SECURE MOBILE NETWORKING

Venturing beyond iOS: Exploring Co-Processor Firmware

#OFTW v4.0 – Lukas Arnold

TECHNISCHE UNIVERSITÄT DARMSTADT

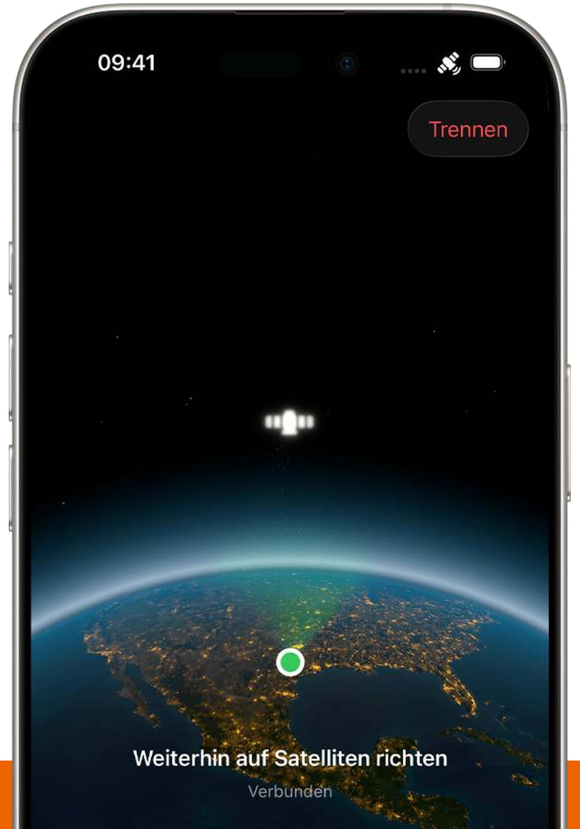
Features Enabled by Coprocessors

Connectivity

AirDrop

Always On

Satellite



Coprocessors from a Security Standpoint

 Platform Security Guide

System Coprocessor Integrity Protection

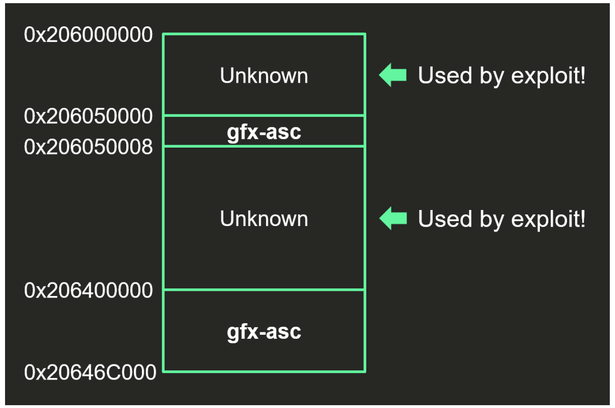
Coprocessor firmware handles many critical system tasks — for example, the Secure Enclave, the image sensor processor and the motion coprocessor. Therefore its security is a key part of the security of the overall system. To prevent modification of coprocessor firmware, Apple uses a mechanism called *System Coprocessor Integrity Protection (SCIP)*.

🤔

What if coprocessor firmware contains bugs?

⚠️

Might impact security of entire device



Operation Triangulation: The last (hardware) mystery

RESEARCH 27 DEC 2023 18 minute read

While analyzing the exploit used in the Operation Triangulation attack, I discovered that most of the MMIOs used by the attackers to bypass the hardware-based kernel memory protection do not belong to any MMIO ranges defined in the device tree. The exploit targets Apple A12–A16 Bionic SoCs, targeting unknown MMIO blocks of registers that are located at the following addresses: 0x206040000, 0x206140000, and 0x206150000.

Why Research Wireless Coprocessors?

Complex legacy protocol stacks

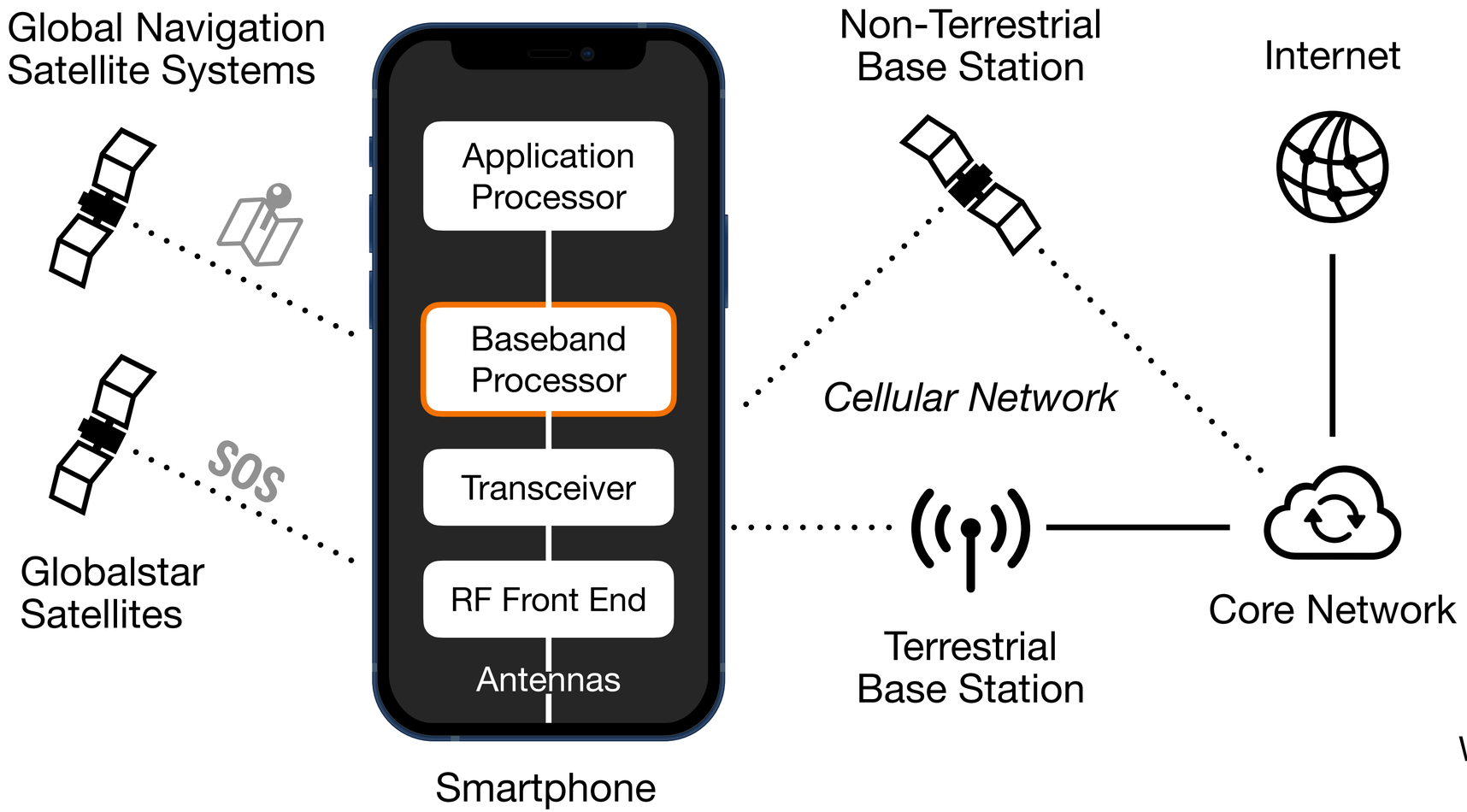
Apple’s approach to vertical integration

