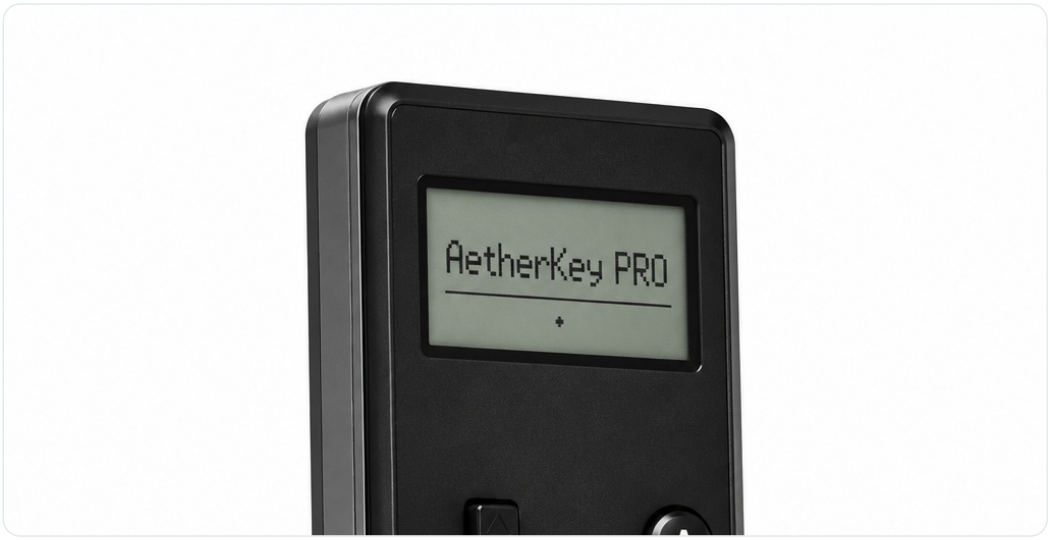
AetherKey

The multi-protocol field toolkit that puts an entire red-team bench in your pocket.



A pocket-sized, ESP32-S3 powered multi-tool that unifies BadUSB payload execution, Wi-Fi and Bluetooth attack tooling, NFC/RFID interaction, sub-GHz radio work and infrared control in a single handheld with an OLED display, D-pad and A/B gamepad controls — engineered as the spiritual successor to the Flipper Zero, USB Rubber Ducky and Marauder family, and shipping in two tiers: **AetherKey Regular** for turnkey operation, and the fully unlocked **AetherKey Pro** for builders who want to make the device their own.

102×38×27	MM FORM FACTOR
5	RADIO & BUS SYSTEMS
<900 ms	PLUG-TO-PAYLOAD
12 h	ACTIVE BATTERY LIFE

CONTENTS & EXECUTIVE SUMMARY

EXECUTIVE SUMMARY

AetherKey is a **102 × 38 × 27 mm handheld multi-protocol security toolkit** built around the Espressif **ESP32-S3-WROOM-1-N8R8** system-on-chip. The device pairs a 1.3-inch monochrome OLED with a four-way D-pad and two gamepad-style action buttons, giving it the unmistakable silhouette of a classic vertical handheld console while concealing a professional field toolkit. A single SoC coordinates every capability: native USB 2.0 On-The-Go for keyboard/mouse emulation and mass-storage payloads, a 2.4 GHz radio for Wi-Fi and Bluetooth (including the full Marauder toolset), a PN532 NFC controller for contactless card work, a CC1101 sub-GHz transceiver for 300–928 MHz remote control traffic, and an infrared blaster array with matching receiver for consumer IR.

What separates AetherKey from every device that came before it is the **control surface that surrounds it**. A dedicated Android companion application provides a full second screen for the device over BLE — a payload studio, a live Marauder console, NFC and IR tools, battery telemetry and an OTA firmware manager. For every other platform, the same capabilities are exposed through a **zero-install web panel** served over Web Bluetooth or a Wi-Fi captive portal, so an iPhone, a Linux laptop or a borrowed classroom PC can drive the device with nothing but a browser. Firmware is a modular, FreeRTOS-based runtime in which every capability is a swappable module, and on the Pro tier that modularity is opened all the way up: AetherSDK exposes the same internal APIs the factory firmware uses, so users can compile their own capability packs, install community modules from a signed registry, and flash through USB or the on-board UART/JTAG pads.

The product ships in two tiers. **AetherKey Regular** is the turnkey edition — every stock tool, factory-fitted and ready to run, with firmware locked to signed official images so the device always boots a known-good state. **AetherKey Pro** is the unlocked edition for engineers and researchers: the expansion edge pads are populated, debug entry points are enabled, and the module registry accepts user-compiled code. Both share identical RF hardware, mechanicals and companion ecosystem, so a team standardises on one tool and chooses per person how open it should be.

