Network Working Group
Internet-Draft
Intended status: Informational
Expires: 22 November 2026

B. Morrison
Alter Meridian Pty Ltd (~truealter)

May 2026

The Compute-Location Gate: Provenance-Class Routing of Identity Inference with Wire-Layer Refusal of Unconsented Provenance Classes

draft-morrison-compute-location-gate-00

Abstract

This memo specifies the compute-location gate: a mechanism by which a client and an identity-inference server negotiate, at the wire layer and before any inference is performed, the location at which an identity inference will compute, as a deterministic function of the provenance class of the input signal. Three provenance classes are distinguished. Active inference, initiated by the inferred-about principal, MAY compute server-side and produce a server-held identity vector. Passive aggregate observation over a cohort no smaller than a declared minimum MAY compute server-side but yields only a population-level observation that is not attributable to an individual. Passive individual observation is local-only: it is computed and retained on the device that observed it and is never transmitted to a server. The gate is enforced by consent-class matching and by a wire-layer refusal returned when a requested provenance class is not consented; it is not enforced by any cryptographic proof concerning data that was not used. The memo is Informational. The wire surface composes with the discovery mechanism of [MCPDNS], the handle namespace of [IDPRONOUNS], and the organisational policy substrate of [POLICYPROV]; no new transport is introduced.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 2 November 2026.

Table of Contents

1. Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

2. Introduction

An identity-inference system derives statements about a principal (traits, competencies, dispositions, belonging measures) from signals the principal emits. Such systems face a question that access control alone does not answer: _who may read_ a derived statement, and _where the derivation itself is permitted to compute_.

These are distinct questions. Access control governs the read path of a datum that already exists. The compute-location question governs whether the datum may be brought into existence on a given machine at all. A system that answers only the first question can decline to serve an inferred trait to an unauthorised reader, but it has, by the time of that refusal, already computed and persisted the trait on its server. The compute-location question, asked earlier, prevents the server-side derivation from occurring when the signal's provenance does not warrant it.

This memo specifies a mechanism, the compute-location gate, that answers the second question at the wire layer. Before an inference is performed, the client and the server negotiate the _provenance class_ of the input signal. The provenance class deterministically selects a _compute location_. Where the negotiated provenance class is not covered by the principal's consent, the server returns a wire-layer refusal, and no inference is performed at any location.

The mechanism rests on a principle the present author has elsewhere termed identity-as-inference: that a principal's identity is inferred from manifestation rather than declared, and that every such inference is admissible only under an explicit gate on the provenance of the signal from which it is drawn. The first clause of that principle is: no inference without a compute-location gate. This memo is the wire-layer codification of that clause.

The mechanism is deliberately narrow. It specifies provenance-class negotiation, the routing function from provenance class to compute location, the consent-class match, and the refusal returned on a consent miss. It does NOT specify, and explicitly excludes from its scope (Section 11), any cryptographic proof that a particular data category was excluded from a derivation. The gate's enforcement model is consent-class matching plus wire-layer rejection. Verification that the gate held is addressed by build-time pipeline checks (Section 8) and by post-hoc audit, not by a cryptographic attestation of absence.

The wire surface composes with four Morrison-family Internet-Drafts: the discovery surface of [MCPDNS], the handle namespace of [IDPRONOUNS], the cross-session coordination posture of [SUBSTRATE], and the organisational policy substrate of [POLICYPROV]. No new transport, no new handle category, and no new discovery record is introduced.

3. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 RFC2119 [RFC8174] when, and only when, they appear in all capitals, as shown here.

The following terms are defined for the purposes of this document. Terms previously defined by the referenced Morrison-family memos retain their established meaning and are reproduced here only when operative for the present specification.

Identity inference A computation that derives a statement about a

principal (a trait value, a competency estimate, a disposition, a belonging measure, or a comparable derived attribute) from one or more signals the principal has emitted.

Provenance class A property of an input signal, established before inference, that records how the signal came to be observed. Three provenance classes are defined in Section 3.

Compute location The machine class on which an identity inference is permitted to execute. Two compute locations are distinguished: server-side, on infrastructure operated by the inference service; and device-local, on the device that observed the input signal.

