OS Age-Signal Architecture: Technical & Privacy Risk Analysis

Companion analysis to the Levy briefing memo and the research archive

Prepared by: Jonathan Westmoreland, Bend Privacy Alliance **Date:** June 18, 2026 **Status:** Source-checked against primary developer documentation (Android Developers; Apple Developer), June 2026. The platform APIs are in beta and changing — confirm against the live docs before formal use.

*This is the technical appendix in a three-part package: the **Levy briefing memo** (the short policy argument), the **plain-language explainer** (`os_age_signals_explained.md`, the jargon-free front door for a general audience), and this document (the detailed backing). Read the explainer first if you want the architecture without the technical vocabulary.*

> **The one question this answers:** If Oregon adopts OS-level age signaling, what exactly happens to a user's age data from collection to use — and where can it be misused, leaked, expanded, or misunderstood? The core concern is not "age verification" in the abstract. It is whether Oregon creates a **reusable age-signal layer at the operating-system / account level that can later expand to apps, browsers, websites, schools, and other services.**

Part 1 — One-page explainer (non-technical)

How OS age signaling actually works. An operating-system age-signal law (California's AB 1043 is the template) does not, by itself, ask anyone to upload an ID. At device or account setup, the account holder *indicates* a birth date or age. The operating system turns that into a **coarse age bracket** — under 13, 13–15, 16–17, or 18+. When an app (and, under AB 1856-style expansions, a browser or website) asks, the OS hands over the bracket through a software interface. Under an AB 1043-style statute, the recipient is then treated as having **"actual knowledge"** of the user's age range, which triggers a chain of legal duties. (Oregon could draft this differently; the "actual knowledge" hook is a feature of the California model, not a given.)

Why that design is deceptively deep. Because nothing dramatic happens at setup — no ID scan, no face scan — an OS age-signal law looks light-touch. But it can quietly normalize a **persistent age attribute attached to the device or account** that can be requested over and over, by more and more recipients, for the life of that account. The privacy question is not the first request. It is everything the architecture makes possible afterward: which parties can ask, what they keep, whether the bracket becomes one more stable identifier in a tracking profile, and whether a "child-safety" signal becomes a general-purpose age-and-identity layer.

The three things to hold onto:

1. **The signal is a starting point, not a safeguard.** The platforms (Apple, Google) hand over a bracket. They do not decide what it means in Oregon law, they do not gate any particular feature, and their signals stop at the edge of one operating system. Everything protective has to be built by the law and the recipient — not the platform.

2. **Scope creep is the main risk.** California started with "OS age signals for apps" (AB 1043) and is now considering AB 1856, which would extend the same signal to browsers and websites. An Oregon bill should be read as the *first* step in that progression unless it contains explicit limits.
3. **The protection has to be statutory.** Because the OS model transmits a signal rather than blocking speech, it may face less courtroom resistance than the app-store laws — which means Oregon cannot count on litigation to supply privacy limits. The guardrails have to be in the statute.

The rest of this document is the technical backing for those three points.

Part 2 — Architecture map

The OS age-signal flow has six stages. At each, the questions that matter are: who controls it, what data exists, where it lives, what is transmitted, what logs are created, who can access or audit it, and whether it can be reused.

Stage	What happens	Who controls it	Key risk introduced
1. Collection	Account holder indicates birth date / age at device or account setup	OS / platform-account provider	Whether adults are pressured to self-identify; whether a real birth date is stored or discarded
2. Derivation	OS derives a coarse age bracket from the birth date / age	OS / platform-account layer	Where the underlying birth date lives after the bracket is derived; whether it is deleted
3. Translation	Bracket is formatted into a standard signal (status + age range + method)	OS / platform	Extra fields beyond the bracket (verification method, supervision status, parent-child link) widen the data surface
4. Request	An app — or, under expansion, a browser/website — asks for the signal	Recipient app/site (and any code inside it)	**Who is allowed to ask** is the central control; third-party code inside a covered app inherits the app's access
5. Delivery	Signal is returned via a platform API	OS / platform	Whether a persistent identifier accompanies the

Stage	What happens	Who controls it	Key risk introduced
			signal (it does, on Android)
6. Legal effect	Under an AB 1043-style statute, the recipient is deemed to have "actual knowledge" of the age range and owes downstream duties	Statute + recipient	"Actual knowledge" pulls in obligations the recipient may meet by collecting *more* data, not less

The architecture's center of gravity is stages 4–5. Collection and derivation are mostly settled by the platform. The privacy outcome turns on *who can request the signal* and *what travels with it.*

Part 3 — Data inventory

Every element that can exist in an OS age-signal system, with a use-grade. "Necessary" = needed for the stated purpose; "Risky" = useful but a real privacy hazard; "Prohibit" = should not be shared/retained absent compelling justification; "Unclear" = depends on implementation.