1 · CONCEPT & POSITIONING

Every generation of field tools has forced a choice. The USB Rubber Ducky is devastatingly effective at HID automation but is blind — it cannot see the network it lands on. The Flipper Zero is a brilliant interaction toolkit for RF and access-control work, but its USB presence is comparatively limited and its Wi-Fi story depends on external companion boards. Dedicated Marauder builds deliver world-class Wi-Fi tooling yet live on breadboards or dev boards with no display, no controls and no enclosure. A penetration tester assembling coverage across all of these domains historically carried a backpack: a ducky, a Flipper, a Wi-Fi Pineapple or bettercap-running laptop, an Proxmark for card work, and a universal remote.

AetherKey collapses that backpack into a single vertical handheld. The design thesis is simple: **one pocketable object, one battery, one display, one control language, every protocol**. The ESP32-S3 was selected precisely because it is unusually capable of carrying several of those domains simultaneously — it is one of the few modules on the market that combines a dual-core 240 MHz application processor, a mature Wi-Fi/BLE stack, native high-speed USB that can enumerate as a keyboard and a mass-storage volume at once, hardware cryptography, and enough GPIO flexibility to front-end external radios like the CC1101 and PN532 without glue logic. Placing all of that on one die eliminates the co-processor latency and power overhead that multi-chip designs suffer, and it means every capability — from ducky payloads to beacon spam — can read and act on each other's state instantly.

The product is positioned squarely as a **red-team multipurpose tool for authorised security testing, research and education**. It is the device an assessor can hand to a junior consultant with the Regular firmware (safe, guided, logged), and the device a hardware researcher flashes with their own code on the Pro. That dual identity is deliberate: the two tiers are not different hardware — they are the same hardware with different permission models, which lets teams buy one SKU set and tune openness per operator.

> *The goal was never to replace any single tool. It was to make the backpack unnecessary — one object that boots in half a second, speaks every protocol on the engagement, and gets out of your way.*

CAPABILITY MAP AT A GLANCE

DOMAIN	WHAT AETHERKEY DOES	PREDECESSOR IT ABSORBS
USB / HID	DuckyScript-compatible payload engine, run-on-plug, mass-storage volume, CDC console	USB Rubber Ducky
Wi-Fi	Scan, evil portal, beacon spam, handshake capture, deauth detect, station/AP/monitor modes	WiFi Marauder / Pineapple
Bluetooth	BLE scan & spoof, GATT server for the web panel, BT device discovery	—
NFC / RFID	Read/write/emulate ISO14443A/B, MIFARE, FeliCa; dictionary attack; LF emulation via app	Flipper Zero / Proxmark (lite)
Sub-GHz	CC1101 300–928 MHz capture, replay, raw frame editor	Flipper Zero
Infrared	Capture, replay, 650-device library, raw timing editor	Flipper Zero / universals
MITM / network	bettercap-compatible flows via USB/Wi-Fi bridge, captive ops, ARP/DNS spoofing through companion	Wi-Fi Pineapple / bettercap
Extensibility	Modular firmware, AetherSDK, signed module registry (Pro)	—

Fig 2 — AetherKey in hand: full control without a host computer.

2 · INDUSTRIAL DESIGN & HUMAN INTERFACE

The enclosure follows a vertical Game Boy-inspired layout, chosen for reasons beyond nostalgia. A tall, narrow body (102 × 38 × 27 mm) fits naturally in one hand with the thumb resting on the controls and fingers wrapping the spine — the grip a field tool needs when it is used standing up, in a rack aisle, or at a door reader. The upper third of the front face is dedicated to a 1.3-inch 128 × 64 monochrome OLED, chosen over colour alternatives because a monochrome panel is readable in direct sunlight, sips power, and renders crisp bitmap UI that is legible at arm's length. Beneath it sit the four controls that define the device: a four-way D-pad on the left, two circular action buttons labelled **A** and **B** on the right, and a green LINK LED between them that glows whenever the device is paired to the companion app or web panel.

The control scheme deliberately mirrors a game console rather than a five-way joystick plus OK button, because two context buttons let menus carry meaning: across the firmware, **A is always "select / act / confirm"** and **B is always "back / cancel / stop"**, with the D-pad navigating lists and adjusting values. A long press of A serves as a universal "run this now", and holding B for two seconds is a global stop that halts any running module — including an in-flight payload — and returns to the home screen. This consistency means the device is operable with muscle memory within minutes, in the dark, and without reading a manual.

Materials are chosen for field durability with a professional finish: a PC/ABS shell with a soft-touch coating, anodised 6061 aluminium rails top and bottom that act as both structural spine and RF window, Torx T3 fasteners into brass inserts (no clips to fatigue, no Phillips to strip — and deliberate: it signals "this is serviceable hardware"). The USB-C receptacle sits on the bottom edge for charging and host work; a microSD slot on the side carries payloads, captures and wordlists; recessed RST and BOOT pads live on the spine for recovery. The Pro edition exposes an additional side window over the expansion edge pads for I²C, UART and spare GPIO.

Fig 3a — Bottom edge: USB-C, microSD, recessed reset.

fig 3b — Pro edition, matte black with bronze accents.

INTERFACE GRAMMAR

D-pad navigates · A confirms and long-press runs · B cancels, backs out, and long-press is a global STOP · LINK LED glows green when paired, pulses while scanning, and goes dark on the hardware radio kill switch. The OLED contrast, orientation and pixel theme are adjustable in settings, and the entire UI can be mirrored to the companion app in real time.

