A Story of an Apple Patch

Or should we trust the vendor’s descriptions?

Digital Security

LEADING INFORMATION SECURITY SERVICES PROVIDER

870+
clients

1802
projects

ТАСС · ФОСАГРО · ВТБ24 · mail.ru group · МТС · Альфа-Банк · Сбербанк · Leroy Merlin · Газпром · QIWI

16
years in information security

19
industries

- Software development
- Banks and finance
- Telecom
- Transport and logistics
- Retail
- Production
- Media
- Energy
- Blockchain, etc.

1300
vulnerabilities found in 2018

Acknowledgments

Adobe · The Apache Software Foundation · hp · Microsoft · vmware · Apple · SAP · Twitter · Oracle · Cisco

75
research papers

100+
experts

165
talks at international conferences

HITB · DEFCON · YSTS · CONFidence
CONFidence · BlackHat · RSA · Infosec in the City ...

Digital Security

Agenda

- Binary diffing
- Apple software and patch diffing
- Our story about CVE-2019-8574

Digital Security

Binary diffing

Introduction to Binary Diffing

Digital Security

Binary and patch diffing

Once Upon a Time ...

- **Binary diffing** is a process of comparing two versions of an executable file.
- **Patch diffing** is a process of comparing two versions of an executable file, one patched and the other unpatched, in order to identify a vulnerability.

- “Xakep” magazine #169
  - Feabrury 2013

Digital Security

Why?

- Analysis of code development
- Importing knowledge
- Train on real software
  - From CTF to Real world tasks
- Create 1-day exploits
  - For both attack and defense
    - Patch-gapping
- 0-day vulnerability search
  - Developer may err in patching
  - Developer may overlook something
  - Studying a new pattern
- …

> Patch-gapping is the practice of exploiting vulnerabilities in open-source software that are already fixed (or are in the process of being fixed) by the developers before the actual patch is shipped to users. This window, in which the issue is semi-public while the user-base remains vulnerable, can range from from days to months. It is increasingly seen as a serious concern, with possible in-the-wild uses detected by Google. In a previous post, we demonstrated the feasibility of developing a 1day exploit for Chrome well before a patch is rolled out to users. In a similar vein, this post details the discovery, analysis and exploitation of another recent 1day vulnerability affecting Chrome.

source

> For many of the exploits it is unclear whether they were originally exploited as 0day or as 1day after a fix had already shipped. It is also unknown how the attackers obtained knowledge of the vulnerabilities in the first place. Generally they could have discovered the vulnerabilities themselves or used public exploits released after a fix had shipped. Furthermore, at least for WebKit, it is often possible to extract details of a vulnerability from the public source code repository **before** the fix has been shipped to users. CVE-2019-8518 can be used to highlight this problem (as can many other recent vulnerabilities). The vulnerability was publicly fixed in WebKit HEAD on Feb 9 2019 with commit 4a23c92e6883. This commit contains a testcase that triggers the issue and causes an out-of-bounds access into a JSArray - a scenario that is usually easy to exploit. However, the fix only shipped to users with the release of iOS 12.2 on March 25 2019, roughly one and a half months after details about the vulnerability were public. An attacker in possession of a working exploit for an older WebKit vulnerability would likely only need a few days to replace the underlying vulnerability and thus gain the capability to exploit up-to-date devices without the need to find new vulnerabilities themselves. It is likely that this happened for at least some of the following exploits.

source

Digital Security

Do not forget about patch diffing

Forgetting about patch diffing

One of the lessons that our industry sometimes (and to my surprise) forgets is: Public availability of a patch is, from the attacker perspective, not much different than a detailed analysis of the vulnerability including a vulnerability trigger.

There used to be a cottage industry of folks that analyze patches and write reports on what the fixed bugs are, whether they were fixed correctly, and how to trigger them. They usually operated away from the spotlight, but that does not mean they do not exist - many were our customers.

People in the offensive business can build infrastructure that helps them rapidly analyze patches and get the information they need out of them. Defenders, mostly due to organizational and not technical reasons, can not do this. This means that in the absence of a full discussion of the vulnerability, defenders will be at a significant information disadvantage compared to attackers.

Without understanding the details of the vulnerability, networks and hosts cannot be monitored for its exploitation, and mitigations-other-than-patching cannot be applied.

Professional attackers, on the other hand, will have all the information about a vulnerability not long after they obtain a patch (if they did not have it beforehand already).