Send custom packet over the air

What is a Baseband?
A wireless coprocessor!

Smartphone Basebands

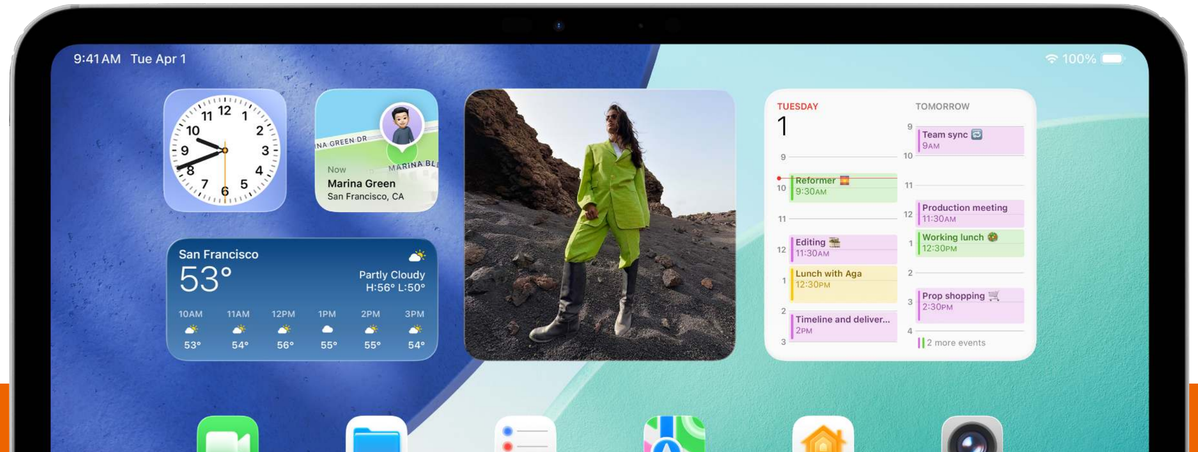
Wireless Interfaces



Smartphone Basebands

In Apple Devices

MEDIATEK Qualcomm



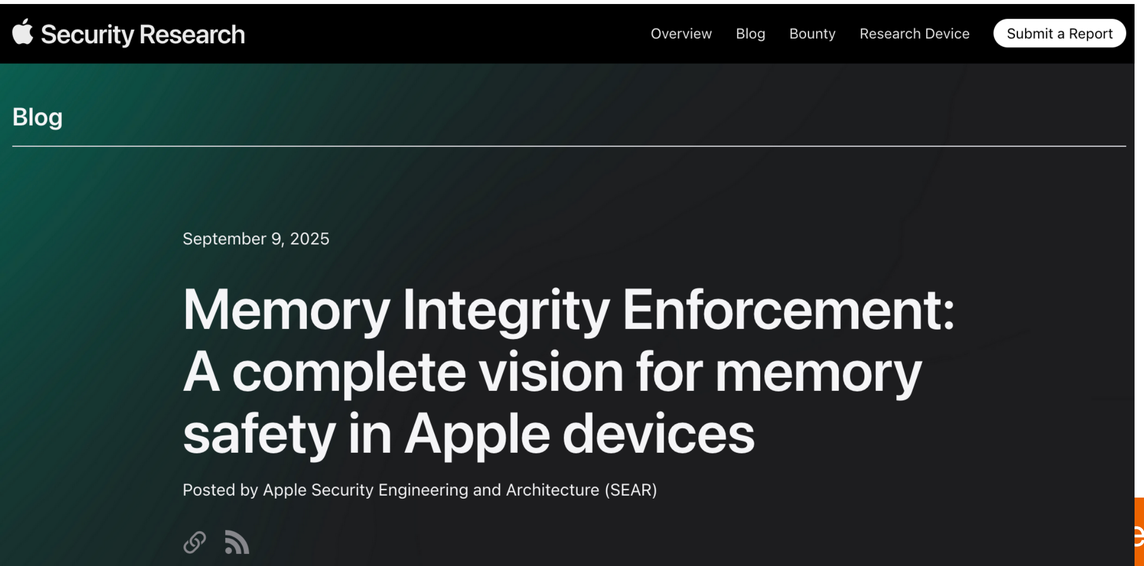
Why Exploit the Baseband?

Weak security
(compared with AP)

Critical data
passes through

Unnoticed
persistence

Jump to
AP possible?



