

Aetheria

Delegated Authority and Execution Governance

Bradley D. Wilson
Aetheria Intelligence

2026 edition

Thirty-seven chapters, cumulative appendices and a complete instructor companion. A case-based study of institutional authority, exact-effect control, execution evidence and recovery.

© 2026 Bradley D. Wilson. All rights reserved. Educational fixtures and sample keys are identified in the companion. Publication identifiers and printer specifications will be assigned for the selected release format.

Source edition and evidence status

Technical cases follow the frozen Aetheria 3.0 Integrated Build Roadmap of 8 October 2026, version 3, interpreted through its R1 source resolution. Operational present tense describes the selected contracts. Worked examples are fictional and their calculations are reproducible; implementation, deployment and StilleNacht qualification require separate acceptance evidence. Manuscript fixture checks are reported as such. [22, 61]

Mara is the fictional council clerk and a separately appointed reviewer of Jonah's outgoing message. Jonah is the mail requester. Imani operates the system; Nadia is a second eligible Operations reviewer. Leena evaluates assurance campaign evidence. Sofia is a fictional field officer, and Ellis is an investigator in the proposed public-safety cases. The names identify case roles, not an installed institution.

Teaching materials

The accompanying Instructor Companion provides a session plan and worked answers for all six exercises in each of Chapters 1–37: 37 sessions and 222 answers. The reproducibility package includes the worked fixtures, a signed evidence laboratory and the complete capability crosswalk. The appendices provide the component glossary, formal-assumptions map, record and timing references, and cumulative bibliography.

How to read this book

This is a technical textbook for engineers designing governed AI actions and auditors evaluating the resulting claims. Institutional staff can follow the fictional cases without implementing the contracts. Start with the failed task in each chapter, use What to retain to consolidate the result, then test it with the six exercises. Appendix A is a cumulative glossary; the Instructor Companion provides worked answers.

People in the cases

Mara researches council minutes and has a separate designated duty reviewing Jonah's outgoing mail. Jonah requests that mail. Imani operates the service and Nadia can perform a second eligible Operations review. Leena evaluates the assurance campaign. Sofia is a field officer, while Ellis investigates an internal case and Officer Reed is affected by its outcome. The latter two domains remain proposed teaching applications. A change of person or task always receives its own authority and evidence checks.

Components at a glance

- Atlas — the user's screen; Access — institutional identity, session and entitlement checks.
- Authorized Context — admitted source reads and later disclosures; Aggregator — bounded computation and model coordination.
- Workflow — durable task progress; Binding — an exact proposed effect and its immutable identity.
- Governance — policy and duties; Human Oversight — the review of one material view.
- Authority — contained delegation and grant issuance; Gateway — the recorded handoff to execution.
- Nullius — isolated grant enforcement and target write; Observer — separately permitted target readback.
- Reconciliation — a claim from admitted attempt and observation evidence; Artifacts — retained exportable evidence.
- Verticals and AppFactory — institution-owned domain definitions and authored applications; StilleNacht — independent assurance campaigns.

What the status words mean

Specified describes a contract or behavior selected in the frozen 3.0 roadmap. Worked means that the printed fixture or calculation has been checked in the supplied companion. Proposed marks a conditional domain or extension that still needs its complete profiles, owners and tests. These labels describe different subjects: a worked fixture does not establish installed acceptance. The source register preserves file paths, while the chapter-end notes collect internal source numbers by section. External literature remains next to the claim it supports. Package identifiers such as C09 and small labels inside diagrams locate source files; the component names above explain the responsibility.

Contents

1 Aetheria and institutional work.....	4
2 Architecture and trust boundaries.....	12
3 Records and verifiable evidence.....	20
4 Atlas sessions and user interaction.....	29
5 Access, tenancy, and current trust.....	35
6 Workflow state, scheduling, and concurrency.....	41
7 Authorized Context and source admission.....	48
8 Snapshots, coverage, and exact council research.....	54
9 The Aggregator and bounded cognition.....	61
10 Model connections, adapters, and credential custody.....	69
11 Inference disclosure, budgets, and uncertain outcomes.....	75
12 Conversation history and governed knowledge reuse.....	82
13 Axiomatrix, ADL, and contract compilation.....	89
14 Machine Binding and exact effects.....	94
15 Governance rules, composition, and obligations.....	99
16 Human Oversight and review validity.....	104
17 Authority mandates and delegation.....	110
18 Grant issuance, currentness, and protected history.....	115
19 Gateway and the Nullius interface.....	124
20 Nullius admission, compilation, and execution.....	130
21 Observation, reconciliation, and supported outcomes.....	136
22 Persistence, ledgers, and evidence history.....	143
23 Evidence exports and offline verification.....	149
24 StilleNacht apparatus and campaign execution.....	157
25 Assurance oracles, adversarial cases, and qualification.....	163
26 Verticals, foundations, and institution profiles.....	171
27 AppFactory authoring, composition, and compilation.....	177
28 Product admission, installation, and updates.....	183
29 Municipal systems, data, and integration.....	189
30 Administration, enrollment, and operational control.....	196
31 Lifecycle, deployment, and recovery.....	203
32 Observability, usage, and operational diagnosis.....	211
33 The clerk research journey.....	218
34 The governed email journey.....	223
35 Field officer analysis as a domain case.....	229
36 Internal Affairs evidence and conduct analysis.....	234
37 Failure analysis and integrated reasoning.....	241
Appendix A Terminology and notation.....	247
Appendix B Record and interface reference.....	260
Appendix C State timing and bounds.....	264
Appendix D Worked fixtures and calculation methods.....	266
Appendix E Exercises and teaching alignment.....	269
Appendix F Sources references and claim traceability.....	271
Appendix G Capability boundaries and open domain specifications.....	274
Cumulative references.....	280

1 Aetheria and institutional work

Mara, a clerk in a fictional municipality, asks an AI assistant to summarize references to artificial intelligence in fifteen years of council minutes. The answer looks useful. Jonah, a colleague, requests an email containing it for his supervisor. Mara is the designated reviewer and approves the message for the exact recipient shown. During final preparation, the assistant adds another recipient because that person seems likely to be interested.

The mail service accepts the changed message, but its response disappears on the network. Jonah sees a timeout and asks the interface to send again. The service now holds two copies of a message addressed to someone Mara never approved.

This imagined failure begins with Mara's research and Jonah's outgoing message. The summary could be perfectly accurate, and the operation would still be wrong. Follow the research request and mail proposal through permission, execution and evidence. Who was allowed to receive the message? What exactly did Mara approve? What happened after the first send?

Learning objectives

- Explain the difference between an answer, an institutional permission and an external effect.
- Trace the major responsibilities from an authenticated request to permitted output.
- Interpret a research count without confusing lexical matches, semantic discussion and complete coverage.
- Explain why current access, execution evidence and independent assurance answer different questions.

1.1 The action that was actually approved

Mara approved Jonah's particular message for a particular recipient. Adding an address changed the proposed action. Losing the response created a second problem: the assistant could no longer tell whether the first attempt had taken effect. Aetheria addresses these problems by preserving the identity of the action through review and execution, then recording what the available evidence supports.

Three statements organize the rest of the chapter. “Here is the summary” describes an answer. “This person may send this message” expresses permission. “The mail service accepted this message” describes an observed result. Each statement calls for its own evidence. The summary needs sources and a method; permission needs institutional rules and a precise action; the result needs evidence from the attempt and the receiving service.

An institution supplies the roles, policies and delegations behind that permission. Consider the difference between a clerk preparing a report and a supervisor approving its distribution. Ferraiolo and Kuhn's work on role-based access control organizes access around the functions people are permitted to perform within an organization. That gives us a useful starting question: which duties and permissions does the clerk's role carry? [40]

Aetheria turns the relevant institutional rules into checks on a request and its proposed action. Authentication establishes the requesting identity; trusted role mappings establish the person's place in the institution. A prompt that says “I am the administrator” remains a statement to evaluate. The system obtains its authority evidence from the institution's configured records.

Lampson's protection model separates the domain making an access from the object and operation being protected. Its rules also govern how access can change. Applied to Jonah's mail case, that directs attention to the sender, recipient and permitted action, and to who may alter those permissions. A broad account label leaves those questions unanswered. [261]

Table 1.1 Three statements about the same task

Statement	Evidence needed	Question to ask
Here is the requested summary.	Permitted source records and a stated research method.	What was counted, and how complete was the search?
This person may send this message.	Current institutional permission for the exact message and recipient.	Does the proposed action match the approved action?
The mail service accepted this message.	Matching evidence from the attempt and the mail service.	Which result does that evidence establish?