Server-held identity vector A persisted, server-side representation of a principal's inferred identity, readable through the inference service's consented query surface. Only inference of the active provenance class (Section 3.1) may write to the server-held identity vector.

Device-local daemon A long-running process executing on a device under the principal's control, on which device-local inference (Section 3.3) is computed and where the resulting representation is retained. The device-local daemon does not transmit individual-attributable derived statements to a server.

Cohort floor The minimum cohort size, declared by the inference service, below which a passive aggregate inference (Section 3.2) MUST NOT be computed. The cohort floor exists so that a population-level observation cannot be narrowed to an individual.

Consent class A unit of consent granted by a principal, keyed to a provenance class and a signal stream. A consent class is the grant against which a negotiated provenance class is matched (Section 6).

Wire-layer refusal A typed response returned by the inference service, before any inference is performed, when a requested provenance class is not covered by a consent class. The refusal terminates the inference request; no inference is performed at any compute location.

~handle A principal identity handle as defined by [IDPRONOUNS].

4. Provenance Classes

Every input signal admitted to an identity inference carries exactly one of three provenance classes. The provenance class is established before inference and is immutable for the lifetime of the signal record. The provenance class, not the signal's content and not the trait category of the prospective output, is the value on which the compute-location gate routes.

4.1. Active Inference

An input signal is of the active provenance class when the inferred-about principal initiated the act that produced it. Challenge-response exchanges, a consented structured assessment, an explicit attestation the principal authored, and inputs the principal supplied to a deliberate identity-elaboration flow are all of the active class. The defining property is principal initiation: the principal performed an act whose purpose, as understood by the principal at the time of the act, was to contribute signal to their own identity inference.

An inference drawn solely from active-class signal MAY compute server-side and MAY write to the server-held identity vector (Section 7.1).

4.2. Passive Aggregate Observation

An input signal is of the passive aggregate provenance class when it was observed without a principal-initiated act, and when the inference drawn from it is computed over a cohort of principals no smaller than the cohort floor (Section 5) and yields only a population-level observation. A passive aggregate inference produces a statement about the cohort (a distribution, a rate, a population parameter) and does not produce a statement attributable to any individual within the cohort.

An inference of the passive aggregate class MAY compute server-side (Section 7.2). Its output is a population-level observation; it MUST NOT write an individual-attributable statement to the server-held identity vector.

4.3. Passive Individual Observation

An input signal is of the passive individual provenance class when it was observed without a principal-initiated act and the inference drawn from it would yield a statement attributable to a single principal.

An inference of the passive individual class is local-only. It MUST be computed on the device-local daemon (Section 4) running on the device that observed the input signal, and the resulting derived statement MUST be retained on that device. An individual-attributable statement of the passive individual provenance class MUST NOT be transmitted to a server, and MUST NOT, by any derivation path, reach the server-held identity vector.

The passive individual class is the load-bearing constraint of this memo. The other two classes describe what server-side inference MAY do; this class describes what server-side inference MUST NOT do. A system that routes passive individual observation to server-side compute has not implemented the compute-location gate, irrespective of any access control it applies to the resulting datum afterward.

5. The Device-Local Daemon

Inference of the passive individual provenance class is computed on a device-local daemon. The device-local daemon is a process under the principal’s control, executing on the device that observed the input signal.

The device-local daemon:

1. Receives passive individual signal observed on its host device.

2. Computes the identity inference locally. No input signal of the passive individual class, and no statement derived from it, leaves the device for the purpose of the inference.

3. Retains the derived statement in device-local storage under the principal’s control.

4. Exposes the derived statement to the principal, and only to the principal, through a device-local interface. The principal MAY, by a subsequent active-class act, elect to contribute a derived statement to a server-held inference; such an act re-enters the pipeline as active-class signal (Section 3.1) and is gated afresh.

The device-local daemon MUST NOT expose an interface by which a server, or any party other than the principal operating the device, can read an individual-attributable statement of the passive individual class. The transition of a passive-individual-derived statement to server-side visibility occurs only through a fresh active-class act by the principal, never through a daemon-exposed read surface.

The device-local daemon participates in the substrate-observation posture of [SUBSTRATE] for the purpose of cross-session coordination of the principal's surfaces; that participation is orthogonal to the present memo and introduces no path by which passive individual derived statements reach a server.