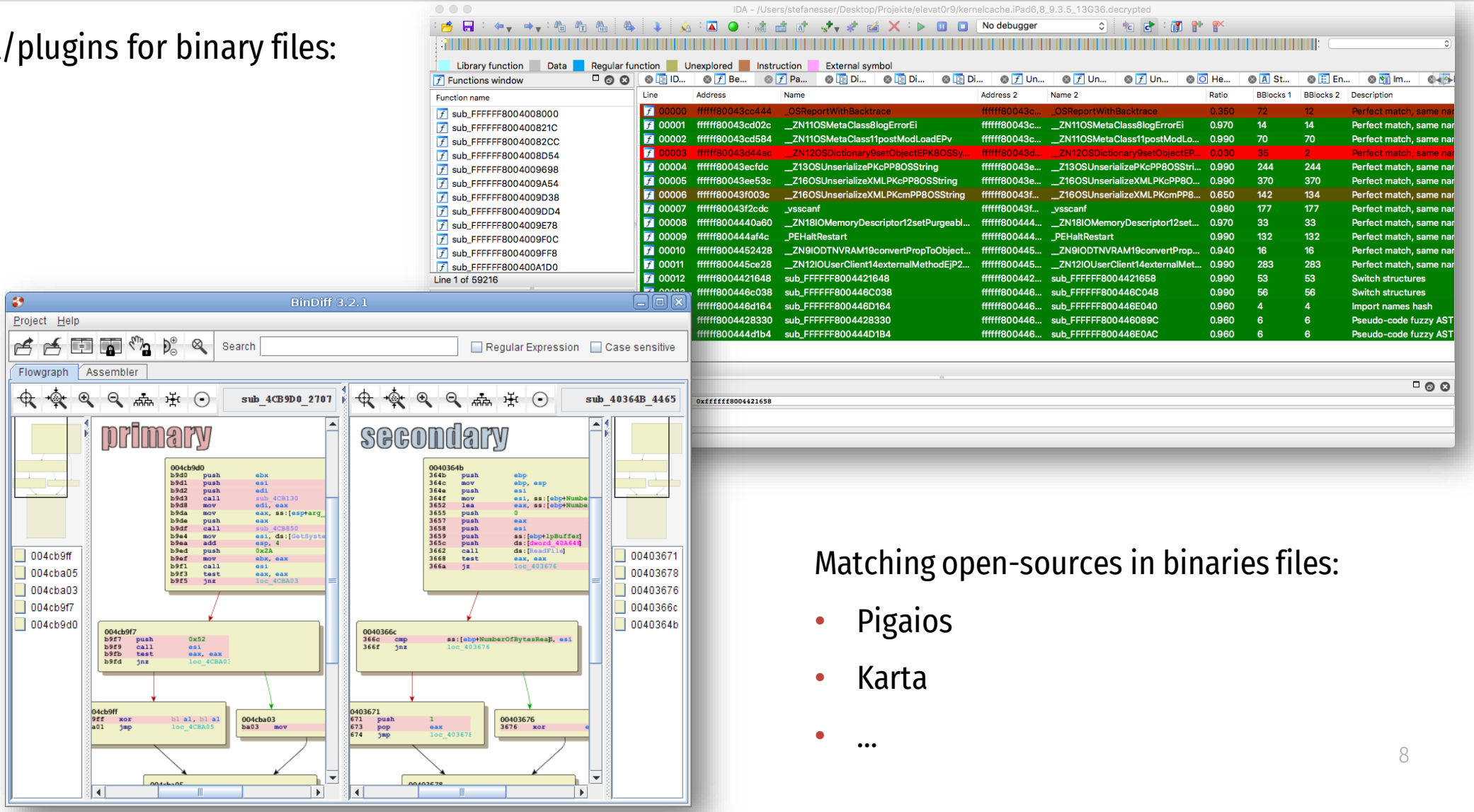
Source: https://twitter.com/halvarflake/status/1162644911931478016

Digital Security

Tools

Comparison tool/plugins for binary files:

- Diaphora
- BinDiff
- DarunGrim
- YaDiff
- rizzo
- Realyze
- Turbodiff
- patchdiff2
- rematch
- …



Matching open-sources in binaries files:

- Pigaios
- Karta
- …

Digital Security

Apple software and binary diffing

Apple-specific or Hello Objective-C !

Objective-C for newbies

A bit of theory

[NSMutable set];

objc_msgSend(NSMutable, @selector(set));

Object name

Method name

```
NSString *outputPath;
NSMutableSet *hiddenSections = [NSMutableSet set];

int ch;
BOOL errorFlag = NO;

struct option longopts[] = {…

if (argc == 1) {…

CDClassDump *classDump = [[CDClassDump alloc] init];
```

```
                        ; char **
call    _objc_autoreleasePoolPush
mov     rdi, cs:classRef_NSMutableSet ; void *
mov     rsi, cs:selRef_set ; char *
call    cs:_objc_msgSend_ptr
= rax                   ; class_dump_Prefix_pch::NSMutableSet *
mov     rdi, hiddenSections
call    _objc_retainAutoreleasedReturnValue
mov     [rbp+var_70], rax
lea     rsi, off_1000594A0 ; void *
lea     rdi, [rbp+longopts] ; void *
mov     edx, 260h       ; size_t
call    _memcpy
cmp     ebx, 1
jz      loc_100013A5F
mov     rdi, cs:classRef_CDClassDump ; void *
mov     rsi, cs:selRef_alloc ; char *
mov     r12, cs:_objc_msgSend_ptr
call    r12 ; _objc_msgSend
mov     rsi, cs:selRef_init ; char *
mov     rdi, rax        ; void *
call    r12 ; _objc_msgSend ; -[CDClassDump init]
mov     [rbp+classDump], rax
mov     rax, cs:selRef_stringWithUTF8String_
```

RE Objective-C binaries:

- "Reversing Objective-C Binaries With the REobjc Module for IDA Pro"
- "Automating Objective-C Code Analysis with Emulation"

