Device Behavioural Blueprint (DB^2): A Risk-Aware Framework for Unique Device Behaviour Profiling Using Microarchitectural Variations*

Muthupavithran Selvam[a,b,*], Safwana Haque[a,b], Amit Kumar Singh[c], Zhan Cui[b] and Rajarajan Muttukrishnan[a]

[a] City St George's, University of London, London, UK
[b] Cyber Immune Security Research, BT Group, London, UK
[c] University of Essex, Essex, UK

ARTICLE INFO

Keywords:
Device Fingerprinting, Microarchitectural Behaviour, CPU–RTC Timing Drift, PMU-Based Identification, Closed-Set Identification, Open-Set Recognition, DAIR Risk Scoring, Edge Device Security.

ABSTRACT

This paper introduces DB^2, a risk-aware behavioural profiling framework that derives device identity from CPU–RTC timing deviation and Performance Monitoring Unit (PMU) microarchitectural events, without relying on GPUs, radios, sensors, or dedicated hardware. The method captures oscillator-coupled timing variation and execution behaviour through a structured signal-processing pipeline, producing device specific behavioural signatures that remain distinguishable across reboots, temperature variation, and core transitions. DB^2 structures identity assurance into three layers: closed-set identification, calibrated open-set rejection, and stability aware risk scoring. Evaluation under a strict three-way split with reboot separation for training, calibration, and unseen testing yields a macro-F_1 of 0.957 on unseen reboots. The open-set layer rejects previously unseen devices with a mean true positive rate of 0.990 at a fixed event-level false-reject rate of approximately 0.12 under strict leave-one-device out validation, with operating point selection performed exclusively on the calibration split. A Dynamic Aware Identification and Risk (DAIR) mechanism decomposes behavioural stability across temperature, reboot, and core factors to provide interpretable posture monitoring for enrolled devices. Under identity claim manipulation via spoofing, Sybil, and cloning relabelling scenarios, targeted identities exhibit reduced identification consistency and elevated risk, while non targeted devices remain stable under identical calibration settings. These results demonstrate that reliable behavioural fingerprints can be engineered from standard CPU and RTC resources on commodity edge devices, supporting server-assisted, risk-aware authentication in IoT and edge environments.

1. Introduction

The rapid growth of the Internet of Things (IoT) is driven by advances in processing and communication technologies. More capable processors and pervasive connectivity now enable deployments across Industry 4.0, smart cities, healthcare, and defence [22, 28]. As a result, heterogeneous edge devices (EDs) have become fundamental system components, offering low cost, flexible deployment, and local computation [8]. Their ubiquity and accessibility, however, make them attractive targets for adversarial exploitation.

The same connectivity and constrained execution environments that enable large-scale IoT deployment also expose EDs to identity centric threats [31]. Compromised single board computers (SBCs) can participate in distributed denial-of-service (DDoS) campaigns, cryptojacking, and lateral movement. More critically, adversaries can introduce unauthorised or cloned devices that impersonate legitimate nodes, a problem observed in industrial automation and mobile communication systems [10, 35]. Identity forgery, device spoofing, and Sybil attacks undermine trust relationships at the network layer [18].

Conventional identity mechanisms, device names, certificates, and stored credentials depend on artefacts that can be copied or extracted [37]. This limitation motivates behavioural fingerprinting, in which device identity is derived from intrinsic hardware variability rather than from stored secrets. Manufacturing imperfections in oscillators, clock sources, and microarchitectural structures produce measurable deviations in timing and performance behaviour [1]. These variations form a behavioural signature that persists even across identical hardware and software images.

Microarchitectural variability manifests through subtle offsets in CPU execution timing, cache behaviour, and clock drift. In particular, interaction between the CPU cycle counter and the Real-Time Clock (RTC) exposes cross-oscillator deviation that reflects physical tolerances in crystal oscillators and clock distribution paths [20]. Unlike external sensing or radio-based techniques, such signals are widely available on commodity processors. Prior research has explored clock-skew estimation, PUF-style extraction, and performance monitoring approaches [26]. However, these studies typically treat fingerprinting as a classification task and rarely evaluate behavioural stability under reboot cycles, core migrations, or thermal variations [23, 25].

* This work was supported by British Telecom (BT).
*Corresponding author
muthupavithran.selvam@citystgeorges.ac.uk (M. Selvam); muthupavithran.selvam@bt.com (M. Selvam); safwana.haque@citystgeorges.ac.uk (S. Haque); a.k.singh@essex.ac.uk (A.K. Singh); zhan.cui@bt.com (Z. Cui); r.muttukrishnan@citystgeorges.ac.uk (R. Muttukrishnan)
ORCID(s):

Several barriers therefore, limit practical deployment on constrained edge platforms: (i) reliance on auxiliary hardware or sensing modules [2]; (ii) dependence on server-class computation; (iii) lack of unified approaches using only ubiquitous CPU and RTC resources; and (iv) insufficient evaluation under realistic operational stressors such as reboots, multicore scheduling, and temperature drift. Without these validations, fingerprinting remains a laboratory result rather than a deployable identity primitive.

Consequently, there remains no unified behavioural identity framework that derives fingerprints exclusively from ubiquitous CPU–RTC resources, rigorously validates stability under realistic operational stressors, and integrates closed-set identification with open-set rejection and continuous behavioural risk assessment within a client–server edge architecture. Rather than treating fingerprinting as a standalone supervised classification task, DB^2 integrates identity establishment, unknown-device rejection, and runtime stability monitoring into a unified framework suitable for any type of edge environment.

To address these challenges, this work makes three main contributions:

- An edge-lightweight behavioural fingerprinting methodology that anchors device identity in cross-oscillator CPU–RTC timing deviation and PMU microarchitectural signals, demonstrating persistent device separability across reboot cycles, multicore execution, and realistic temperature variation.
- An identity-layer validation framework that integrates calibrated open-set rejection with structured identity manipulation testing (spoofing, Sybil, cloning), enabling detection of both previously unseen devices and forged identity claims.
- A Dynamic Aware Identification and Risk (DAIR) framework, which extends static fingerprinting into continuous behavioural assurance by quantifying stability across thermal, reboot, and core transitions and mapping it to interpretable risk levels.

2. Related Work

This section summarises work on device fingerprinting, focusing on identification techniques that do not require additional external hardware. Only limited research has explored whether microarchitectural noise from on-board components can generate reliable device fingerprints. Most existing work relies on accelerator subsystems such as Graphics Processing Units (GPUs), Digital Signal Processors (DSPs), and Radio Frequency (RF) front ends rather than the more ubiquitous Central Processing Unit (CPU) and Real Time Clock (RTC) resources available across both edge devices and conventional personal computers (PCs).

The work most closely related to ours is by Sánchez Sánchez et al. [32], who fingerprint SBCs using oscillator-induced timing variations combined with GPU execution characteristics. Although effective on SBCs, the study primarily evaluates discrimination performance and does not explicitly analyse cross-reboot persistence, core migration effects, workload variability, or temperature-driven stability. Although it acknowledges degradation after reboot, it does not evaluate persistence. Furthermore, the approach depends on GPUs, which restricts its applicability across edge and embedded platforms.

2.1. Cyber-Physical and Type Level Fingerprinting

Babun et al. [4] proposed a fingerprinting framework for CPS that employed operating system and kernel level observables within a challenge-response protocol to generate type level signatures. This was effective for identifying device types rather than identical units of the same make and model. Additionally, Kumar et al. [17] provide a comprehensive survey on fingerprinting methodologies, outlining application-specific selection criteria and design considerations. These works, however, do not target instance-level uniqueness or address stability in edge environments.

2.2. Physical Unclonable Function (PUF)

PUFs generate device-specific responses to each applied challenge and have been widely deployed across devices for many years [3]. Recently, PUF realisations have been extended to diverse materials and form factors [38], and the distinction between strong and weak PUFs is determined by the number of challenge–response pairs (CRPs) [7]. However, PUFs require dedicated on-board circuitry, which reduces scalability, increases cost, and necessitates a dedicated CRP provisioning and management system.

2.3. Radio Frequency (RF)-Based Fingerprinting

Wireless RF-based fingerprinting exploits front-end imperfections to identify device-specific traits using raw baseband captures (in-phase and quadrature, I/Q) and channel state information (CSI) from mixers, oscillators, power amplifiers, and antenna paths. An early work [15] used beacon timing to identify access points. More recently, Yang et al. [36] have learned device characteristics from raw I/Q traces to handle cross-receiver variability. Other studies explore transmitter distortions and examine reliability under real-world conditions [14]. Defence studies such as HidePrint [19] inject controlled physical-layer noise to mask RF signatures, while Blind Spots [30] shows that hardware warm-up and environmental drift degrade stability and proposes sequential transfer learning to recover accuracy.

Despite these advancements, RF approaches face scalability and deployment challenges, as fingerprints often exhibit receiver dependence (requiring adaptation or re-enrolment across setups), may require specialised software-defined radio hardware and calibration, have near-field requirements, and remain sensitive to channel dynamics and warm-up effects [14, 30]. These constraints suggest that RF-based fingerprinting may require complementary behavioural signals to ensure stability across heterogeneous deployments.

2.4. Sensor-Based Fingerprinting

Sensor-based approach exploits manufacturing-induced imperfections in Micro-Electro-Mechanical Systems (MEMS) sensors, which manifest as small but consistent biases, scale errors, and noise asymmetries that are often dismissed as "sensor noise" but can reveal an intrinsic signature. SensorID [39] showed that smartphone inertial sensor calibration parameters yield stable device-specific signatures even among identical models. In another study, ICMetric [34, 33] derives cryptographic keys from measurable sensor characteristics, demonstrating feasibility, though key reproducibility and manipulation remain concerns. More recently, SenSig [11] applied statistical modelling to classify stand-alone Inertial Measurement Units (IMUs), such as the MPU-6050, from their intrinsic imperfections. However, these studies evaluate sensors in isolation under controlled conditions and rarely explore stability across temperature, workload, and time. Additionally, many edge platforms lack these sensors, which limits deployment.

2.5. Power, Acoustic, and Imaging-Based Fingerprinting

Some studies have utilised power, audio, and imaging to expose device-specific traits. V-Health [13] uses a relaxation-voltage curve after charge/discharge to enable low-cost battery fingerprinting, though signatures drift with ageing. Sweep-to-Unlock fingerprinting of smartphones using loudspeaker frequency-response slopes, influenced by room acoustics and speaker wear, introduces variability [6]. In imaging, Berdich and Groza effectively identify smartphones using mid-frequency Discrete Cosine Transform (DCT) coefficients of dark images to isolate sensor-fixed pattern and DSNU artefacts, though this technique is sensitive to ambient light, camera tuning, and sensor ageing, which affect stability and motivate calibration and fusion with other modalities for long-term robustness [5].

2.6. On-Board Timing and Processor-Imperfection-Based Fingerprinting

An early timing work [24] measured the drift between RTCs and digital signal processing (DSP) units against the CPU Time Stamp Counter (TSC), reporting 98.7% (RTC) and 93.3% (DSP) uniqueness across 703 PC pairs. However, reliance on discrete DSPs and specific RTC implementations limits applicability on edge devices, where such components are often absent or inaccessible. More recently, CADFA [29] measures device clock skew using microcontroller timers/RTCs within a simple challenge-response routine, achieving high accuracy on Arduino boards; however, it does not account for multicore scheduling, thermal effects, or CPU frequency scaling, which affect stability and repeatability on SBCs and PCs.

Our early work on the continuous device-to-device authentication (CD2A) framework exploited crystal-oscillator imperfections observable on CPU (per-core) and GPUs to build behavioural fingerprints and track identity over time [27]. CD2A established per-device identity, monitored behavioural change via drift metrics (DAS and DRF), applied an authentication timeline, and clustered events under policy control. The study reported high accuracy but also highlighted two practical limitations: limited GPU availability on edge nodes and incomplete characterisation of reboot and temperature-induced variability. These limitations motivate the present paper's focus on ubiquitous on-board resources and an explicit assessment of cross-reboot, thermal, and workload robustness. These observations motivate a design that (i) removes dependency on accelerators, (ii) anchors identity in CPU–RTC interaction available on commodity processors, and (iii) explicitly evaluates behavioural persistence under operational stress.

Device fingerprinting has evolved from early hardware-centric approaches to more recent software-driven, statistical, and learning-based techniques, as summarised in Table 1. Despite this progress, several recurring limitations remain. Many approaches depend on specialised sensors, radios, or accelerators such as GPUs or DSPs that are not universally available on resource constrained edge and IoT devices (LM1). Others provide limited analysis of fingerprint stability across reboots, temperature variations, and workload changes (LM2), or do not account for multicore variability and core migration effects (LM3). In addition, few studies incorporate mechanisms to monitor behavioural drift over time (LM4). These recurring gaps motivate a fingerprinting framework that relies solely on ubiquitous on-board components and explicitly validates behavioural persistence under realistic operational variation. To the best of our knowledge, no prior work reports a CPU–RTC-based behavioural pipeline evaluated under unified operational-stability and identity-manipulation criteria. Table 1 makes this gap explicit by mapping representative studies to LM1–LM4.

Rather than proposing a new sensing modality, DB² focuses on operationalising ubiquitous CPU and RTC primitives into a complete identity and assurance framework.

Table 1 consolidates these observations by mapping key studies against their algorithms, behavioural sources, features, outcomes, and limitations (LM1–LM4).

3. Problem Definition

In distributed IoT and edge environments, logical device identifiers such as MAC addresses or certificates can be forged or replicated without modifying the underlying hardware. Consequently, identity layer authentication alone cannot ensure that a claimed identity corresponds to its physical device.