Cheap exploits, but
expensive deployment

PAPER (YEAR)	METHODOLOGY	VENDORS	PROTO	OS
2011				
SMS of Death [185]	OTA Fuzzing	(Feature Phones)	GSM	○
2012				
Baseband Attacks [236]	Manual Static Analysis	Intel, Qualcomm	GSM	○
2016				
Breaking Band [114]	Manual Static Analysis	Samsung	LTE	●
2017				
Path of Least Resistance [182]	RIL Fuzzing	MediaTek	GSM	○
2018				
Exploiting Basebands [120]	Manual Static Analysis	Huawei	CDMA	○
2019				
LTEFuzz [152]	OTA Fuzzing	Qualcomm, HiSilicon	LTE	○
2020				
BaseSAFE [177]	Emu-based Fuzzing	MediaTek	LTE	●
Exploring MediaTek [119]	Emu-based Fuzzing	MediaTek	LTE	○
2021				
BaseSpec [150]	Automated Static Analysis	MediaTek, Samsung	GSM – NR	●
Intel Fuzzing [219]	Emu-based Fuzzing	Intel	RIL	○
JMPscare [176]	Emu-based Fuzzing	MediaTek	LTE	●
2022				
DoLTEst [194]	Spec-based OTA Testing	Intel, HiSilicon, MediaTek, Samsung, Qualcomm	LTE	●
FirmWire [134]	Emu-based Fuzzing	MediaTek, Samsung	GSM & LTE	●
2023				
BaseComp [149]	Symbolic Analysis	MediaTek, Samsung	LTE & NR	●
Internet-to-BB RCE [251]	Manual Static Analysis	Samsung	VoLTE	●
Securing Pixel 6 [254]	Emu-based Fuzzing	Samsung	GSM – NR	○
2024				
BaseMirror [168]	Automated Static Analysis	Samsung	RIL	●
BVFinder [173]	Automated Static Analysis	MediaTek, Samsung	GSM – NR	○
5GBaseChecker [232]	Differential OTA testing	HiSilicon, MediaTek, Samsung, Unisoc, Qcom	NR	●
SIMurai [172]	Hardware & Emulation-based Fuzzing	Intel, MediaTek, Samsung, Unisoc, Qualcomm	SIM APDU	●
2025				
5Ghoul [110]	OTA Fuzzing	MediaTek, Qualcomm	NR	●

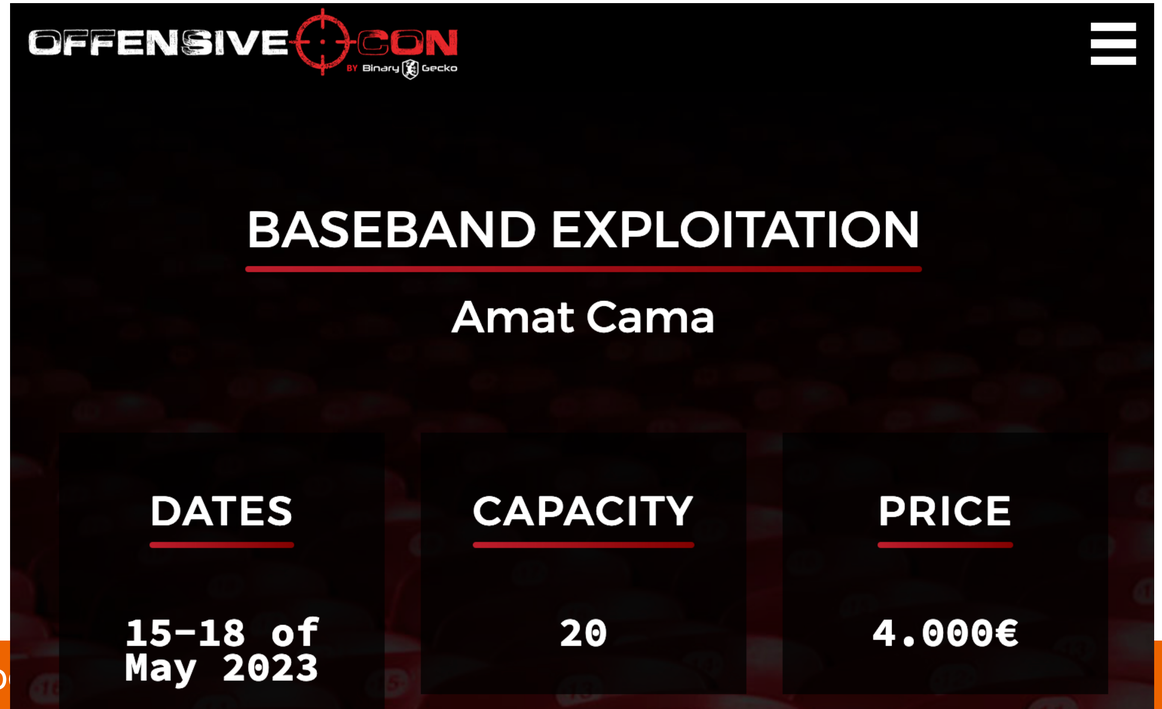
Multiple Internet to Baseband Remote Code Execution Vulnerabilities in Exynos Modems

2023-MAR-16 Tim Willis

Posted by Tim Willis, Project Zero

In late 2022 and early 2023, Project Zero reported eighteen 0-day vulnerabilities in Exynos Modems produced by Samsung Semiconductor. The four most severe of these eighteen vulnerabilities (CVE-2023-24033, CVE-2023-26496, CVE-2023-26497 and CVE-2023-26498) allowed for Internet-to-baseband remote code execution. Tests conducted by Project Zero confirm that those four vulnerabilities allow an attacker to remotely compromise a phone at the baseband level with no user interaction, and require only that the attacker know the victim's phone number. With limited additional research and development, we believe that skilled attackers would be able to quickly create an operational exploit to compromise affected devices silently and remotely.

The fourteen other related vulnerabilities (CVE-2023-26072, CVE-2023-26073, CVE-2023-26074, CVE-2023-26075, CVE-2023-26076 and nine other vulnerabilities that are yet to be assigned CVE-IDs) were not as severe, as they require either a malicious mobile network operator or an attacker with local access to the device.



Related Work

Exploitation Mitigations

Google 2023

„Regardless of whether it is remote code execution within the WiFi SoC or within the cellular baseband, a common and resonating theme has been the **consistent lack of exploit mitigations** in firmware.“

TASZK WE FIND NEEDLES 2023

„[…] competing chipset manufacturers don’t always publicize their own baseband exploit mitigations. […] It’s clear that vendors have **definitely started working on baseband exploit mitigations**."