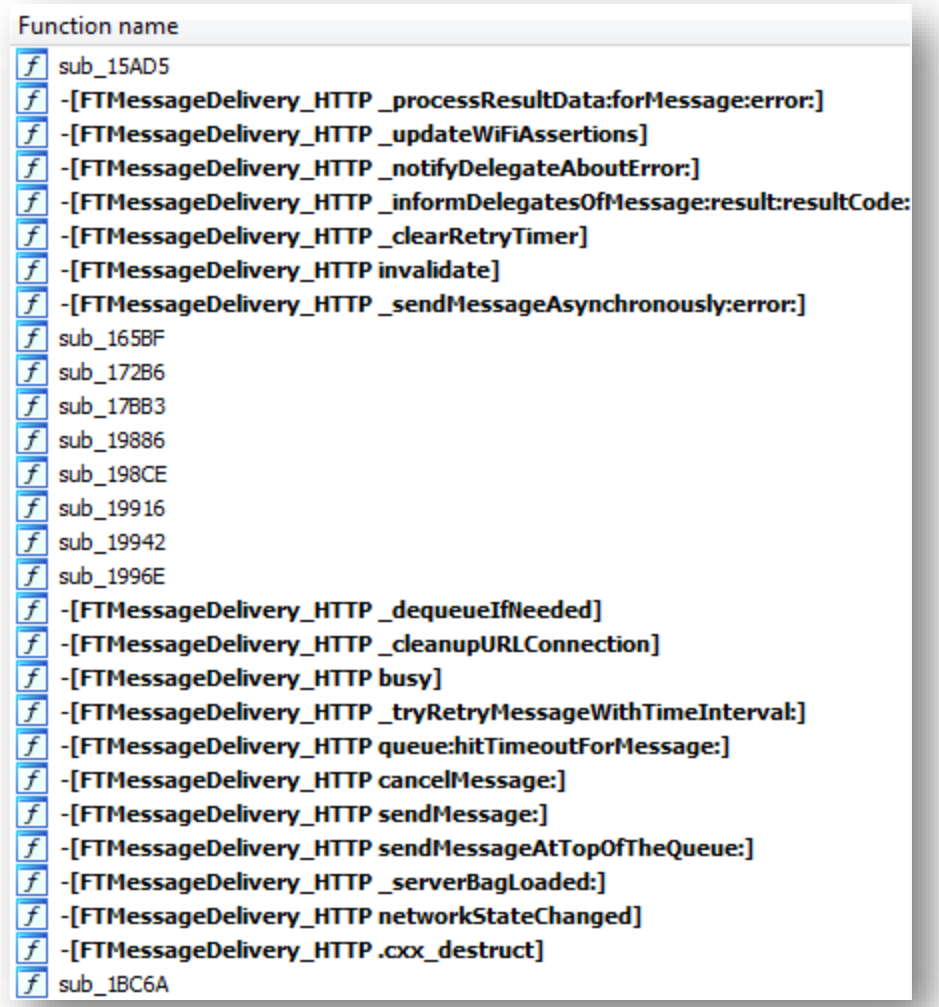
Digital Security

Objective-C vs. C\C++

Strings!

C/C++ methods

Objective-C methods



Digital Security

CVE-2019-8605

Fun =)

iOS 12.3

Step 1

Дата выпуска: 13 мая 2019 г.

Ядро

Целевые продукты: iPhone 5s и более поздние модели, iPad Air и более поздние модели, iPod touch (6-го поколения)

Воздействие. Вредоносная программа может выполнять произвольный код с системными привилегиями.

Описание. Проблема с использованием памяти после ее освобождения устранена путем улучшенного управления памятью.

CVE-2019-8605: Нед Уильямсон (Ned Williamson), сотрудничающий с подразделением Google Project Zero

iOS 12.4

Released July 22, 2019

Step 2

Step 3

iOS 12.4.1

Released August 26, 2019

Kernel

Available for: iPhone 5s and later, iPad Air and later, and iPod touch 6th generation

Impact: A malicious application may be able to execute arbitrary code with system privileges

Description: A use after free issue was addressed with improved memory management.

CVE-2019-8605: Ned Williamson working with Google Project Zero

CVE-2019-7287 and CVE-2019-7286

In-the-wild iOS Exploit Chain

- CVE-2019-7287 and CVE-2019-7286
- HITB 2019 Stefan Esser "Recreating an iOS 0-day jailbreak out of Apple's security patches"
- Ian Beer, Project Zero "A very deep dive into iOS Exploit chains found in the wild"

Foundation

Available for: iPhone 5s and later, iPad Air and later, and iPod touch 6th generation

Impact: An application may be able to gain elevated privileges

Description: A memory corruption issue was addressed with improved input validation.

CVE-2019-7286: an anonymous researcher, Clement Lecigne of Google Threat Analysis Group, Ian Beer of Google Project Zero, and Samuel Groß of Google Project Zero

IOKit

Available for: iPhone 5s and later, iPad Air and later, and iPod touch 6th generation

Impact: An application may be able to execute arbitrary code with kernel privileges

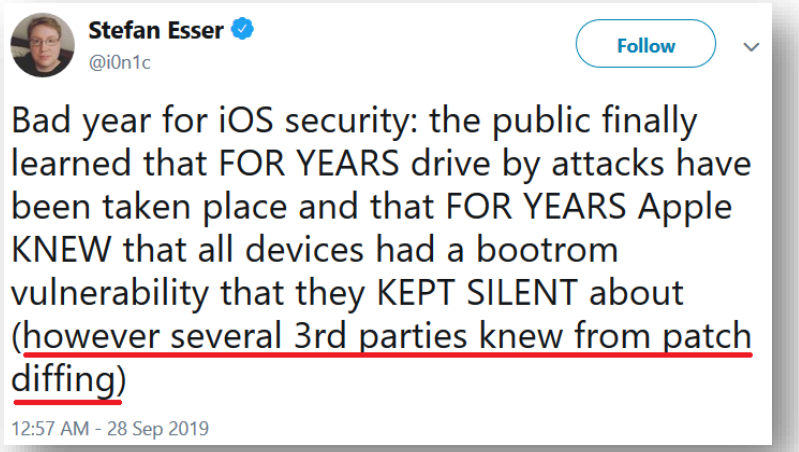
Description: A memory corruption issue was addressed with improved input validation.