This work addresses the problem of establishing and maintaining behavioural device identity under realistic operational variation. Given behavioural traces derived from CPU–RTC interaction and PMU signals, the task is threefold: (i) to determine whether a device can be reliably distinguished from other enrolled devices (closed-set identification), (ii) to detect behavioural traces that do not correspond

Table 1
Summary of device-fingerprinting works and their corresponding limitations (LM1–LM4)

WORKS	YEAR	DEVICE TYPE	ALGORITHM	BEHAVIOUR SOURCES	FEATURES	RESULTS	LIMITATIONS (LM1–LM4)
[24]	2007	PC	Statistical	RTC and DSP	RTC/DSP drift	93%+ in 38 PCs	LM1, LM2, LM3, LM4
[39]	2019	Mobile	Calibration error modelling	IMU	Sensor bias (offset, gain, scale factor)	Uniquely ID >100 devices	LM1, LM2, LM4
[4]	2021	CPS	Correlation-based	OS/Kernel	Sys calls, memory, CPU, timer	Device type ID	LM2, LM3, LM4
[34]	2021	IoMT	Statistical	IMU	Noise or bias	Unique key per device	LM1, LM2, LM4
[32]	2022	SBC	XGBoost	GPU counters	Window-based GPU features	91.9% avg TPR	LM1, LM2, LM3, LM4
[11]	2022	Sensor	Quantization & Hamming distance	MPU6050	Noise	Unique ID for 4 sensors	LM1, LM2, LM4
[5]	2022	Smartphone camera	Wide NN (low/mid-freq DCT)	Dark images	DSNU cues	Camera ID	LM1, LM2, LM4
[13]	2023	Smartphone battery	SoH mapping	Open-circuit voltage during rest	Relaxation-curve params	Only temporary	LM2, LM4
[6]	2023	Smartphone loudspeaker	FR estimation	Audio playback/recording	Roll-off slope, spectral envelope	Per-device ID; room/volume effects	LM1, LM2, LM4
[14]	2024	RF-dev-board	Few-shot / meta learning	Raw I/Q	Channel attention	83.6% accuracy	LM2, LM4
[29]	2024	Microcontroller	Timer/RTC skew	RTC & CPU timer	Clock skew/drift	Good accuracy but no repeatability	LM2, LM4
[36]	2025	RF-dev-board	Source-free adaptation	Raw I/Q	CFO & IQ imbalance	98% accuracy	LM2, LM4
[27]	2025	IoT	Classifier	CPU & GPU	Sliding-window stats	TPR 99.96%; robust to CA	LM1, LM2

to any enrolled identity (open-set rejection), and (iii) to assess whether a claimed identity remains stable over time despite reboots, workload changes, temperature variation, and multicore scheduling.

The core challenge is to distinguish legitimate operational variability from adversarial identity misuse without relying on additional hardware or stored secrets.

3.1. Threat Model

We consider adversaries that target the identity layer of a distributed IoT or edge system by manipulating logical identifiers such as MAC addresses, certificates, or software credentials. The attacker's objective is to gain unauthorised access, bypass identity-based controls, or distort system behaviour while operating on different physical hardware.

The following threat scenarios are examined:

- **TH1 (Device Spoofing):** An attacker adopts the identifiers of a legitimate device to impersonate it. By presenting valid credentials while operating on different hardware, the adversary attempts to bypass identity-based access control [9].
- **TH2 (Sybil Attacks):** A malicious device presents multiple fabricated identities simultaneously. These forged identities can bias distributed decision processes or distort trust metrics in decentralised systems [21].
- **TH3 (Cloning):** An adversary replicates the software stack and identifiers of a legitimate device to create a *logical replica*. Although the claimed identity appears valid, the physical hardware substrate differs, creating a mismatch at the behavioural level [12, 16].

In all cases, the attack targets the logical identity rather than directly modifying microarchitectural behaviour. The adversary exploits weaknesses in identity management rather than altering CPU execution paths, PMU counters, or RTC timing signals.

3.1.1. Assumptions and Trust Boundary

The following assumptions define the trust boundary of this study:

- The attacker cannot physically modify the hardware or replace components.
- The measurement agent and data collection pipeline are trusted and not tampered with.
- The attacker does not obtain kernel-level, root-level, or firmware-level control that would allow manipulation of performance counters, clock sources, or the measurement loop.

The CPU, RTC, and feature extraction pipeline are therefore treated as part of the trusted computing base. If administrative control of a device is lost, identity integrity cannot be guaranteed, and the device should be treated as untrusted (e.g., quarantined by policy).

This threat model explicitly isolates identity claim forgery from measurement-chain compromise and feature space adversarial manipulation, which constitute stronger attacker capabilities and are treated as separate problem classes.

Attacks that directly alter PMU counters, inject synthetic timing traces, or compromise the trusted measurement path fall outside the scope of this work and require separate defensive mechanisms.

4. Proposed Framework

DB^2 is a behavioural identity and assurance framework that unifies closed-set identification, open-set rejection, runtime stability assessment, and structured threat evaluation using CPU–RTC timing interaction and PMU microarchitectural events exposed by standard performance monitoring interfaces.

DB^2 derives identity from inherent microarchitectural variability arising from manufacturing tolerances, which introduce subtle yet repeatable differences in oscillator drift, execution timing, and cache behaviour. From these effects, the framework captures CPU cycle deviation relative to an independent RTC reference, together with instruction counts and L1/L2 cache-miss behaviour, to form a behavioural fingerprint anchored in physical execution behaviour rather than copyable identifiers. This fingerprinting layer is complemented by stability analysis across temperature variation, reboot sessions, and core placement, which underpins the Dynamic-Aware Identification and Risk (DAIR) scoring mechanism.

At a high level, DB² operates in two operational phases: fingerprint establishment and behavioural assurance. During fingerprint establishment, a lightweight on-device agent collects CPU–RTC timing and PMU measurements under controlled yet realistic runtime conditions. A signal-processing pipeline converts raw traces into structured feature representations and trains a closed-set classifier that models each enrolled device's behavioural signature. During behavioural assurance, newly observed traces are evaluated within the learned embedding space to (i) verify closed-set identity consistency, (ii) reject unseen devices through open-set detection, (iii) quantify behavioural stability across temperature, reboots, and cores using the DAIR mechanism, and (iv) assess identity-manipulation scenarios such as spoofing, Sybil, and cloning.

As illustrated in Fig. 1, these two phases are implemented within a client–server architecture. During the fingerprint establishment phase (enrolment), devices generate behavioural traces that are aggregated on a central server to learn device specific behavioural models. During behavioural assurance, new traces are processed through the same pipeline to (i) verify closed set identity consistency, (ii) reject non enrolled behaviour, and (iii) quantify runtime stability using DAIR.

For experimental reproducibility, devices run the feature-collection agent with sufficient privileges to access PMU counters and RTC timing. Collected measurements are transmitted to a central server for aggregation, training, and evaluation. Transport security (e.g., SSH/TLS) is treated as an implementation requirement rather than a core contribution.

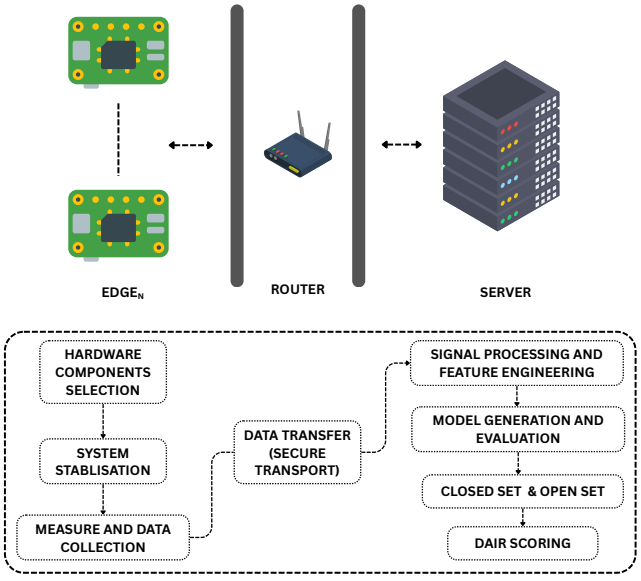

Figure 1: Overview of the DB^2 behavioural identity framework.

4.1. Fingerprint Establishment

The fingerprint establishment phase corresponds to the first stage illustrated in Fig. 1. It defines the hardware components, stabilisation controls, and measurement procedures used to construct each device's behavioural blueprint. The following subsections describe these elements in detail.

4.1.1. Hardware Components Selection

The first design decision is to identify on board components whose intrinsic physical variation can be observed without additional hardware. Microscopic manufacturing tolerances introduce small but measurable differences in oscillator behaviour and execution timing. Although negligible for functional correctness, these differences influence cycle counts, instruction timing, and memory access behaviour, providing a stable source of device-level variability.

Reliable extraction of such variability requires comparison between independent timing sources. A single oscillator cannot reveal its own drift without an external reference. DB^2 therefore leverages two physically distinct clock domains: the CPU core clock and the Real-Time Clock (RTC). The CPU clock governs instruction execution and microarchitectural activity, while the RTC is driven by a separate crystal source. Cross-comparison of these oscillators reveals device-specific timing deviations rooted in hardware tolerances.

This choice directly addresses the availability limitation (LM1) observed in prior work. Unlike GPU, RF, or sensor based approaches, both CPU performance counters and a hardware backed Real-Time Clock (RTC) are present on many modern single-board and embedded platforms, although access mechanisms and privilege requirements may vary across operating systems.

Although experimental validation is performed on Raspberry Pi single-board computers, the method relies solely

on standard CPU performance counters and an RTC source, both of which are commonly available on modern ARM-based and embedded processors. No additional accelerators, sensors, or external hardware components are required. This makes the approach portable across platforms that expose standard performance-monitoring interfaces.

By measuring CPU execution behaviour relative to an independent RTC reference, DB^2 isolates cross-oscillator timing deviations and evaluates their stability across reboot sessions, temperature variation, and core placement. In our implementation, the RTC reference is accessed through `/dev/rtc0`, ensuring that timing measurements are anchored to a hardware crystal distinct from the CPU core clock domain. This oscillator level comparison serves as the physical anchor for the behavioural fingerprint.

4.1.2. System Stabilisation

After selecting the hardware components to be monitored, the system is configured to improve measurement repeatability during fingerprint acquisition. This stabilisation phase reduces variability introduced by background processes, dynamic frequency scaling, and scheduler interference, ensuring that collected traces primarily reflect intrinsic hardware behaviour rather than transient software noise.

During data collection, the CPU frequency is fixed to avoid non-deterministic timing variation caused by dynamic scaling mechanisms. In addition, dedicated cores are isolated for measurement to minimise interference from interrupts and unrelated workloads. These controls improve signal consistency without altering the device's underlying physical characteristics. Prior work [32, 27] similarly reports that execution-context control and frequency stabilisation enhance reproducibility in hardware-based identification, directly addressing limitations related to repeatability and multicore variability (LM2, LM3).

Importantly, stabilisation does not eliminate operational variation. Temperature drift and reboot effects remain present and are explicitly evaluated. Cross-core variability is assessed by repeating the measurement procedure on each core under pinned affinity, rather than by allowing uncontrolled scheduler migration. Rather than suppressing these factors, DB^2 measures their impact and later quantifies behavioural persistence through the DAIR scoring framework.

4.1.3. Measurement and Data Collection

Once stability conditions are established, the framework measures the joint behaviour of the CPU and RTC using a fixed duration measurement routine that operates under normal execution conditions. A lightweight C-based measurement agent, deployed through an API-driven script, configures the environment by enabling PMU registers, fixing the CPU frequency, and isolating the target core. It then uses direct register access to monitor four primary PMU events: CPU cycles, retired instructions, L1 cache misses, and L2 cache misses, while the RTC provides an external oscillator reference for timing.

Each measurement session runs within a fixed 120-second RTC window. This duration was selected to ensure sufficient integration time for stable estimation of cross-oscillator deviation while keeping collection overhead acceptable for resource-constrained edge devices. Shorter windows produced higher dispersion in preliminary testing, whereas substantially longer windows increased acquisition time without measurable improvement in separability. For every core, the agent performs 1,000 iterations per reboot, computing deltas for cycles, instructions, and cache-miss events between the start and end of each window, and recording temperature, timestamp, reboot index, and device type. This process is repeated across all four cores and over 10 reboots to analyse reboot persistence and cross core variability without injecting synthetic stress workloads or intrusive instrumentation.

Ten reboot cycles provide sufficient statistical coverage of cold-start variability, oscillator settling effects, and kernel reinitialisation noise to evaluate behavioural persistence. Empirical variance analysis showed diminishing marginal changes in inter reboot dispersion beyond approximately eight cycles, indicating that ten reboots provide stable persistence estimation without unnecessary data redundancy. Increasing the number of reboots did not materially change stability metrics but significantly increased collection time.

Let R denote the total number of reboot cycles, and N the number of measurement iterations performed per core within each reboot. Let $c \in \{0, 1, 2, 3\}$ denote the CPU core index and $r \in \{1, \ldots, R\}$ the reboot index. Let $i \in \{1, \ldots, N\}$ denote the iteration index within a reboot.

For each measurement window, let ΔC denote elapsed CPU cycles, ΔI elapsed retired instructions, ΔM_{L1} elapsed L1 cache misses, and ΔM_{L2} elapsed L2 cache misses. Let $Temp$ denote the recorded CPU temperature and ts the measurement timestamp. The collected records are aggregated into a dataset $\mathcal{R}$ for secure transmission and subsequent processing.

Algorithm 1 outlines the full collection procedure. For each core and iteration, PMU counters are sampled at the start and end of the RTC window, and elapsed values for CPU cycles, instructions, and cache-miss events are computed to capture execution behaviour. After each iteration, PMU counters are reset to avoid cumulative accumulation and potential overflow, ensuring that each record represents an independent 120-second measurement window. Each record is annotated with the device's MAC address, core ID, temperature, timestamp, reboot index, and device type before being securely transmitted to the server for aggregation and analysis.

Let f_{CPU} denote the fixed CPU frequency during stabilised execution, and let T_{RTC} denote the duration of the RTC-referenced measurement window. Let ΔC represent the observed cycle accumulation during the window, and let $E[C]$ denote the nominal expected cycle count under ideal fixed-frequency operation.