No systematic review of mitigations 😕

Diving into the C1
Apple’s own baseband

Analysis Perspectives

To Decipher Security Model of Apple's C1 Baseband

Hardware iOS Firmware

+

Security Model & Mitigations

Analysis: Hardware 🛠️

Official Specifications

PRESS RELEASE
February 19, 2025

Apple debuts iPhone 16e:

Expanding the benefits of Apple silicon, C1 is the first modem designed by Apple and the most power-efficient modem ever on an iPhone, delivering fast and reliable 5G cellular connectivity. Apple silicon — including C1 — the all-new internal design, and the advanced power management of iOS 18 all contribute to extraordinary battery life.

PRESS RELEASE
September 9, 2025

Introducing iPhone Air,

generation of wireless technologies, N1 improves the overall performance and reliability of features like Personal Hotspot and AirDrop. iPhone Air also features C1X, a new cellular modem designed by Apple. C1X is up to 2x faster than C1, and for the same cellular technologies, it is even faster than the modem in iPhone 16 Pro, while using 30 percent less energy overall. This makes C1X the most power-efficient modem in an iPhone.

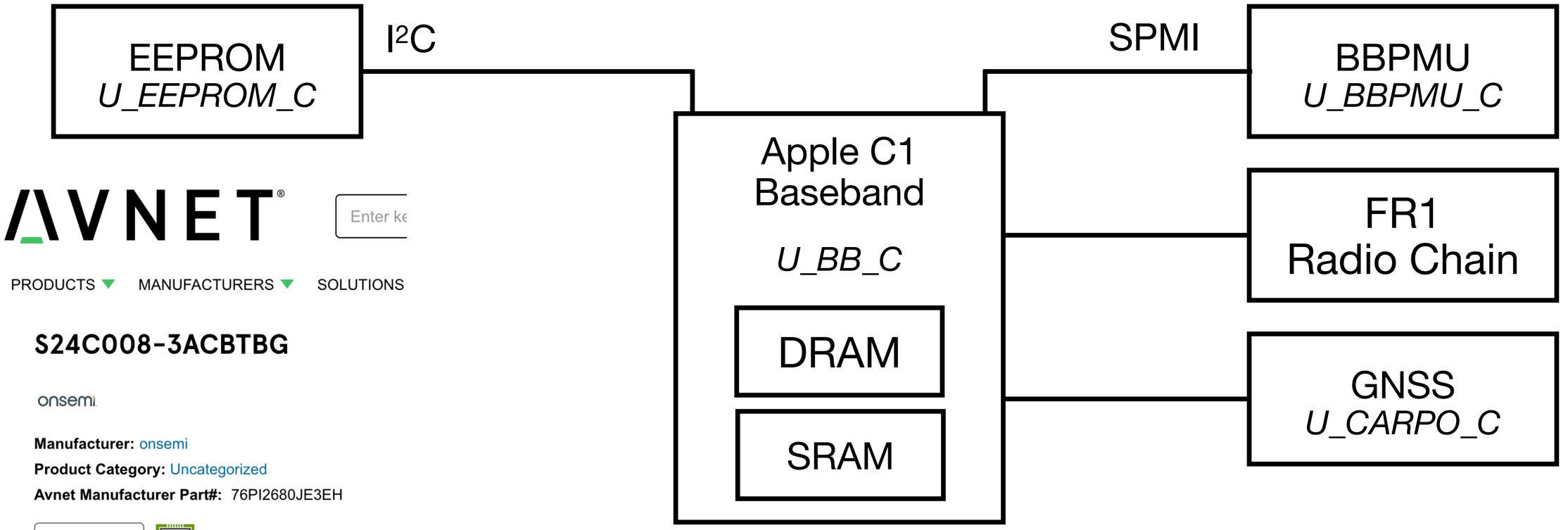
iPhone 16e: Technical Specifications

Cellular and Wireless	**Model A3212***
	5G NR (Bands n1, n2, n3, n5, n7, n8, n12, n14, n20, n25, n26, n28, n29, n30, n38, n40, n41, n48, n53, n66, n70, n71, n75, n76, n77, n78, n79)
	FDD-LTE (Bands 1, 2, 3, 4, 5, 7, 8, 12, 13, 14, 17, 18, 19, 20, 25, 26, 28, 29, 30, 32, 66, 71)
	TD-LTE (Bands 34, 38, 39, 40, 41, 42, 48, 53)
	UMTS/HSPA+ (850, 900, 1700/2100, 1900, 2100 MHz)
	GSM/EDGE (850, 900, 1800, 1900 MHz)
	Apple C1 cellular modem
	5G (sub-6 GHz) with 4x4 MIMO[10]
	Gigabit LTE with 4x4 MIMO[10]
Location	GPS, GLONASS, Galileo, QZSS, BeiDou, and NavIC

If we only had more details … 😘

Hardware Architecture

10-Jul-2025	85423100	PMIC TMQA52B0-APP-S2 TAIWAN SEMIC/WLCSP1790664-01TN0AC\\\\\ Integrated CircuitPMIC TMQA52B0-APP-S2 TAIWAN SEMIC/WLCSP1790664-01TN0AC\\\\\	Taiwan	India	700 NOS	$864.36