CVE-2019-7287: an anonymous researcher, Clement Lecigne of Google Threat Analysis Group, Ian Beer of Google Project Zero, and Samuel Groß of Google Project Zero

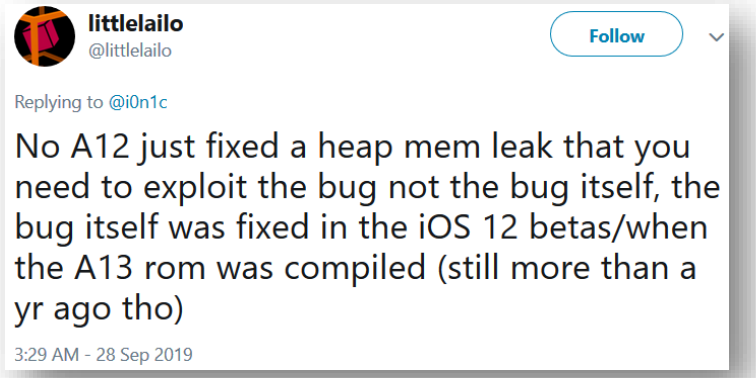
Boot ROM vulnerability's

Jailbreak happiness

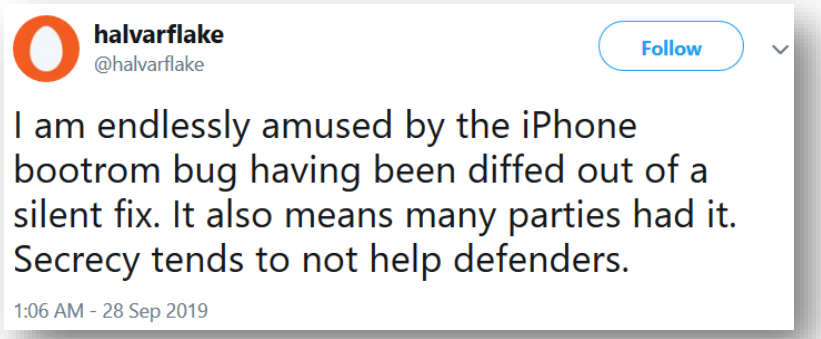
- checkm8 exploit
- Affecting the iPhone 4S (A5 chip) through the iPhone X (A11 chip)
  - iWatch, iPads, …
  - Fixed in A12 and A13 CPUs



https://twitter.com/i0n1c/status/1177854926191579138



https://twitter.com/littlelailo/status/1177893159428329473



https://twitter.com/halvarflake/status/1177857185725984768

Digital Security

CVE-2019-8574

Our case/story

Digital Security

CVE-2019-8574

First information from Apple

About the security content of iOS 12.3

This document describes the security content of iOS 12.3.

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 updates page.

Apple security documents reference vulnerabilities by CVE-ID when possible.

For more information about security, see the Apple Product Security page.

iOS 12.3

Released May 13, 2019

sysdiagnose

Available for: iPhone 5s and later, iPad Air and later, and iPod touch 6th generation

Impact: An application may be able to execute arbitrary code with system privileges

Description: A memory corruption issue was addressed with improved memory handling.

CVE-2019-8574: Dayton Pidhirney (@_watbulb) of Seekintoo (@seekintoo)

About the security content of macOS Mojave 10.14.5, Security Update 2019-003 High Sierra, Security Update 2019-003 Sierra

This document describes the security content of macOS Mojave 10.14.5, Security Update 2019-003 High Sierra, Security Update 2019-003 Sierra.

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 updates page.

Apple security documents reference vulnerabilities by CVE-ID when possible.

For more information about security, see the Apple Product Security page.

macOS Mojave 10.14.5, Security Update 2019-003 High Sierra, Security Update 2019-003 Sierra

Released May 13, 2019

sysdiagnose

Available for: macOS Sierra 10.12.6, macOS High Sierra 10.13.6, macOS Mojave 10.14.4

Impact: An application may be able to execute arbitrary code with system privileges

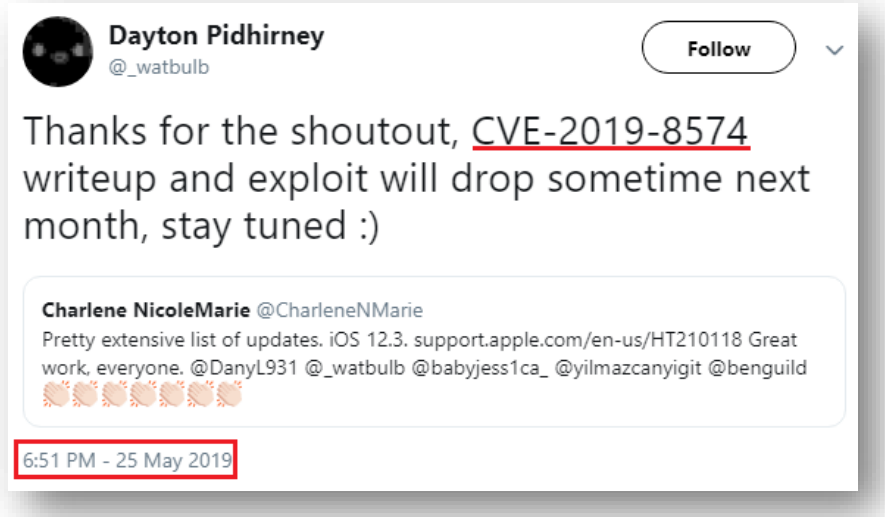
Description: A memory corruption issue was addressed with improved memory handling.