Algorithm 1: On-Device Data Acquisition

Input: R (reboots), N (iterations per core), $Cores$ (e.g., $\{0, 1, 2, 3\}$), T_{RTC} (e.g., 120s), f_{CPU} (fixed), $\mathcal{E} = \{\text{Cycles, Instr., L1-Miss, L2-Miss}\}$
Output: Batch $\mathcal{R}$ of records $\langle \texttt{MAC}, i, c, \Delta C, \Delta I, \Delta M_{L1}, \Delta M_{L2}, E[C], D, Temp, ts, Type, r \rangle$

```
MAC ← get_mac_address();  Type ← get_device_type();
for r ← 1 to R do
    foreach c ∈ Cores do
        sched_setaffinity(c);
        lock CPU frequency to f_CPU and enable PMU E on core c;
        open /dev/rtc0;
        for i ← 1 to N do
            if i=1 then
                wait for next RTC tick;
            C_0 ← read_pmccntr_el0(); I_0 ← read_pmevcntr0_el0();
              M_L1,0 ← read_pmevcntr1_el0(); M_L2,0 ← read_pmevcntr2_el0();
            t_0 ← read_rtc();
            while read_rtc() − t_0 < T_RTC do
                No-Operation Loop (active polling)
            ;
            C_1 ← read_pmccntr_el0(); I_1 ← read_pmevcntr0_el0();
              M_L1,1 ← read_pmevcntr1_el0(); M_L2,1 ← read_pmevcntr2_el0();
            Temp ← read_cpu_temperature();  ts ← get_current_timestamp();
            ΔC ← C_1 − C_0;  ΔI ← I_1 − I_0;  ΔM_L1 ← M_L1,1 − M_L1,0;
              ΔM_L2 ← M_L2,1 − M_L2,0;
            E[C] ← f_CPU · T_RTC;  D ← ΔC − E[C];
            append
              ⟨MAC, i, c, ΔC, ΔI, ΔM_L1, ΔM_L2, E[C], D, Temp, ts, Type, r⟩ to
              R;
            reset PMU counters to zero for next iteration;
        close /dev/rtc0;
    Secure export:;
    transmit R to server (SSH/TLS);
    R ← ∅;
    if r < R then
        RebootDevice()
```

The expected cycle count $E[C] = f_{\text{CPU}} \cdot T_{\text{RTC}}$ represents the nominal cycle accumulation under ideal fixed-frequency execution. The deviation term $D = \Delta C - E[C]$ therefore captures the aggregate effect of oscillator tolerances, interrupt overhead, and residual timing variability under stabilised conditions. Importantly, D is not interpreted as pure clock drift; instead, it constitutes one component of a broader behavioural vector that also incorporates instruction counts and cache-miss activity. Together, these PMU-derived signals characterise device-specific execution behaviour relative to an independent RTC reference.

This methodology extends earlier oscillator-comparison approaches by integrating fine-grained PMU instrumentation with RTC-referenced timing on modern single-board computers. Consistent with [27, 32], fixed timing windows and core isolation improve repeatability and reduce noise. In the prototype implementation, traces are transmitted using standard secure transport mechanisms (e.g., SSH/TLS). Transport security is treated as an implementation requirement rather than a core contribution of the framework. Although the current implementation runs directly on the OS, the design is compatible with deployment inside a Trusted Execution Environment (TEE). Such environments can further protect the fingerprinting logic and measurement routines from compromise, thereby strengthening the framework's security posture in real-world installations.

While stabilisation reduces noise, residual variation from temperature drift, core migration, and reboot differences cannot be eliminated. These variations represent intrinsic device behaviour and are quantified within the DAIR scoring framework to assess long term stability. In deployment, the full stabilisation profile used in our experiments is not strictly required; lightweight CPU affinity pinning and periodic calibration are sufficient to maintain a usable behavioural signal on typical edge platforms.

Although the final representation includes 636 derived statistical features per observation, these are computed from a small set of primary physical signals (cycles, instructions, L1/L2 misses, and deviation). The feature expansion captures complementary temporal and statistical views of the same underlying oscillator-coupled execution behaviour.

The 636 features result from computing $|S_f|$ short window and $|S_h|$ long-window statistics across the base channel set $\mathcal{B}$ for each rolling window size in $W_f \cup W_h$, combined with selected interaction ratios. Dimensionality is reduced by removing redundant maxima features before residualisation and centring.

4.1.4. Signal Processing and Feature Engineering

After data acquisition, the behavioural traces undergo a signal processing pipeline that converts raw PMU measurements into consistent and discriminative features. This stage suppresses transient measurement artefacts and scale inconsistencies while preserving the intrinsic cross oscillator deviation structure that underpins device separability. The resulting feature matrices, produced as shown in Algorithm 2, provide compact statistical representations that capture distinctive and persistent device characteristics and form the foundation for subsequent identification and risk-aware analysis.

Importantly, the objective of this stage is not to introduce new behavioural signals, but to stabilise and characterise the intrinsic CPU–RTC deviation structure across temporal scales.

Each record contains the core behavioural metrics: elapsed cycles, instructions, L1/L2 cache misses, temperature, and reboot index. Reboot indices are assigned sequentially per device at collection time and define the temporal axis used for partitioning and stability analysis. Before feature extraction, all records are validated to ensure the presence of required identifiers (MAC address, reboot index, core ID, and iteration). Incomplete or duplicate entries are removed. The remaining data are grouped by device, reboot, and core, and sorted by iteration to preserve temporal ordering. Features that exhibit no variation or near-constant behaviour are removed, and the same feature set is enforced across FIT, CAL, and TEST to maintain consistency and prevent leakage.

For each measurement, statistical descriptors are computed across window lengths of 10–200 samples. Short windows capture micro-level timing dispersion, while longer windows capture low-frequency drift and thermal effects. We use two window families: shorter windows (W_f = 10–100 samples) for local dispersion analysis, and longer windows (W_h = 60–200 samples) for low-frequency drift and stability assessment. The computed statistics include the

Algorithm 2: Post-Collection Processing

Input: Raw PMU tuples (`MAC`, `Reboot`, `Core`, `Iteration`, $\Delta C, \Delta I, \Delta M_{L1}, \Delta M_{L2}, Temp, ts$)
Output: Corrected feature matrices for FIT (1–6), CAL (7–8), TEST (9–10)

Definitions: $W_f \leftarrow \{10, 20, \ldots, 100\}$
$W_h \leftarrow \{60, 80, 100, 120, 140, 160, 180, 200\}$
$S_f \leftarrow \{\text{mean}, \text{min}, \text{max}, \text{median}, \text{sum}, \text{std}\}$,
$S_h \leftarrow \{\text{skew}, \text{kurt}, \text{q10}, \text{q25}, \text{q75}, \text{q90}, \text{iqr}\}$;
$T^\star \leftarrow$ `TempC__roll100__median` (if available);

1. Normalise & deduplicate
Rename MAC ← `MAC Address`; ensure `Reboot`, `Core`, `Iteration`; derive `TempC` if needed;
Drop duplicates by (`MAC Address`, `Reboot`, `Core`, `Iteration`); check stream length ≈ 1000;
2. Select base channels
$\mathcal{B} \leftarrow$ numeric columns common to all splits; prefer `TempC`; remove constants/near-constants; cast to `float32`;
3. Rolling-window features (per stream)
foreach $g \in$ groups(`MAC Address`, `Reboot`, `Core`) **do**
  sort g by `Iteration`;
  foreach $b \in \mathcal{B}$ **do**
    foreach $w \in W_f$ **do**
      append $b\_\_roll w\_\_s$ for all $s \in S_f$
    foreach $w \in W_h$ **do**
      append $b\_\_roll w\_\_s$ for all $s \in S_h$
  align rows to window end `IterEnd` and keep IDs;
concatenate all streams into matrix X;
4. Split (leak-free) FIT ← {`Reboot` ∈ {1, …, 6}}, CAL ← {`Reboot` ∈ {7, 8}}, TEST ← {`Reboot` ∈ {9, 10}};
5. v3 corrections
(a) Drop maxima: remove all columns with suffix `__max`;
(b) Temperature residuals (FIT-only fit): **foreach** *feature y in X not prefixed by* `TempC` **do**
  estimate a pooled slope β_y using FIT only;;
  estimate a per-core intercept $b_{y,c}$ for each `Core` c using FIT only;;
  set $r \leftarrow y - (\beta_y T^\star + b_{y,c})$ for FIT, CAL, and TEST;;
keep `TempC` features unchanged ($r \leftarrow y$);
(c) Per-device centering: **foreach** (`MAC`, `Core`) **do**
  compute $\tilde{y} \leftarrow \text{median}_{\text{FIT}}(r)$ elementwise; if missing, use the per-core FIT median;
  set $z \leftarrow r/\tilde{y} - 1$ (safe denom: if $\tilde{y} \notin \mathbb{R}$ or 0, use 1);
replace features by z in FIT, CAL, TEST;
6. Output emit corrected matrices with IDs (`MAC Address`, `Reboot`, `Core`, `IterEnd`) and all corrected features;

mean, minimum, median, sum, and standard deviation for W_f, and skewness, kurtosis, quantiles (q10, q25, q75, q90), and the interquartile range for W_h. These descriptors capture short-term dispersion and longer-term drift while remaining consistent across devices and reboots.

Temperature residualisation reduces direct linear temperature scaling in feature space, while thermal stability within the DAIR framework is evaluated on calibrated identity confidence under temperature variation, ensuring that behavioural persistence is assessed at the decision level rather than through raw feature correlation. Each feature vector is tagged with the device MAC address, core ID, and reboot index to maintain traceability.

To ensure reproducible evaluation, the processed data are partitioned into three non overlapping subsets: FIT (reboots 1–6) for model training, CAL (reboots 7–8) for calibration and threshold tuning, and TEST (reboots 9–10) for final validation. This temporal separation prevents information leakage across stages and ensures that each model is evaluated on previously unseen reboot conditions. All temperature regression parameters are estimated exclusively on the FIT partition and then applied unchanged to CAL and TEST to prevent distributional leakage. Before training, spike-prone maxima are removed to reduce the influence of transient bursts, and all features are centred relative to the FIT median to maintain intra-device coherence across reboots without masking thermal or cross core drift. The resulting matrices represent each device's behavioural blueprint and serve as input to the closed-set model training stage within the fingerprint establishment phase.

4.1.5. Closed-Set Fingerprinting

Closed set fingerprinting is formulated as a multi class classification task in which each enrolled device corresponds to a class label. Following feature engineering, the refined behavioural vectors are standardised using parameters fitted exclusively on the FIT partition (StandardScaler) and then projected via Principal Component Analysis (PCA) to reduce dimensionality while preserving the dominant variance structure. The learned scaler and PCA transformation are applied unchanged to CAL and TEST to prevent information leakage. Before scaling, features are centred using a per-device-per-core mean estimated exclusively on FIT and applied unchanged to CAL and TEST, ensuring consistent normalisation without cross partition leakage.

Several machine-learning models are evaluated, including Extra Trees (ET), Quadratic Discriminant Analysis (QDA), Random Forest, Multi-Layer Perceptron (MLP), and Support Vector Machines with an RBF kernel. All models are trained on FIT (reboots 1–6). Where required, hyperparameters are selected using CAL (reboots 7–8) based on Macro-F1 performance. For the ET reference model, fixed hyperparameters are used as reported in Table 4. The final configuration is fixed before evaluation on TEST (reboots 9–10), which contains behavioural traces from unseen reboot sessions.

Among the evaluated models, Extra Trees (ET) is selected as the primary reference due to its robustness to high dimensional feature spaces, stable performance across reboot separated splits, and favourable calibration behaviour.

The resulting closed-set package comprises the fitted scaler, the PCA transformation, and the trained classifier. During deployment, new behavioural traces are processed through the identical transformation pipeline before classification. This closed-set layer provides the identity baseline for open-set rejection and DAIR-based behavioural risk assessment.

4.2. Behavioural Assurance

4.2.1. Open-Set Identification of Unknown Devices

Closed-set classifiers assume that all possible devices are enrolled during training. In practical deployments, however, traces may originate from previously unseen devices, newly introduced nodes, or spoofed identities. A closed-set model is forced to assign such traces to one of the enrolled classes, potentially yielding confident but incorrect matches. To address this limitation, DB^2 integrates an open-set rejection layer that evaluates whether a trace is consistent with the enrolled device population.

The complete procedure is summarised in Algorithm 3.

Algorithm 3: Open-Set Identification under Strict Leave-One-Device-Out (LOO)

Input: $\mathcal{D}_{FIT}, \mathcal{D}_{CAL}, \mathcal{D}_{TEST}$ with identifiers (m, c, r, t); target known-event reject rate ρ; K; smoothing window w_s; voting (ν, N_c, w_v)
Output: Event-level decisions on $\mathcal{D}_{TEST}$; FPR_{known}, $\text{TPR}_{unknown}$, Kept_{known}

1. **Strict LOO.** Select held-out device $d^\star$. Remove $d^\star$ from $\mathcal{D}_{FIT}$ and $\mathcal{D}_{CAL}$.
2. **Embedding (FIT-only).** Fit scaler and PCA on $\mathcal{D}_{FIT}$. Transform all partitions and obtain embeddings $\mathbf{z}_i$ and whitened $\mathbf{z}_i^w$.
3. **Closed-set backbone.** Train classifier on $\mathbf{z}_i$ (enrolled devices only). Compute class means and fit KNN model in whitened space.
4. **Novelty indicators.** For each sample compute: nearest-centroid distance, KNN distance, PCA reconstruction error, entropy, probability-margin uncertainty, top-k gap, cosine gap.
5. **Rejection head (CAL only).** Normalise indicators using CAL percentiles. Construct pseudo-unknown samples from CAL (upper-tail push). Train logistic rejection head to obtain novelty score u_i.
6. **Smoothing and calibration (CAL only).** Apply temporal smoothing over (m, c, r). Fit ECDF on CAL and map scores to $\hat{u}_i \in [0, 1]$.
7. **Operating point (CAL only).** Select quantile $q^\star$ such that, after per-core thresholding and ν-of-N_c voting with window w_v, the CAL known-event reject rate equals ρ.
8. **TEST evaluation.** Freeze $q^\star$ and apply identical thresholding and voting to $\mathcal{D}_{TEST}$. Report FPR_{known}, $\text{TPR}_{unknown}$, and Kept_{known}.