1.2 Orientation before the machinery

Follow the work in ordinary language first. Mara asks a question. The system establishes who she is, retrieves records she may use, computes a result and presents an answer she may see. If Jonah asks it to send the resulting email, the work expands: fix the message and recipients, check permission, attempt the send and determine the outcome. These are the responsibilities to recognize before memorizing component names.

A request describes work someone wants done. Context is the information made available for that work. An effect is a proposed change to something outside the system's own decision history, such as creating a message in a mail service. A target is the service or resource that receives that change. Reading minutes and creating an outgoing message therefore raise different questions about access and permission.

Two other terms will recur. A credential is technical material a service uses to authenticate or exercise an allowed operation; a source-reading credential, for example, gives access to a defined read interface. A record is a structured account of a request, decision or event. Later chapters explain how records identify their contents and how signatures protect the statements they carry.

When the text calls a component an owner, it means the component is responsible for a particular decision or kind of evidence. Scope describes the boundaries of that responsibility: which institution, environment, data and operation are involved. Admission is the decision to accept an input for a defined use. In the opening examples, read these terms as “who decides,” “which boundaries apply” and “what has passed the required checks.”

We can now name the first three pieces. Atlas is the working screen where Mara submits her research and Jonah submits his mail request. Authorized Context controls the source-reading path. The Aggregator coordinates reasoning over the information that path releases. Their common home, Aetheria Core, also contains the decision and evidence machinery we will meet as the task develops. For now, follow the request from the screen to permitted records and back to an answer.

1.3 The question must become an exact task

Mara asks, “How often has artificial intelligence been discussed?” What should we count? Meetings containing a mention, matching phrases and minutes spent discussing the subject would produce different answers. Aetheria’s specified council research method counts lexical matches in the authoritative final minutes for eligible meetings. Lexical means that the method follows stated rules for matching words and phrases. An analysis of substantive discussion would need its own declared method.

Under this method, the token ai and the adjacent token pair artificial intelligence count after the prescribed text normalization. The letters inside chair and paid contribute no match. A.I. becomes the separate tokens a and i, so it also contributes none. By contrast, the hyphen in artificial-intelligence separates two adjacent words that still form the required phrase. We can predict these outcomes before asking a model to explain them.

The denominator is the complete population of final, noncancelled meetings in Mara’s permitted catalog and requested time interval. Each meeting contributes one designated authoritative revision of its final minutes at the snapshot. An agenda copy and a transcript of that meeting remain supporting documents rather than extra meetings to count. If two revisions both claim to be authoritative, the conflict appears as a source or completeness problem.

A snapshot fixes the revisions and query context used for the calculation. Imagine collecting the first half of the results on Monday and the rest on Tuesday: the snapshot keeps that work tied to one defined set of revisions. Access remains current throughout. If the relevant access or policy epoch changes, the protocol invalidates continuation and unreleased results. The snapshot stabilizes the evidence while those checks govern its continued use.

1.4 A small research example

Table 1.2 gives Mara five fictional meetings so that we can check the method by hand. For this exercise, they are the complete permitted population of final, noncancelled meetings. Each has one authoritative final minute. All fall in the interval beginning 8 October 2011 at 00:00 UTC and ending 8 October 2026 at 00:00 UTC. The interval includes its starting instant and excludes its ending instant; this is the meaning of half-open.

Table 1.2 Fictional authorized meeting fixture

Meeting	Year	Final minute text	Matches
M001	2012	Council discussed AI.	1
M002	2015	Artificial intelligence tools were discussed. AI was mentioned again.	2
M003	2018	The chair paid the annual fee.	0
M004	2021	The A.I. committee met.	0
M005	2025	Artificial-intelligence procurement was reviewed.	1

Start with M002. Its text contains artificial intelligence and a later AI: two occurrences in one meeting. M001 and M005 each add one occurrence. Three meetings therefore match, and the total is four occurrences. Divide three matching meetings by five eligible meetings to obtain 60 percent. Meeting

membership and phrase frequency answer different questions; measuring discussion time would require different evidence.

A useful result reads: “The complete permitted fixture contains five eligible meetings. Three contain a match under the lexical method, with four matching occurrences in total.” Mara can inspect the method, coverage and permitted citations to the original revisions. The model’s explanation presents the verified calculation in readable language and preserves its quantities.

Now interrupt the scan after M001 and M002. A required page has become unavailable. We have observed two matching meetings and three occurrences in the scanned subset. A partial answer can report those observations with incomplete coverage explicit. A count that depends on the missing material remains null. Zero is reserved for a complete permitted scan that finds zero matches.

Suppose M002 receives a corrected final minute and a later task uses the new snapshot. Its count may change. Readers can reconstruct the difference when each result identifies its source revision and method. This is why the earlier task stays tied to its own snapshot and the later task identifies the revised evidence.

1.5 The same clerk requests restricted phone logs

Keep Mara’s identity unchanged and replace her research question with a request for the mayor’s phone logs. In this scenario, her role permits council research and excludes those logs. Authorized Context evaluates the requested source under that role. It refuses the unsupported read, or returns the appropriate governed result, before the protected contents enter the reasoning process.

Why check at this point? Because putting a record into a model payload already releases it to that processing path. Removing a sentence from the final screen leaves the earlier disclosure intact. Reusable conversations, caches and exported evidence create further release points. Aetheria evaluates the applicable current permissions at source admission and at each later disclosure boundary.

Even the refusal needs care. “You cannot read the seven restricted logs” reveals the number seven. The ordinary clerk response must withhold protected membership, excluded-source counts and restricted identifiers. An administrative investigator may have a separately authorized view, but Mara’s response follows her own entitlement.

We can test this boundary with a simple comparison. Prepare two source states that contain the same permitted records and different hidden records. Submit the same currently authorized request to both. The permitted content, counts, citations and provider payloads must agree under the specified contract. Timing leakage needs a separate measurement: equality of those outputs answers the narrower disclosure question we have actually tested.

1.6 From the draft to a controlled send

Return to the email. Jonah’s draft is candidate content Mara can inspect as reviewer. Sending asks an external service to create a message. Machine Binding fixes that proposal as an immutable effect: the exact target, parameters and payload identity. Governance evaluates the applicable rules; required human review evaluates the displayed proposal; Authority evaluates delegation and issuance conditions. The same effect must survive each check. Adding the extra recipient would require the changed action to be evaluated again.

When the authority conditions hold, Authority issues a grant: bounded permission for the exact execution identity and profile. Gateway carries the approved handoff to Nullius, the protected executor.

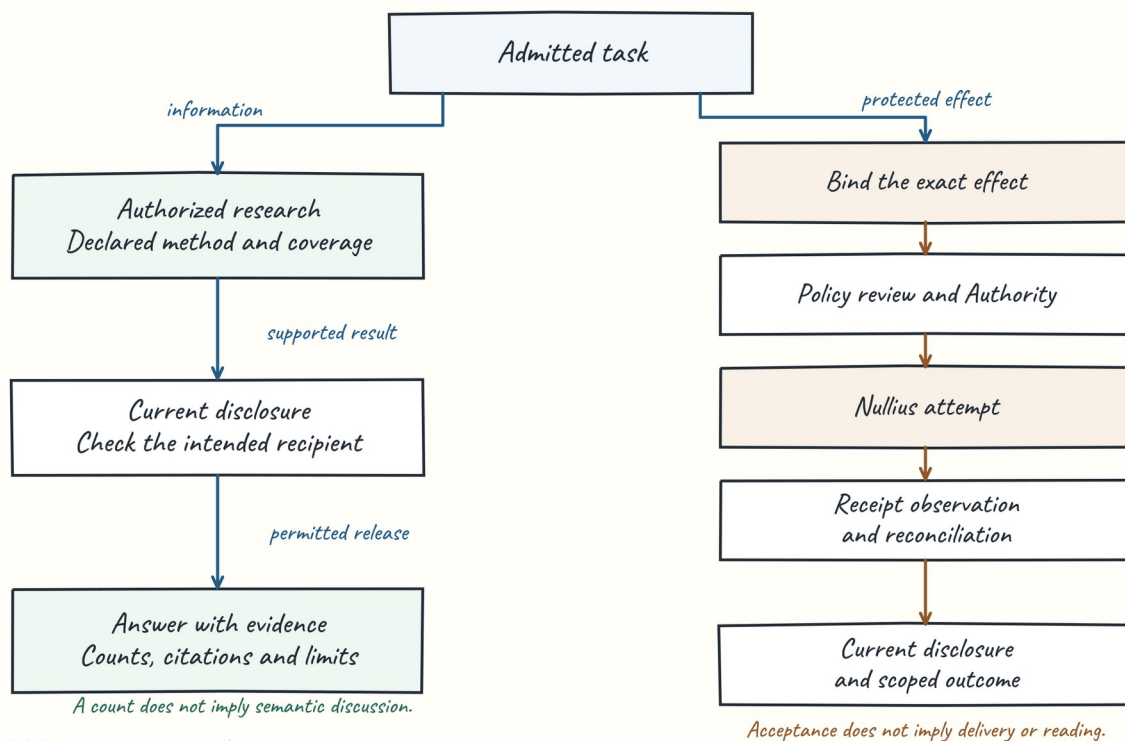
Nullius checks the grant and its own local safety conditions, durably records consumption and the attempt, and retains the resulting evidence. Core controls the decision machinery; Nullius holds the protected writer credential. This division gives the institution a controlled route from permission to an external change.