3 · HARDWARE ARCHITECTURE

The internal architecture is a disciplined single-SoC star topology. Everything orbits the ESP32-S3, and every satellite is on a bus the SoC can drive with hardware peripherals rather than bit-banged GPIO: the OLED and PN532 share the I²C bus; the CC1101 and microSD share SPI; the IR blaster and receiver use the RMT peripheral's eight channels for nanosecond-accurate edge timing; the buttons and LEDs are plain GPIO with hardware debouncing. Power is a three-chip chain — a BQ25895 charger/power-path controller on VBUS, a MAX17048 fuel gauge feeding battery state into the UI, and a TPS62841 buck producing the 3.3 V system rail with a 60 µA quiescent draw that lets the device sit in standby for days.

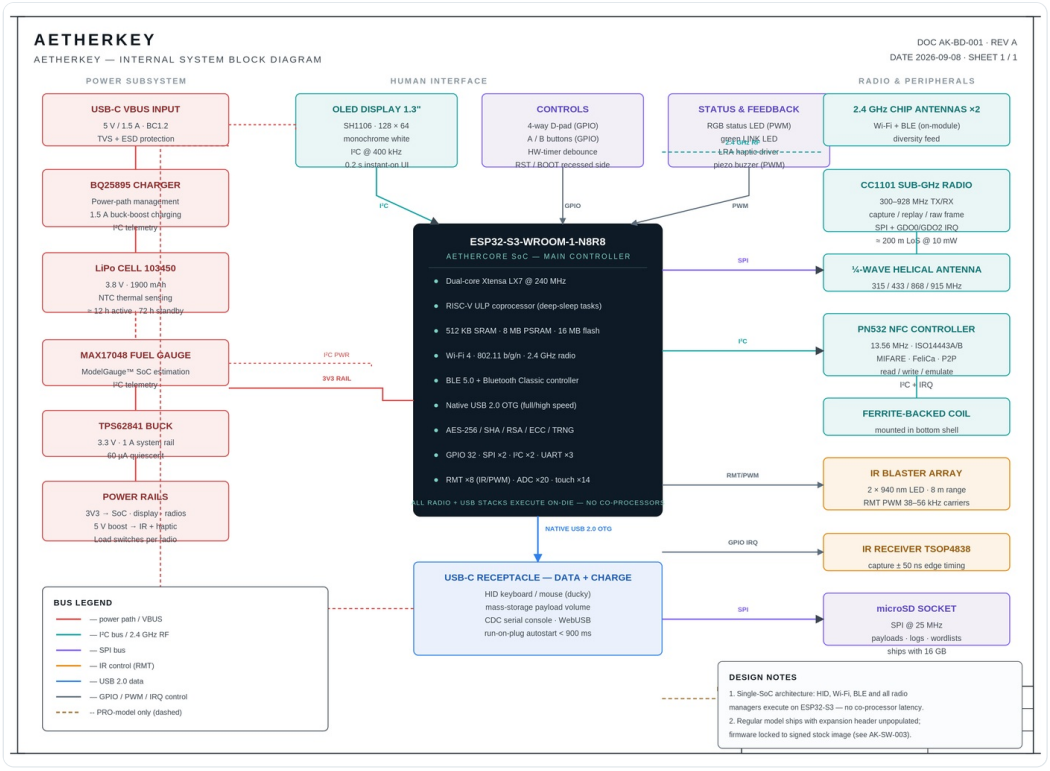


Fig 4 — Internal system block diagram (AK-BD-001). Full-resolution PNG included with this document.

Two design decisions in this diagram deserve comment. First, the **native USB 2.0 OTG** path runs straight from the SoC to the USB-C receptacle: there is no USB hub or bridge chip, which is why plug-to-payload time is under 900 ms and why the device can composite HID and mass-storage in a single enumeration. Second, **each radio domain has its own power path through a load switch** — the CC1101, PN532 and the 5 V IR boost can each be physically de-powered from firmware, which both conserves battery and gives the device a defensible "radios off" state that the kill switch enforces.

The expansion header shown dashed on the right of the diagram is populated only on the Pro edition: a Qwiic/Stemma-compatible I²C connector, eight spare GPIO with UART0, and 3V3/5V takeoff pads. On Regular hardware the header footprints exist on the PCB but are unpopulated and the firmware refuses to load unsigned modules — the hardware is identical, the doors are simply locked.

4 · EXPLODED MODULE VIEW

The dissection below shows the assembly stack from top shell to bottom shell along the device's central axis. Four M1.4 Torx T3 screws thread from the bottom enclosure into brass bosses captured in the top shell, clamping the display FPC, main PCB and battery sandwich between them. The NFC antenna is not on the PCB at all — it is a four-turn, ferrite-backed coil laminated to the inside of the bottom shell, which both frees PCB area and lifts the antenna away from the battery's ground plane for a stronger read field.