CVE-2019-8574: Dayton Pidhirney (@_watbulb) of Seekintoo (@seekintoo)

Digital Security

Disclosure timeline

- As of today, 19 October 2019
- No details from the author yet...



`sysdiagnose` helps collect data and send bug reports to Apple.

Если вы обнаружили программу, работающую с ошибками, и хотите сообщить разработчикам программы о найденной ошибке, то лучше включить в ваш отчет и диагностический снимок системы. Это упростит разработчикам поиск и устранение ошибок. Сделать это можно следующими способами.

Первый способ

Воспользуйтесь комбинацией клавиш *Shift+Ctrl+Alt+Cmd+.* и такой отчет будет создан.

Второй способ

Откройте *Терминал* и выполните команду:

```
sudo sysdiagnose
```

Введите ваш пароль. Программа предупредит, что в отчет будут включены некоторые приватные данные, такие как серийный номер компьютера, его имя и имя пользователя. Если вы согласны с этим, то просто нажмите *Enter* (картинки кликабельны):

Apple Tech Support · macOS (operating system)

What is a sysdiagnose?

1 Answer

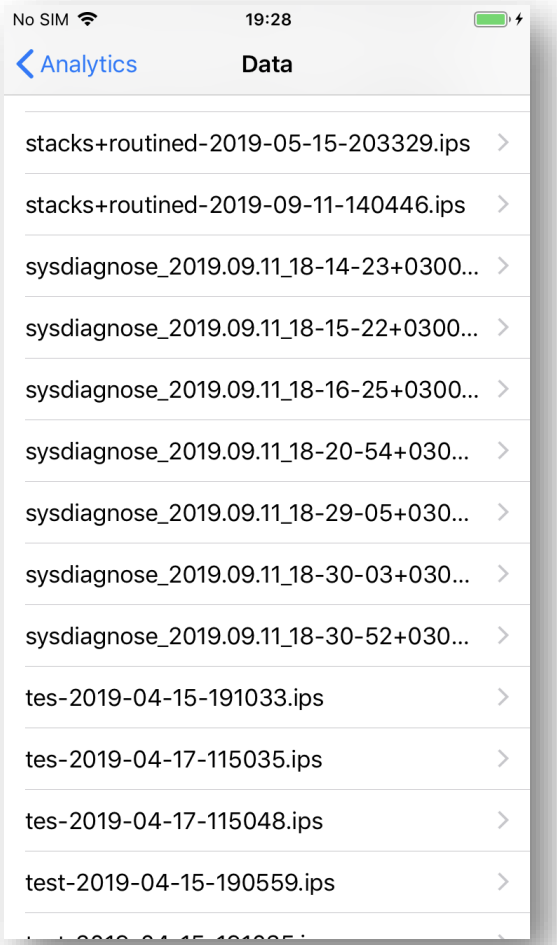
Aaron Paul, I can has Mac?
Answered Jan 1, 2018 · Author has **151** answers and **280.7k** answer views

A sysdiagnose is a report of the state of a device. It includes logging from different services and reports of the state of systems. What exactly is contained in a sysdiagnose will vary depending on what type of device and which version of the OS. They are used by senior support personnel and engineers to determine root cause of issues occurring. For example if a Mac is experiencing unexpected restarts, the logs may reveal the previous shutdown reasons which could tell you if the issue was related to hardware or software.

1k views · View 1 Upvoter

sysdiagnose on iPhone

- Settings > Privacy > Analytics > Analytics Data



Digital Security

Preparing for Apple diffing

- Get update files
  - MacOS
    - Analysis of Software Update Catalog files (.sucatalog)
    - /Library/Updates
    - …
  - iOS
    - ipsw.me
    - …

Preparing for Apple diffing

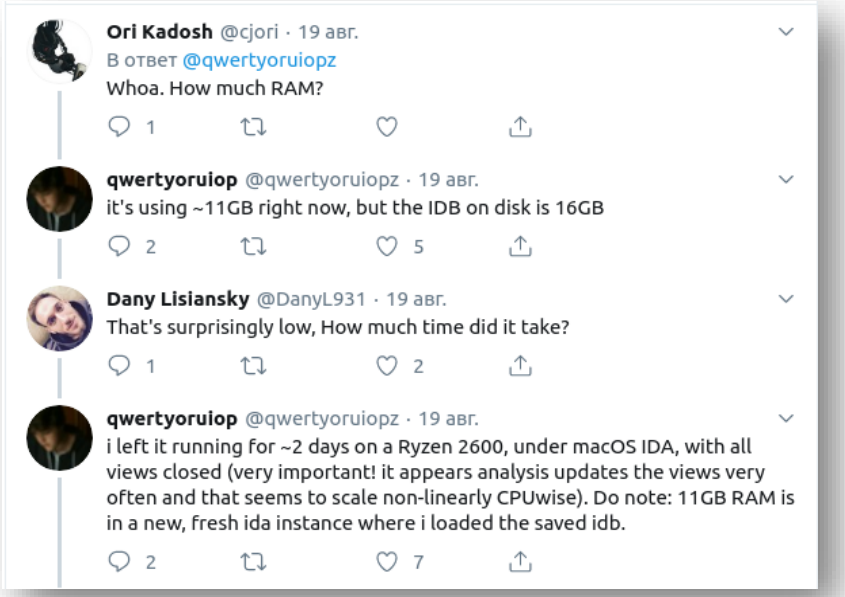
MacOS

- Extract necessary files from the update (Suspicious Package, ...)
  - Executable files
  - Libraries
  - Frameworks
  - /System/Library/Kernels/kernel
  - ...