Let $\mathcal{D}_{FIT}$, $\mathcal{D}_{CAL}$, and $\mathcal{D}_{TEST}$ denote the training, calibration, and evaluation partitions, respectively. Each sample is represented by a feature vector $\mathbf{x}_i$ and identifiers (m_i, c_i, r_i, t_i) corresponding to MAC address, CPU core, reboot index, and event index.

The embedding function obtained from scaling and PCA is denoted by $\phi(\cdot)$, producing embeddings $\mathbf{z}_i = \phi(\mathbf{x}_i)$. The closed-set classifier is denoted by $\mathcal{C}$, and the rejection head by $\mathcal{G}$.

Strict leave-one-device-out protocol. Open-set evaluation follows a strict leave-one-device-out (LOO) procedure. For each fold, one device $d^\star$ is removed from $\mathcal{D}_{FIT}$ and $\mathcal{D}_{CAL}$. All model components scaler, PCA (including whitening), classifier, KNN model, rejection head, and ECDF calibration are refit using only the remaining enrolled devices. The held-out device appears only in $\mathcal{D}_{TEST}$ and is treated as unknown during evaluation.

Embedding and closed-set backbone. A scaler and PCA are fitted on $\mathcal{D}_{FIT}$ and applied to all partitions. This produces an embedding space in which each enrolled device forms a compact cluster. An Extra Trees classifier is trained on the embeddings of enrolled devices. In parallel, class means and a K-nearest neighbour (KNN) model are constructed in the whitened embedding space.

Novelty indicators. To determine whether a trace is consistent with enrolled behaviour, a compact set of indicators is computed:

(i) distance to the nearest class mean, (ii) average KNN distance, (iii) PCA reconstruction error, (iv) classifier entropy, (v) probability-margin uncertainty, (vi) top-k probability gap, and (vii) cosine gap to the nearest class mean.

These indicators capture both geometric deviation in the embedding space and the classifier's predictive uncertainty.

Rejection head and calibration. Each indicator is normalised using percentile bounds computed exclusively from $\mathcal{D}_{CAL}$. Because true unknown samples are unavailable during training, conservative pseudo-unknown examples are constructed from $\mathcal{D}_{CAL}$ by pushing indicator values toward their upper tails. A logistic regression model $\mathcal{G}$ is trained to produce a scalar novelty score u_i.

Novelty scores are temporally smoothed within each (m, c, r) stream using a fixed window. An empirical cumulative distribution function (ECDF), fitted only on smoothed CAL scores, maps u_i to calibrated novelty values $\hat{u}_i \in [0, 1]$. The learned mapping is applied unchanged to $\mathcal{D}_{TEST}$.

Operating-point selection. The rejection threshold is selected using $\mathcal{D}_{CAL}$ only. A quantile $q^\star$ is chosen such that, after per-core thresholding and ν-of-N_c voting with temporal window w_v, the event-level known-device reject rate equals the predefined operating point ρ.

Here, the known-device reject rate represents the fraction of enrolled-device events incorrectly rejected.

Event-level decision. For evaluation, per-row threshold exceedances are aggregated at the event level by counting exceedances across cores for each (m, r, t) and applying the fixed ν-of-N_c rule. A temporal window of length w_v is then applied over consecutive events within each (m, r) stream.

Using the frozen threshold $q^\star$, the final TEST metrics are computed:

- FPR_{known}: fraction of known-device TEST events rejected,
- $\text{TPR}_{unknown}$: fraction of held-out-device TEST events rejected,
- Kept_{known}: fraction of known-device TEST events accepted.

Within DB², this calibrated rejection layer defines a behavioural boundary that separates enrolled identity clusters from structurally inconsistent behaviour under strict retraining conditions.

4.2.2. The DAIR Scoring Framework

The open-set module rejects behaviour that does not match the enrolled population, but it does not quantify how

stable an enrolled device remains over time. To address this, the *Dynamic-Aware Identification and Risk (DAIR)* framework evaluates behavioural stability across temperature variation, reboot sessions, and CPU-core execution. DAIR operates on calibrated closed-set predictions and produces per-device stability scores together with a composite risk value.

For each behavioural record i, the closed-set model provides a predicted label $\hat{y}_i$ and a calibrated confidence $\tilde{p}_i \in [0, 1]$ obtained using isotonic regression on the CAL partition. Contextual metadata includes temperature T_i, reboot index r_i, and core index c_i. From these signals, DAIR computes four bounded stability indicators in $[0, 1]$: identification consistency (S_{true}), thermal stability (S_{temp}), reboot stability (S_{reboot}), and core stability (S_{core}). Higher values indicate stronger behavioural stability.

1) Identification consistency (S_{true}).

Identification consistency captures both prediction correctness and confidence reliability. For each device, the median calibrated confidence assigned to the true class is computed. In addition, high-confidence misclassifications are identified using a combined confidence score derived from calibrated top-1 probability and decision margin. Misclassifications that exceed a predefined confidence threshold are treated as overconfident errors.

S_{true} increases when correct predictions have high calibrated confidence and decreases when confident misclassifications occur. This ensures that both accuracy and confidence calibration contribute to identity stability.

2) Thermal stability (S_{temp}).

Thermal stability evaluates how sensitive calibrated identity confidence is to temperature variation. Two complementary statistics are computed per device: (i) the absolute Spearman rank correlation between calibrated confidence and temperature, and (ii) the magnitude of the linear regression slope of confidence with respect to temperature over the observed temperature range.

These measures are mapped to the range $[0, 1]$ using bounded inverse transforms, producing a temperature stability score where values close to 1 indicate low temperature sensitivity. Complementary feature-level analysis indicates that elevated temperature induces variance inflation in high order microarchitectural statistics, particularly L2 cache-miss quantiles and skew based metrics. Although linear temperature residualisation reduces mean drift, variance expansion in the hot regime leads to reduced classifier confidence stability. This effect is captured by S_{temp}, which decreases when calibrated confidence becomes temperature sensitive.

3) Reboot stability (S_{reboot}).

Reboot stability measures behavioural persistence across reboot sessions. For each reboot, classification accuracy and median calibrated confidence are computed. Variability across reboots is quantified using the standard deviation of these metrics. In addition, distributional shift is measured by computing the Jensen–Shannon divergence between each reboot's predicted-label distribution and the device's overall prediction distribution.

Algorithm 4: DAIR: Dynamic-Aware Identification and Risk Scoring

Input: Per-event outputs for device d: $(\hat{y}_i, p_i, T_i, r_i, c_i)$
Output: $S_{\text{true}}, S_{\text{temp}}, S_{\text{reboot}}, S_{\text{core}}$, Risk($d$), Band($d$)

1) Confidence calibration
Fit isotonic regression on CAL partition
Apply calibration to obtain $\tilde{p}_i$ for all events of device d

2) Identification consistency
Compute median calibrated confidence assigned to the true class
Identify confident misclassifications using combined confidence and margin criteria
Aggregate into stability score $S_{\text{true}} \in [0, 1]$

3) Thermal stability
Compute Spearman correlation between $\tilde{p}_i$ and T_i
Estimate linear confidence–temperature slope over observed range
Map temperature sensitivity to bounded stability score S_{temp}

4) Reboot stability
For each reboot r, compute accuracy and median calibrated confidence
Compute standard deviation across reboots
Compute Jensen–Shannon divergence between per-reboot and overall prediction distributions
Aggregate dispersion into bounded stability score S_{reboot}

5) Core stability
Repeat reboot procedure using core index c
Aggregate dispersion into bounded stability score S_{core}

6) Composite risk and banding
Compute Risk(d) using weighted sum in (1)
Assign qualitative band using fixed quantile thresholds derived from the enrolled device risk distribution.

These dispersion statistics are mapped to a bounded instability measure and converted into a stability score in $[0, 1]$. Lower variability and smaller distributional divergence yield higher reboot stability.

4) Core stability (S_{core}).

Core stability is computed analogously to reboot stability, replacing reboot index with core index. Accuracy variability, confidence variability, and Jensen Shannon divergence of per core prediction distributions are aggregated into a bounded instability measure. The resulting score reflects whether the device's behavioural signature remains consistent across different CPU cores.

5) Composite risk.

The four stability indicators are combined into a composite risk value:

$$\begin{aligned}\text{Risk}(d) = W_{\text{true}}(1 - S_{\text{true}}) + W_{\text{temp}}(1 - S_{\text{temp}}) \\ + W_{\text{reboot}}(1 - S_{\text{reboot}}) + W_{\text{core}}(1 - S_{\text{core}}),\end{aligned} \tag{1}$$

where the weights sum to one. Identification consistency is prioritised in the weighting scheme, while the remaining terms capture environmental and execution-context stability.

For interpretability, risk values are mapped to four qualitative bands (*Low*, *Medium*, *High*, *Critical*) using empirical quantiles computed from the enrolled device risk distribution, and these are held fixed across the weight sensitivity analysis. These thresholds are then held fixed during TEST evaluation.

DAIR extends the pipeline from static identification to continuous behavioural assurance. Rather than reporting

classification accuracy alone, it decomposes stability across temperature, reboot cycles, and core transitions, producing interpretable indicators of identity persistence under realistic operational variation. This transforms DB^2 from a pure fingerprinting mechanism into a structured behavioural assurance framework.

5. Experimental Validation

This section presents the experimental evaluation of the DB^2 framework through a proof-of-concept deployment on a controlled edge device testbed. The objective is to quantify the accuracy of closed-set identification, the open-set rejection capability, and behavioural stability under realistic operational variation.

The evaluation proceeds in three stages. First, closed-set experiments assess per device recognition performance across reboot and core variations. Second, open-set experiments evaluate the framework's ability to reject previously unseen devices while maintaining a controlled false-positive rate. Third, the DAIR framework quantifies behavioural stability and assigns risk scores under both normal operation and controlled identity-manipulation scenarios.

5.1. Experimental Setup

Experiments were conducted on nine Raspberry Pi edge devices comprising Raspberry Pi 4 Model B (Rev 1.5) and Raspberry Pi 5 Model B (Rev 1.0). Pi 4 units use Arm Cortex–A72 quad-core processors fixed at 1.8 GHz with 4 GB RAM, while Pi 5 units use Arm Cortex–A76 quad-core processors fixed at 2.4 GHz with 4 GB RAM. All devices ran Debian GNU/Linux 12 (Bookworm, 64-bit) with kernel 6.6 to ensure consistent operating-system behaviour across the testbed.

To reduce uncontrolled variability in execution, dynamic frequency scaling was disabled (`force_turbo=1`) and the Performance Monitoring Unit (PMU) was enabled (`enable_pmu=1`). A hardware RTC (`ds1307`) provided an independent oscillator reference accessed via `/dev/rtc0`. Core and interrupt isolation parameters (`isolcpus`, `nohz_full`, `rcu_nocbs`, `irqaffinity`) were applied to isolate the active measurement core. All four cores (0–3) were profiled under pinned CPU affinity. Configuration integrity was verified after each reboot. Thermal throttling was mitigated using aluminium heatsinks and an actively controlled 5 V PWM fan. These controls reduce measurement noise during evaluation; deployment environments may relax some constraints at the cost of increased variance.

Data collection was automated using a lightweight C-based measurement agent orchestrated by a shell-level supervisor. At startup, the agent verified access to the PMU and hardware RTC. For each acquisition cycle, CPU affinity was fixed, a brief cache warm-up was performed, and a controlled measurement loop recorded elapsed cycles, retired instructions, L1/L2 cache misses, RTC timestamps, core identifiers, reboot indices, and temperature. Logs were written locally in CSV format and transferred to a central collector for aggregation.

Table 2
Statistical descriptors and feature counts.

Statistic	Functional Role	Count per Base
Mean (μ)	Average value within each window	10
Median ($\tilde{x}$)	Robust central tendency	10
Standard deviation (σ)	Dispersion / noise amplitude	10
Minimum (min)	Lower envelope behaviour	10
Interquartile range (IQR)	Robust spread, outlier-resistant	8
Quantiles ($q_{10}, q_{25}, q_{75}, q_{90}$)	Distribution spread at multiple percentiles	32
Skewness (γ_1)	Distribution asymmetry	8
Kurtosis (γ_2)	Distribution peakedness	8
Sum (Σ)	Aggregate magnitude within each window	10
	Per-base total	**106**

Table 3
Base metrics and window configuration.

Metric	Symbol	Behaviour Captured	Features
Elapsed cycles	f_{cyc}	Oscillator stability and timing drift	106
Instructions	f_{inst}	Pipeline efficiency and execution variation	106
L1 misses	f_{L1}	Memory-access latency and scheduling behaviour	106
L2 misses	f_{L2}	Memory-hierarchy dynamics	106
Deviation	f_{dev}	RTC–CPU drift under fixed f_{CPU}	106
Temperature	f_{temp}	Thermal operating state	106
		Total ($6 \times 106 = 636$)	**636**

Each reboot initiated a new block of 1,000 measurement iterations per core. This block size was selected based on empirical analysis in the present study, where performance variance stabilised beyond approximately 600–800 iterations. The procedure was repeated across ten reboots and four cores for all nine devices, yielding

$$1000 \times 10 \times 4 \times 9 = 360{,}000$$

behavioural records.

Each record included *MAC Address, Core, Reboot, IterEnd, Elapsed Cycles, Instructions, L1 Misses, L2 Misses, Temperature*, and *Timestamp*. All devices communicated over an isolated wireless network to prevent external traffic interference during data collection.

5.2. Dataset Description

The dataset consists of 360,000 behavioural records collected from nine devices across ten reboot sessions and four CPU cores. Each reboot block contains 1,000 measurement iterations per core.

After feature extraction, each observation is represented by a 636-dimensional statistical feature vector derived from six base PMU metrics: *Elapsed Cycles*, *Instructions*, *L1 Misses*, *L2 Misses*, *Deviation*, and *Temperature*.

