behind the scenes work performed by AMFI
  - Setting stored in NVRAM, requires reboot

Developer Mode

- `amfid` checks developer mode status on boot
- GUI handler uses CFUserNotificationCreate to prompt
- Enable/disable via User Client methods
- New entitlements (q.v. http://NewOSXBook.com/ent.jl)

```
morpheus@Bifröst (~/Documents/OSXBook/2nd/src/jtool2) %jtool2 --ent /Volumes/Sydney20A380.D730S/usr/libexec/amfid
com.apple.private.CoreAuthentication.SPI: true
com.apple.private.security.storage.amfid: true
com.apple.springboard.CFUserNotification: true
com.apple.private.iokit.nvram-read-access: true
com.apple.private.iokit.nvram-write-access: true
com.apple.private.LocalAuthentication.Storage: true
com.apple.private.amfi.developer-mode-control: true
com.apple.private.security.storage.driverkitd: true
com.apple.private.LocalAuthentication.CallerName: true
com.apple.private.CoreAuthentication.BackgroundUI: true
com.apple.private.security.storage.AppDataContainers: true
com.apple.security.exception.iokit-user-client-class:
        AppleMobileFileIntegrityUserClient
com.apple.private.usernotifications.bundle-identifiers:
        com.apple.amfi.usernotifications
```

Developer Mode

```
_developerModeArmInKernel:
100005ae0       0xaa0103e0  _MOV_R      X0, X1                  R0 = R1 (0x0)
100005ae4       0x52800161  MOVZ        W1, 0xb                  ; R1 = 0xb
100005ae8       0xd2800002  MOVZ        X2, 0x0                  ; R2 = 0x0
100005aec       0x52800003  MOVZ        W3, 0x0                  ; R3 = 0x0
100005af0       0xd2800004  MOVZ        X4, 0x0                  ; R4 = 0x0
100005af4       0xd2800005  MOVZ        X5, 0x0                  ; R5 = 0x0
100005af8       0x14001be2  B           0x10000ca80              _IOConnectCallScalarMethod
        _IOConnectCallScalarMethod(0 ,0xb,0 ,0 ,0 ,0 );
_developerModeDisableInKernel:
100005afc       0xaa0103e0  _MOV_R      X0, X1                  R0 = R1 (0x0)
100005b00       0x52800181  MOVZ        W1, 0xc                  ; R1 = 0xc
100005b04       0xd2800002  MOVZ        X2, 0x0                  ; R2 = 0x0
100005b08       0x52800003  MOVZ        W3, 0x0                  ; R3 = 0x0
100005b0c       0xd2800004  MOVZ        X4, 0x0                  ; R4 = 0x0
100005b10       0xd2800005  MOVZ        X5, 0x0                  ; R5 = 0x0
100005b14       0x14001bdb  B           0x10000ca80              _IOConnectCallScalarMethod
        _IOConnectCallScalarMethod(0 ,0xc,0 ,0 ,0 ,0 );
```

Developer Mode

- From KEXT Side, UserClient is extended:

```
morpheus@Bifröst (~/Documents/OSXBook/2nd/src/jtool2) %jtool2 -S /tmp/extracted/AppleMobileFileIntegrity.kext|
pipe> c++filt|
pipe pipe> grep Dev| grep Mode
ffffff008ff7ac4 short AppleMobileFileIntegrityUserClient::armDeveloperMode(OSObject*, void*, IOExternalMethodArguments*)
ffffff008ff7b08 short AppleMobileFileIntegrityUserClient::turnOffDeveloperMode(OSObject*, void*, IOExternalMethodArguments*)
ffffff008ff7b84 short AppleMobileFileIntegrityUserClient::turnOnDeveloperMode(OSObject*, void*, IOExternalMethodArguments*)
ffffff008ffadac short amfiDeveloperModeStatus(sysctl_oid*, void*, int, sysctl_req*)
ffffff008ffb024 short AMFILoadDevModeStatus_Thread(void*, int)
ffffff008ffb09c short loadDevModeStatus()
ffffff008ffb2c8 short armDeveloperMode()
ffffff008ffb43c short turnOnDeveloperMode()
ffffff008ffb468 short turnOffDeveloperMode()
ffffff00900fc94 short loadDevModeStatus() (.cold.1)
ffffff00900fcec short turnOnDeveloperMode() (.cold.1)
```