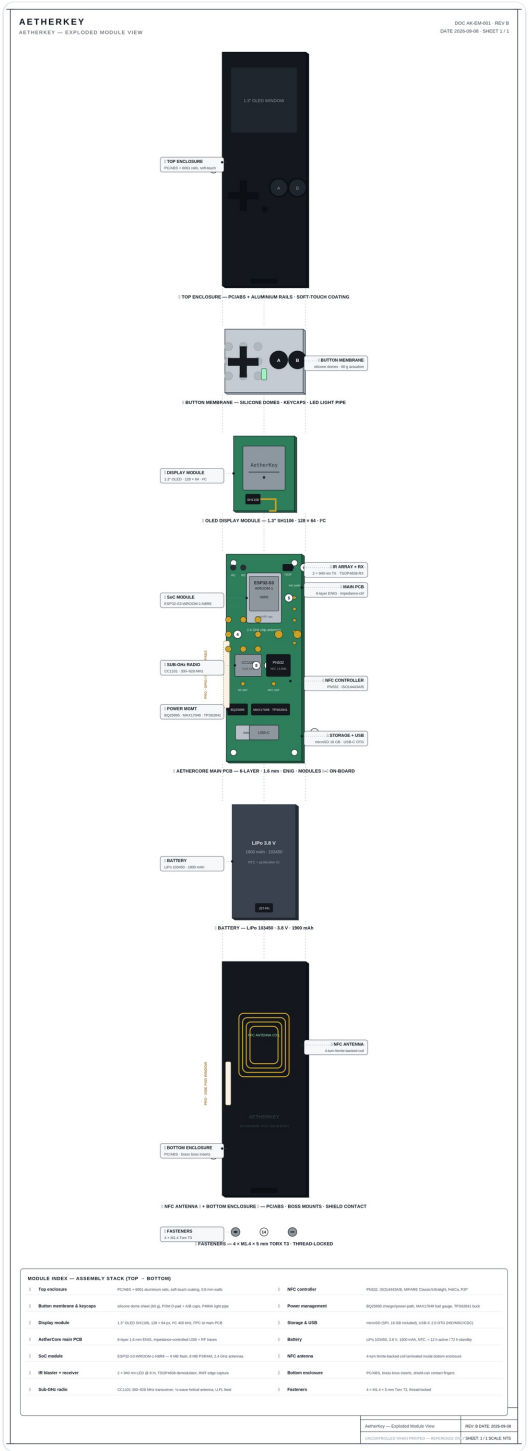


Fig 5 — Exploded module view with module index ①–⑭ (AK-EM-001 rev B). Full-resolution PNG supplied with this document.

The photorealistic render below shows the same assembly as it separates in space, and the AI-rendered companion piece follows. Note how the display FPC folds behind the OLED before mating to the main board's ZIF socket, how the battery occupies the mid-volume between PCB and bottom shell, and how the helical antenna exits the PCB's top edge through the aluminium rail window.

Fig 6 — Photoreal exploded render of the same assembly stack.

5 · TECHNICAL SPECIFICATIONS

MECHANICAL	
Dimensions	102 × 38 × 27 mm (W×D×H as held: 38 wide × 102 tall × 27 thick)
Mass	118 g including battery
Enclosure	PC/ABS, soft-touch coating, 6061 aluminium rails, Torx T3 service entry
Display	1.3″ OLED (SH1106), 128 × 64 monochrome, I²C 400 kHz
Controls	4-way D-pad, A/B action buttons, recessed RST/BOOT, RF kill switch
Indicators	LINK LED (green), RGB status LED, LRA haptic, piezo buzzer
COMPUTE & STORAGE	
SoC	ESP32-S3-WROOM-1-N8R8 — dual-core Xtensa LX7 @ 240 MHz, RISC-V ULP
Memory	512 KB SRAM, 8 MB PSRAM, 16 MB flash
Crypto	AES-256, SHA, RSA, ECC hardware acceleration, TRNG
Removable storage	microSD via SPI, 16 GB card included
RADIOS	
Wi-Fi	802.11 b/g/n 2.4 GHz, station / soft-AP / promiscuous monitor
Bluetooth	BLE 5.0 (+ Classic controller), GATT client & server
Sub-GHz	CC1101, 300–928 MHz TX/RX, ≈200 m line-of-sight @ 10 mW, helical antenna
NFC / RFID	PN532, 13.56 MHz ISO14443A/B, MIFARE Classic/Ultralight, FeliCa, P2P; LF via companion
Infrared	2 × 940 nm TX LEDs (≈8 m), TSOP4838 receiver, 38–56 kHz carriers
USB & POWER	
USB	USB-C 2.0 OTG (480 Mbps) — HID keyboard/mouse, mass-storage, CDC, WebUSB
Battery	LiPo 103450, 3.8 V, 1900 mAh — ≈12 h active, ≈72 h standby
Charging	BQ25895, 5 V/1.5 A input, full in ≈2.5 h; MAX17048 fuel gauge
ENVIRONMENT & COMPLIANCE	
Operating	−10 °C to +45 °C, IP54 splash-resistant
Compliance	Designed to FCC Part 15 / CE / UKCA / RoHS; region-locked TX power tables

MODEL DIFFERENTIATION

CAPABILITY	AETHERKEY REGULAR	AETHERKEY PRO
All stock tools (payloads, Marauder, NFC, sub-GHz, IR)	●	●
Android companion app + BLE web panel	●	●
OTA firmware updates (official signed)	●	●
Payload authoring (DuckyScript + Aether extensions)	●	●
Custom modules / capability packs (.AEM)	—	●
AetherSDK source access & toolchain	—	●
Community module registry install	—	●
Expansion edge pads (I²C, GPIO, UART0) populated	—	●
USB debug / UART/JTAG boot entry enabled	—	●
Bootloader unlocked for custom firmware	—	●
Boot state	factory-signed image only	user-controlled

WHY LOCK THE REGULAR EDITION?