The dataset is partitioned temporally into three non-overlapping subsets: FIT (reboots 1–6), CAL (reboots 7–8), and TEST (reboots 9–10). This separation ensures that evaluation is performed on previously unseen reboot sessions, preventing temporal leakage across training, calibration, and validation stages.

Tables 2 and 3 summarise the statistical descriptors and base metrics used to construct the 636-dimensional behavioural fingerprints.

5.3. Closed-Set Identification

Table 4 reports closed-set identification performance on the TEST partition (reboots 9–10).

The leading models (QDA, Extra Trees, and MLP) exceed 0.91 macro-F1 on unseen reboot sessions, with average

Table 4
Closed-set device fingerprinting performance across models.

Model	Key Hyperparameters	Test Accuracy	Test Macro-F1	Avg TPR	Avg FPR
ET	n_estimators = 800, max_features = sqrt, min_samples_leaf = 2	**0.9574**	**0.9575**	**0.9574**	**0.0053**
MLP	[512, 256], lr = 1e-3, epochs = 40	0.9572	0.9569	0.9572	0.0054
QDA	reg_param = 0.001	0.9424	0.9420	0.9424	0.0072
RF	n_estimators = 600, min_samples_leaf = 1	0.911	0.911	0.878	0.017
GB	HistGB, max_depth = 10, lr = 0.1, early_stopping = True	0.910	0.910	0.908	0.015
KNN	k = 3, weights = distance	0.812	0.810	0.847	0.021
SVC-RBF	C = 5, γ = auto	0.745	0.756	0.751	0.031
DT	min_samples_leaf = 5	0.667	0.667	0.672	0.049
GNB	var_smoothing = 1e-9	0.446	0.449	0.456	0.074
SGD	α = 0.001, loss = modified_huber	0.335	0.335	0.339	0.091
LDA	solver = svd	0.156	0.144	0.165	0.118
LinSVC	C = 0.5, dual = False	0.129	0.100	0.139	0.124
Ridge	α = 1.0	0.131	0.123	0.141	0.106
LogReg	C = 2, penalty = l2	0.111	0.022	0.111	0.500

false-positive rates below 1%. These results indicate strong device separability under reboot transitions and natural thermal variation. In contrast, linear models (LDA, Ridge, Logistic Regression, LinSVC) degrade substantially, revealing that the fingerprint space is inherently non-linear.

To assess statistical stability, 2000 stratified bootstrap resamples of the TEST partition were evaluated using the Extra Trees backbone. The 95% confidence interval for macro-F1 was [0.9263, 0.9304] (point estimate 0.9284), and for accuracy [0.9257, 0.9298] (point estimate 0.9277). The narrow intervals confirm that closed-set separability remains stable under resampling.

5.3.1. Model-wise Performance

A clear hierarchy emerges across model families. Extra Trees (ET) and MLP achieve the highest performance on unseen reboot sessions, with macro-F1≈0.957 and accuracy≈0.957. QDA follows at macro-F1≈0.942, while all remaining baselines fall substantially below this level. In particular, classical linear classifiers (LDA, Linear SVM, Ridge, Logistic Regression) degrade sharply in both macro-F1 and average true-positive rate.

This ranking reveals an important structural property of the fingerprint space. Device identities are not separable by simple linear decision boundaries. Instead, separability arises from higher-order interactions embedded within the engineered rolling-window descriptors. Ensemble models and neural networks are able to capture these interactions effectively, whereas global linear separators fail to model the underlying structure.

Although ET and MLP exhibit nearly identical closed-set accuracy, Extra Trees is selected as the backbone model for the subsequent open-set and DAIR stages. ET provides equivalent discriminative performance while offering stronger robustness to feature collinearity, reduced sensitivity to distributional shifts, and more stable class probability outputs for downstream calibration and out-of-distribution scoring. Its ensemble structure also facilitates interpretability and risk-aware extensions within the DB^2 framework.

Interestingly, the RBF-kernel SVM underperforms relative to tree-based ensembles, suggesting that global kernel similarity alone is insufficient to capture the structured statistical heterogeneity encoded in the engineered feature space.

5.3.2. Behavioural Consistency Across Devices

Figure 2 presents the row-normalised confusion matrix for the Extra Trees classifier evaluated on the TEST partition (reboots 9–10). The matrix exhibits a dominant diagonal structure, indicating that the majority of behavioural traces are correctly assigned to their originating device. Off-diagonal leakage is limited and concentrated between a small subset of architecturally similar devices, indicating localised pairwise overlap without evidence of cluster collapse or global identity confusion.

Importantly, the lowest per-device recall remains above 0.89, while most devices exceed 0.94. This confirms that the learned fingerprints remain distinguishable across independent reboot cycles and cross-core execution, demonstrating stable identity separability under unseen operating conditions.

Figure 3 further illustrates per-device true positive rates (TPR) for ET, QDA, and MLP. All three models maintain consistently high recall across devices, with ET and MLP exhibiting similarly uniform behaviour. Although minor degradation appears for a small number of device pairs, no single device dominates the error profile, indicating balanced separability across the enrolled population.

In this testbed, devices of the same model remain separable, indicating that the engineered features capture device specific variation rather than only model-level behaviour. This supports the hypothesis that the extracted fingerprint encodes fine-grained microarchitectural variation rather than superficial timing artefacts or model-specific bias. While minor performance degradation appears under specific device pairs or elevated operating conditions, no scenario results in widespread misclassification across devices, indicating that intrinsic microarchitectural variation remains detectable despite environmental noise within realistic bounds.

5.3.3. Robustness to Temperature, Core, and Reboot Variation

To evaluate operational robustness, TEST data were grouped by temperature range, CPU core index, and reboot session. Figure 4 reports the absolute macro-F1 scores per condition, while Figure 5 shows the deviation from the overall TEST macro-F1.

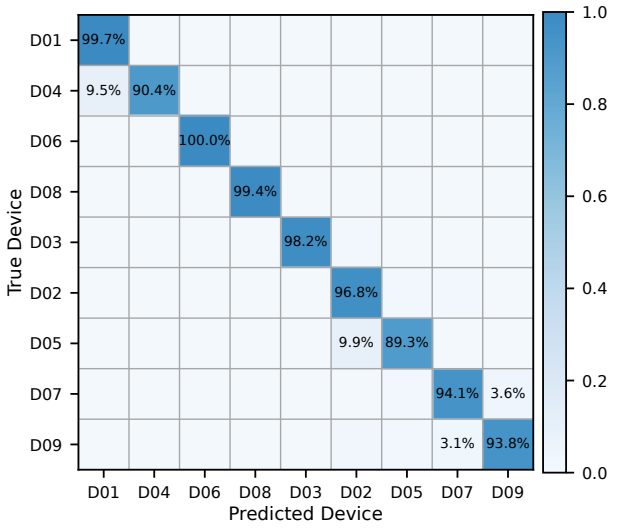

Figure 2: Row-normalised confusion matrix for ExtraTrees on TEST reboots 9–10. The dominant diagonal indicates strong device separability under held-out reboots.

Performance remains stable across reboot sessions (0.947 – 0.968 macro-F1), indicating that device identity persists after power cycling. Core-wise variation is limited, with macro-F1 ranging from 0.935 to 0.992, confirming that the fingerprint is not tied to a single execution core and remains discriminative across core placement.

Temperature introduces the largest variation. Macro-F1 reaches 0.944 in the 41–48°C range, decreases to 0.848 at ≤41°C, and drops to 0.583 above 48°C. The deviation plot shows a −0.375 difference relative to the overall TEST score at the highest temperature bin. This indicates that elevated thermal conditions increase behavioural variability and reduce classification margin, while device-specific patterns remain observable. Importantly, separability is reduced but not eliminated in the high-temperature regime. Confusion remains structured rather than random, and device-level centroids remain distinct in the representation space (Figure 6). The degradation, therefore, reflects margin compression under thermal stress rather than the collapse of the identity manifold.

5.3.4. Feature-Space Structure

To examine structural separability under realistic operating variation, we analyse inter-device centroid distances and sample-level margins in the TEST set (reboots 9–10). All measurements are computed directly in the PCA-transformed representation space used for inference.

Figure 6 shows the pairwise centroid distances between device classes. The matrix reveals structured proximity relationships. While many device pairs exhibit substantial separation, several pairs (e.g., D08–D07, D02–D07) show comparatively smaller centroid distances, indicating locally compact identity clusters. This indicates that identity separability is device-dependent and influenced by intrinsic hardware similarity rather than random variation.

Figure 7 presents the per-device margin distribution, defined as the ratio between the nearest competing centroid distance and the intra-class distance. Margin values near or above unity indicate that samples are closer to their own centroid than to the nearest competing centroid. Devices with median margins below one exhibit tighter inter-device proximity and reduced separation confidence. Margin distributions vary across devices, with some exhibiting strong, consistent separation (e.g., D08, D02), whereas others show reduced margins due to closer neighbouring centroids. However, variation across devices reflects differences in compactness and inter-device proximity.

These results indicate that device identity stability is strongly supported by measurable geometric structure in the representation space, rather than being solely an artefact of classifier decision boundaries. The structured separability observed here explains both the high closed-set accuracy and the device-dependent robustness patterns analysed in the following section.

5.3.5. Ablation Study

To quantify the contribution of individual feature groups in DB², we conduct a structured ablation study using an ExtraTrees classifier on the TEST split (reboots 9–10). In addition to overall Macro-F1, we report the mean classification margin as a geometric indicator of class separation, and thermal degradation defined as

$$\Delta_{\text{th}} = \left| F1_{\text{hot}} - F1_{\text{cold}} \right|$$

Temperature bins are derived from the measured on-device trace, where $F1_{\text{cold}}$ corresponds to $T \leq 41°\text{C}$ and $F1_{\text{hot}}$ to $T > 48°\text{C}$.

Table 5 summarises the results. The *No Deviation* configuration (530 features) achieves the highest Macro-F1 (0.966) and the lowest thermal degradation ($\Delta_{\text{th}} = 0.220$), indicating that deviation-based descriptors are not strictly required for maximising closed-set accuracy on this split.

The full DB² configuration (636 features) achieves a slightly lower Macro-F1 (0.957) with comparable thermal stability ($\Delta_{\text{th}} = 0.236$). The difference in Macro-F1 between the two configurations is approximately 0.9%, indicating that discriminative strength remains largely preserved when deviation features are included.

Importantly, the objective of DB² is not to optimise a single closed-set metric, but to provide a structurally complete behavioural representation suitable for downstream modules, including open-set rejection and risk-aware scoring. Deviation-based descriptors capture cross-clock drift and execution irregularities that may not maximise closed-set accuracy, but contribute to behavioural coverage and deployment consistency.

The *Timing-only* subset (212 features) achieves competitive Macro-F1 (0.931) but exhibits lower geometric margins

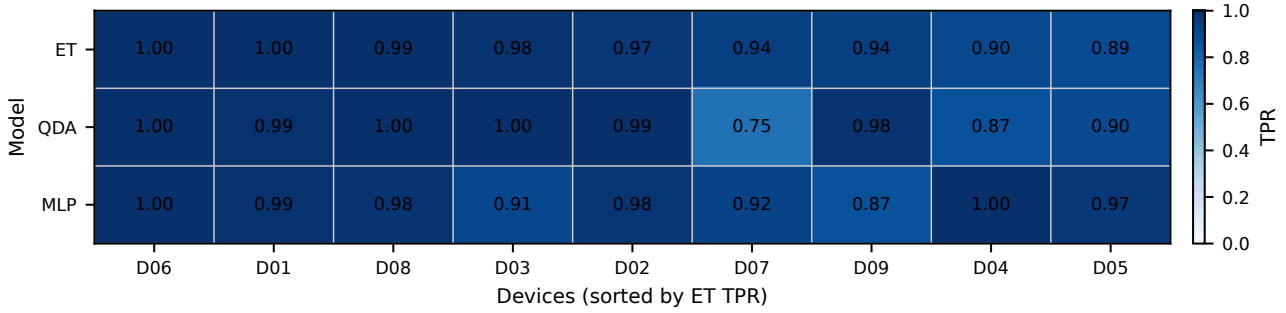

Figure 3: Per-device TPR comparison for ET, QDA, and MLP. Top models achieve consistently high reproducibility across all devices.

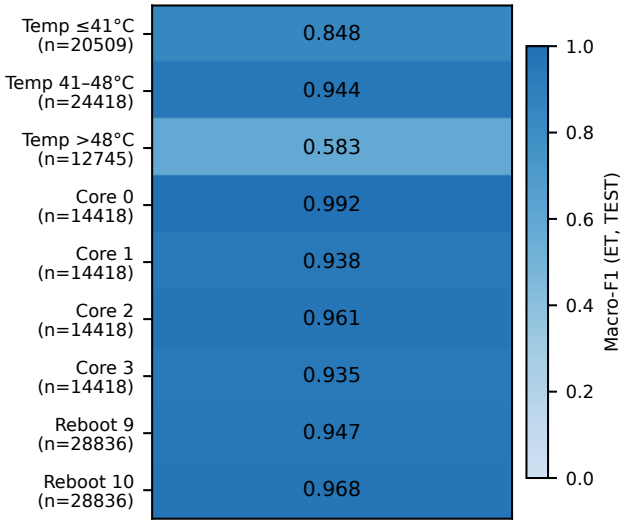

Figure 4: Condition-wise macro-F1 performance of the Extra Trees (ET) classifier on the TEST partition (reboots 9–10), grouped by temperature range, CPU core, and reboot index. Performance remains stable across reboots and cores. Elevated temperatures (>48°C) introduce substantial variation in classification performance. Results confirm that device fingerprints persist under realistic operational variation.