Data element	Typically held by	Use-grade	Note
Exact birth date	OS / account provider	**Prohibit (sharing)**	Should never leave the OS layer; only the bracket should travel downstream
Exact age	OS / account provider	**Prohibit (sharing)**	Same as above
Coarse age bracket (e.g., 13–15)	OS → recipient	**Necessary**	The minimum needed to determine which protections apply
Over/under a single threshold	OS → recipient	**Necessary (preferred)**	A yes/no answer (e.g., "18+?") is lower-risk than a bracket; prefer where the law allows
Verification method (self / guardian / payment / gov-ID)	OS → recipient	**Risky**	Reveals more than age; can imply ID was collected somewhere
Child / supervised-account status	OS → recipient	**Risky**	Exposes that the user is a supervised minor

Data element	Typically held by	Use-grade	Note
Parental-consent status	OS → recipient	**Risky**	Useful for the law, but a sensitive flag
Parent–child relationship	OS / account graph	**Risky**	Relationship data is independently sensitive
Device ID	recipient / OS	**Risky**	Linkability vector when combined with the bracket
Account ID	OS / recipient	**Risky**	Same
Install ID (Android `installId`)	recipient (stored)	**Risky**	Persistent per-install identifier the developer is told to retain
App-specific token	recipient	**Unclear**	Depends on whether it is stable and cross-referenced
Error / refusal / unknown status	OS → recipient	**Necessary**	Must be handled gracefully; refusal should not itself be penalized
Timestamp / request logs	OS and/or recipient	**Risky**	Repeated requests build a behavioral log of when age was checked

Design rule that falls out of this table: only the *coarse bracket* (or, better, an over/under-threshold answer) should leave the OS layer. Birth date and exact age should be derived-and-discarded. Every "Risky" row that a bill allows to travel widens the breach and tracking surface without adding to the stated child-safety purpose.

Part 4 — Request-control analysis (the most important section)

The single most consequential technical question: **can only a first-party covered service request the signal, or can third-party SDKs, ad networks, analytics tools, and embedded code also obtain it?**

What the platforms actually do today:

- **Google Play Age Signals API** restricts the returned information to "the requesting app," limits use to the requesting app, and its terms bar use for advertising, marketing, user profiling, or analytics. It recommends pairing with the Play Integrity API to confirm the call comes from an untampered app.

- **Apple's Declared Age Range API** returns the band only when the user (or a parent/guardian) agrees to share it, with a parent-controlled choice of *always / per-request / never*. Apple does not reveal the birth date.

Where the control actually leaks: both platforms scope access to *the app*, not to a first-party identity within the app. A third-party SDK — an ad network, analytics library, attribution tool, or embedded component — runs **inside the app's process and inherits the app's access.** So the platform's "only the requesting app" boundary is an **app-level** boundary, not a "first-party-only" boundary. If a covered app contains an advertising or analytics SDK, that SDK sits inside the boundary that received the age signal. Industry compliance guidance already flags this: developers are told to inventory every third-party SDK and disable non-essential identifiers (ad IDs, fingerprinting, retargeting) for minors. That advice exists precisely *because* the technical boundary does not stop in-app third-party code by itself.

The legislative implication: a "child-safety" signal can become a **tracking attribute** the moment it sits next to ad-tech inside an app, unless the statute (not the platform terms) forbids the signal — and anything derived from it — from reaching advertising, analytics, profiling, or third-party SDKs. Platform terms-of-service restrictions are enforced by policy and audits, not by technical impossibility.

Questions a bill must answer: Which recipients may request the signal — all apps, only "covered" apps, app stores, browsers, websites, school apps, social platforms, AI companions? May embedded third-party code receive or infer it? Is advertising/analytics/profiling use prohibited by statute, not just by platform policy?

Part 5 — Persistence and linkability

This is the technical heart of the privacy concern: **can the age signal become a fingerprint or a durable tracking attribute?**

- **Stability:** the bracket is low-entropy on its own (four values), so it is not an identifier by itself. The risk is **combination** — a stored bracket sitting next to a device ID, account ID, or install ID becomes one more stable attribute in a per-user profile.
- **Persistent identifiers attached to the signal:** on Android, the API returns a **Play-generated `installId`** that developers are told to **store** (to reconcile against parent-revocation reports). That is a persistent, per-install identifier tied to the user's age/approval status. On Apple, the signal carries no persistent identifier of its own — but Apple's own developer guidance suggests persisting consent information to the developer's servers, so **persistence becomes developer-created** rather than platform-prevented.
- **Real-time vs. cached:** Google prohibits long-term storage of age-signal data and directs developers to query in real time and decide in real time; Texas-style rules require deleting age-related personal data once verification is complete. These are good defaults — but they are *platform/contract* rules, and a state law should encode the same deletion duty so it does not depend on a vendor's terms.