AVNET®

Enter ke

PRODUCTS ▼ MANUFACTURERS ▼ SOLUTIONS

S24C008-3ACBTBG

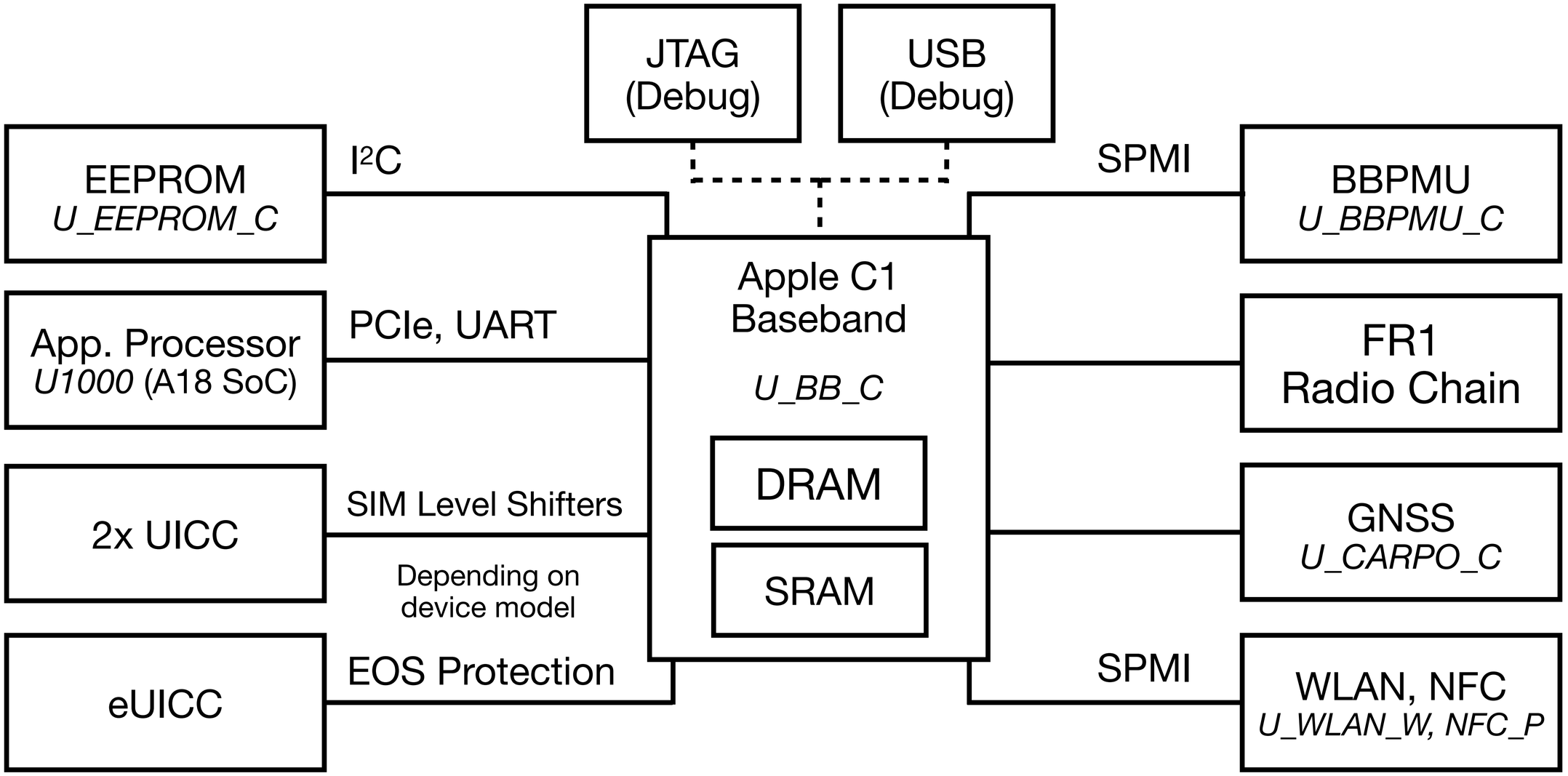
onsemi

Manufacturer: onsemi
Product Category: Uncategorized
Avnet Manufacturer Part#: 76PI2680JE3EH