The target's reply tells us what kind of result we have. A positive mail response establishes acceptance by the selected service. Delivery to a person, and that person's reading of the message, require further evidence. In the synthetic mailbox profile, a separately authorized read can establish the exact stored content when the complete matching chain is present. The reported outcome follows the evidence available for that profile.

Now let the response disappear as it did in the opening story. The operation stays under its original identity and fence while permitted reconciliation gathers evidence. Here, a fence is a retained control that prevents an unresolved operation from being treated as a fresh, safe send. A timeout does not prove that nothing happened. The interface preserves that uncertainty so Jonah can distinguish an unresolved attempt from a confirmed failure.

When is the work complete

Two reasoning paths · Each outcome retains its own evidence and disclosure conditions



F01B / Completion conditions · Not a full workflow transition graph

Figure 1.1 Information and action completion. Each arrow follows the work toward its stated completion condition. Both paths include current disclosure controls. This is a reasoning diagram; the detailed workflow specifies its full set of states and transitions.

1.7 Moving from the clerk to public safety

Next, move to an officer asking for current information about whether a proposed search is lawful. The task requires reported facts and applicable legal and policy sources. Across the office, an Internal Affairs investigator submits a case file concerning alleged excessive force. That analysis requires a way to

distinguish allegations, observations, supporting evidence and contrary evidence. Each task adds demanding domain questions to the access and evidence controls already introduced.

These two cases are conditional teaching applications. The specified expanded workflows cover council research, mail and generic model tasks; a generic task alone does not supply a validated legal-analysis or case-assessment service. Such a service needs its own task and result contracts, authoritative source ownership, jurisdiction and effective-date handling, demonstrated coverage of its files and media, validation, and defined human review and disposition responsibilities.

Keep two questions visible in the officer's example. "May this service retrieve the case file?" is a software access decision. "May the officer conduct this search?" concerns human conduct under the relevant facts and law. Likewise, an investigator's final institutional disposition remains separate from analytical material offered for review. Later case chapters use these distinctions to examine the additional domain machinery.

1.8 Reading the whole map

Wilson's manuscripts offer a conceptual way to revisit the opening story. The Execution Authority Gap separates the ability to cause an action from technical permission, institutional authority and normative legitimacy. Intelligence Is Not Authority develops five families of continuity failure: identifying the act; establishing its basis, competence and authorization; controlling the transition; supporting conclusions with evidence; and preserving responsibility under uncertainty. The extra recipient raises an action-identity problem; the lost response raises an evidence and recovery problem.

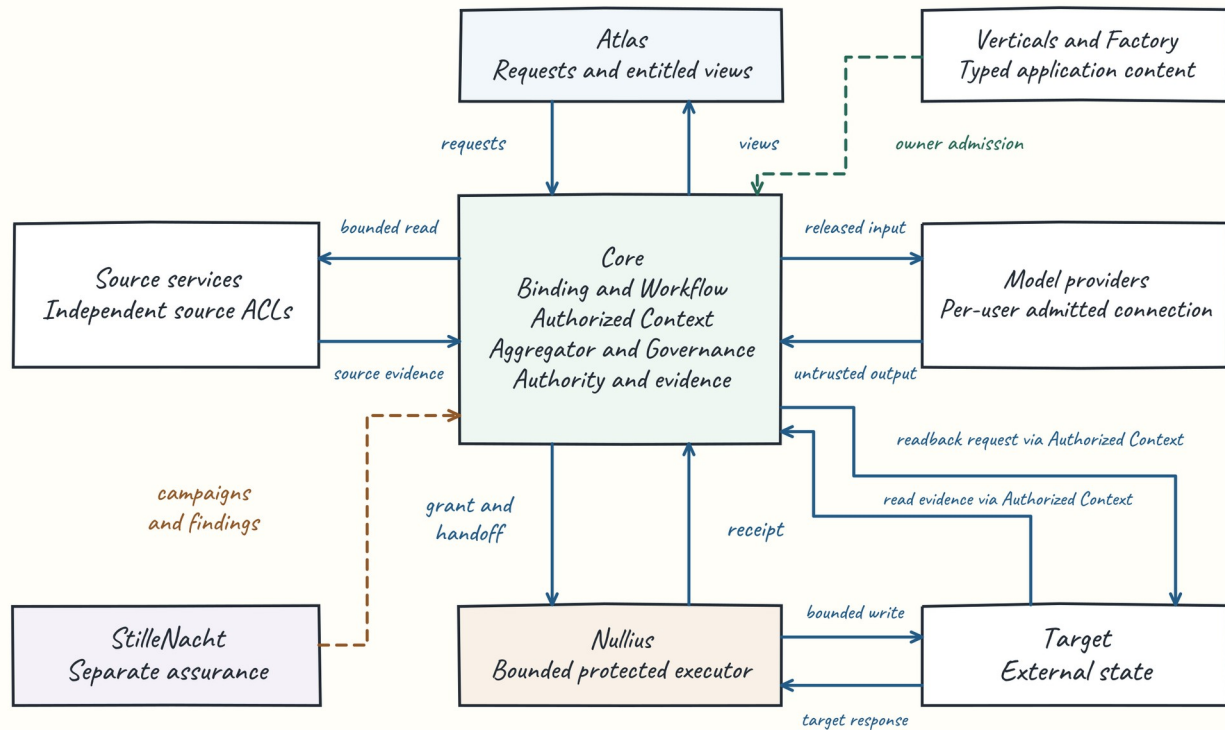
This is a proposed analytical framework. It helps us ask where responsibility could break as work passes between people and software. Institutions still have to judge the legitimacy of their rules, while engineering evidence shows how the represented rules and actions were handled. We will use both kinds of reasoning throughout the book, beginning with the concrete question that follows from the research and mail cases: which component may make each decision?

We have followed a question, a restricted read and an outgoing message. Figure 1.2 now puts that work into a wider picture. Core contains the components that interpret requests, obtain permitted context, evaluate decisions and retain evidence. The Observer gathers separately permitted readback through Authorized Context; reconciliation compares the attempt and observation with the operation's requirements. Notice the return arrows. An account of what happened is part of the work.

Two parts of the map support the wider system. Verticals provide domain definitions and institutional configuration; AppFactory authors and compiles application content. StilleNacht conducts assurance campaigns that assess specified behavior. Installed applications continue through the runtime controls while the authoring Factory is offline, and assurance findings inform lifecycle decisions. Chapter 2 examines these responsibilities and their credential boundaries.

Aetheria as a system of responsibilities

Solid arrows carry requests or evidence • Dashed arrows show admitted content or assurance



F01A / Responsibility map • Core internals and API sequences are intentionally expanded later

Figure 1.2 Whole system responsibilities. Solid arrows carry the requests and evidence identified by their labels. Dashed arrows show content or assurance dependencies. Core is expanded in Chapter 2 and later component chapters. The map groups responsibilities; it leaves deployment and detailed API order to their own diagrams.

What to retain

The council count needs an entitled source set and a declared method. Jonah's proposed mail needs an exact review, and a lost reply leaves the send outcome open.