Turnkey devices are handed to juniors, lent to colleagues, and left in bags. The Regular edition always boots a state the owner can reason about: firmware is factory-signed, modules cannot be replaced, and recovery is a single official reflash. The Pro edition is for people who accept the responsibility (and the fun) of running their own code — the same philosophy that separates a games console from a dev kit.

6 · SOFTWARE & FIRMWARE ARCHITECTURE

AetherCore, the device firmware, is a FreeRTOS-based modular runtime. Every capability — the payload engine, the Marauder Wi-Fi suite, the NFC manager, the IR manager, the web panel — is a **module** that registers itself with a central registry and declares the buses, GPIO and UI surfaces it needs. Modules can be enabled, disabled or (on Pro) replaced at runtime without reflashing, because the registry arbitrates hardware ownership: two modules cannot both claim the CC1101, and a disabled module's GPIO is released back to the pool. This is the architectural feature that makes the whole product coherent — the device is not a menu of frozen apps but a small operating system for radio tools.

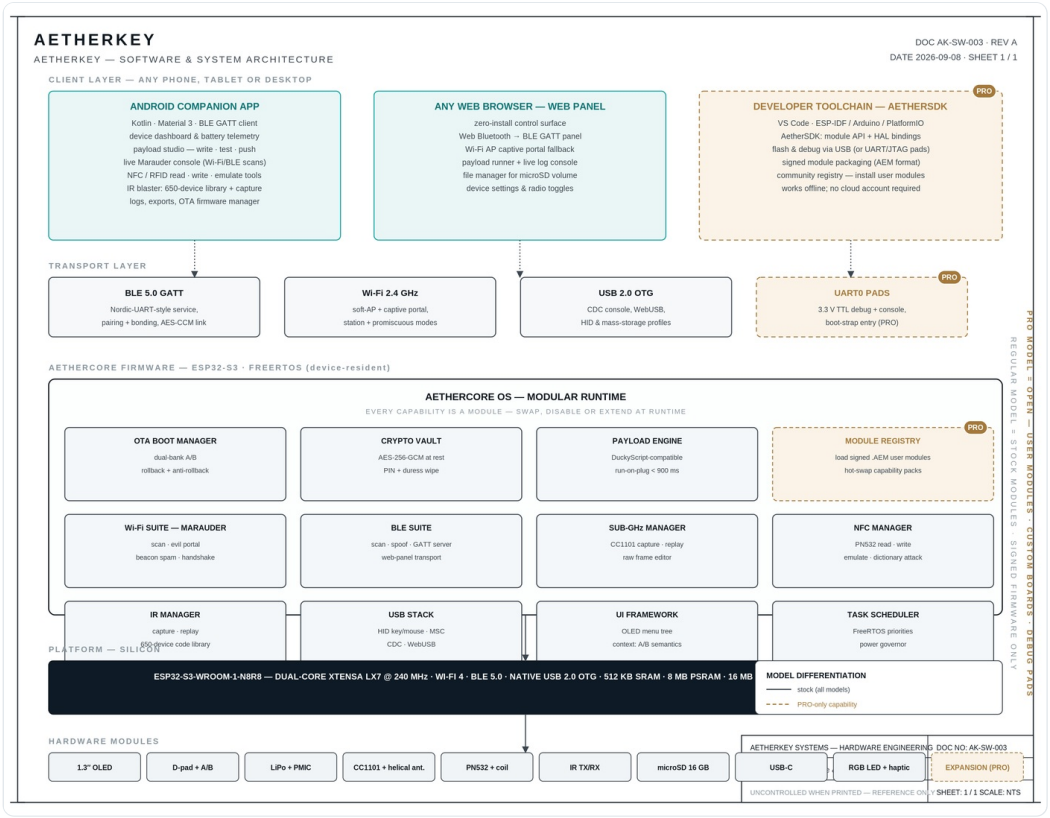


Fig 7 — Software & system architecture (AK-SW-003), showing client, transport, firmware and hardware layers with PRO-only elements dashed.

The firmware is served by a small set of core services. The **crypto vault** stores captures, credentials and payload secrets AES-256-GCM encrypted at rest, unlocked by PIN on boot, with a configurable duress PIN that performs a cryptographic wipe instead. The **OTA boot manager** maintains A/B firmware banks with rollback and anti-rollback counters, so a bad flash or brown-out during update can never brick the device — it simply boots the previous bank. The **task scheduler** is a power governor that aggressively duty-cycles radios and parks the ULP coprocessor as a watchdog for timed tasks (such as a delayed payload or scheduled scan) while the main cores sleep.