Compare Request Free CAD Models

RoHS 6 Compliant Lifecycle

Hardware Architecture



Analysis: iOS 

Capabilities on Stock iOS

We **can**

Read system logs
Create sysdiagnoses

Install debug profiles

Read ARI packets

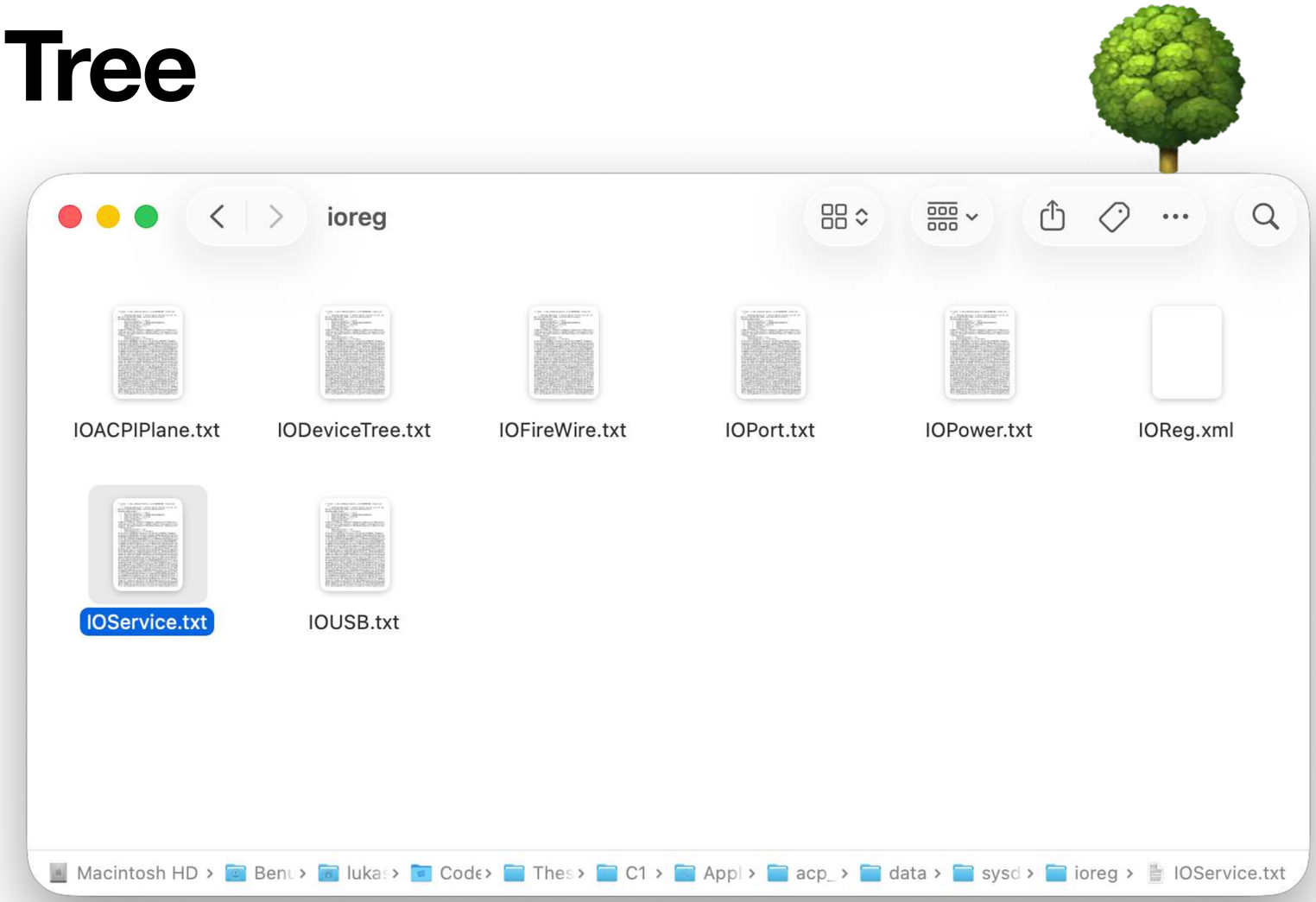
Send OTA packets

We **cannot**

Use system-level debuggers

Run arbitrary code
Change system files

Device Tree



Application processor & baseband exchange data over **PCIe**

Analysis: Firmware 💾

Extracting C1 Firmware

/Firmware/c4000v59/Release/patched/ftab.bin

```
$ ipsw fw c1 ftab.bin
```

iOS Image

ftab.bin

extracted

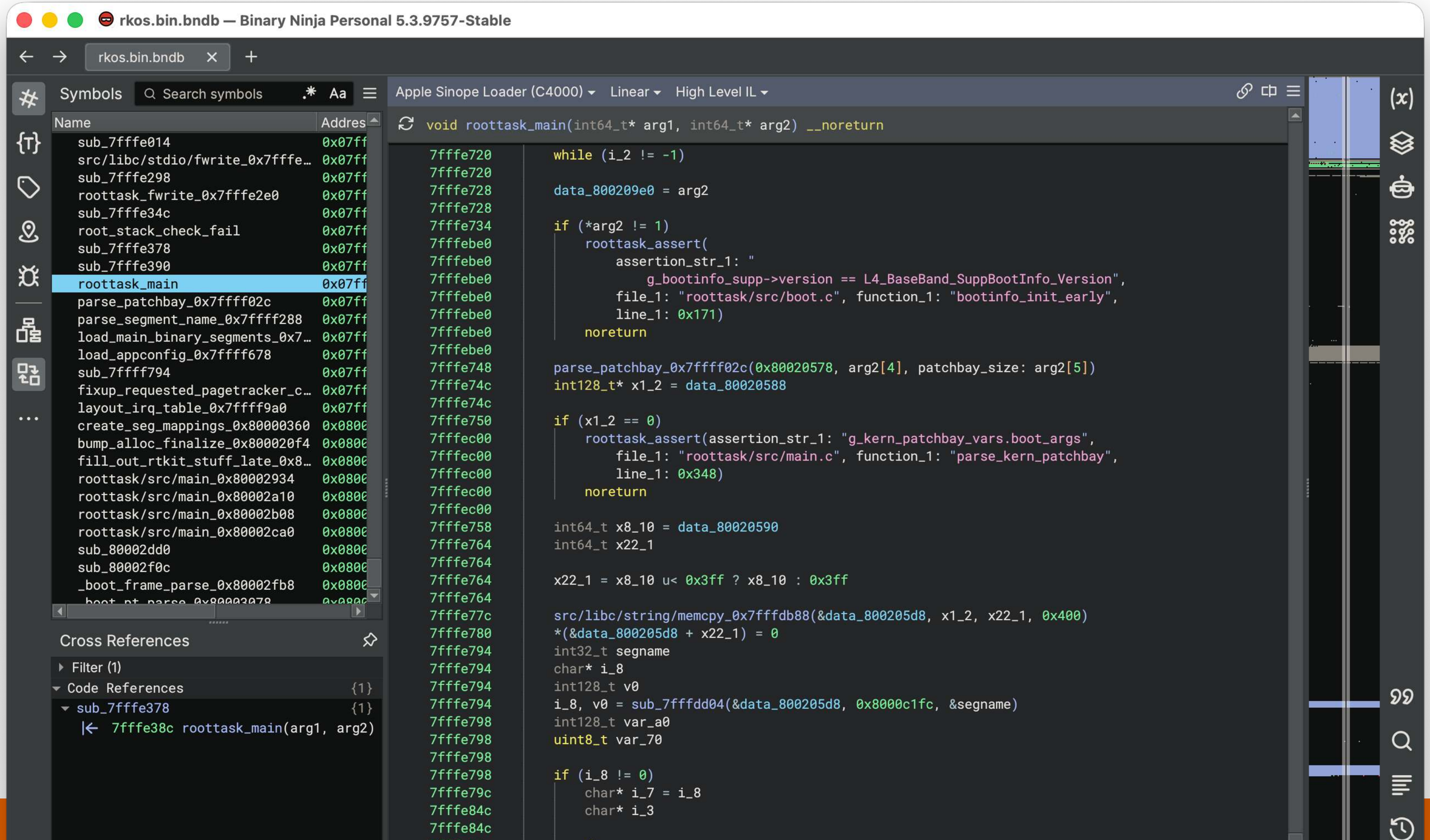
file.bin

file.lzfse.bin

Firmware Architecture

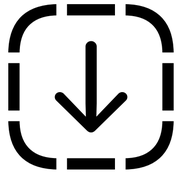
- **rkos:** Operating system & cellular protocol stack
- **cdpd:** Cellular data processing for downlink
- **cdpu:** Cellular data processing for uplink
- **l1cs:** Lower levels of cellular protocol stack
- **gns1:** Global navigation satellite system (ARCv2)