Preparing for Apple diffing

iOS

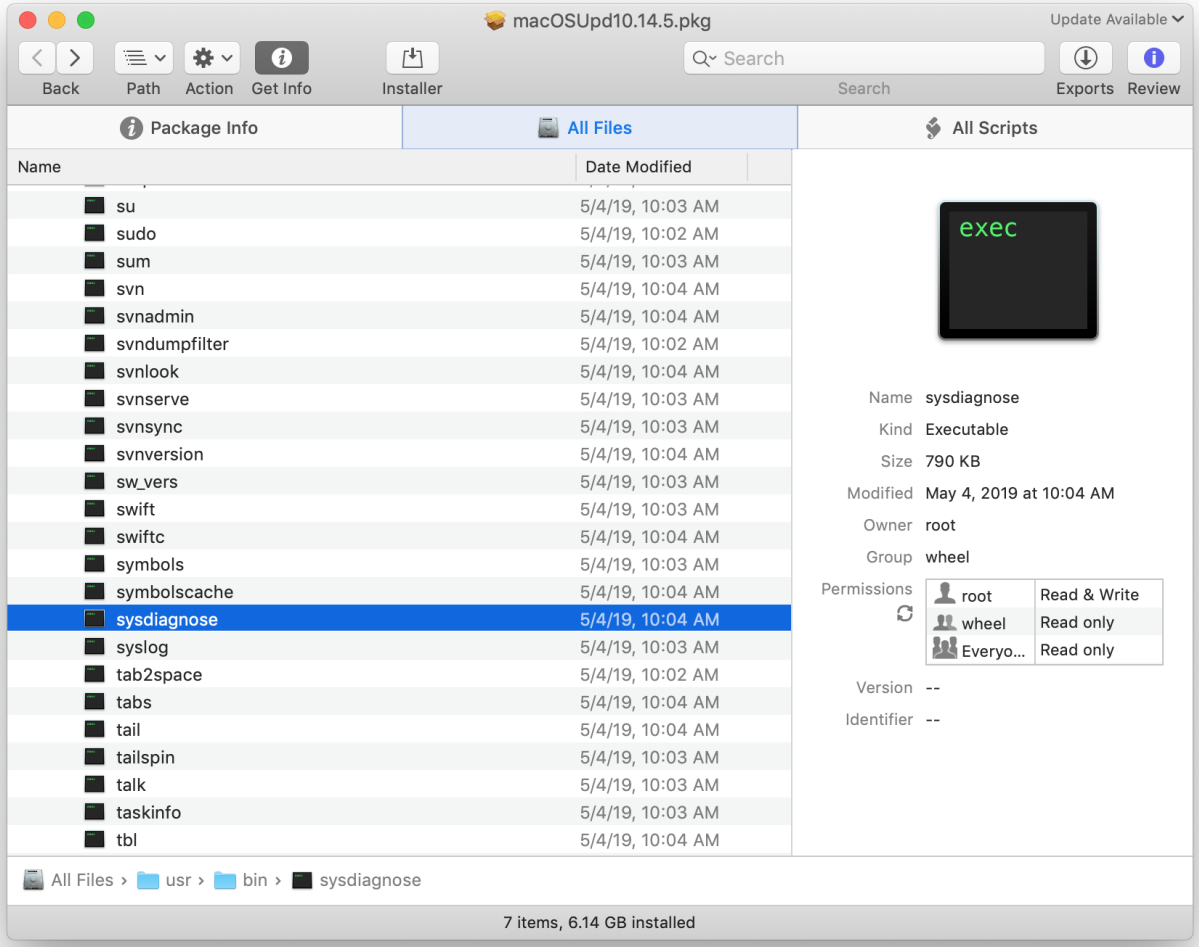
- Kernel
  - Kernel and modules are stored kernelcache
  - For analysis, we have to extract the Mach-O kernel file from kernelcache
    - img4tool
    - joker
    - jtool2
    - ...
- user land
  - Rootfs is in one of the .dmg files in ipsw
  - Most libraries and frameworks are distributed as dyld_shared_cache file
    - > 1gb :(
    - Hard to analyze
- ipsw makes it easier to get ipsw-files and extract files from them



Digital Security

Extracting sysdiagnose from updates

- Update files acquired from /Library/Updates
- `sysdiagnose` was extracted with the Suspicious package utility



Diffing

Let's go!