6. The Cohort Floor

A passive aggregate inference (Section 3.2) MUST NOT be computed unless the cohort over which it is computed is no smaller than the cohort floor. The cohort floor is a positive integer declared by the inference service and exposed at the negotiation surface (Section 6) so that a client can determine, before requesting an inference, whether a passive aggregate request is admissible.

The cohort floor exists to ensure that a passive aggregate inference yields a population-level observation and not an individual-attributable one. An inference computed over a cohort smaller than the floor risks re-identification: with a sufficiently small cohort, a population parameter is a near-individual statement. The floor is the structural boundary between the passive aggregate class, which MAY compute server-side, and the passive individual class, which MUST NOT.

The cohort floor is a parameter of this mechanism, not a fixed constant of this memo. An inference service SHALL declare its cohort floor and SHALL NOT compute a passive aggregate inference over a cohort below it. A service whose declared cohort floor is so low that a population parameter computed at that size is individually identifying has not satisfied the intent of this section; the floor is a number, but a number chosen so that the resulting observation is genuinely population-level. The reference deployment described in Section 12 declares a cohort floor of 1000.

The cohort floor governs the _aggregate_ boundary only. It is not a privacy budget, it is not a noise-calibration parameter, and it does not compose additively across queries. The relationship of the cohort floor to the statistical-disclosure-control literature is discussed in Section 10.

7. Wire-Layer Negotiation

Before an identity inference is performed, the client and the inference service negotiate the provenance class of the input signal and, by the routing function of Section 7, the compute location. The negotiation is a request-response exchange carried over the client's existing transport to the inference service; this memo introduces no new transport. Where the inference service is reached as a Model

Context Protocol [MCP] surface, the negotiation is a tool invocation and its response; where it is reached over another transport, the negotiation is the corresponding request-response pair.

7.1. The Inference Request

A client requesting an identity inference SHALL include, in the request, an inference-negotiation object carrying at minimum the following fields.

provenance_class (enum, REQUIRED) One of active, passive-aggregate, passive-individual. The provenance class the client asserts for the input signal of the prospective inference.

signal_stream (string, REQUIRED) A stable identifier for the stream from which the input signal is drawn. Consent classes (Section 6.2) are keyed to the pair (provenance_class, signal_stream); the stream identifier is the second key.

requested_output (string, REQUIRED) An identifier for the category of derived statement the inference is to produce. The requested output does not select the compute location (the provenance class does), but it is carried so that the consent match (Section 6) and the audit record (Section 9) can record what was requested.

cohort_size (integer, OPTIONAL) Present only when provenance_class is passive-aggregate. The size of the cohort over which the aggregate inference is to be computed. The service SHALL reject the request if cohort_size is below the declared cohort floor (Section 5).

principal (string, OPTIONAL) The ~handle of the inferred-about principal, present when the inference is attributable to a named principal. Absent for a passive aggregate inference, which is by construction not attributable to an individual.

7.2. The Negotiation Response

The inference service SHALL respond to an inference-negotiation request with one of three typed responses.

accepted The asserted provenance class is consented (Section 6), the routing function (Section 7) has selected a compute location, and the inference will proceed at that location. The response carries the selected compute_location (server-side or device-local), so that the client and any audit observer record where the inference computed.

refused The asserted provenance class is not covered by a consent class. No inference is performed at any compute location. The structure of the refusal is specified in Section 6.3.

redirected The asserted provenance class is consented, but the routing function has selected a compute location other than the one the client is positioned to satisfy. The most common case is a client requesting a server-side inference on signal the service classifies as passive-individual: the service does not perform the inference server-side, and the response directs the client to perform the inference on the device-local daemon (Section 4). A redirected A redirected response is not a refusal; the inference is admissible. It is not a server-side acceptance either.

The negotiation response, in every case, is returned _before_ any inference is performed. An inference service MUST NOT compute an identity inference and then decide, from the result, whether to return it; the compute-location gate is evaluated on the negotiation object alone, ahead of the inference.

8. Consent-Class Matching

The compute-location gate is enforced by consent-class matching. The asserted provenance class of an inference request is matched against the consent classes the principal has granted; an inference proceeds only when a matching consent class is found.