Above the firmware sit three client surfaces, deliberately kept equivalent in capability. The Android app is the flagship client; the web panel exists so that iOS, desktop and restricted machines never need an install; and AetherSDK (Pro) is the developer surface — a VS Code toolchain on top of ESP-IDF/Arduino/PlatformIO that exposes the exact module API the factory firmware uses, packages user modules into signed **.AEM** files, and publishes to a community registry. Every surface speaks to the device over the same GATT/USB transport, so a module written once is usable from the on-device UI, the app, and the web panel without porting.

7 · COMPANION APP & WEB PANEL

The Android companion (Kotlin, Material 3) turns the phone into a full second screen for the device over an encrypted BLE link. The dashboard surfaces live battery telemetry, link quality and radio state at a glance; the payload studio is where most field work happens — scripts are written or imported, syntax-checked against the device's parser, dry-run tested, and pushed to the AetherKey's microSD with one tap. The Marauder screen is a live console streaming scan results as the device sees them, with captured networks and clients select-able as targets for evil portals or beacon spam. NFC and IR tools mirror their on-device counterparts with the extra screen real estate used for tag databases, signal history and bulk actions. The log view exports everything the device has done in forensic-friendly format.

Fig 8a — Companion dashboard: battery, radio state, tool grid.

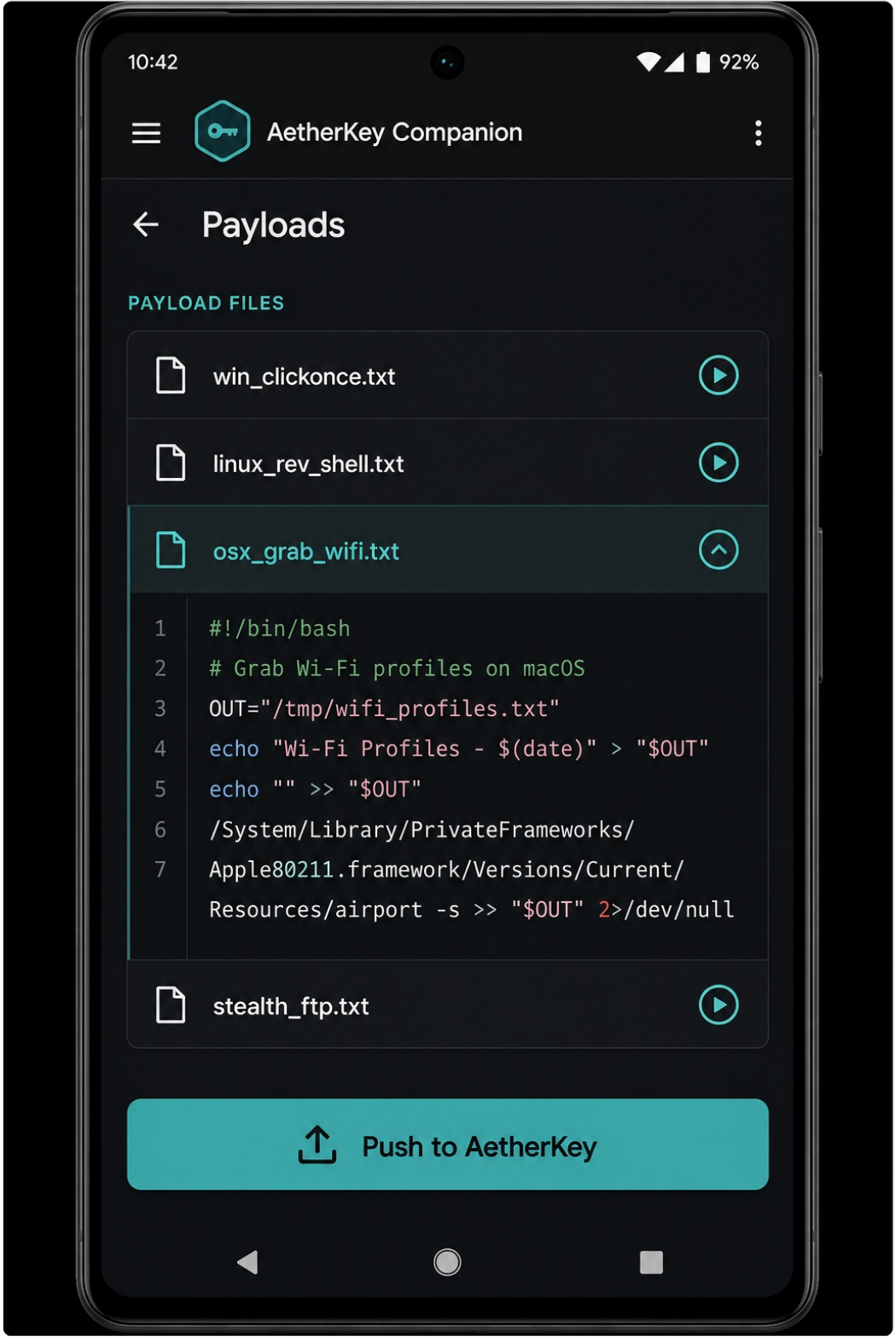


Fig 8b — Payload studio: write, test, push to device.