- Answer — a supported informational result
- Permission — an institutional decision about an exact action
- Effect — the change attempted at an external target

Exercises

Q1.1 Classify each statement as content, permission or outcome: a draft council summary; an approval of an exact outgoing email; a report that a target accepted the email. What additional evidence would be needed for each stronger claim?

Q1.2 Using Table 1.2, calculate eligible meetings, meetings with matches, total matching occurrences and the proportion of meetings with a match. Explain the treatment of M003 and M004.

Q1.3 After M001 and M002 are scanned, a required page becomes unavailable. What can an answer claim, and what must remain incomplete?

Q1.4 A developer proposes loading all records into the model and filtering its answer afterward. Explain the violated boundary and two disclosures that could occur before the final display.

Q1.5 A mail attempt times out. The interface offers a button labeled Resend safely because no receipt arrived. Evaluate that claim.

Q1.6 Explain why permission to retrieve a case file cannot itself answer whether an officer may conduct a search.

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 1.1: [1], [2], [5], [6], [8], [9], [19]; 1.2: [1], [2], [3], [4], [5], [7], [8], [9], [12], [14], [19]; 1.3: [4], [6]; 1.4: [4], [18]; 1.5: [2], [4], [8], [18]; 1.6: [1], [3], [4], [5], [6], [9], [10], [19], [31]; 1.7: [2], [7], [8], [18], [36], [37]; 1.8: [1], [3], [7], [10], [11], [17], [19], [20], [32], [33].

2 Architecture and trust boundaries

Suppose we give the program that drafts Jonah's email both a mail-service credential and the power to approve its own sends. It now has a direct route from a suggestion to an external action. Moving that credential into a protected executor changes what the drafting program can do, even if the screen looks exactly the same. This chapter follows those credentials and decisions through Aetheria. We will ask who holds each capability, which evidence travels between components, and what survives a failure or restore.

Learning objectives

- Assign identity, read access, cognition, policy, authority, execution and observation to their proper owners.
- Distinguish logical modules, credential boundaries, process isolation and administrative independence.
- Trace source reads and observation without granting direct source credentials to the Aggregator or Observer.
- Explain why recovery depends on separate state and restore domains.

2.1 Who decides and who can act

Begin with the word owner. In Aetheria, the owner of a decision is the component responsible for evaluating its conditions and producing its authoritative record. Atlas presents review material; the relevant server-side components establish the reviewer's role and evaluate the response. A database retains the resulting record. This assignment lets us locate the evidence for a decision and the machinery that must defend it.

A trust boundary is crossed when a receiving component must check the sender's input before relying on it. The check depends on the use: an authenticated identity for a request, an allowed source for a read, or a matching signed grant for execution. Scope, exact references, current policy and local safety conditions may all matter. The boundary describes a trust obligation, including when both components share a computer.

Saltzer and Schroeder's protection principles provide an established vocabulary here. Least privilege limits a program to the privileges its job needs. Complete mediation calls for checking authority at every access, including when earlier decisions have been cached. Fail-safe defaults make permission the basis for access. These principles give us questions to ask of an architecture and its implementation. [38]

Apply those questions to Mara's task. Authorized Context controls source reads; Authority issues bounded execution permission; Nullius holds the protected writer. Informational work can finish after the required source and disclosure checks. A protected write additionally passes through the grant and execution path. This assignment is Aetheria's specified arrangement, and its strength depends on the actual credential and process boundaries an installation enforces.

2.2 Atlas and the first binding

Mara starts in Atlas. The interface presents tasks, evidence, review material and outcomes obtained through Core's APIs, limited to what she may see. A browser cache helps restore a view. If her role changes while a tab remains open, the next consequential request is evaluated against current server-side evidence. The older screen is useful for understanding how a stale approval label can arise.

Initial request Binding attaches the authenticated actor and permitted context to the work. Tenant, organization and environment identify security boundaries. The system resolves them through trusted identity mappings rather than accepting whichever labels the caller supplies. A workload acting through a service identity likewise needs its configured role and scope. These bindings make the request accountable to a particular institutional context.

Workflow keeps track of durable progress and coordinates work between components. It can correlate results and resume permitted steps after interruption. The authoritative decision still comes from the component responsible for it: a grant from Authority, source access from the reading path, and an outcome from the required execution and observation evidence. Workflow records progress using those results.

Binding occurs again later for a different reason. The first pass establishes who is asking, in which context, and for what work. Machine Binding fixes the exact proposed effect after a candidate exists, including its target, route, parameters and payload identities. A well-formed proposal therefore still needs this second pass. It gives policy evaluation, review and execution a common action to inspect.

2.3 Reading sources and coordinating reasoning

Authorized Context holds the source reader credential. It evaluates the requesting principal, permitted source boundary and purpose, issues a bounded read, and returns the allowed evidence with its provenance. The source service also enforces its own access rules. Later releases of those bytes are evaluated for their particular recipients and purposes, including disclosure to a model provider.

The Aggregator asks for that context and coordinates reasoning over what is released. It can synthesize results, consult specialists, retain disagreements and propose further work. Source credentials, institutional grant-signing authority and the protected writer remain with their designated components. If the model suggests joining council minutes with phone logs, the new read goes back through Authorized Context under the applicable permissions.

Norm Hardy's confused-deputy example explains a related danger. A compiler with privileged access accepted a caller-supplied output filename and overwrote a billing file. It had applied one source of authority while serving a request that called for another. Hardy's capability solution makes the authority used for an object explicit: the capability identifies the object and authorizes access to it. The lesson for this reading path is to preserve whose request and authority govern each operation. [39]

Miller, Yee and Shapiro develop this argument through object capabilities: explicit authority to a particular resource helps a deputy act with the rights intended for that request. Their analysis also shows why a static list of permissions omits important rules about how authority propagates. For this book, the question becomes concrete: which exact object, operation and delegation reached the component that will act? Aetheria's signed grants have their own checks; the comparison supplies a design question rather than an equivalence claim. [264]

In Aetheria's own contract, a model-generated plan remains a proposal. Authorized Context evaluates the requested read under the relevant principal and scope; a refusal constrains the continuing plan. The DATA_ONLY interpretation also keeps instructions found inside a retrieved document in the role of source content. Authority to change rules or issue grants comes through the separately controlled decision paths.

Sending data to a model introduces another boundary. Governed egress evaluates the connection, model capability, user entitlement and payload release. A per-user connection uses a supported provider credential or connection mode. An ordinary consumer chat subscription is not assumed to supply API

authorization. Local and specialty models likewise require declared capabilities and qualification for their execution profile. This is how the system accommodates different models while retaining control of the data passed to them.

2.4 From policy review to execution

The proposed email is now exact. Governance evaluates the applicable rules and required institutional context. Human review, when required, concerns the displayed proposal and the reviewer's permitted role. Authority evaluates delegation and the conditions for issuing a grant for that same proposal. A policy result, a human response and a grant each record a different decision in this process.

Issuing the grant binds the operation to its permitted execution profile and scope. Two stores take part: Core's ordinary transaction store and the protected issuer journal. Their coordinated protocol has recovery obligations across separate durability domains; it is not one atomic transaction spanning both stores. Issuance records permission. Nullius later records consumption, and target evidence establishes the result of an attempt.

Gateway delivers the handoff to Nullius. The executor verifies the issuer and bindings, applies local stop and safety conditions, durably controls consumption and the original attempt, and uses its constrained adapter to call the target. A valid grant can still meet a local hold. The executor then follows that hold instead of attempting the operation. Any changed target or operation has to return through the applicable authorization process.

Notice the custody arrangement in Figure 2.1. Core's authorizing process has no protected target writer credential, and Nullius has no institutional grant-signing authority. Each can produce the evidence assigned to its role. The separation restricts the damage available through either path; assessing a compromised administrator or a shared host requires the broader threat and custody analysis developed below.

2.5 Finding out what happened

An executor receipt reports what the execution boundary recorded about an attempt. An observation reports what a separately permitted read found at the target. Reconciliation compares these records, their exact references and the result rules for the operation. Those rules determine the evidentiary conclusion. Arrival order alone carries no special authority: a late receipt must pass the same relevant checks as an earlier one.

Trace the readback precisely: Observer sends a request to Authorized Context, which reads the target. Evidence returns through Authorized Context to Observer, which produces the observation record used in reconciliation. The read credential stays with Authorized Context. Observer is independent of the executor's report and writer credential; independence from a compromised Core process or a common administrator is a separate property to demonstrate.