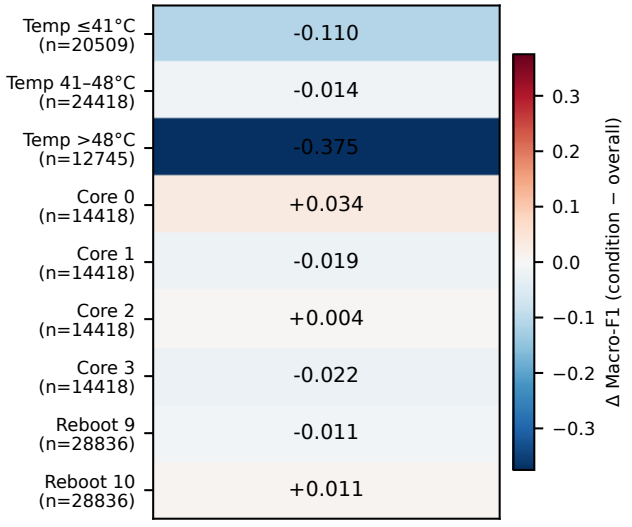

Figure 5: Deviation in macro-F1 relative to the overall TEST score (condition – overall) for the Extra Trees classifier. Most operating conditions show minor variation (within ±0.38), indicating stable behaviour across cores and reboots. The largest deviation occurs at high temperature (>48°C), reflecting thermal sensitivity of oscillator-based timing features without loss of identity separability.

(0.323), reflecting reduced inter-device separation. Cache-inclusive configurations substantially increase margin values (up to 0.600), indicating stronger geometric separation in representation space despite slightly lower classification accuracy.

Overall, the ablation results show that different feature groups influence discriminative strength, geometric structure, and environmental stability in distinct ways. While the *No Deviation* subset yields the highest closed-set accuracy on this split, the full DB^2 configuration is retained to preserve representational completeness, maintain compatibility across evaluation stages, and ensure consistent deployment behaviour under varied operating conditions.

Although deviation-based features do not maximise closed-set Macro-F1 on this split, their inclusion improves sensitivity to identity–behaviour misalignment under adversarial relabelling (Section 5.6). In particular, deviation descriptors contribute to confidence instability and margin compression under identity-forgery scenarios, supporting DAIR-based detection even when closed-set accuracy alone remains high. The full DB^2 configuration is therefore retained to preserve behavioural coverage rather than to optimise a single classification metric.

5.3.6. *Confidence, Reliability, and Cross-Model Consistency*

Closed-set identification is not only a question of accuracy, but also of confidence reliability. Because both the

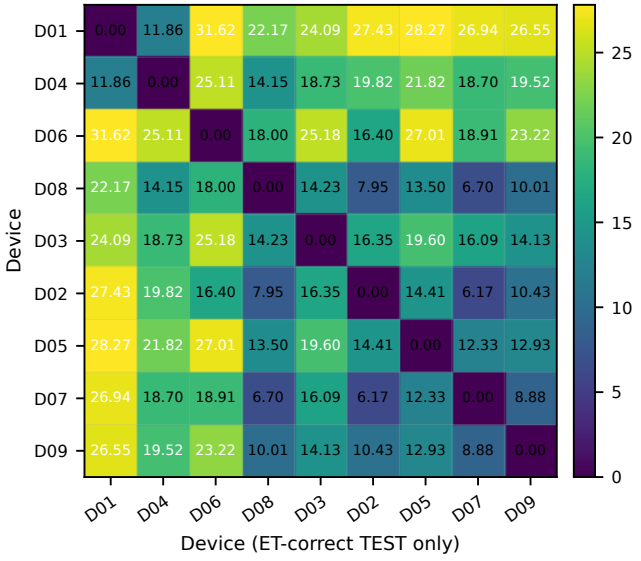

Figure 6: Pairwise centroid distances between device classes computed over the TEST set (reboots 9–10). Smaller distances indicate structurally closer identity manifolds, which explain device-dependent margin variation and the observed confusion behaviour.

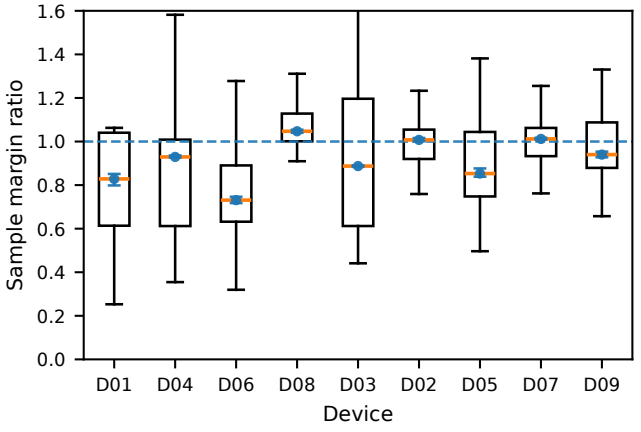

Figure 7: Per-device sample margin distribution on the TEST set. Margin values greater than one indicate that samples are closer to their own class centroid than to the nearest competing class centroid. Variability across devices reflects differences in structural compactness and inter-device proximity.

open-set rejection mechanism and the DAIR scoring framework depend on calibrated confidence estimates, we evaluate selective prediction behaviour using accuracy–coverage analysis.

Figure 8 presents the accuracy–coverage curves for the three strongest models: ExtraTrees (ET), QDA, and MLP. Accuracy–coverage analysis measures how classification accuracy increases as low-confidence predictions are progressively rejected. The area under the accuracy–coverage curve summarises selective classification behaviour and should not be confused with ROC-AUC. ET and MLP

Table 5
Ablation results on the TEST split (reboots 9–10) using ExtraTrees. Margin Mean reflects average classification margin. Thermal degradation is $\Delta_{th} = |F1_{hot} - F1_{cold}|$.

Configuration (features)	Macro-F1	Margin Mean	Δ_{th}
No Deviation (530)	**0.966**	0.446	**0.220**
Full DB² (636)	0.957	0.442	0.236
Timing Only (212)	0.931	0.323	0.237
Timing + Cache (424)	0.921	0.529	0.340
Cache Only (212)	0.918	**0.600**	0.297
No Temp (530)	0.894	0.540	0.295

exhibit near-monotonic accuracy improvement as coverage decreases, while QDA shows slightly lower accuracy at comparable coverage levels. This behaviour confirms that lower-confidence predictions are more error-prone, supporting selective rejection and downstream stability scoring. Such behaviour is essential for calibrated rejection thresholds in the open-set stage and for reliable stability scoring in DAIR.

Beyond confidence calibration, we examine structural consistency across model families. Figure 9 compares ET, QDA, and MLP in terms of mean accuracy, macro-F1, and worst-case device-level TPR and FPR. All three models exhibit closely aligned mean performance, and their worst-case FPR remains below 0.01. The consistency across ensemble (ET), probabilistic (QDA), and neural (MLP) classifiers indicates that separability is primarily driven by the engineered feature space rather than by a specific learning paradigm.

Together, these results complete the closed-set validation of DB². Feature-space analysis demonstrates structured device separability, the ablation study clarifies the role of individual feature groups, robustness analysis characterises sensitivity under thermal stress, and accuracy–coverage behaviour confirms that confidence estimates are statistically meaningful. These properties provide a stable foundation for the open-set rejection mechanism and the DAIR stability framework presented next.

5.4. Open-Set Identification of Unknown Devices

Open-set performance is evaluated under a strict leave-one-device-out (LOO) protocol on the TEST partition (reboots 9–10). In each fold, one device is removed entirely from FIT and CAL and treated as unknown. The full ET+OOD pipeline described in Section 4.2.1 is retrained using only the remaining eight devices.

All threshold selection is performed exclusively on CAL. After temporal smoothing and applying the fixed two-of-four (m=2, C=4) voting rule, a quantile threshold is selected on CAL such that the event-level false-reject rate of enrolled devices matches the predefined operating point (approximately 0.12). The selected threshold is then frozen and applied unchanged to TEST. TEST data are used solely for final evaluation and play no role in operating-point tuning.

Table 6 reports strict LOO results at this operating point. For each held-out device, the table shows the selected quantile q, the false-reject rate of enrolled devices on TEST

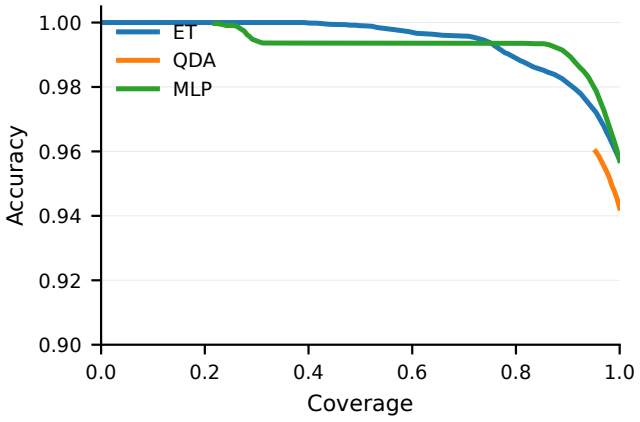

Figure 8: Accuracy–coverage curves for ET, QDA, and MLP on the TEST split (reboots 9–10). Coverage denotes the fraction of samples retained after rejecting low-confidence predictions. As coverage decreases, accuracy increases, confirming that calibrated confidence provides a meaningful ranking for selective prediction and downstream stability analysis.

(TEST-FPR), the rejection rate of the held-out device (TPR), and the known-device retention after smoothing and per-core voting (Kept). Here, TEST-FPR measures event-level false rejection of enrolled devices, TPR measures rejection of the held-out device treated as unknown, and Kept reports the acceptance rate for enrolled devices.

Figure 10 shows that around the selected operating point (FPR≈0.12), the trade-off curve flattens, indicating that further relaxing the threshold increases false rejects with only marginal improvement in unknown rejection.

The target operating point (FPR≈0.12) is chosen to strike a balance between unknown-rejection performance and enrolled-device retention. As shown in Figure 10, moving to stricter thresholds (e.g., FPR=0.05) reduces unknown rejection while increasing enrolled-device retention, whereas relaxing beyond 0.12 yields negligible additional TPR gain.

To quantify uncertainty without reusing correlated event-level samples across folds, each held-out device is treated as one independent unit. Global metrics are computed as averages across the nine folds, and 5000 bootstrap resamples over held-out devices are used to obtain 95% confidence intervals. Confidence intervals are computed over held-out devices rather than correlated event-level samples, ensuring statistically conservative uncertainty estimates.

At the target operating point (CAL-FPR≈0.12), defined prior to TEST evaluation, the global mean TPR is 0.9900 [0.9800, 0.9950], the global mean TEST-FPR is 0.1201 [0.1051, 0.1343], and the known-device retention is 0.8799 [0.8655, 0.8951]. Across folds, CAL-FPR remains tightly concentrated around the target value (0.11995 [0.11991, 0.12001]), confirming stable operating-point selection under strict LOO retraining.

Most folds achieve near-complete rejection when held out (TPR≈0.995). One fold (D04) exhibits reduced rejection (TPR=0.9501), indicating closer feature-space proximity to

Table 6
Strict leave-one-device-out (LOO) open-set performance on TEST (reboots 9–10). Thresholds are tuned on CAL to target event-level FPR≈0.12 under temporal smoothing and two-of-four core voting.

Held-out	q	TEST-FPR	TPR	Kept
D01	0.8753	0.1378	0.9950	0.8622
D04	0.8685	0.0858	0.9501	0.9142
D06	0.8663	0.1166	0.9950	0.8834
D08	0.8718	0.1304	0.9950	0.8696
D03	0.8761	0.1423	0.9950	0.8577
D02	0.8785	0.1197	0.9950	0.8803
D05	0.8738	0.1142	0.9950	0.8858
D07	0.8671	0.1522	0.9950	0.8478
D09	0.8710	0.0821	0.9950	0.9179
Mean	–	0.1201	0.9900	0.8799

the enrolled population under strict retraining. False-reject control and known-device retention remain stable in this case, suggesting that the variation reflects intrinsic geometric similarity rather than instability in the threshold.

Importantly, despite variation in TPR across folds, the CAL-based thresholding procedure maintains stable false-reject control and consistent known-device retention. The worst-case fold does not destabilise global performance, and confidence intervals remain tight, confirming repeatable and stable rejection behaviour under strict retraining conditions.

Minor variations in TEST-FPR across folds reflect distributional shifts between CAL and TEST under strict LOO retraining. Since threshold selection is performed exclusively on CAL, small deviations on TEST are expected and are absorbed within the reported confidence intervals.

5.5. Risk-Aware Analysis (DAIR Scoring Mechanism)

DAIR is evaluated on the TEST partition (reboots 9–10) to quantify behavioural stability of enrolled devices under operational variation. Stability indicators S_{true}, S_{temp}, S_{reboot}, and S_{core} are derived from the calibrated ET backbone described earlier.

The four indicators are aggregated into a composite risk score using fixed weights:

$$\begin{aligned}\text{Risk}(d) = {} & 0.55(1 - S_{\text{true}}) + 0.20(1 - S_{\text{temp}}) \\ & + 0.15(1 - S_{\text{reboot}}) + 0.10(1 - S_{\text{core}}).\end{aligned} \tag{2}$$

The dominant weight on S_{true} (0.55) reflects that identity consistency is the primary assurance signal. Temperature (0.20), reboot (0.15), and core (0.10) weights quantify contextual sensitivity. These weights were fixed prior to TEST evaluation and not tuned post hoc. Sensitivity analysis confirms that relative device ranking remains stable under alternative weight configurations. Risk values are mapped to Low, Medium, High, and Critical bands using empirical quantiles computed from the enrolled device risk distribution, which are held fixed across the sensitivity analysis. Bootstrap resampling of the enrolled device distribution confirms that band assignments remain stable under small

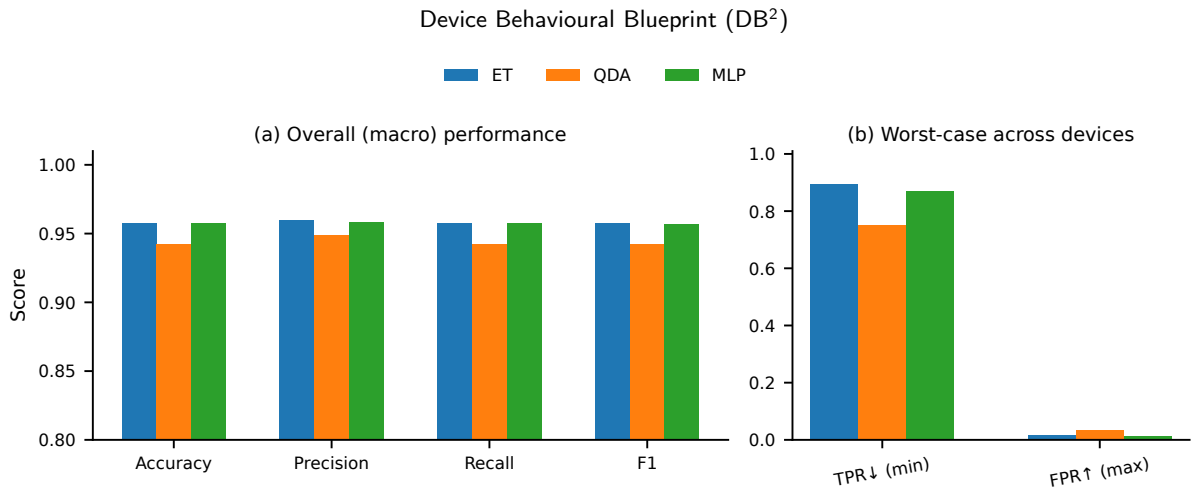

Figure 9: Performance comparison of the three strongest classifiers (ET, QDA, MLP). Bars show mean accuracy and macro-F1 across devices, together with worst-case device-level TPR and FPR. Close alignment across model families demonstrates that device separability is inherent to the feature space rather than model-specific.