8.1. Provenance Class Is Distinct from Trait Category

A consent class is keyed to a provenance class and a signal stream, not to the category of the derived output. A principal who has consented to active-class inference of a disposition has not thereby consented to passive-individual-class inference of the same disposition. Consent granted for one provenance class does not generalise to another. This is the defining property of provenance-class consent: the _how it was observed_ is consented separately from the _what is derived_.

It follows that an inference service MUST carry the provenance class on every signal record and on every derived statement throughout its internal architecture, so that the consent match can be evaluated and so that the audit record (Section 9) can record the provenance class that was matched. A derived statement that has lost its provenance class is not consent-checkable and MUST NOT be served.

8.2. The Consent Class

A consent class is a grant, authored by the principal, carrying at minimum:

* The provenance_class the grant covers.

* The signal_stream the grant covers.

* The set of requested_output categories the grant admits, or an explicit marker that the grant admits all output categories for the covered (provenance class, signal stream) pair.

* A revocation marker; consent classes are revocable, and a revoked consent class MUST NOT satisfy a subsequent match.

A consent class for the passive-individual provenance class authorises device-local inference (Section 4) only. No consent class, of any provenance class, authorises the transmission of an individual-attributable passive-individual derived statement to a server; that transmission is precluded by Section 3.3 and is not a grantable scope.

Consent for inference from a third-party signal stream is a separate consent class from any authorisation the principal may have granted the third-party platform itself. A principal’s authorisation of a platform’s data-access grant is not a consent class for ALTER-side or any inference-service-side inference from that platform’s stream; the inference service SHALL require a distinct, separately revocable consent class, keyed to the (provenance_class, signal_stream) pair, before inferring from a third-party stream.

8.3. The Wire-Layer Refusal

When the asserted provenance class of an inference request is not covered by a consent class, the inference service SHALL return a refused negotiation response. The refusal carries at minimum:

reason (enum, REQUIRED) One of no-consent-class (no consent class covers the asserted provenance class and signal stream), consent-revoked (a consent class existed but has been revoked), cohort-floor (a passive-aggregate request carried a cohort_size below the declared floor), or provenance-not-routable (the asserted provenance class is not one the service admits for the requested output).

provenance_class (enum, REQUIRED) The provenance class that was

asserted and refused, echoed so that the client and audit observer record what was requested.

explanation (string, REQUIRED) A human-readable explanation, sufficient for the client's reasoning surface to present to the principal without a further round-trip.

A refused response is terminal for the inference request. The inference service MUST NOT, having returned a refusal, perform the refused inference at any compute location, and MUST NOT perform a substitute inference of a different provenance class without a fresh negotiation. The refusal is a wire-layer event: it is observable to the client, it is recorded in the audit log (Section 9), and it occurs before any inference computation.

The refusal model of this memo is consent-class matching plus wire-layer rejection. It is not, and does not rely on, any cryptographic attestation that a refused signal was absent from a computation. The refusal asserts that the inference was not performed; it does not produce a proof, verifiable without access to the underlying data, that a particular data category did not contribute to some other computation. The boundary between the mechanism this memo specifies and the cryptographic-attestation question it does not address is stated in Section 11.

9. Routing and Compute-Location Selection

The routing function maps a consented provenance class to a compute location. The function is total over the three provenance classes and is deterministic: the same provenance class always selects the same compute location.

9.1. Active Class to Server-Side

A consented inference of the active provenance class is routed to server-side compute. Its output MAY be written to the server-held identity vector and MAY be served, subject to the inference service's ordinary read-path consent, through the service's consented query surface. The principal initiated the act that produced the signal; server-side derivation and persistence is the routing outcome.

9.2. Passive Aggregate Class to Server-Side, Population-Level Output

A consented inference of the passive-aggregate provenance class, whose cohort_size is no smaller than the declared cohort floor, is routed to server-side compute. Its output is a population-level observation. The output MUST NOT be written to the server-held identity vector as an individual-attributable statement; the server-held identity vector is, by Section 3, the destination of active-class inference alone. A passive aggregate output is a statement about the cohort, retained as such.