Who owns the credential or decision

Selected production responsibilities • Evidence-signing keys are not grant keys

<i>Owner</i>	<i>Source read credential</i>	<i>Target writer credential</i>	<i>Grant issuance</i>	<i>Primary contribution</i>
<i>Atlas</i>	<i>No</i>	<i>No</i>	<i>No</i>	<i>Entitled view</i>
<i>Aggregator</i>	<i>No</i>	<i>No</i>	<i>No</i>	<i>Scoped cognition</i>
<i>Authorized Context</i>	<i>Owner</i>	<i>No</i>	<i>No</i>	<i>Admitted reads</i>
<i>Authority</i>	<i>No</i>	<i>No</i>	<i>Owner</i>	<i>Exact grant</i>
<i>Nullius</i>	<i>No</i>	<i>Owner</i>	<i>No</i>	<i>Bounded attempt</i>
<i>Observer</i>	<i>Via owner</i>	<i>No</i>	<i>No</i>	<i>Observation evidence</i>
<i>StilleNacht</i>	<i>No production</i>	<i>No production</i>	<i>No</i>	<i>Assurance finding</i>

Observer readback travels through Authorized Context and its independently admitted reader.

FO2A / Ownership matrix • No protocol arrows are used in this figure

Figure 2.1 Credential and decision ownership. “Via owner” denotes a request mediated by the credential holder. The matrix concerns the named responsibilities; evidence-signing keys remain specific to their purposes.

Who is allowed to perform that read? The answer may be a separately configured observation service. Suppose Mara loses access after an earlier send. The service can continue permitted reconciliation under its own current read rights, while any disclosure back to Mara is checked against hers. The operation can remain accountable after the initiating user’s permissions change.

For the specified mail profile, DENIED or CANCELLED_PRE_EFFECT requires affirmative protected evidence excluding an attempt. An empty target view or a missing receipt leaves that question open. Conversely, a matching observation in the synthetic mailbox can support verified success when the required authority, attempt and result chain also matches. Both conclusions depend on their stated evidence conditions.

2.6 What a restore must preserve

BasalDB provides typed persistence for Core’s records and views. The architecture separates ordinary Core history and outbox state from protected control state, Nullius consumption and send state, the external target’s state, and assurance records. These stores serve different recovery obligations. We need to know which of them a restore can roll back together.

Take a backup of ordinary Core state, attempt a send, and then restore that backup. The restored history predates the send. Nullius must still remember the consumed grant or unresolved attempt. Protected consumption and recovery fences therefore survive outside the ordinary Core restore domain. The same requirement applies to the relevant protected issuance floors and currentness evidence.

A restore domain is the set of state that one restore operation or administrative power can roll back. Separate database schemas share a recovery risk if one restore-all operation rewinds them together. The specified design calls for distinct credentials, storage and backup boundaries appropriate to these responsibilities. Resisting an administrator who controls every local store also requires a surviving external monotonic reference.

Monotonic means that the protected reference retains the relevant advancement within the stated threat boundary. It prevents a recovery claim from quietly assuming that earlier use can be forgotten. An engineering host under one operator can demonstrate functional behavior; independent custody and release qualification require additional evidence about who controls the relevant stores, assessment machinery and surviving anchors. Figure 2.2 separates those questions.

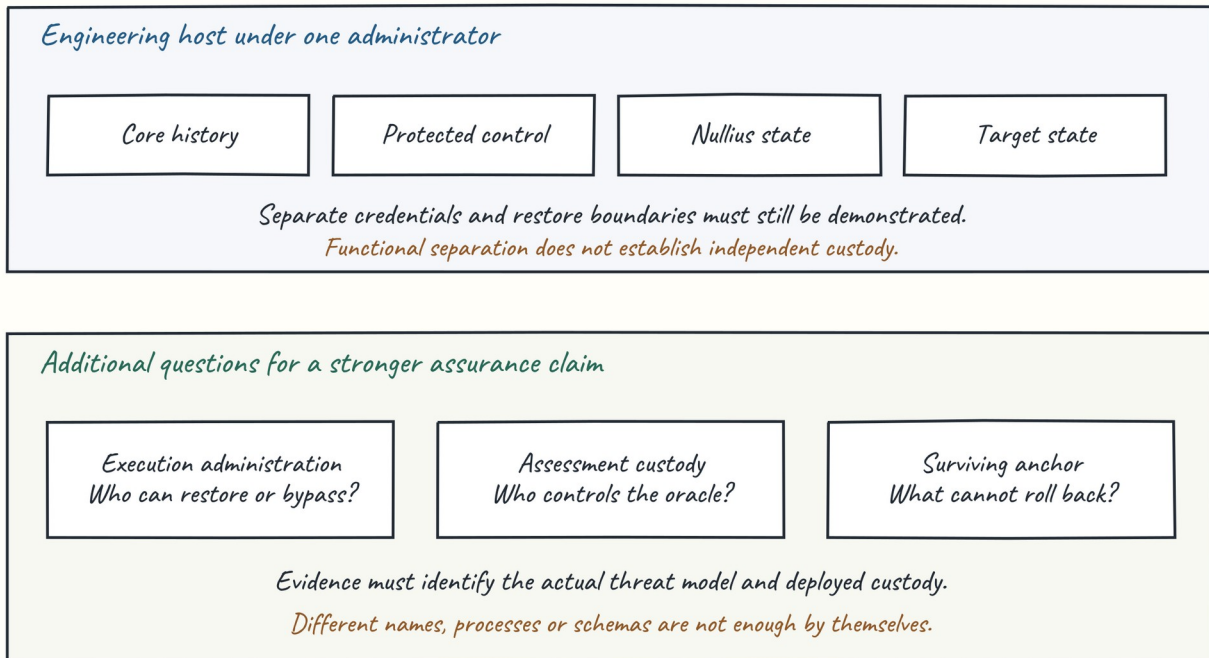
2.7 Assurance and application content

StilleNacht assesses bounded claims through assurance campaigns, reference checks, conformance work and adjudication. Its critical oracle needs an independent basis for judgment. Simply calling Core's own acceptance function would repeat the answer being assessed. The claimed assurance level therefore determines what must be demonstrated about both implementation independence and custody. StilleNacht has neither production mutation authority nor institutional grant authority.

Assurance findings can support lifecycle promotion or holds. A release decision concerns a declared system, configuration, evidence set and set of claims. An action decision concerns an exact operation under current institutional conditions. Keeping these assessment units distinct allows campaigns to inform lifecycle controls while the runtime components evaluate individual actions.

Separation depends on the claim

Logical owners / credential custody / restore domains / administrator powers



F02B / Containment and review questions • Not an installed deployment or certification claim

Figure 2.2 Custody and recovery questions. The upper panel groups responsibilities on a common engineering host. The lower panel identifies additional administrative and custody questions for a stronger assurance claim. Boxes denote containment and responsibility.

Verticals supply typed domain meaning, workflow profiles and institution-specific configuration. AppFactory authors and compiles that application content. Publication, installation and activation are separate events, each with its required controls. Once deployed, an application uses the runtime decision components and continues to enforce their contracts while the Factory is offline.

This division also guides architectural change. Two pure utility modules may share a process when their combination preserves the required privilege boundary. Untrusted model, tool or plug-in execution must stay outside the authorization signer's process. Evaluate a proposed change by the capabilities it joins and the failures it can contain. The appropriate number of services follows from that analysis.

2.8 Reviewing the boundary in practice

Let us test the email path with four interruptions. First, the draft is complete but a required reviewer is unavailable. The permitted draft and its evidence can remain available while the send waits for the required review. Second, a grant reaches Nullius after a local hold takes effect. The executor applies its current local admission rules and records the resulting state.

Third, the target may have accepted the operation, but its reply is lost. Protected send state and the original operation identity remain in force during reconciliation. Fourth, observation establishes the required target state after Mara has lost permission to see the result. The operation's outcome and disclosure to the requester now have different answers. A useful status display preserves both.

A design review can follow the same route. Start with the authenticated caller and permitted source population. Follow the actual bytes through release checks. For an effect, identify the target, payload, policy and authority evidence, protected writer route and retained attempt state. Finally, ask what observation establishes and what recovery must preserve. Every box and arrow in the architecture should help answer one of these concrete questions.