Developer Mode

- `armDeveloperMode()` stashes in NVRAM

```
morpheus@Bifröst (~/Documents/OSXBook/2nd/src/jtool2) %jtool2 -dt __Z16armDeveloperModev \
> /tmp/extracted/AppleMobileFileIntegrity.kext  | grep -v ^fff
opened companion file AppleMobileFileIntegrity.kext.ARM64.608F8748-3A3F-3B92-A5CE-A4B2BC37A37C
Disassembling 86068 bytes from address 0xffffff008ffb2c8 (offset 0xf498):
__Z16armDeveloperModev:
        __ZN9IOService15serviceMatchingEPKcP12OSDictionary("IODTNVRAM",0 );
        __ZN9IOService22waitForMatchingServiceEP12OSDictionaryy(?,0 );
        __ZN15OSMetaClassBase12safeMetaCastEPKS_PK11OSMetaClass(?,0xffffff007a22c80);
        __ZN8OSSymbol11withCStringEPKc("security-mode-change-enable");
        __ZN8OSString11withCStringEPKc("true");
        _IOLog("AMFI: Failed to set NVRAM variable\n");
```

Developer Mode

- Entitlements now subject to validation of `developer_mode_state()`

```
_developerModeEntitlements:
0xfffffff007a23830: 00 00 00 00 00 00 00 00    ........
0xfffffff007a23838: 0xfffffff00733a79c          "get-task-allow"
0xfffffff007a23840: 00 00 00 00 00 00 00 00    ........
0xfffffff007a23848: 0xfffffff00733a7ab          "task_for_pid-allow"
0xfffffff007a23850: 00 00 00 00 00 00 00 00    ........
0xfffffff007a23858: 0xfffffff00733a780          "com.apple.system-task-ports"
0xfffffff007a23860: 00 00 00 00 00 00 00 00    ........
0xfffffff007a23868: 0xfffffff00733a7be          "com.apple.system-task-ports.control"
0xfffffff007a23870: 00 00 00 00 00 00 00 00    ........
0xfffffff007a23878: 0xfffffff00733a7e2          "com.apple.system-task-ports.token.control"
0xfffffff007a23880: 00 00 00 00 00 00 00 00    ........
0xfffffff007a23888: 0xfffffff00733a80c          "com.apple.private.cs.debugger"
0xfffffff007a23890: 00 00 00 00 00 00 00 00    ........
0xfffffff007a23898: 00 00 00 00 00 00 00 00    ........
```

Developer Mode

- `developer_mode_state()` controlled by PPL

```
_enable_developer_mode:
fffffff0081c05cc        0x52800020  MOVZ        W0, 0x1                 ; R0 = 0x1
fffffff0081c05d0        0x17ed8d0d  B           0xfffffff007d23a04      _pmap_toggle_developer_mode_ppl
        _pmap_toggle_developer_mode_ppl(0x1);
_disable_developer_mode:
fffffff0081c05d4        0x52800000  MOVZ        W0, 0x0                 ; R0 = 0x0
fffffff0081c05d8        0x17ed8d0b  B           0xfffffff007d23a04      _pmap_toggle_developer_mode_ppl
        _pmap_toggle_developer_mode_ppl(0 );
_developer_mode_state:
fffffff0081c05dc        0xf00106c8  ADRP        X8, 8411                R8 = 0xfffffff00a29b000
fffffff0081c05e0        0x9127e108  ADD         X8, X8, #2552           R8 = R8 + 0x9f8 = 0xfffffff00a29b9f8
fffffff0081c05e4        0x38bfc108  LDAPRB  W8, [X8]
fffffff0081c05e8        0x12000100  AND         W0, W8, #0x1
fffffff0081c05ec        0xd65f03c0  RET                             ..
```