ARM

Similarity	Confidence	Address	Primary Name
0.35	0.73	0000000100...	isFactory block invoke
0.79	0.97	0000000100...	factoryDisable
0.92	0.98	0000000100...	-[SDTask start]
0.95	0.95	0000000100...	-[SystemDiagnostic iOSgetLogCopyingContainers]
1.00	0.99	0000000100...	-[SDTask endStatus]
1.00	0.99	0000000100...	-[SystemDiagnostic setTarballPath:]

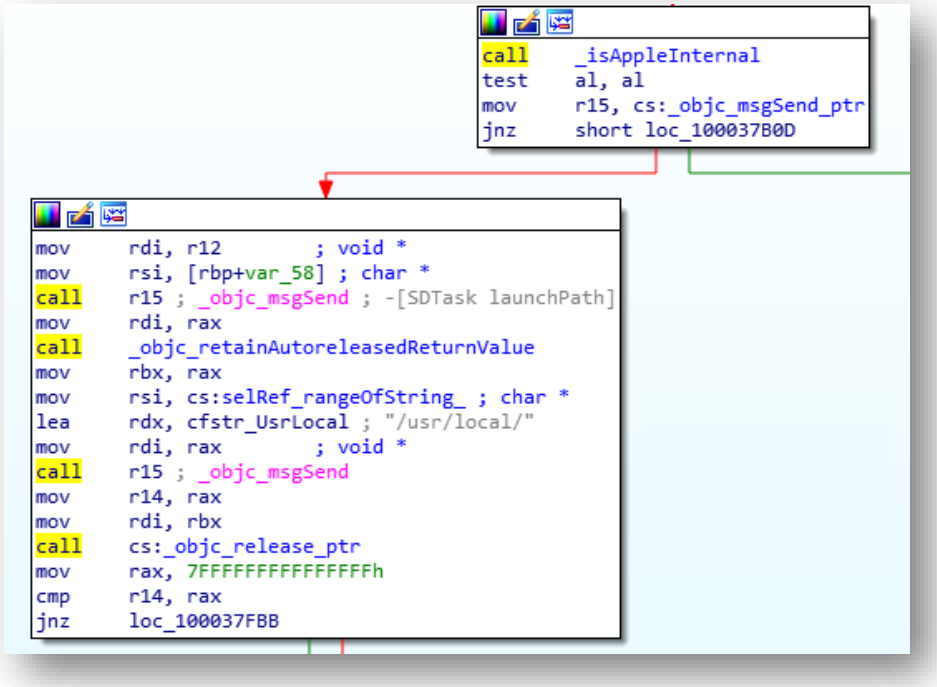
X86_64

Similarity	Confidence	Address	Primary Name
0.93	0.99	0000000100...	-[SDTask start]
1.00	0.99	0000000100...	-[SystemDiagnostic setTmpO...
1.00	0.99	0000000100...	-[SystemDiagnostic setShou...
1.00	0.99	0000000100...	-[SystemDiagnosticBT remo...
1.00	0.99	0000000100...	NSFilePosixPermissions

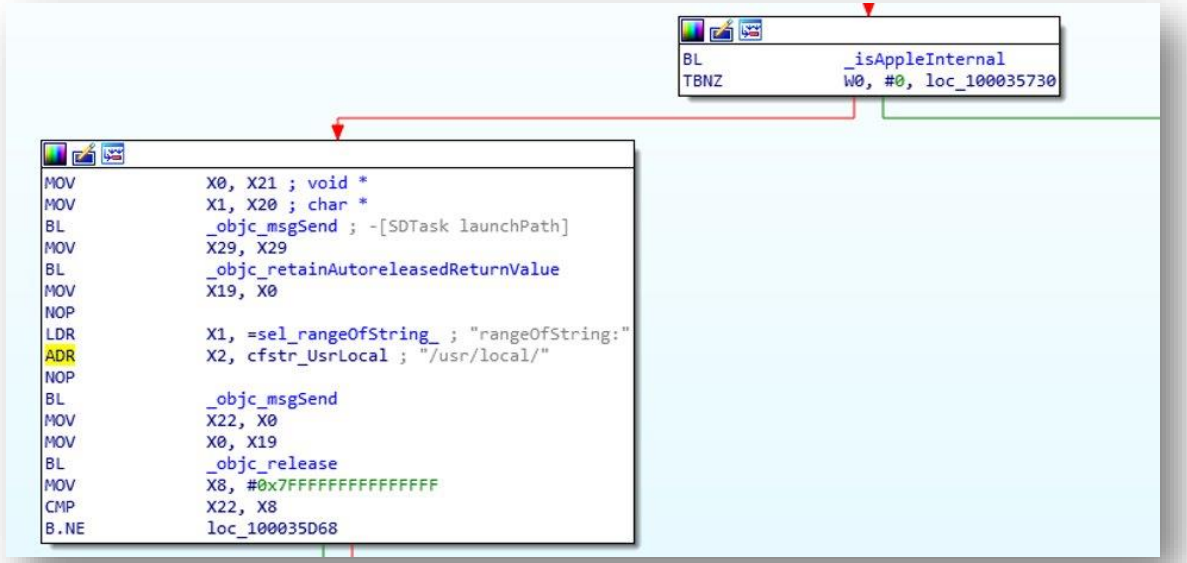
- `__isFactory_block_invoke` – insignificant changes
- `_factoryDisable` - insignificant changes
- `-[SystemDiagnostic _iOSgetLogCopyingContainers]` – added call `-[SystemDiagnostic _copyAstroLogsContainer]` for an additional source of information for system diagnostics
- `-[SDTask _start]` – additional check!!!