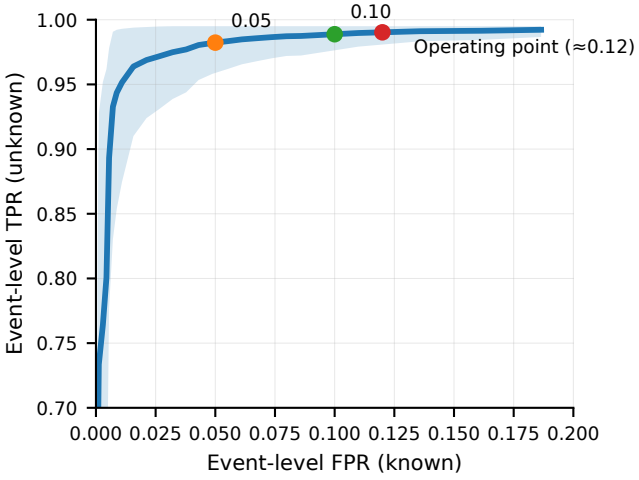

Figure 10: Event-level open-set operating trade-off under strict LOO retraining. Thresholds are selected exclusively on CAL and frozen for TEST. The shaded region indicates 95% confidence intervals across held-out devices.

perturbations of quantile thresholds, indicating that qualitative risk categorisation is not sensitive to minor numerical variation.

Table 7 summarises the DAIR outputs on TEST. S_{true} remains consistently high across devices (all above 0.97), confirming stable recognition under held-out reboots. In contrast, S_{temp} exhibits the largest variation (0.6609–0.9954), indicating device-specific thermal sensitivity. Both S_{reboot} and S_{core} remain close to one for most devices, confirming persistence across power cycles and core allocation.

Most devices fall into the *Low* band, confirming reproducible behaviour under nominal TEST variation. D03 is assigned to the Critical band, and D04 and D05 fall into the Medium band. In all three cases, elevated Risk is driven primarily by reduced thermal stability (S_{temp}), while S_{true} remains above 0.97. This behaviour is consistent with the observed increase in feature dispersion under high-temperature conditions. Devices exhibiting stronger variance inflation in cache-miss and deviation-based features correspondingly show greater reductions in thermal stability, confirming that DAIR captures environmental sensitivity rather than identity misclassification. This confirms that the increased Risk reflects environmental sensitivity rather than misidentification or identity collapse.

Table 7
DAIR summary on TEST (reboots 9–10). Higher S indicates stronger behavioural stability; lower Risk indicates higher reliability. Risk bands are derived from baseline quantiles (Low< 0.0616, Medium< 0.0765, High< 0.0779, Critical≥ 0.0779).

Device	S_{true}	S_{temp}	S_{reboot}	S_{core}	Risk	Band
D03	0.9886	0.6609	0.9957	0.9636	0.0784	Critical
D04	0.9810	0.7310	0.9989	0.8841	0.0760	Medium
D05	0.9749	0.8503	0.9434	0.9067	0.0616	Medium
D07	0.9849	0.8717	0.9872	0.9431	0.0416	Low
D09	0.9866	0.9473	0.9898	0.9042	0.0290	Low
D01	0.9998	0.8784	0.9995	0.9955	0.0249	Low
D06	0.9968	0.9135	0.9941	0.9928	0.0207	Low
D02	0.9939	0.9389	0.9901	0.9655	0.0205	Low
D08	0.9910	0.9954	0.9946	0.9838	0.0083	Low

Importantly, DAIR does not modify classification decisions; it provides a stability overlay that quantifies behavioural persistence around correct recognition. The resulting bands, therefore, represent operational posture rather than immediate compromise signals. On clean TEST traces, Low and Medium bands correspond to baseline variation observed during enrolment. Sustained upward transitions into High or Critical, particularly when accompanied by degradation in S_{true}, indicate structural instability and justify stricter authentication policy or investigation.

5.6. Threat-Modelled Evaluation

The final validation stage evaluates DB² under controlled identity manipulation. Whereas the closed-set and

open-set stages assess separability and novelty detection under nominal conditions, this section examines whether the framework can detect behavioural inconsistency when identity labels are deliberately forged.

Three attack scenarios are reproduced on the held-out TEST partition (reboots 9–10):

- **TH1 (Spoofing / misbinding):** Two legitimate devices impersonate each other via label swapping.
- **TH2 (Cloning / identity-claim impersonation):** A single physical device claims the identity of one legitimate victim device by relabelling its traces to that target identity.
- **TH3 (Sybil):** A single physical device presents multiple logical identities by distributing its traces across multiple victim identities.

In all cases, only the identity labels are altered; the PMU and RTC features remain unchanged. This models an adversary capable of forging identity claims without modifying the local measurement pipeline. The objective is therefore not feature corruption detection, but behavioural identity mismatch detection.

Attack Construction. Attacks are implemented by controlled relabelling of TEST traces. For TH1 and TH2, 70% of the selected traces are reassigned. For TH3, 80% of the source traces are relabelled and distributed across multiple victim identities. The remaining clean samples preserve a baseline for comparison. This produces a substantial but structured perturbation of the identity distribution without altering the underlying hardware behaviour.

Evaluation Protocol. All experiments reuse the previously calibrated **ExtraTrees (ET)** backbone (clean TEST accuracy: 0.9574, consistent with Section 5.3). Preprocessing, per-core normalisation, PCA projection, and isotonic calibration remain unchanged. Detection leverages three complementary DAIR-derived signals:

1. Calibrated classifier confidence and probability margin (identity consistency).
2. DAIR stability indicators, particularly S_{true}.
3. Aggregated attack-salience score S_{attack}, combining identity inconsistency signals.

An adaptive threshold is derived from the distribution of S_{attack} on non-targeted enrolled devices (clean reference set), targeting a 1% false-positive rate and enforcing a minimum floor of 0.10. Devices exceeding this threshold are flagged as exhibiting identity–behaviour conflict.

Attack Impact. Under the combined TH1–TH3 scenario, closed-set accuracy drops from 0.9574 to 0.6622 ($\Delta = -0.2952$), reflecting large-scale identity corruption. Despite identical behavioural features, manipulated devices exceed the calibrated S_{attack} threshold, while clean devices remain below it under the selected operating point. This confirms that detection is driven by identity–behaviour misalignment rather than feature perturbation. Detection is therefore based on structural identity–behaviour inconsistency rather than memorisation of specific attack patterns, making the mechanism agnostic to the particular relabelling configuration.

Table 8 summarises representative DAIR posture shifts. Affected devices show sharp reductions in S_{true}, elevated composite *Risk*, and escalation into higher risk bands. In contrast, clean devices remain below the calibrated S_{attack} threshold under the selected operating point. Some clean devices may still exhibit elevated baseline Risk due to temperature sensitivity (captured by S_{temp}), but without triggering attack salience.

Threat-Specific Behaviour. In the **TH1** spoofing case, both swapped identities are flagged by S_{attack} and transition to the *Critical* band, with pronounced reductions in S_{true}. This reflects strong inconsistencies in identity behaviour under label swapping.

Under **TH2** impersonation (single-victim cloning), the claimed victim identity exhibits reduced S_{true} and elevated composite *Risk*, while the physical source device remains stable when evaluated under its true identity.

In **TH3** Sybil behaviour, all fabricated victim identities are detected by S_{attack} and escalate to Medium or Critical risk bands depending on the magnitude of identity inconsistency.

System-Level Implication. These results validate the layered progression of DB^2. The **closed-set stage** establishes identity separability; the **open-set layer** rejects unfamiliar devices; the **DAIR mechanism** quantifies behavioural stability; and the present **threat-modelled evaluation** demonstrates resilience under identity forgery at the behavioural decision level. Collectively, these findings demonstrate that DB^2 provides accurate identification, reliable unknown rejection, and measurable behavioural assurance under controlled adversarial identity manipulation.

5.7. Computational and Deployment Overhead

Feature measurement is performed on-device using lightweight counters, while identity modelling, open-set scoring, and DAIR aggregation are executed on the backend server. All computational overhead figures reported below therefore correspond exclusively to server-side processing. Timing measurements were obtained on a server-class CPU environment using the FIT/CAL/TEST split described earlier (636 engineered features, PCA projection, and calibrated ExtraTrees backbone); the same configuration was used for all reported runs to ensure comparability.

Table 9 summarises the computational characteristics. Training the ExtraTrees classifier on 173,016 FIT samples required 37.8 seconds. Inference on the 57,672-sample TEST partition required 0.0177 ms per sample for prediction and 0.0252 ms per sample for the complete end-to-end pipeline (StandardScaler + PCA + ExtraTrees).

PCA reduced the feature space from 636 to 256 components, retaining 99.89% of the total variance. The serialised ExtraTrees model occupies approximately 885 MB,

Table 8
Complete DAIR posture and attack-salience snapshot under combined TH1–TH3 attacks (all 9 enrolled devices). Targeted devices are flagged by S_{attack}, while clean devices remain below threshold. Risk-band thresholds are derived from baseline DAIR quantiles.

Device	S_{true}	S_{reboot}	S_{temp}	S_{core}	Risk	Band	S_{attack}	Attack Flag	Threat Role
D05	0.3229	0.9794	0.9142	0.9621	0.3965	Critical	1.0000	True	TH1 target
D02	0.4945	0.9608	0.8758	0.9352	0.3152	Critical	1.0000	True	TH1 target
D08	0.7779	0.9962	0.9363	0.9840	0.1371	Critical	1.0000	True	TH2 victim
D04	0.8989	0.9928	0.8108	0.9014	0.1044	Critical	1.0000	True	TH3 victim
D09	0.9066	0.9946	0.9446	0.9338	0.0699	Medium	1.0000	True	TH3 victim
D06	0.9120	0.9886	0.9195	0.9866	0.0676	Medium	1.0000	True	TH3 victim
D03	0.9886	0.9957	0.6609	0.9636	0.0784	Critical	0.0000	False	Clean (untargeted)
D07	0.9848	0.9866	0.8668	0.9402	0.0430	Low	0.0245	False	TH3 source (clean)
D01	1.0000	0.9998	0.8918	0.9962	0.0221	Low	0.0000	False	TH2 source (clean)

Adaptive S_{attack} threshold: > 0.10 (99th percentile on clean reference set, with floor). All attacked identities are detected; clean identities remain below threshold.

while the full preprocessing + classifier pipeline occupies 886 MB. The model footprint is primarily driven by the ensemble structure and the required tree depth for multi-device discrimination. Model size can be reduced using fewer estimators, depth constraints, or post-training compression techniques; we report the uncompressed backbone to reflect the evaluated configuration.

The reported footprint corresponds to an uncompressed research configuration optimised for separability analysis. In deployment scenarios, estimator count, tree depth, or post-training compression can substantially reduce memory usage without altering the architectural principles of the behavioural pipeline.

These results demonstrate that DB² is operationally feasible under a server-assisted architecture. IoT devices perform lightweight measurement and transmission, while identity modelling and risk aggregation are executed centrally with sub-millisecond per-sample latency. Although the evaluated research configuration produces a large ensemble footprint, model compression or estimator reduction can significantly reduce memory requirements in deployment without altering the behavioural architecture.

Table 9
Computational overhead of the DB² backbone under server-side execution.

Component	Value	Description
Training time (ET, FIT)	37.8 s	173,016 samples
Inference time (predict)	0.0177 ms/sample	57,672 TEST samples
End-to-end pipeline	0.0252 ms/sample	Scaler + PCA + ET
PCA components retained	256	99.88% variance explained
Model size (ET)	885 MB	Serialized ExtraTrees model
Full pipeline size	886 MB	Scaler + PCA + ET

6. Discussion

6.1. DB² as a Layered Behavioural Assurance Framework

The evaluation shows that DB² functions not merely as a fingerprint classifier, but as an integrated behavioural assurance architecture in which separability, rejection, and stability monitoring operate as complementary control layers. Each layer addresses a distinct failure mode beyond classification accuracy.

Closed set identification confirms that device signatures occupy structurally separable regions in the engineered feature space. The consistency of this separability across diverse model families indicates that identity information is embedded in the physical signal rather than tied to a specific learning algorithm. This model invariance strengthens deployment robustness by reducing sensitivity to classifier substitution or architectural updates.

Separability alone cannot prevent forced identity assignment when unfamiliar devices appear. The open-set layer introduces calibrated boundary enforcement through strict CAL-based operating-point selection under leave-one-device-out retraining, converting the classifier into a rejection-capable gatekeeper and preventing silent identity co-option.

A further failure mode occurs when an enrolled device remains correctly classified but becomes behaviourally fragile under operational variation. The DAIR mechanism decomposes stability into independent dimensions, exposing latent sensitivity that aggregate accuracy conceals. Under identity manipulation, this layered structure produces coherent posture shifts without feature perturbation, demonstrating that identity behaviour mismatch is detectable at the confidence and stability level.