Developer Mode

```
_pmap_toggle_developer_mode_internal:
fffffff00846ba3c        0xd503237f  PACIBSP
fffffff00846ba40        0xd10083ff  SUB         SP, SP, 32              R31 = SP - 0x20 (0xfffffffffffffe0)
fffffff00846ba44        0xa9017bfd  STP         X29, X30, [SP, #16]         ppl_developer_mode_set
fffffff00846ba48        0x910043fd  ADD         X29, SP, #16
fffffff00846ba4c        0xd000f168  ADRP        X8, 7726                R8 = 0xfffffff00a299000
fffffff00846ba50        0x394e4109  LDRB        W9, [X8, #912]
fffffff00846ba54        0x9000f18a  ADRP        X10, 7728               R10 = 0xfffffff00a29b000
fffffff00846ba58        0x3967e14a  LDRB        W10, [X10, #2552]           ppl_developer_mode_storage
fffffff00846ba5c        0x3700006a  TBNZ        W10, #0, 0xfffffff00846ba68
fffffff00846ba60        0x34000040  CBZ         X0, 0xfffffff00846ba68
fffffff00846ba64        0x37000129  TBNZ        W9, #0, 0xfffffff00846ba88
fffffff00846ba68        0x52800029  MOVZ        W9, 0x1                  ; R9 = 0x1
fffffff00846ba6c        0x390e4109  STRB        W9, [X8, #912]
fffffff00846ba70        0x9000f188  ADRP        X8, 7728                R8 = 0xfffffff00a29b000
fffffff00846ba74        0x9127e108  ADD         X8, X8, #2552           R8 = R8 + 0x9f8 = 0xfffffff00a29b9f8
fffffff00846ba78        0x089ffd00  STLXR       W31, X0, [X8]               ppl_developer_mode_storage
fffffff00846ba7c        0xa9417bfd  LDP         X29, X30, [SP, #0x10]
fffffff00846ba80        0x910083ff  ADD         X31, SP, #32
fffffff00846ba84        0xd65f0fff  RETAB
fffffff00846ba88        0x90ffc1a8  ADRP        X8, 2095156             R8 = 0xfffffff007c9f000
fffffff00846ba8c        0x91104508  ADD         X8, X8, #1041           R8 = R8 + 0x411 = 0xfffffff007c9f411
fffffff00846ba90        0x528aa929  MOVZ        W9, 0x5549               ; R9 = 0x5549
fffffff00846ba94        0xa90027e8  STP         X8, X9, [SP, #0]
fffffff00846ba98        0x90ffc1c0  ADRP        X0, 2095160             R0 = 0xfffffff007ca3000
fffffff00846ba9c        0x91251800  ADD         X0, X0, #2374           R0 = R0 + 0x946 = 0xfffffff007ca3946
fffffff00846baa0        0x97ffa1f6  BL          0xfffffff008454278      _panic
        _panic("PMAP_CS: attempted to enable developer mode incorrectly @%s:%d");
```

Trust Caches

```
0xffffff0077ea888: 0xffffff007cd571b          "personalized.engineering-root"
0xffffff0077ea890: 0xffffff007e7a4b0          _func_ffffff007e7a4b0
0xffffff0077ea898: 0xffffff007e7a498          _func_ffffff007e7a498
0xffffff0077ea8a0: 73 72 74 6C 00 00 00 00    srtl....
0xffffff0077ea8a8: 00 00 00 00 00 00 00 00    ........
0xffffff0077ea8b0: 0xffffff007cd5739          "personalized.trust-cache"
0xffffff0077ea8b8: 0xffffff007e7a4b0          _func_ffffff007e7a4b0
0xffffff0077ea8c0: 0xffffff007e7a488          _func_ffffff007e7a488
0xffffff0077ea8c8: 73 72 74 6C 00 00 00 00    srtl....
0xffffff0077ea8d0: 00 00 00 00 00 00 00 00    ........
0xffffff0077ea8d8: 0xffffff007cd5752          "personalized.pdi"
0xffffff0077ea8e0: 0xffffff007e7a474          _func_ffffff007e7a474
0xffffff0077ea8e8: 0xffffff007e7a464          _func_ffffff007e7a464
0xffffff0077ea8f0: 76 64 72 74 00 00 00 00    vdrt....
0xffffff0077ea8f8: 00 00 00 00 00 00 00 00    ........
0xffffff0077ea900: 0xffffff007cd5763          "personalized.ddi"
0xffffff0077ea908: 0xffffff007e7a4b0          _func_ffffff007e7a4b0
0xffffff0077ea910: 0xffffff007e7a464          _func_ffffff007e7a464
```