Fig 8c — Live Marauder console with target selection.

Fig 8d — NFC/RFID tools: read, write, emulate, brute.

The **web panel** is the same feature set without an app store. Over Web Bluetooth (Chrome/Edge desktop, Android Chrome) the panel connects to the device's GATT service directly; over Wi-Fi the device hosts a soft-AP captive portal so any browser — including iOS Safari — can drive it after joining the access point. The panel carries the payload runner, a live log console, a file manager for the microSD volume, and settings for radios, UI and firmware. Nothing is installed, nothing phones home, and the panel is served from the device itself so it works fully offline.

Fig 9 — The BLE web panel running on a phone browser beside the device.

8 · PAYLOAD ENGINE

Plug the AetherKey into any unlocked host and it enumerates as a composite USB device: a keyboard, a mouse and a mass-storage volume, presented in under 900 milliseconds from insertion. The payload engine interprets a DuckyScript-compatible language — the de-facto standard for HID automation, meaning the enormous existing corpus of community payloads runs unmodified — extended with AetherKey-specific commands that reach beyond a classic ducky because the interpreter can call on every other module in the system.

AETHER EXTENSIONS (BEYOND STOCK DUCKYSCRIPT)

- **WAIT_FOR_NETWORK / WAIT_FOR_DRIVE** — gate execution on host conditions
- **STACK_RUN** — execute payloads conditionally based on OS fingerprint
- **HTTP_EXFIL** — stage results over the device's own Wi-Fi instead of USB
- **BLE_LOG** — mirror live execution to the companion console
- **RANDOM_DELAY / JITTER** — humanised typing cadence

Payloads live on the microSD in a simple directory structure, can be selected and run from the on-device UI without a host at all (useful for staging), and can be marked as **run-on-plug** so the device behaves exactly like a Rubber Ducky when the engagement calls for that. On the Pro edition, the engine also accepts compiled executable payload stages that run from the mass-storage volume, and payload authors can push scripts from the phone to the device over BLE without ever touching the SD card.

Because the engine is a module in the registry rather than baked into the firmware, the interpreter itself is upgradable independently of the OS — the community can extend the scripting language without waiting for a full firmware release.

9 · WIRELESS TOOLKIT — MARAUDER, MITM & CAPTIVE OPS

The Wi-Fi suite ships the full **Marauder** toolset — the reference ESP32 Wi-Fi attack toolkit — as a first-class firmware module, not a bolted-on app. From the on-device UI or the companion console this covers station and AP scanning with per-client association tracking, beacon spam (targeted or flood), evil-portal hosting with credential harvesting templates, PMKID and EAPOL handshake capture, probe-request sniffing, and deauthentication detection with alerting. Raw captures land on the microSD in PCAPNG format ready for offline cracking, and the companion app will hand a capture straight to a cracking workflow or export it.

Bluetooth tooling covers BLE scanning with service enumeration, advertisement spoofing for device-impersonation testing, and the GATT server that underpins the web panel — the same infrastructure that makes the device controllable makes it a testbed for BLE attacks. As with Wi-Fi, TX behaviour is bounded by region-locked power tables and a hardware kill switch.

For man-in-the-middle work the device plays two roles. Tethered over USB to a laptop it acts as a **bettercap-compatible attack radio**: the laptop runs bettercap with the AetherKey as its interface, combining the pineapple-style captive ops the device does standalone with the full suite bettercap brings (ARP/DNS spoofing, credential sniffer, proxy modules). Standalone, the device's own soft-AP plus captive portal covers the common "rogue network" engagements without any host computer at all — check a box in the web panel and the AetherKey becomes the test access point, logging every association attempt and template-served credential to encrypted storage.

OPERATIONAL SAFETY RAILS

All attack modes refuse to run until the operator acknowledges an authorisation banner and sets an engagement window; every action is logged locally with a timestamp; and the RF kill switch physically disables all transmitters when the engagement is over. Region tables limit TX power and channel use to what local regulations permit, and restricted modes (jamming class interference) are not implemented at all.

10 · NFC / RFID, SUB-GHZ & INFRARED

Contactless work runs on the PN532, the industry-standard 13.56 MHz controller. The device reads and writes ISO14443A/B tags — MIFARE Classic and Ultralight families, plus FeliCa — and can emulate a tag itself so a saved credential can be presented to a reader. A dictionary attack module automates key recovery against Classic tags using configurable key lists held on the microSD, and the companion app adds a tag database with comparison, notes and export. For low-frequency 125 kHz credential testing, the companion app pairs the device with optional LF accessory hardware over the expansion pads (Pro) for reading and emulating EM4100-class badges.

The sub-GHz side is a CC1101 transceiver covering 300–928 MHz on a quarter-wave helical antenna, feeding the same capture/replay model the Flipper popularised: record a remote's transmission, inspect the raw frame (modulation, deviation, sync word, payload), edit it, and retransmit — with saved signal files on the microSD and a library of common protocols (fixed-code remotes, rolling-code captures for analysis, weather sensors, doorbell and garage openers for authorised testing). Transmissions are region-gated by frequency table, and a rolling-code replay guard warns before resending captured rolling frames.