9.3. Passive Individual Class to Device-Local

A consented inference of the passive-individual provenance class is routed to device-local compute on the device-local daemon (Section 4). The inference service does not perform the inference. Where a client requests a server-side inference on signal the service classifies as passive-individual, the service returns a redirected negotiation response (Section 6) directing the client to the device-local daemon.

The routing function has no branch by which a passive-individual inference computes server-side. A server-side compute location is not a selectable outcome for the passive-individual provenance class under any consent configuration. This is the structural property that distinguishes the compute-location gate from an access-control mechanism: access control could, in principle, be configured to admit a reader of a server-side passive-individual trait; the routing function of this section has no such configuration, because the trait is never computed server-side to be read.

10. Audit and Build-Time Verification

The compute-location gate is verifiable in two complementary ways: by a per-event audit record written at negotiation time, and by a build-time check on the inference pipeline’s data-flow.

10.1. Per-Event Audit Record

Every negotiation, whether it resolves to accepted, refused, or redirected, SHALL produce an append-only audit record carrying at minimum: the asserted provenance_class; the signal_stream; the requested_output; the negotiation outcome; the selected compute_location where the outcome was accepted; the reason where the outcome was refused; an [RFC3339] timestamp; and the attribution of the requesting party. The audit record is written to an append-only log; admitted records are not retractable or amendable. Where the inference service is operated as, or alongside, an organisational

identity substrate, the audit record SHOULD be emitted to that substrate's audit-signal ingestion surface as specified by [POLICYPROV].

The audit record makes the gate's operation observable after the fact: an auditor can determine, for any inference the service performed, what provenance class was asserted, what compute location was selected, and which consent class was matched.

10.2. Build-Time Data-Flow Verification

In addition to the per-event audit record, an inference service SHOULD verify, at the time its inference pipeline is built, that the pipeline contains no data-flow path by which a signal of the passive-individual provenance class reaches the server-held identity vector.

The verification treats the inference pipeline as a directed graph whose nodes are pipeline stages and whose edges are data-flow dependencies between stages. The verification is the property: for every signal node of the passive-individual provenance class, and for every node representing a write to the server-held identity vector, there exists no directed path from the former to the latter. A pipeline that fails this property has a route by which a device-local-only inference could reach the server, and the build SHOULD fail.

This build-time verification is a check on the pipeline's structure, performed before the pipeline runs; it is independent of, and complementary to, the per-event audit record, which observes the pipeline's behaviour after each negotiation. The combination of a structural check at build time and a behavioural record at run time is the verification model of the compute-location gate. Neither component is a cryptographic proof, and the verification model does not depend on one; the boundary is stated in Section 11.

11. Relation to Prior Art

The compute-location gate is structurally distinct from the prior-art families with which it is most likely to be confused.

Differential privacy [DWORK] and the broader statistical-disclosure-control literature address the population-versus-individual boundary at the algorithmic layer: a query mechanism adds calibrated noise so that the presence or absence of any single record is statistically masked, and a privacy budget composes the guarantee across queries. The compute-location gate addresses a different boundary at a different layer. The cohort floor of Section 5 is not a privacy budget: it does not compose additively across queries, and it

calibrates no noise. It is a categorical admissibility threshold: below the floor, the passive aggregate class is simply not the applicable provenance class, and the inference is not computed server-side at all. Differential privacy makes a server-side individual-sensitive computation safe to release; the compute-location gate routes the individual-attributable computation off the server entirely. The two are composable: a passive aggregate inference admitted by the cohort floor MAY additionally apply differential privacy to its population-level output. They are not substitutes.

Decentralised personal-data architectures (Solid [SOLID] and comparable personal-data-store designs) move the _storage_ of personal data to a principal-controlled pod and govern the _read path_ through access control. The compute-location gate governs the _compute path_: it determines where a derivation is permitted to run, not merely where its result is stored or who may read it. A personal-data store can hold a passive-individual trait that was nonetheless computed server-side and copied to the pod; the compute-location gate precludes the server-side computation in the first place. The two are complementary (a device-local daemon (Section 4) may use a personal-data store as its retention surface), but the gate's contribution is the routing of computation, which a storage-layer architecture does not address.