Wilson's systems manuscript studies exact-effect authorization using failure classes that include changes after approval, replay, bypass and uncertain provider responses. Its evaluation concerns a declared prototype and controlled distribution. That historical work helps frame these review questions. Evidence for the present system must address its own installation, isolation, custody and operational conditions.

2.9 Carrying a formal claim into an implementation

A formal model is useful when its assumptions can be located in the system being built. In *From Thought to Action*, assumptions A1–A10 define the conditions under which the governed-execution transition rules support their safety claims. The paper supplies definitions and proof sketches. Appendix B.4 maps those assumptions to the owners and controls taught in this book.

The mapping needs more than a change of component names. The paper places verification at a gateway in its abstract transition system. The selected architecture distributes the corresponding work across request admission, Binding, Governance, Authority, Gateway, Nullius and their protected state. An implementation must show that these boundaries collectively enforce the modeled transition and that no credential or alternate route bypasses them.

Consider revocation. The model assumes that verification observes relevant revocation state before dispatch. The selected system has immediate local restrictions and bounded evidence of remote state. A five-second permitted status window admits the possibility that remote state changes after the observation. A theorem about globally current revocation therefore requires a stronger synchronization assumption than that window supplies. A bounded-freshness claim instead names the admitted observation, its age, the local restrictions and the dispatch boundary at which they were checked.

The same care applies to the time of an external effect. A valid local dispatch can precede a provider's later acceptance or commit. An effect-time authorization claim needs participation from that provider or a separately justified protocol. The local safety argument remains valuable when stated at its actual boundary.

To review a build, write the claim, its assumptions, the mechanism enforcing each assumption and the evidence that the mechanism is installed. Appendix B.4 supplies the map; Chapters 24 and 25 explain how to challenge the resulting claims. The textbook's worked calculations support their declared examples. Installed-system evidence supplies the next step.

What to retain

Trace a permission and a writer credential to different owners. An architecture drawing becomes a stronger claim only when the installed controls and evidence are identified.

- Trust boundary — where one owner must check another's input
- Writer credential — the protected means to change a target
- Assumption — a condition a stated guarantee depends on

Exercises

Q2.1 An Atlas page shows Approved in browser storage. The server has since revoked the relevant role. Which state controls a new consequential request, and why?

Q2.2 Draw the governed observation path and name the holder of the target read credential. What independence does the path provide, and what independence does it not prove?

Q2.3 A deployment places Core and Nullius tables in different schemas inside one database with one restore-all backup. Explain the unresolved recovery risk.

Q2.4 An Aggregator specialist recommends a different email recipient after approval. Which earlier fact can it reuse, and which authority claim can it not reuse?

Q2.5 Explain why a successful StilleNacht campaign is neither a grant for all future actions nor a call that every task must synchronously make.

Q2.6 A developer wants to combine two deterministic contract utilities in one process and also load arbitrary specialist plug-ins beside the grant signer. Evaluate the proposals separately.

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 2.1: [2], [3], [7], [8], [9], [10], [12], [28]; 2.2: [1], [5], [6], [7], [8], [9], [12], [30]; 2.3: [2], [3], [4], [8], [12], [16], [18]; 2.4: [1], [2], [3], [5], [6], [9], [10], [31]; 2.5: [3], [8], [9], [10], [12], [17], [19]; 2.6: [3], [6], [21], [22], [28], [31]; 2.7: [3], [6], [7], [11], [12], [20], [21], [22]; 2.8: [1], [3], [5], [6], [10], [17], [19], [31], [34]; 2.9: [61], [265].

3 Records and verifiable evidence

A reviewer approves one message, and the executor receives another. To detect the substitution, the system needs a reliable way to compare their contents. Yet two copies of a structured record can also differ harmlessly in spacing or property order. Which differences matter? This chapter works from that question to the exact bytes behind Aetheria's records, digests and signatures. We will inspect a small specimen, change it in controlled ways and predict which checks succeed.

Learning objectives

- Distinguish a schema, a record, a payload, a canonical digest and a digital signature.
- Construct the selected native signature message and distinguish it from a stored-record digest.
- Explain strict parsing, canonicalization, trusted schema selection and bounded reference closure.
- Evaluate what structural validity and cryptographic integrity establish, and what they leave to semantic owners.

3.1 Records connect the work to its evidence

A record is a structured statement with a defined purpose. A request identifies wanted work; a grant states bounded permission; a receipt reports an attempt; an observation reports a read of the target. Each kind has a producer and consumers with specified responsibilities. References connect the records into an account of the work. Their kinds tell the reader which question each statement can answer.

A schema describes the permitted shape of a record: its required properties, value types, constants and other structural constraints. A contract also defines meaning, relationships, producer roles and behavior. Think of checking an expiry field. A schema can establish that the field has the right form; a later decision must evaluate the time it names. A record that has passed structural validation is ready for the remaining checks required by its use.

Axiomatrix supplies deterministic interpretation and verification operations for these records. ADL describes and compiles declarative contract and profile content. Together they provide machinery the decision components can use. Policy evaluation, institutional grant issuance and target execution remain with their respective components. When an interface returns a value named `ValidatedRecord`, the name denotes structural validation; an authorized effect requires the additional authorization conditions.

Consider `source_blob`, the record used here for source material. It carries provenance, scope, a content digest, media type, source identity and revision, an access-policy reference, and an interpretation. `DATA_ONLY` identifies its contents as data. Thus an instruction found inside a document can be quoted or analyzed while the system's installed rules continue to govern its use.

The W3C PROV-DM model places provenance in a broader vocabulary of entities, activities and agents. It describes relationships such as use, generation and derivation. We can use that vocabulary to ask how a set of council minutes contributed to a calculation and which service performed it. Recording a derivation helps a reader trace the result to its inputs. Aetheria's native contracts specify the particular records and verification rules used in this chapter. [41]

3.2 Three objects that need an identity

A computer stores these objects as bytes. A byte contains eight bits; a bit is a binary value, 0 or 1. A cryptographic hash maps a byte sequence to a fixed-length digest. Here, we use that digest to identify

the bytes under a stated representation rule. Before comparing two hashes, ask what each one covers. Aetheria uses three different inputs in the example that follows. [27]

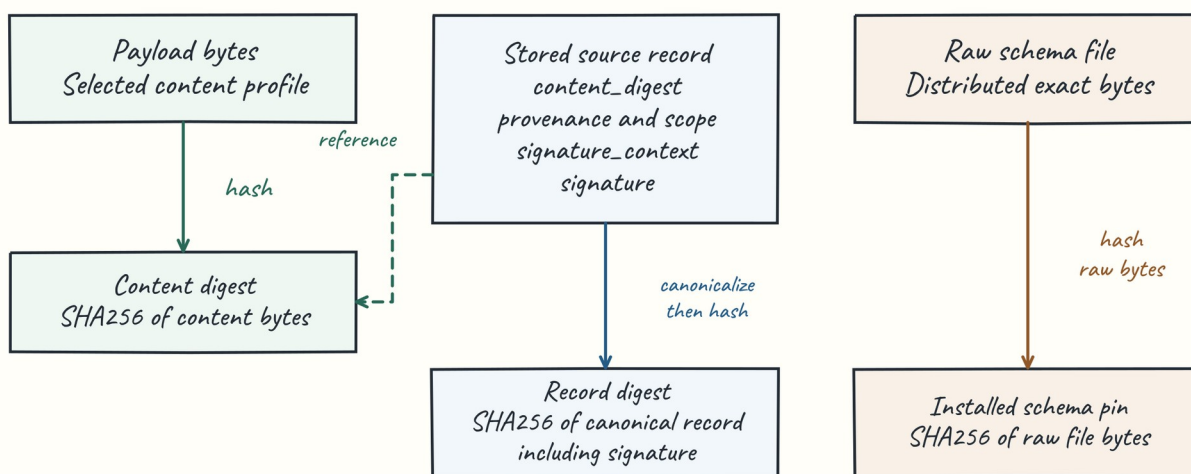
First comes the content payload: the material the record refers to. Its digest is stored in `source_blob` as `content_digest`. For an arbitrary source file, the relevant profile can identify the exact file bytes; a structured `DATA_ONLY` profile may instead require canonical JSON bytes. Canonicalization supplies a deterministic representation of the permitted structured value. The content's declared profile selects the rule.