File	Type	Size
CRnn	Data (lzfse, fwsg)	5.7 MB
RPnn	Data (lzfse, fwsg)	3.9 MB
R1nn	Data (lzfse, fwsg)	12 MB
R203	Data (lzfse, fwsg)	4.1 MB
rcpi	Recipe File Format	5 KB
bver	Version String	81 B
ibdt	iBootData	272 B
l1c2	Zeros	6 KB
ARC2	"empty"	7 KB
CAR2	0xFEEDCAFE	487 KB
CAR3	0xFEEDCAFE	487 KB
GNS1	ARCv2	480 KB
apmu	ARMv7	7 KB
pmfw	ARMv7	10 KB
illb	ARMv8	555 KB
cdph	ARMv7 (fwsg)	917 KB
cdpd	ARMv8 (fwsg)	15 MB
cdpu	ARMv8 (fwsg)	18 MB
rkos	ARMv8 (fwsg)	75 MB
l1cs	ARMv8 (fwsg)	68 MB

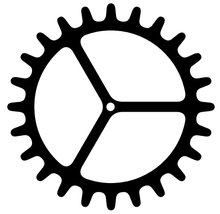

rkos.bin.bndb — Binary Ninja Personal 5.3.9757-Stable
rkos.bin.bndb
Symbols
Search symbols
Name
Addres
sub_7fffe014
src/libc/stdio/fwrite_0x7fffe…
sub_7fffe298
roottask_fwrite_0x7fffe2e0
sub_7fffe34c
root_stack_check_fail
sub_7fffe378
sub_7fffe390
roottask_main
parse_patchbay_0x7ffff02c
parse_segment_name_0x7ffff288
load_main_binary_segments_0x7…
load_appconfig_0x7ffff678
sub_7ffff794
fixup_requested_pagetracker_c…
layout_irq_table_0x7ffff9a0
create_seg_mappings_0x80000360
bump_alloc_finalize_0x800020f4
fill_out_rtkit_stuff_late_0x8…
roottask/src/main_0x80002934
roottask/src/main_0x80002a10
roottask/src/main_0x80002b08
roottask/src/main_0x80002ca0
sub_80002dd0
sub_80002f0c
_boot_frame_parse_0x80002fb8
Cross References
Filter (1)
Code References {1}
sub_7fffe378 {1}
7fffe38c roottask_main(arg1, arg2)
Apple Sinope Loader (C4000)
Linear
High Level IL
void roottask_main(int64_t* arg1, int64_t* arg2) __noreturn
7fffe720 while (i_2 != -1)
7fffe720
7fffe728 data_800209e0 = arg2
7fffe728
7fffe734 if (*arg2 != 1)
7fffebe0 roottask_assert(
7fffebe0 assertion_str_1: "
7fffebe0 g_bootinfo_supp->version == L4_BaseBand_SuppBootInfo_Version",
7fffebe0 file_1: "roottask/src/boot.c", function_1: "bootinfo_init_early",
7fffebe0 line_1: 0x171)
7fffebe0 noreturn
7fffebe0
7fffe748 parse_patchbay_0x7ffff02c(0x80020578, arg2[4], patchbay_size: arg2[5])
7fffe74c int128_t* x1_2 = data_80020588
7fffe74c
7fffe750 if (x1_2 == 0)
7fffec00 roottask_assert(assertion_str_1: "g_kern_patchbay_vars.boot_args",
7fffec00 file_1: "roottask/src/main.c", function_1: "parse_kern_patchbay",
7fffec00 line_1: 0x348)
7fffec00 noreturn
7fffec00
7fffe758 int64_t x8_10 = data_80020590
7fffe764 int64_t x22_1
7fffe764
7fffe764 x22_1 = x8_10 u< 0x3ff ? x8_10 : 0x3ff
7fffe764
7fffe77c src/libc/string/memcpy_0x7fffdb88(&data_800205d8, x1_2, x22_1, 0x400)
7fffe780 *(&data_800205d8 + x22_1) = 0
7fffe794 int32_t segname
7fffe794 char* i_8
7fffe794 int128_t v0
7fffe794 i_8, v0 = sub_7fffdd04(&data_800205d8, 0x8000c1fc, &segname)
7fffe798 int128_t var_a0
7fffe798 uint8_t var_70
7fffe798
7fffe798 if (i_8 != 0)
7fffe79c char* i_7 = i_8
7fffe84c char* i_3
7fffe84c