Consent-management and access-control frameworks generally govern which parties may read which data. The compute-location gate's consent classes (Section 6) govern, in addition, the provenance class under which an inference may be _brought into existence_. A consent framework that admits a reader of a derived trait has not, by that admission, said anything about whether the trait may be derived server-side from passively observed individual signal; that is the question the provenance-class consent of this memo answers.

The contribution of this memo is the wire-layer composition of the compute-path question with provenance-class consent: a negotiation, ahead of inference, that routes computation by provenance class and refuses at the wire layer when the provenance class is not consented.

12. Scope Boundary: What This Memo Does Not Specify

This memo specifies the compute-location gate: provenance-class negotiation, the routing function, consent-class matching, and the wire-layer refusal. It deliberately does not specify several adjacent mechanisms, and an implementer SHOULD NOT read the gate as providing them.

This memo does not specify a cryptographic proof, attestation, or guarantee that any specified data category, provenance class, or derivation pathway was excluded from a particular computation. The gate's enforcement model is consent-class matching plus wire-layer rejection (Section 6) and its verification model is per-event audit plus build-time data-flow checking (Section 8). A refusal under this memo asserts that an inference was not performed; it does not produce, and does not rely on, any object that proves (verifiably without access to the underlying data) that a data category did not contribute to some output. Such a proof is a different mechanism, addressed by separate work, and is outside the scope of this specification. An implementer requiring a cryptographic exclusion guarantee MUST NOT infer one from the audit record or the build-time check described here; neither is such a proof, and the compute-location gate does not become one by composition.

This memo does not specify the internal inference algorithms, the trait taxonomy, the identity-vector representation, or the device-local daemon's storage format. These are implementation matters of an inference service; the memo specifies only the wire surface at which compute location is negotiated.

This memo does not specify a discovery mechanism, a transport, or a handle namespace. It composes with [MCPDNS], the principal's existing transport, and [IDPRONOUNS] respectively.

13. Implementation Status

A reference implementation of the compute-location gate is operated by the present author against a production identity-inference service reachable at the ~truealter.com substrate. The reference deployment classifies input signals into the three provenance classes of Section 3, performs the negotiation of Section 6 ahead of inference, routes by the function of Section 7, returns the wire-layer refusal of Section 6.3 on a consent miss, declares a cohort floor of 1000 for passive aggregate inference, and runs the build-time data-flow verification of Section 8.2 as a gate on its inference-pipeline build. The device-local daemon of Section 4 is the reference deployment's local-execution surface for passive individual inference.

In the spirit of [RFC7942], the present author notes that this section documents implementation experience and is expected to be removed before the document advances beyond the Independent Stream. No claim of interoperability is made; the reference deployment is a single service operated by the specification's author.

14. IANA Considerations

This memo requests no IANA action.

The negotiation field names and the negotiation-response type names used in Sections 6 and 7 are illustrative of the reference deployment described in Section 12. Conforming inference services MAY name the corresponding fields and response types by any convention consistent with their transport's addressing primitive; the load-bearing contribution of this memo is the three-provenance-class routing function and the wire-layer refusal, not the field names. No new DNS record types, transport identifiers, port numbers, URI schemes, or media types are introduced.

15. Security Considerations

The compute-location gate concentrates an admissibility decision (where an identity inference may compute) at a wire-layer negotiation performed ahead of inference. The following considerations arise.

15.1. Provenance-Class Misassertion

A client may assert a provenance class that does not correspond to how the input signal was actually observed; for example, asserting the active class for a signal that was passively observed, so as to route an individual-attributable inference to server-side compute. The asserted provenance class is a claim, and the inference service MUST NOT treat it as self-certifying. The service SHALL establish the provenance class from substrate it observes (the act that produced the signal, the stream the signal arrived on, the presence or absence of a principal-initiated trigger) and SHALL refuse a request whose asserted provenance class is inconsistent with the observed substrate. A provenance class established from observed substrate, rather than accepted from a client assertion, is the integrity foundation of the gate.