syscalls/traps

- Several new syscalls:

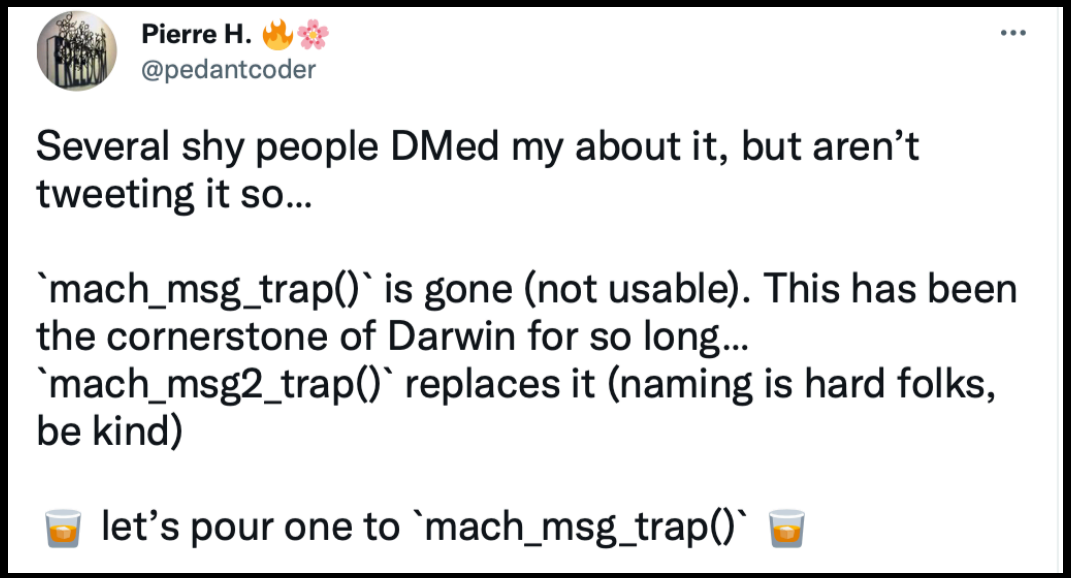
```
#define     SYS_debug_syscall_reject_config 548
#define     SYS_graftdmg        549
#define     SYS_map_with_linking_np 550
#define     SYS_freadlink       551
#define     SYS_record_system_event 552
#define     SYS_mkfifoat        553
#define     SYS_mknodat         554
#define     SYS_ungraftdmg      555
```

- Two new Mach traps: 13 and 47
- Panacea Protection Layer calls up to 106

mach_msg2

(this one's for you, @doadam)

- Lots of buzz about mach_msg revamp (as trap#47)
  - mach_msg_overwrite_trap is deprecated!
  - https://twitter.com/pedantcoder/status/1534971013225517056



- TL;DR: pedant coders should pay even more attention to detail

Launch Constraints

- Superbly detailed by Linus Henze*
- https://gist.github.com/LinusHenze/4cd5d7ef057a144cda7234e2c247c056

* - wen ETA Fugu iOS16??

Probably missed stuff..

- ..but hey, it's been a few years..
- I'm comparing iOS and Android tomorrow, so may talk about cryptex, XPC/mach_msg/syscall filtering, etc then.
- That's all for now