Together, separability, boundary enforcement, and stability monitoring form a coherent behavioural identity governance structure. DB² therefore represents a proof-of-concept behavioural identity governance framework in which classification, rejection, and stability assessment are structurally interdependent rather than isolated metrics.

6.2. Physical Basis of Separability and Stability

Device separability originates from physical manufacturing variation. Even within the same product family, crystal oscillators and microarchitectural timing paths exhibit small but persistent differences. When CPU execution timing is compared against an independent RTC reference, these offsets accumulate into measurable cross-oscillator deviation. Because this behaviour is hardware rooted rather than software induced, it persists across reboots, changes in core placement, and operating system resets.

Thermal degradation reflects the same physical origin. Oscillator frequency exhibits nonlinear temperature dependence, and elevated temperatures amplify timing variability. As thermal noise increases, the geometric margins between nearby device identities narrow, explaining the observed reduction in separability under high-temperature conditions.

The combination of timing driven deviation and cache derived microarchitectural dynamics creates a balance between discrimination strength and environmental robustness. Devices with smaller structural margins tend to exhibit greater thermal sensitivity, indicating that separability and stability are physically coupled properties rather than independent artefacts of the classifier.

6.3. Deployment Implications

The overhead analysis in Section 5.7 indicates that server-side inference is negligible relative to the 120-second acquisition window. The 120-second window reflects a conservative evaluation setting chosen to maximise signal stability; shorter windows are feasible with reduced confidence and do not require architectural modification. As a result, optimisation efforts should focus on reducing acquisition time rather than accelerating classification. The separation between on-device measurement and backend modelling enables progressive identification and lightweight client integration, while centralised processing accommodates the model footprint. DB^2 is therefore best suited to server-assisted edge deployments rather than fully on-device execution.

6.4. Scope and Future Extensions

The present evaluation considers nine Raspberry Pi devices drawn from two closely related model families under a uniform software stack. This controlled setup enables isolation of intrinsic behavioural separability across reboot cycles while avoiding confounding architectural variation. Extending validation to larger and more heterogeneous fleets including x86 and RISC-V platforms would provide additional insight into cross-architecture generalisation, while the core training, calibration, and evaluation protocol remains unchanged.

The adversarial analysis focuses on logical identity forgery (TH1–TH3), where labels are manipulated without altering the measurement path. More capable adversaries able to shape workloads, approximate feature distributions, or interfere with the local sensing pipeline fall outside the current trust boundary and motivate future robustness analysis.

Finally, the evaluated ExtraTrees backbone occupies approximately 885 MB in uncompressed form. This footprint reflects a research configuration optimised for separability analysis. In deployment scenarios, estimator reduction, depth constraints, or post-training compression can substantially reduce memory requirements while preserving the architectural principles of behavioural fingerprinting. Future work may explore adaptive operating-point selection and broader environmental stress evaluation to further characterise stability under extreme operating regimes.

7. Conclusion

Establishing reliable device identity on resource-constrained edge platforms without dedicated hardware remains a fundamental challenge. This work introduced DB^2, a risk-aware behavioural identity framework that derives fingerprints exclusively from CPU–RTC cross-oscillator deviation and PMU microarchitectural events available on commodity processors. By structuring identity assurance into closed-set separability, calibrated open-set rejection, and behavioural stability assessment through DAIR, the framework advances beyond standalone classification toward layered identity governance. Evaluation across nine Raspberry Pi devices and 360,000 behavioural records demonstrates consistent separability across reboots, cores, and thermal variation, together with strong unknown-device rejection under strict leave-one-device-out validation and CAL-only operating-point selection. These results demonstrate that robust behavioural fingerprints can be engineered from ubiquitous on-board resources within a statistically disciplined and deployment-ready calibration framework.

Data Availability

The engineered dataset supporting this study is archived on Zenodo (DOI: 10.5281/zenodo.18805876). The record is publicly accessible, with files under embargo pending journal publication. The dataset contains anonymised device identifiers (D01–D09), engineered rolling-window features derived from CPU–RTC and PMU measurements, and documented FIT/CAL/TEST partitions. Original hardware identifiers have been removed for privacy.

References

[1] Al-Omary, A., Othman, A., AlSabbagh, H.M., Al-Rizzo, H., 2018. Survey of hardware-based security support for iot/cps systems. KnE Engineering , 52–70.

[2] Babaei, A., Schiele, G., 2019a. Physical unclonable functions in the internet of things: State of the art and open challenges. Sensors 19, 3208.

[3] Babaei, A., Schiele, G., 2019b. Physical unclonable functions in the internet of things: State of the art and open challenges. Sensors 19. URL: https://www.mdpi.com/1424-8220/19/14/3208, doi:10.3390/s19143208.

[4] Babun, L., Aksu, H., Uluagac, A.S., 2021. Cps device-class identification via behavioral fingerprinting: From theory to practice. IEEE Transactions on Information Forensics and Security 16, 2413–2428. doi:10.1109/TIFS.2021.3054968.

[5] Berdich, A., Groza, B., 2022. Smartphone camera identification from low–mid frequency dct coefficients of dark images. Entropy 24, 1158. URL: https://doi.org/10.3390/e24081158, doi:10.3390/e24081158.

[6] Berdich, A., Groza, B., Mayrhofer, R., Levy, E., Shabtai, A., Elovici, Y., 2023. Sweep-to-unlock: Fingerprinting smartphones based on loudspeaker roll-off characteristics. IEEE Transactions on Mobile Computing doi:10.1109/TMC.2023.3259386. early Access.

[7] Cao, Y., Liu, W., Qin, L., Liu, B., Chen, S., Ye, J., Xia, X., Wang, C., 2022. Entropy sources based on silicon chips: True random number generator and physical unclonable function. Entropy 24, 1566. URL: https://doi.org/10.3390/e24111566, doi:10.3390/e24111566.

[8] Celdrán, A.H., von der Assen, J., Moser, K., Sánchez, P.M.S., Bovet, G., Pérez, G.M., Stiller, B., 2023. Early detection of cryptojacker malicious behaviors on iot crowdsensing devices, in: NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium, IEEE. pp. 1–8.

[9] Dabaghchian, M., Khazaei, H., Khorshed, M.T., 2020. Efficient identity spoofing attack detection for iot. IEEE Internet of Things Journal 7, 3965–3977. URL: https://mason.gmu.edu/~mdabaghc/publications/Efficient%20Identity%20Spoofing%20Attack%20Detection%20for%20IoT.pdf, doi:10.1109/JIOT.2019.2947411.

[10] EE Times Europe, 2020. The role of hardware root of trust in edge devices. URL: https://www.eetimes.eu/the-role-of-hardware-root-of-trust-in-edge-devices/. accessed: 2025-08-16.

[11] Gray, D., Mehrnezhad, M., Shafik, R., 2022. Sensig: Practical iot sensor fingerprinting using calibration data, in: 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), IEEE, Genoa, Italy. pp. 66–81. doi:10.1109/EuroSPW55150.2022.00014.

[12] Hameed, F., Garg, S., Amin, M.B., Kang, B., Khan, A., 2021. A context-aware information-based clone node attack detection scheme in internet of things. IEEE Access 9, 147074–147089. doi:10.1109/ACCESS.2021.3123194.

[13] He, L., Shin, K.G., 2023. Fingerprinting battery health using relaxing voltages, in: Proceedings of the 14th ACM International Conference on Future Energy Systems (e-Energy '23), ACM, Orlando, FL, USA. pp. 1–12. doi:10.1145/3575813.3597363.

[14] Irfan, M., Rehman, Z.U., Khan, Z., Khalil, M.I., Choi, J., 2024. Device fingerprinting for cyber-physical systems: Reliability analysis and graph-based modeling. Symmetry 16, 846. URL: https://www.mdpi.com/2073-8994/16/7/846, doi:10.3390/sym16070846.

[15] Jana, S., Kasera, S.K., 2008. On fast and accurate detection of unauthorized wireless access points using clock skews, in: Proceedings of the 14th ACM international conference on Mobile computing and networking, pp. 104–115.

[16] Kalinin, S., Gribkov, N., 2024. Syntactic–semantic detection of clone-caused vulnerabilities in iot devices. Sensors 24, 7251. doi:10.3390/s24227251.

[17] Kumar, V., Paul, K., 2023. Device fingerprinting for cyber-physical systems: A survey. ACM Comput. Surv. 55. URL: https://doi.org/10.1145/3584944, doi:10.1145/3584944.

[18] Nosouhi, M.R., Sood, K., Grobler, M., Doss, R., 2022. Towards spoofing resistant next generation iot networks. IEEE Transactions on Information Forensics and Security 17, 1669–1683.

[19] Oligeri, G., Sciancalepore, S., 2024. Hideprint: Evading radio fingerprinting identification systems with adversarial noise injection. arXiv preprint arXiv:2408.09179 URL: https://arxiv.org/abs/2408.09179.

[20] Polcák, L., Franková, B., 2015. Clock-skew-based computer identification: Traps and pitfalls. J. Univers. Comput. Sci. 21, 1210–1233.

[21] Rajan, A., Jithish, J., Sankaran, S., 2017. Sybil attack in iot: Modelling and defenses, in: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 2323–2327. doi:10.1109/ICACCI.2017.8126193.

[22] Riad, K., Huang, T., Ke, L., 2020. A dynamic and hierarchical access control for iot in multi-authority cloud storage. Journal of Network and Computer Applications 160, 102633.

[23] Rührmair, U., Devadas, S., Koushanfar, F., 2011. Security based on physical unclonability and disorder, in: Introduction to hardware security and trust. Springer, pp. 65–102.

[24] Salo, T.J., 2007. Multi-factor fingerprints for personal computer hardware, in: MILCOM 2007-IEEE Military Communications Conference, IEEE. pp. 1–7.

[25] Sánchez, P.M.S., Celdrán, A.H., Bovet, G., Pérez, G.M., 2022. Adversarial attacks and defenses on ml- and hardware-based iot device fingerprinting and identification. arXiv preprint arXiv:2212.14677 URL: https://arxiv.org/abs/2212.14677, doi:10.48550/arXiv.2212.14677.

[26] Sanchez-Rola, I., Santos, I., Balzarotti, D., 2018. Clock around the clock: Time-based device fingerprinting, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, pp. 1502–1514.

[27] Selvam, M., Khanam, Z., Singh, A.K., Cui, Z., Muttukrishnan, R., 2025. Cd2a: Continuous device-to-device authentication exploiting crystal oscillator impurities, in: 2025 12th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp. 377–385. doi:10.1109/NTMS65597.2025.11076929.

[28] Shafique, K., Khawaja, B.A., Sabir, F., Qazi, S., Mustaqim, M., 2020. Internet of things (iot) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5g-iot scenarios. IEEE access 8, 23022–23040.

[29] Shang, C., Cao, J., Zhu, T., Zhang, Y., Niu, B., Li, H., 2024. Cadfa: A clock skew-based active device fingerprint authentication scheme for class-1 iot devices. IEEE Systems Journal 18, 590–599. doi:10.1109/JSYST.2024.3351222.

[30] names not provided in your snippet, A., 2024. Blind spots: On the resiliency of device fingerprints to hardware warm-up through sequential transfer learning, in: Proceedings of the 17th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '24), ACM, New York, NY, USA. doi:10.1145/3643833.3656138. cC BY 4.0.

[31] Srivastava, A., Gupta, S., Quamara, M., Chaudhary, P., Aski, V.J., 2020. Future iot-enabled threats and vulnerabilities: State of the art, challenges, and future prospects. International Journal of Communication Systems 33, e4443.

[32] Sánchez Sánchez, P.M., Jorquera Valero, J.M., Huertas Celdrán, A., Bovet, G., Gil Pérez, M., Pérez, G.M., 2023. A methodology to identify identical single-board computers based on hardware behavior fingerprinting. Journal of Network and Computer Applications 212, 103579. URL: https://www.sciencedirect.com/science/article/pii/S108480452200220X, doi:https://doi.org/10.1016/j.jnca.2022.103579.

[33] Tahir, R., Tahir, H., McDonald-Maier, K., 2015. Securing health sensing using integrated circuit metric. Sensors 15, 26621–26642. URL: https://doi.org/10.3390/s151026621, doi:10.3390/s151026621.

[34] Tahir, R., Tahir, S., Tahir, H., McDonald-Maier, K., Howells, G., Sajjad, A., 2021. A novel icmetric public key framework for secure communication. Journal of Network and Computer Applications 195, 103235. URL: https://doi.org/10.1016/j.jnca.2021.103235, doi:10.1016/j.jnca.2021.103235.

[35] Threatpost, 2020. Bluetooth spoofing bug exposes iot devices to attacks. URL: https://threatpost.com/bluetooth-spoofing-bug-iot-devices/159291/. accessed: 2025-08-16.

[36] Yang, J., Zhu, S., Wen, Z., Li, Q., 2025. Cross-receiver radio frequency fingerprint identification: A source-free adaptation approach. Sensors 25, 4451. URL: https://www.mdpi.com/1424-8220/25/14/4451, doi:10.3390/s25144451.

[37] Yousefnezhad, N., Malhi, A., Främling, K., 2020. Security in product lifecycle of iot devices: A survey. Journal of Network and Computer Applications 171, 102779.

[38] Yun, H.S., Wei, D., Yang, S., Park, G., Kim, M.S., Shin, T.J., Walba, D.M., Han, M.J., Yoon, D.K., 2025. Reconfigurable liquid crystal-based physical unclonable function integrating optical and electrical responses. Advanced Materials , e2504288URL: https://doi.org/10.1002/adma.202504288, doi:10.1002/adma.202504288. advance online publication.

[39] Zhang, J., Beresford, A.R., Sheret, I., 2019. Sensorid: Sensor calibration fingerprinting for smartphones, in: 2019 IEEE Symposium

on Security and Privacy (SP), IEEE, San Francisco, CA, USA. pp. 638–655. doi:10.1109/SP.2019.00072.