15.2. Cohort-Floor Evasion

An attacker may attempt to extract an individual-attributable statement through repeated passive aggregate queries whose cohorts overlap such that the difference between two near-identical cohorts isolates an individual. The cohort floor of Section 5 bounds the size of any single cohort but does not, alone, bound a differencing attack across cohorts. An inference service SHOULD additionally constrain the _composition_ of passive aggregate queries, for example by refusing aggregate queries whose cohorts differ by fewer than the cohort floor, and SHOULD record passive aggregate cohort definitions in the audit log (Section 9) so that a differencing pattern is

detectable post-hoc.

15.3. Redirect Suppression

A redirected negotiation response (Section 6) directs a client to perform a passive-individual inference on the device-local daemon rather than server-side. An attacker positioned between the client and the inference service might suppress or rewrite the redirected response so that the client believes a server-side inference is unavailable and abandons the inference, or believes a server-side inference occurred when it did not. Negotiation responses SHOULD be carried over a channel authenticated by the cryptographic identity envelope of [MCPDNS], so that a consuming client can verify the response bears the inference service's declared signing key, and a suppressed or rewritten response is detectable.

15.4. Device-Local Daemon Compromise

The device-local daemon (Section 4) holds passive-individual derived statements that, by Section 3.3, never reach a server. Compromise of the device-local daemon exposes those statements. The gate does not make the device-local daemon's storage secure (that is the device's responsibility), but it does ensure that the blast radius of a device-local compromise is confined to a single device's passive-individual derivations and does not extend to a server-held aggregate of many principals' passive-individual traits, because no such server-held aggregate exists. The compute-location gate's routing is itself a blast-radius mitigation.

15.5. Audit-Log Integrity

The per-event audit record (Section 9.1) is the after-the-fact evidence that the gate operated. An attacker able to amend or delete audit records could conceal a server-side passive-individual inference. The audit log MUST be append-only; admitted records MUST NOT be retractable or amendable by the inference service operator. Where the audit record is emitted to an organisational identity substrate per [POLICYPROV], the append-only property of that substrate's ingestion surface carries the integrity requirement.

16. Privacy Considerations

The compute-location gate is, in its substance, a privacy mechanism; its routing function is a privacy decision. The considerations of [RFC6973] apply, and three are operative here.

16.1. Provenance Class as a Consent Boundary

The gate’s central privacy property is that consent is keyed to provenance class, not to trait category (Section 6.1). A principal controls not merely _what_ is inferred about them but _from what manner of observation_. A principal may consent to active-class inference of a disposition while withholding consent for passive-individual inference of the same disposition; the gate honours that distinction at the wire layer. An inference service that collapses the distinction, treating consent for a trait category as consent for all provenance classes of that category, has not implemented the privacy property this memo specifies.

16.2. Local-Only Retention of Passive Individual Inference

The passive individual provenance class is routed to device-local compute and device-local retention (Sections 3.3, 4, 7.3). The privacy consequence is that the inference service holds no server-side, individual-attributable representation derived from passively observed individual signal. A principal’s passive individual derivations are, by construction, not present in the inference service’s data holdings, not subject to the service’s breach exposure, and not reachable by a server-side query however authorised. The principal’s later election to contribute such a derivation to a server-held inference is an active-class act (Section 4) and is independently consented.

16.3. Third-Party Stream Inference

A principal’s authorisation of a third-party platform’s data-access grant is not consent for an inference service to infer identity statements from that platform’s stream (Section 6.2). Inference from a third-party stream requires a consent class distinct from, and separately revocable from, the platform authorisation. This separation matters because a principal’s mental model of a platform authorisation is "this platform may use my data on this platform", not "any inference service may derive identity traits from my behaviour on this platform". The gate’s consent classes make the second a separate, explicit, revocable grant. Inference from a third-party stream conducted without such a separate consent class is a privacy violation that the gate is specifically structured to prevent.

16.4. Regulatory Context

The compute-location gate's routing of passive individual inference off the server is consistent with data-minimisation expectations under data-protection regimes generally, and with [GDPR] in particular: a derivation that is never computed server-side produces no server-side personal datum to minimise, retain, or erase. Implementers operating in jurisdictions that categorically prohibit certain inferences in certain contexts (for instance the prohibition under [EUAIACT] of emotion inference in workplace and education settings) should note that the compute-location gate is a routing-and-consent mechanism and does not itself satisfy a categorical prohibition: a categorical prohibition bites regardless of provenance class or compute location, and an inference service subject to one MUST refuse the prohibited inference outright rather than route it. The gate composes with a categorical refusal; it does not replace one.