Data Structures

Automated Extraction

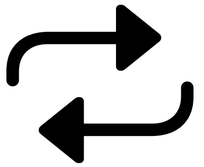


Hardware

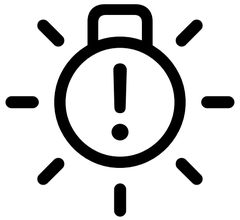


Operating System

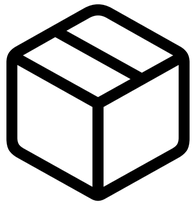
intel®



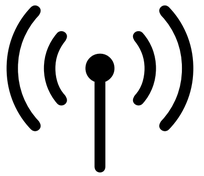
Tasks



Trace



Protocols

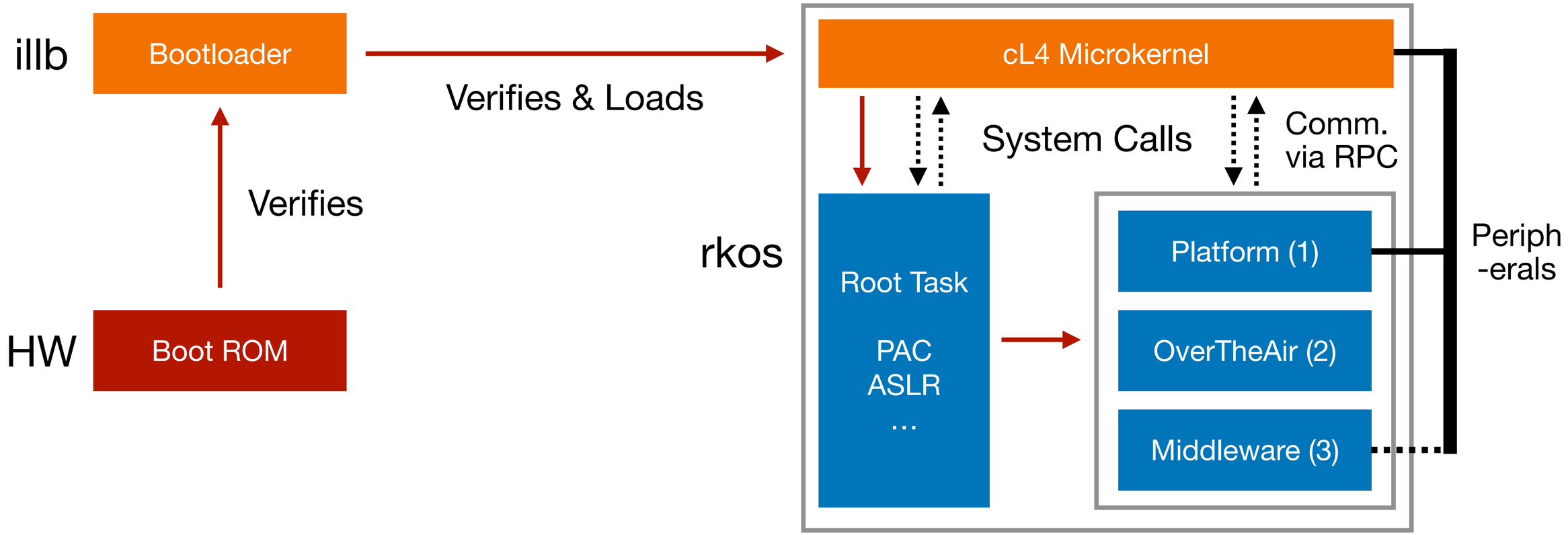


Cellular FSMs

Security: Model & Mitigations 

Exploitation Security Mitigations

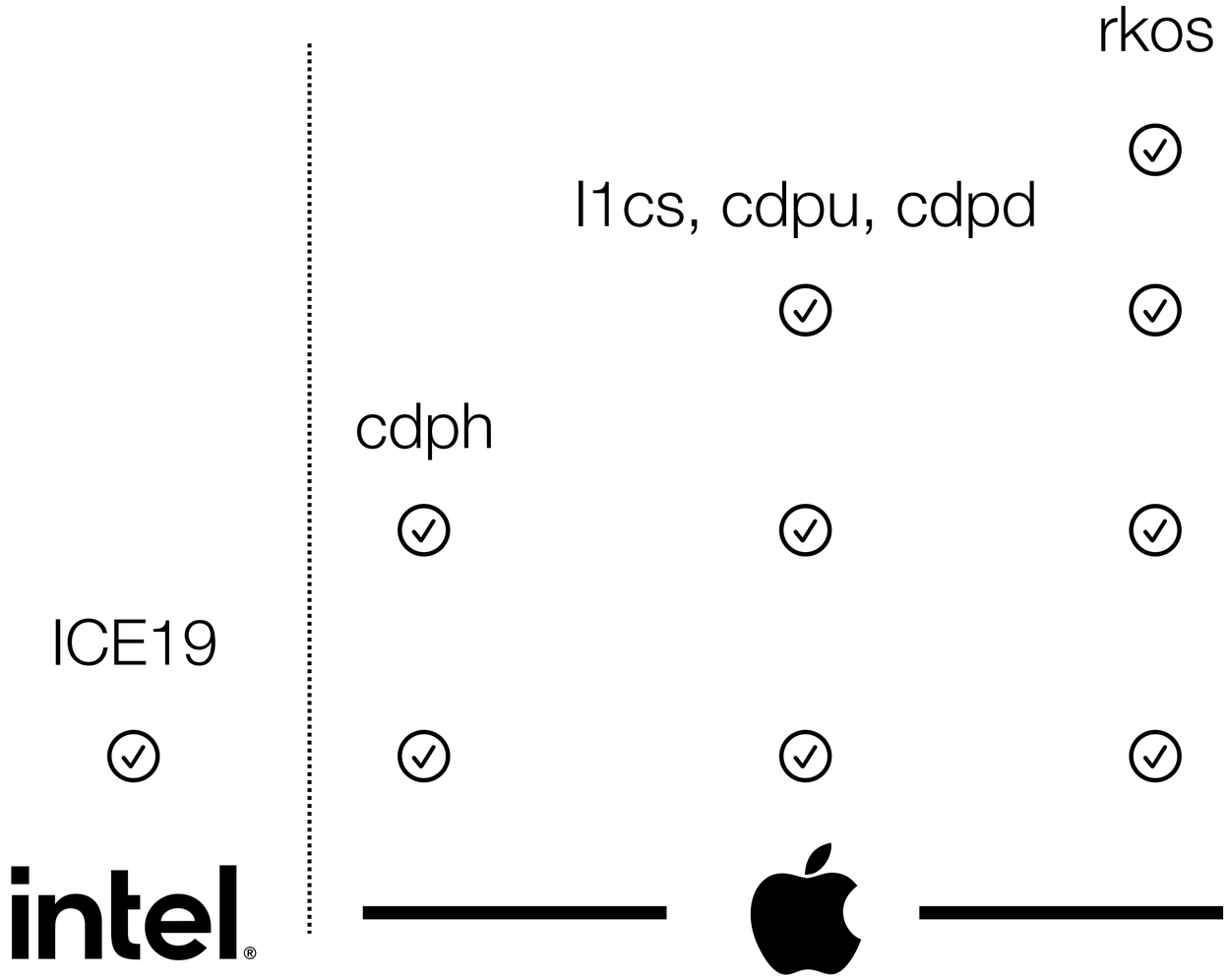
Boot Chain (with Chain of Trust)



Find more details in pwnlambda's work

Operating System Compartment Kit

Exploitation Security Mitigations



cL4 Microkernel (+ CTRR) Compartmentalization
Branch Target Identification Pointer Authentication
Assertions Write XOR Execute
Address Space Layout Randomization Safe String Library Chain of Trust Stack Guards

Exploitation Security Mitigations

ICE19

~ Application Processor

adapted to embedded context

cL4 Microkernel (+ CTRR) Compartmentalization
Branch Target Identification Pointer Authentication
Assertions Write XOR Execute
Address Space Layout Randomization Safe String Library Chain of Trust Stack Guards

Thanks!

Questions?

📪 larnold@seemoo.de – 🦣 mastodon.social/@lukasarnld – 🌎 lukasarnold.de

SEEMOO SECURE MOBILE NETWORKING

TECHNISCHE UNIVERSITÄT DARMSTADT

institute IMDEA networks