- **Revocation and transitions:** what happens when a child turns 13, 16, or 18? The bracket changes, and the signal must be re-queried; whether previously stored brackets are refreshed is a developer-implementation matter. A parent can withdraw consent (on Apple, this can block app launch); the system must handle revocation cleanly.
- **Cross-device and cross-platform:** each platform's signal covers **one operating system** — Apple's is iOS/iPadOS/macOS only; Google's is Android. A child uses phones, tablets, Chromebooks, consoles, and the open web. This gap has spawned **third-party "compliance" middleware that aggregates Apple, Google, console, and web age signals into a single cross-platform session** — which solves a developer headache but creates a new **centralization and linkability** point that no single platform's privacy promises cover.

The legislative implication: the bill should bar the signal and any derived value from being stored as, or joined to, a persistent identifier; require real-time query with prompt deletion of underlying proof; and treat any third-party aggregation layer as a covered entity subject to the same limits.

Part 6 — Edge-case testing

Difficult cases where an OS age signal misfires. These are concrete enough to raise directly with Rep. Levy.

Scenario	Technical problem
Shared family device	The OS/account signal may reflect the account holder, not the person actually using the device at that moment
Parent's account used by a child	An adult (18+) signal may be delivered for a child user
Child's account used by a parent	A minor signal may restrict legitimate adult use
Multiple profiles on one device	The signal may vary by profile, or fail when profiles are switched
School-managed / district device	The district may become the de facto age/account authority, creating new student-data exposure
Foster care / custody dispute	"Parental consent" and parent-child linkage become legally and emotionally fraught
Emancipated minor	A parental-consent model is legally inappropriate for the user
Library / public computer	No stable personal account; the signal is meaningless or wrong
Open-source operating system	May not collect age or identity at all (the reason AB 1856 adds an open-source carve-

Scenario	Technical problem
	out)
Assistive technology	Age-gating flows may interfere with accessibility or block access
No stable identity documents / no platform account	Users without accounts or ID cannot produce a reliable signal and may be excluded
Traveler / VPN / wrong-region account	Region detection drives whether the signal even applies; mismatches produce wrong results

The pattern: the signal assumes **one account = one known-age user on one device.** Every case where that assumption breaks is a case where the system either over-restricts a legitimate user or mis-labels a minor as an adult.

Part 7 — Platform implementation review

Google and Apple are **implementation examples beneath the OS architecture**, not the architecture itself. Both are in beta and changing.

Google Play Age Signals API (Android) — a client-side runtime API the app calls (`checkAgeSignals`). Returns a **user status** (VERIFIED, SUPERVISED, SUPERVISED_APPROVAL_PENDING, SUPERVISED_APPROVAL_DENIED, DECLARED, UNKNOWN), an **age range** via `ageLower`/`ageUpper` (default brackets 0–12, 13–15, 16–17, 18+; customizable in Play Console), and a persistent **`installId`**. It returns data only in regions where Play is legally required to provide it; bars use for advertising/marketing/profiling/analytics; limits the data to the requesting app; prohibits long-term storage; and recommends the Play Integrity API against spoofing. Google's own Play Console help confirms the API began returning signals for new Texas accounts after May 28, 2026, following the Fifth Circuit stay.

Apple Declared Age Range API (iOS / iPadOS / macOS only) — the app calls `requestAgeRange` / `isEligibleForAgeFeatures`. Returns an **age band** (under 13, 13–15, 16–17, 18+), **not** the birth date; the **method** of age assurance (self-declared, guardian-declared, or verified via payment/government ID); whether **parental controls** are enabled; and feature eligibility. Sharing is parent-controlled (always / per-request / never; managed-Screen-Time children cannot change it). Paired with **PermissionKit** and a **Significant Change** flow for parental consent on major updates, with server notifications when a parent withdraws consent (which can block app launch). Apple's own developer news confirms Texas new-account coverage as of January 1, 2026.

For each platform, the diagnostic questions (and current answers where known): *What does it return?* (a bracket/status, not birth date). *Who can call it?* (the app — and any code inside it). *Is it app-specific?* (the data is, but Android attaches a persistent installId). *Is it jurisdiction-limited?* (yes — only where legally required). *Does it include consent / supervision status?* (yes). *What are the data-use restrictions?* (no ads/analytics/profiling; real-time, no

long-term storage). *What happens if the signal is unknown/unavailable?* (an UNKNOWN/error state the developer must handle).

Still to confirm against primary docs (open task): Microsoft / Windows / Xbox family-safety age signals; whether Meta, Snap, TikTok, Discord, or Roblox expose any *developer-callable* age-assurance signal (these largely do age assurance internally for their own platforms rather than offering an OS-style signal API); and the browser-level relay model that AB 1856 points toward. These should be treated as *to-be-verified* rather than assumed.