Second comes the complete stored record. Its native digest covers the canonical JSON representation of the whole signed record, including its signature. A signed metadata change or a changed signature therefore changes the stored-record identity even when the payload stays the same. A consumer resolves the exact kind and digest named by its reference.

Third comes the schema file installed from the catalog. Its pin covers the raw distributed file bytes. Reformatting that file changes the hash even if parsing it would produce the same object. Updating the schema file therefore requires the applicable new pin and controlled adoption. The three identities belong to the payload, the signed record and the installed schema file respectively.

Three byte identities

Every digest must say which bytes it covers



Same parsed JSON can have different raw file bytes.

A signature message is a fourth byte sequence – see Section 3.5.

F03A / Computation and reference relationships • Not network traffic

Figure 3.1 Three byte identities. Solid arrows denote digest computations. The dashed arrow is the content reference carried inside the source record. Each digest identifies the bytes named by its own rule.

Table 3.1 What is being identified?

Object	Bytes used for identity	Further checks required
Content payload	Exact content bytes, or canonical bytes if the selected payload profile requires them	Source reliability, permitted use and current access.
Stored signed record	Canonical JSON of the whole record, including signature	The authority and outcome conditions relevant to its use.
Pinned schema file	Raw distributed file bytes	The full contract semantics for each record instance.

3.3 Parse the input before computing its identity

Suppose an input contains the same property twice with different values. One parser might retain the first value and another the last. Hashing after those two interpretations would identify different objects. Aetheria's native profile avoids that ambiguity by requiring bounded, strict UTF-8 JSON parsing before canonicalization. It rejects representations outside the profile at this first stage.

The permitted value set consists of null, booleans, strings, safe integers, arrays and objects with string keys. Floating-point values, nonfinite numbers, decimal or exponent numeric tokens, duplicate keys, a byte-order mark, lone surrogate characters and trailing documents are rejected. The parser preserves the accepted input for validation. It must not insert defaults, coerce strings into numbers or discard unfamiliar fields to make an input pass.

The safe-integer interval runs from $-9,007,199,254,740,991$ through $9,007,199,254,740,991$. The bound concerns exact numeric representation across the languages governed by the profile. A value beyond it is rejected rather than rounded. Identifiers represented as strings stay strings, including an identifier that happens to contain only digits.

Two public specifications provide the background: RFC 8259 defines the JSON interchange format, and RFC 8785 defines the JSON Canonicalization Scheme, or JCS. Aetheria's installed native profile imposes the additional restrictions just listed. Table 3.2 applies that profile to several inputs that may look similar to a human reader. [24, 26]

Table 3.2 Parse outcomes in the selected native profile

Input fragment	Outcome	Reason
<code>{"n":1}</code>	Potentially admissible shape	Integer token; full schema and semantics still apply.
<code>{"n":1.0}</code>	Reject	Decimal numeric token is not admitted.
<code>{"n":1e0}</code>	Reject	Exponent numeric token is not admitted.
<code>{"n":1,"n":2}</code>	Reject	Duplicate keys cannot be resolved by last-value wins.
<code>{"n":"1"}</code>	Keep as a string	Do not coerce; a schema requiring an integer then rejects it.
<code>{"n":1} {"n":2}</code>	Reject	A second trailing JSON document is not permitted.

3.4 What canonicalization preserves

Canonicalization removes insignificant JSON whitespace and orders object properties deterministically, while preserving array order. Aetheria's JCS subset orders property names by UTF-16 code units. This rule matters when implementations use different programming languages: a default sort by Unicode code point can give a different result for some characters. [24]

Compare the keys U+10000 and U+E000. In UTF-16, U+10000 begins with the code unit D800, so it sorts before E000. A numeric code-point comparison reverses the pair because E000 is smaller than 10000. The example uses the code-point labels so that we can inspect the ordering even if a particular font cannot display the characters.

Strings retain their exact Unicode contents. A composed accented character and a visually equivalent base character followed by a combining mark can therefore yield different canonical bytes. The council matcher's normalization serves a different purpose: matching words according to its domain method. Record canonicalization preserves the values being signed, including distinctions that the search method may treat differently.

Array order is preserved too. Sorting a recipient list, evidence list or step list after signing changes the signed value. If a domain permits a transformation before binding, its effect profile must define that transformation. We can then distinguish the domain's rule for producing the value from the generic rule for representing it as bytes.

The source files use two labels for the same declared safe-integer algorithm: RFC8785+safe-integers-v1 on the wire and rfc8785-ijson-safe-integer-v1 in the SDK. The explicit alias establishes algorithm agreement where the profile allows it. Verification still preserves the actual signed signature_context. Rewriting the label inside a signed record would alter the value being verified.

3.5 Constructing the signature message

A digest identifies bytes under its hashing rule. A digital signature lets a verifier check a particular message against a public key. The native profile uses SHA-256 for 256-bit digests and Ed25519 for signatures. It defines an exact signing message that binds the record's content and declared kind. We can construct that message step by step. [25, 27]

Call the complete stored signed record R. Make U by removing only the top-level signature property from R; retain signature_context and every other field. Let JCS(U) denote the canonical UTF-8 bytes of U under the installed profile. To form M, take the UTF-8 encoding of R.schema_version, add one newline byte 0A, and append JCS(U). Ed25519 signs M directly.

```
U = R with only top-level "signature" removed
M = UTF8(R.schema_version + "\n") || JCS(U)
S = Ed25519.Sign(private_key, M)
R.signature = unpadded_base64url(S)
record_digest = SHA256(JCS(R))
```

Specification notation. The symbol || means byte concatenation. The displayed \n represents one newline byte in the signed message, not two literal characters. Private-key custody belongs to the owning signer.

The resulting signature has 64 bytes and is encoded in the record as 86 unpadded base64url characters. signature_context is an Aetheria JSON field included in M; it does not select the separate Ed25519ctx algorithm. Likewise, signing SHA256(M) would change the specified protocol. Reproducing the exact message construction is essential for two implementations to verify one another's records. [25]

Once the signature has been added to R, compute the stored-record digest over JCS(R). The order resolves the apparent circularity: first sign the unsigned form, then identify the complete signed result. Later progress is retained in the separately defined state or records. The signed immutable payload keeps the values whose identity and signature were established.

3.6 Examining a classroom specimen

The examples folder contains a fictional source_blob, its content payload, exact signature-message bytes, a teaching public key and a verification report. It was signed with a temporary teaching key whose private part is omitted from the package. The key is deliberately uninstalled. We can use it to check the cryptography while keeping trust admission as a separate exercise.

Begin with the payload below. JSON represents an object as named properties inside braces; here the names are meeting_id and text, and both values are strings. All characters in this small example belong to ASCII and occupy one byte each in UTF-8. The specimen uses only ASCII keys and strings, making its canonical bytes straightforward to inspect. Its authoring checks cover this restricted fixture rather than a complete canonicalizer conformance suite.

```
{"meeting_id":"M001","text":"AI"}
```

Classroom payload. The stored payload has no final newline. Exact byte lengths, digests and the public verification key are supplied in examples/verification.json.

Count the characters in the payload, including quotation marks, braces, the colon after each property name and the comma between properties. There are 33 bytes, with no final newline. The supplied signature message contains 718 bytes; the full canonical signed record contains 784. Each length concerns a different input. The payload digest begins 3dbc16a4e4d15d54; the example files retain its full 64 hexadecimal digits for verification.

Read the surrounding record next. Its binding names teaching, fictional-city and classroom as tenant, organization and environment. Its source revision is revision-1, its policy reference is classroom-policy-uninstalled, and its producer role is SOURCE_INGRESS. These values make the exercise's context explicit. In an installation, the relevant identity and trust checks would evaluate such fields against configured evidence.

Now perform three comparisons. Hash the payload and compare it with content_digest. Reconstruct M from the record and verify its signature with the teaching public key. Finally, compute the complete stored-record digest and compare it with the supplied reference. Keeping the inputs separate makes it possible to explain precisely why a controlled change passes one check and fails another.

Table 3.3 Controlled changes to the specimen

Change	Expected result	Interpretation
Reorder JSON properties and add formatting whitespace	Canonical record digest unchanged; signature still verifies	Equivalent permitted representation of the same parsed record.
Change payload AI to ML, leaving the record untouched	Content digest mismatch; record signature still verifies	The signed assertion is intact, but these are not its referenced content bytes.
Change source_revision in the signed record	Signature verification fails	A signed field was altered after signing.
Present the valid teaching key as an installed issuer	No admission follows from mathematical validity	Trusted role, scope, purpose and currentness are separate inputs.
Reformat the pinned schema file	Raw schema file digest changes	A semantic comparison does not satisfy an exact file pin.