Digital Security

-[SDTask _start] patch



X86_64



ARM

Digital Security

Creating diagnostic tasks

```
v17 = +[SDTask taskWithCommand:arguments:outputFile:](
        &OBJC_CLASS___SDTask,
        "taskWithCommand:arguments:outputFile:",
        CFSTR("/System/Library/PrivateFrameworks/ApplePushService.framework/apsctl"),
        v16,
        CFSTR("apsd-status.txt"));
v62 = (void *)objc_retainAutoreleasedReturnValue(v17);
tasks[2] = v62;
v90 = CFSTR("--get");
v18 = objc_msgSend(&OBJC_CLASS___NSArray, "arrayWithObjects:count:", &v90, 1LL);
v19 = objc_retainAutoreleasedReturnValue(v18);
v63 = v19;
v20 = +[SDTask taskWithCommand:arguments:outputFile:](
        &OBJC_CLASS___SDTask,
        "taskWithCommand:arguments:outputFile:",
        CFSTR("/usr/local/bin/pmtool"),
        v19,
        CFSTR("pmtool.txt"));
v64 = (void *)objc_retainAutoreleasedReturnValue(v20);
tasks[3] = v64;
v89 = CFSTR("/Library/SystemMigration/History");
v21 = objc_msgSend(&OBJC_CLASS___NSArray, "arrayWithObjects:count:", &v89, 1LL);
v22 = objc_retainAutoreleasedReturnValue(v21);
v65 = v22;
v23 = +[SDTask taskWithCommand:arguments:outputFile:](
        &OBJC_CLASS___SDTask,
        "taskWithCommand:arguments:outputFile:",
        CFSTR("/usr/bin/find"),
        v22,
        CFSTR("find-system-migration-history.txt"));
```

Digital Security

Patch description

- Before
  - Tasks are run regardless of the path to an executable file
- After
  - If it's not an `isAppleInternal` device and the path to the executable file contains `/usr/local/`, the task won't be executed
  - If the check for `/usr/local/` is not passed, the following message is logged:
    - `Error: Blocked launching %@ on t his build.`

The list of task executable files from `/usr/local/` for Mac:

- `/usr/local/bin/airplayutil`
- `/usr/local/bin/amstool`
- `/usr/local/bin/apsclient`
- `/usr/local/bin/audioDeviceDump`
- `/usr/local/bin/cdcontexttool`
- `/usr/local/bin/cddebug`
- `/usr/local/bin/cdknowledgetool`
- `/usr/local/bin/dastool`
- `/usr/local/bin/idstool`
- `/usr/local/bin/imtool`
- `/usr/local/bin/keystorectl`
- `/usr/local/bin/pmtool`
- `/usr/local/bin/xcpm`
- `/usr/local/efi/bin/efi-perf`

Digital Security

Attack: ~~DLL~~ Bin Hijacking

1. We can create any of the files listed above without `sudo`.

- Works only for systems with `brew` package manager
  - `brew` allows a user to write in these directories

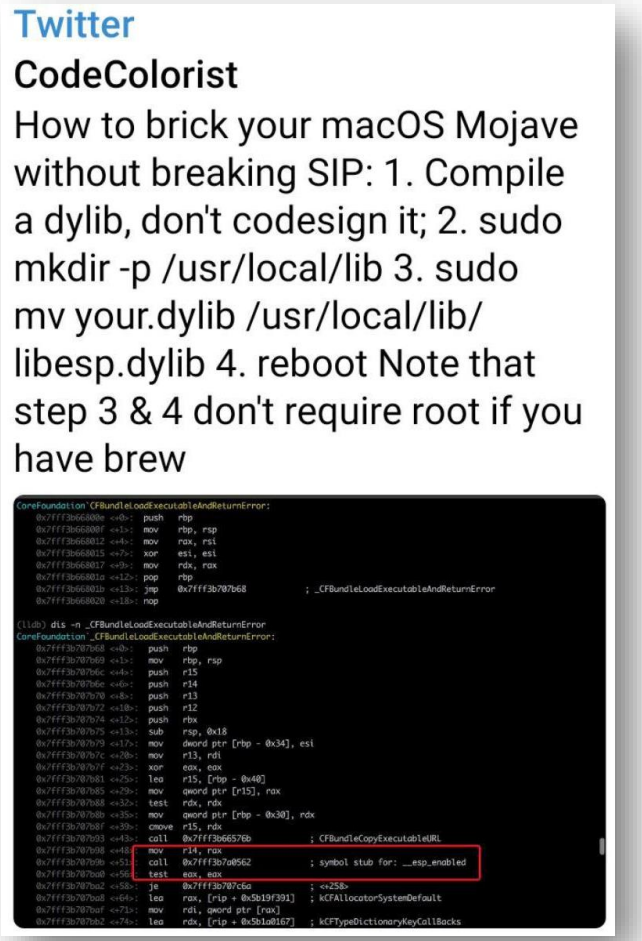
2. When calling `sysdiagnose`, a user file will be executed by `root`, and its output will be contained in the `sysdiagnose` archive, in a text file of the same name.

- To run `sysdiagnose` from terminal, we need `root` (`sudo sysdiagnose`)
  - `sysdiagnose` can also be run with a shortcut `Control-Option-Command-Shift-Period` without password and even from the lock screen
- This vulnerability can also be used by malicious software on Mac for escalating privileges

brew

Yet another problem

- A trick using the same `brew` feature
  - SIP - System Integrity Protection
  - https://twitter.com/CodeColorist/status/1151033366960984064 *



*the tweet was deleted shortly after its publication

Digital Security

Conclusion

Conclusion

- Binary diffing is a great approach for development
- Patch diffing allows you to discover 1-day and even 0-day vulnerabilities
  - Double check the patches for your vulnerabilities issued by the vendor
  - After updates, double check your releases for old vulnerabilities

Digital Security

Thanks for your attention!


+7 (495) 223-07-86


sales@dsec.ru


DSecRU


company/dsec/

Digital Security

Thanks for your attention!