Part 8 — Security and abuse model

For each threat, the statutory safeguard that would reduce it (the technical risk table the memo's questions imply).

Threat	What goes wrong	Statutory safeguard that reduces it
Breach of an age/identity store	Birth dates or brackets + identifiers leak	No exact-birthdate storage downstream; no centralized state database; real-time query + prompt deletion
Subpoena / warrant / civil discovery	Age and parent-child data pulled in litigation	Warrant/court-order requirement; data-minimization so little exists to produce
Immigration enforcement	Age/identity/relationship data repurposed	Explicit prohibition on non-safety government use; purpose limitation
School discipline	District-held signals used punitively	Bar on disciplinary use of age-signal data; school-managed-device protections
Advertiser / developer misuse	Bracket joins an ad profile	Statutory ban on advertising/analytics/profiling use — not just platform terms
Third-party SDK access	Embedded ad/analytics code sees the signal	Bar on the signal reaching third-party SDKs; SDK inventory/segregation duty
Replay / spoofing	Forged or replayed signals	Integrity-attestation requirement (e.g., Play Integrity-style)
False declaration	Self-reported age is simply false	Accept as a known limit; do not "fix" it by mandating ID/biometrics

Threat	What goes wrong	Statutory safeguard that reduces it
Parent-child relationship exposure	Family graph leaks or is subpoenaed	Minimize relationship data; treat it as sensitive
Data-broker enrichment	Bracket sold/merged into broker profiles	Ban on sale and on combining with broker data; tie to OCPA
Cross-app / cross-platform fingerprinting	Bracket + IDs across apps identify the user	Bar joining the signal to persistent identifiers; cover aggregation middleware

Part 9 — Oregon-specific implementation questions

Tying the technical analysis back to a possible Oregon bill:

1. Would Oregon require OS providers to **collect** age, or only to **pass a signal if age is already known**? (The second is far less invasive.)
2. Which **devices** are covered — phones, tablets, laptops, Chromebooks, desktops, game consoles, smart TVs?
3. Does it apply to **school-managed devices**, and if so, how does it interact with OSIPA and FERPA?
4. Does it reach **browsers and websites** now, or **prohibit expansion** without new legislation? (The AB 1856 question.)
5. Is the signal limited to a **coarse bracket or over/under-threshold**, with **exact birth date and age barred** from traveling downstream?
6. Are **advertising, analytics, profiling, and third-party-SDK** uses prohibited **by statute**?
7. Is there a **real-time-query + prompt-deletion** rule, and a **bar on persistent-identifier linkage**?
8. Is **government access** gated behind a **warrant or court order**, with non-safety uses (immigration, discipline) prohibited?
9. Who **enforces** — Attorney General, private right of action, school districts, or a combination? (Note Oregon's recent pattern: SB 1546 used a private right of action; the app-store states have moved toward AG-exclusive or private-only models.)
10. Is a **privacy impact assessment** and an **independent technical-feasibility review** required **before** rulemaking?

Sponsor question checklist (one-page takeaway for Rep. Levy)

A short list to take to legislative counsel:

- **Which layer** — school EdTech, app-store, OS signal, or proof-of-age?

- **Collect or only pass-through?** Does the OS have to collect age, or only relay an existing signal?
- **What travels?** Bracket only, or also birth date, method, supervision status, parent-child link?
- **Who may request?** First-party covered services only, or also browsers, websites, and in-app third-party SDKs?
- **Scope lock?** Does the bill prevent expansion from apps to browsers/websites without new legislation?
- **Reuse ban?** Are advertising, analytics, profiling, discipline, immigration, and law-enforcement uses prohibited by statute?
- **Persistence?** Real-time query with prompt deletion, and no linkage to persistent identifiers?
- **Government access?** Warrant or court order required?
- **Devices and schools?** Which devices, and how does it interact with school-managed devices, OSIPA, and FERPA?
- **Enforcement?** AG, private right of action, or both?
- **Process?** Privacy impact assessment and independent technical review before rulemaking?

Source basis: Google Play Age Signals API documentation (Play Age Signals overview; Use Play Age Signals API (beta); release notes; Play Console Help "Changes to Google Play for upcoming app store bills"), Android Developers, accessed June 2026. Apple Declared Age Range and PermissionKit documentation (Declared Age Range; Age assurance developer Q&A; "Apple expands tools to help parents protect kids and teens online"; "Next steps for apps distributed in Texas"), Apple Developer, accessed June 2026. California AB 1043 / Civil Code §§1798.500 et seq. and AB 1856 (Legislative Counsel's Digest) per the research archive. Platform APIs are in beta and changing; Microsoft/Meta/Snap/TikTok/Discord/Roblox and browser-level behavior remain to be confirmed against primary docs. Confirm current litigation posture (especially Texas SB 2420) the morning of any formal use.