Infrared is the most universal and least controversial of the device's radios: two 940 nm emitters give roughly eight metres of range, the TSOP4838 receiver captures signals with edge timing good enough to discriminate protocol families, and a built-in library covers 650+ consumer devices (TVs, projectors, HVAC, AV racks) with a raw timing editor for the long tail. In practice this is the feature that gets the device into the most buildings — it is the universal conference-room remote that also happens to be a security tool.

11 · SECURITY MODEL & RESPONSIBLE USE

A tool whose purpose is to test security must itself be hardened, and the model is layered. At rest, everything sensitive — captures, credentials, payload secrets — lives in the crypto vault, AES-256-GCM encrypted with a key sealed by PIN; a configurable duress PIN triggers a cryptographic wipe rather than unlock. Over the air, BLE pairing is bonded and encrypted, the Wi-Fi portal uses WPA2 with a device-printed key, and USB CDC requires explicit unlock. At boot, the Regular edition verifies a factory signature on the firmware bank and refuses to run unsigned images; the Pro edition's bootloader is unlocked by its owner's explicit action (with the tamper-evident seal broken and the warranty converted to a dev-kit warranty), so openness is a decision, not an accident.

Operationally, the device is built for authorised use. First-boot setup requires acknowledging a responsible-use agreement; every module run stamps an entry in a tamper-evident local log (hash-chained, exportable as evidence); the RF kill switch is a physical sliding contact that cuts power to every transmitter through those per-radio load switches; and regional compliance tables enforce permitted frequencies and power levels per market. The product ships without any capability whose only realistic use is interference (jamming), and documentation throughout frames every feature in a testing-and-authorisation context.

THREAT MODEL FOR THE DEVICE ITSELF

Assumed adversaries: casual theft (mitigated by PIN + vault), a paired-host compromise (BLE bonding keys, no host command executes without on-device confirmation for destructive operations), and physical access with the case open (Regular: debug entry disabled and unsigned code refused; Pro: accepted — the owner chose an open platform). Side-channel and nation-state attacks are explicitly out of scope for this generation.

12 · LINEAGE & COMPETITIVE POSITION

AetherKey is honest about its lineage — it is a synthesis of tools the community already loves, executed on one SoC with a modern control ecosystem. From the Rubber Ducky family it takes instant HID automation and the DuckyScript corpus. From Flipper Zero it takes the capture/replay model, the sub-GHz and IR interaction toolkit, and the philosophy that a security tool can be friendly enough to live on a keychain. From the Marauder project it takes the definitive ESP32 Wi-Fi attack suite, integrated natively rather than as an add-on board. From the Pineapple/bettercap world it takes captive ops and the MITM workflow. What none of those offer — and what AetherKey adds — is all of it simultaneously, plus a phone and browser control surface, plus a genuinely modular firmware, plus an open developer tier.

	AETHERKEY	FLIPPER ZERO	USB RUBBER DUCKY	MARAUDER BOARD
HID payloads, run-on-plug	●	Partial	●	—
Wi-Fi suite (evil portal, beacon spam, handshakes)	●	add-on	—	●
NFC read/write/emulate	●	●	—	—
Sub-GHz capture/replay	●	●	—	—
Infrared library	●	●	—	—
Phone app control	●	●	—	—
Browser control (zero install)	●	—	—	—
Modular firmware / user modules	**Pro**	—	—	●
MITM workflow (bettercap host)	●	—	—	—
Single device, single battery	●	●	●	bare board

13 · MODELS, BUNDLING & ROADMAP

AetherKey Regular ships with the device, a 16 GB microSD preloaded with payload templates and wordlists, a braided USB-C cable, a lanyard, and the quick-start card — everything needed to run every stock capability out of the box. **AetherKey Pro** ships with all of that plus the populated expansion header, a Qwiic jumper kit, the AetherSDK development licence, and a tamper-evident seal over the service screws whose breaking activates the developer warranty tier. Both editions share the same storefront, the same OTA channel for official firmware, and the same accessory range: LF 125 kHz companion hardware, replacement shells in alternate colours, and a field case.

The roadmap after launch concentrates on the ecosystem rather than the hardware: expansion of the module registry with first-party reference modules (a SDR-bridge for the expansion bus, a Zigbee/BLE-Mesh sniffer module, a CAN-bus interface), the community payload exchange built into the companion app, and an optional self-hosted fleet server for teams that want central logging of engagement history. Hardware revisions will track Espressif's module roadmap so the platform gains capability without changing its shape.

THE PITCH, IN ONE PARAGRAPH

AetherKey is what happens when the ducky, the dolphin, the pineapple and the universal remote stop being four objects in a backpack and become one object in a pocket — a 102-millimetre handheld that boots in half a second, automates a workstation over USB, owns a wireless environment over Wi-Fi and Bluetooth, talks to cards, remotes and IR devices, answers to your phone or any browser, and on the Pro tier opens its firmware and its hardware so thoroughly that the only limit on what it can become is what you write for it. It is not a toy that happens to be a tool; it is a tool that happens to be a joy to hold.