17. Relation to Companion Memos

This memo composes with four Morrison-family Internet-Drafts.

[MCPDNS] supplies the DNS-based discovery surface by which a client locates an inference service, and the cryptographic identity envelope referenced in Section 13.3. This memo introduces no new DNS records or labels.

[IDPRONOUNS] supplies the ~handle namespace by which principals and substrates are named. This memo introduces no new handle category.

[SUBSTRATE] supplies the substrate-observation posture under which the device-local daemon (Section 4) coordinates with the principal's other sessions; that coordination introduces no path by which passive-individual derived statements reach a server.

[POLICYPROV] supplies the organisational identity substrate to whose audit-signal ingestion surface the per-event audit record (Section 9.1) SHOULD be emitted. An inference service operated alongside an organisational identity substrate inherits that substrate's append-only audit posture for the compute-location gate's audit trail.

The compute-location gate is the wire-layer codification of the first clause of the identity-as-inference principle: no inference without a compute-location gate. The companion memos codify adjacent clauses and surfaces of the same principle; this memo is the clause concerning where inference is permitted to compute.

18. Document History

draft-morrison-compute-location-gate-00 (May 2026):

* Initial submission.

* Defines the three provenance classes (active, passive aggregate, passive individual) and the immutability of a signal’s provenance class.

* Specifies the device-local daemon as the compute location for passive individual inference.

* Specifies the cohort floor for passive aggregate inference.

* Specifies the wire-layer negotiation (inference request, negotiation response) performed ahead of inference.

* Specifies consent-class matching, the distinctness of provenance class from trait category, and the wire-layer refusal.

* Specifies the routing function from provenance class to compute location.

* Specifies the per-event audit record and the build-time data-flow verification.

* States the scope boundary excluding cryptographic exclusion guarantees from the specification.

19. References

19.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[MCPDNS] Morrison, B., "Discovery of Model Context Protocol Servers via DNS TXT Records", 2026, <https://datatracker.ietf.org/doc/draft-morrison-mcp-dns-discovery/>.

[IDPRONOUNS]
Morrison, B., "Identity Pronouns: A Reference-Axis Extension to ~handle Identity Systems", 2026, <https://datatracker.ietf.org/doc/draft-morrison-identity-pronouns/>.

[SUBSTRATE]
Morrison, B., "Substrate-Observation as an Alternative to Envelope Coordination for Concurrent Sessions", 2026, <https://datatracker.ietf.org/doc/draft-morrison-substrate-observation/>.

[POLICYPROV]
Morrison, B., "Policy Provision and Governance Inheritance from an Organisational Identity Substrate", 2026, <https://datatracker.ietf.org/doc/draft-morrison-org-alter-policy-provision/>.

[MCP] Agentic AI Foundation, "Model Context Protocol Specification", 2026, <https://modelcontextprotocol.io>.

19.2. Informative References

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/info/rfc6973>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

[GDPR] European Parliament and Council, "Regulation (EU) 2016/679 (General Data Protection Regulation)", 2016.

[EUAIACT] European Parliament and Council, "Regulation (EU) 2024/1689 Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act)", 2024.

[DWORK] Dwork, C., McSherry, F., Nissim, K., and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis", 2006, <https://www.iacr.org/archive/tcc2006/38760266/38760266.pdf>.

[SOLID] Sambra, A., Mansour, E., Hawke, S., Zereba, M., Capadisli, S., Ghanem, A., Aboulnaga, A., and T. Berners-Lee, "Solid: A Platform for Decentralized Social Applications Based on Linked Data", 2016, <https://dig.csail.mit.edu/2016/solid/>.

Acknowledgements

This memo grew out of internal architectural work on the question of how an identity-inference system should decide not merely who may read a derived trait, but where the derivation is permitted to compute. The realisation that the compute-location question is prior to the access-control question: an access-control refusal arrives after the server has already computed and persisted the datum it declines to serve. That observation is the load-bearing insight behind this specification.

Author’s Address

Blake Morrison
Alter Meridian Pty Ltd (~truealter)
Email: blake@truealter.com
URI: alter:~blake