The payload change exposes an easy mistake. If we replace AI with ML while leaving the record untouched, the signed statement inside that record is unchanged. Its signature can still verify. The replacement payload, however, has a different digest and fails the content reference. A complete verification follows the signed reference to the supplied bytes and checks the relationship between them.

3.7 Where the verifier obtains trust

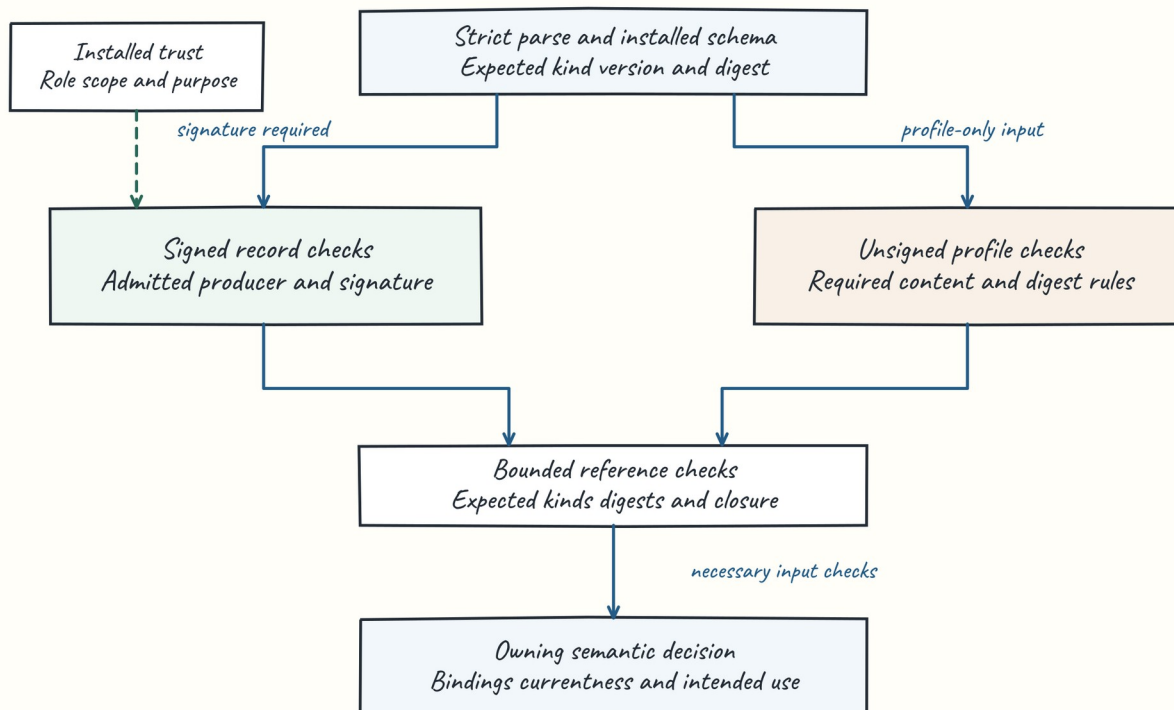
An incoming key_id tells the verifier which key is being claimed. Identity and Trust supply the installed registry that assigns a public key its permitted producer role, algorithm, record kind, tenant, organization, environment, purpose, profile and lifecycle conditions as applicable. The registry supplies the trust basis. Verification evaluates the producer against the relevant boundaries and time: a historical evidence check and a current execution admission can require different temporal conditions.

The role attached to a key matters as much as the mathematics. An observation producer signs observation evidence; a Nullius receipt key signs receipts. Institutional grant issuance requires the appropriate grant authority. Custody adds a second question: which process or administrator can actually use each key? Different key identifiers can coexist within the same compromised process, so a custody assessment follows access to the keys themselves.

Schema resolution follows an equally precise route. The consumer obtains the expected kind, exact version and digest from an installed registry snapshot. It resolves that contract locally within the declared bounds. An unknown version fails this requirement; choosing a nearby version or fetching a schema proposed by incoming data would change the contract being applied. Figure 3.2 shows how these prerequisites support the later decision.

From incoming bytes to a usable owner input

Dependency checks · Failure stops the affected use · No branch issues authority



F03B / Validation dependencies · Not a mandated call order or a generic permission service

Figure 3.2 Dependencies for relying on a record. The signature requirement selects the applicable branch; DATA_ONLY may itself be signed. Each branch leads through the required reference checks to the responsible semantic decision. Arrows express validation dependencies. A failed prerequisite stops the affected use.

3.8 Following references and evaluating meaning

A record may refer to an effect, policy decision or earlier observation. Reference closure means resolving the bounded set of records required for the declared verification purpose and checking their kinds, digests and relationships. The reference names the expected object precisely. Missing records remain visible as missing evidence, and a convenient substitute with a different kind or digest fails the reference check.

The profile also sets cycle rules and resource limits. Some graphs require acyclicity; that requirement belongs to the particular contract. The verifier works on caller-supplied local material within the relevant bounds. It produces a reference report that the responsible decision component can use alongside the remaining semantic checks.

A predicate is a condition that evaluates to true or false. Semantic predicates connect the records to the decision: does the grant bind this exact effect? Do actor and execution identities match the relevant rules? Are time, revocation, usage state and local stop conditions satisfied at the required boundaries? The typed records supply inputs for these comparisons.

Try the recipient substitution from Chapter 1. One record identifies the approved payload; another identifies the changed payload actually proposed for execution. Both can be structurally valid and correctly signed. Their different payload identities still fail the required equality check. By contrast, a

fully supported chain may record a denial or an unresolved attempt. The conclusion follows the relationships and outcome evidence, including when the documented outcome is unknown.

3.9 Reading the verification result

Offline verification examines the supplied evidence artifact without contacting the target or creating another effect. `VALID_FOR_DECLARED_SCOPE` states that the evidence is valid for its declared purpose; the supported outcome may be denial, uncertainty or another permitted result. It does not mean that execution succeeded. Current target state remains `NOT_OBSERVED_OFFLINE` when no new observation has been made.

Integrity and confidentiality answer different questions. Digests and signatures let us examine identity and integrity relationships. Access, disclosure and storage controls protect the information itself. A digest of a guessable value can be compared with candidate guesses, so the digest should not be treated as encryption. The truth of a source statement and the legitimacy of an institutional rule likewise require their relevant substantive assessment.

Return to the three objects on the desk: payload, signed record and schema file. For each, you can now name the bytes behind its identity. You can also explain where the signing message comes from, which registry supplies trust, and why a valid signature can accompany an unresolved outcome. Those distinctions let us examine the later machinery with a concrete question in hand: what evidence makes this particular use of this particular record justified?

What to retain

Name the bytes behind each digest and the trust inputs behind each signature. A valid signature proves a signed statement's origin under its key policy; semantic use needs further checks.

- Canonical bytes — the exact representation being hashed
- Digest — the computed identity of those bytes
- Reference closure — validation of the records a claim depends on

Exercises

Q3.1 A source record signature verifies, but its supplied payload was edited. Which checks fail and which may still pass?

Q3.2 Write the exact native signature message construction. Explain the roles of `signature_context`, the newline and the final stored-record digest.

Q3.3 Why can reformatting an instance leave its canonical digest unchanged while reformatting a pinned schema file invalidates the installed pin?

Q3.4 An implementation sorts all JSON keys by Unicode code point and normalizes strings before signing. Identify the two problems using the examples in this chapter.

Q3.5 A complete artifact verifies offline and includes a well-supported unknown outcome. May the interface label it Verified successful? Explain.

Q3.6 A receipt producer signs a grant-shaped JSON object using a mathematically valid key. Assume its schema shape passes. Why is that still insufficient?

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 3.1: [9], [12], [14], [16]; 3.2: [12], [13], [14], [16], [23]; 3.3: [12], [13], [15], [35]; 3.4: [4], [12], [13], [14], [15], [35]; 3.5: [5],

[6], [12], [13], [14], [15]; 3.6: [8], [9], [12], [13], [16]; 3.7: [3], [8], [9], [12], [13], [14], [29]; 3.8: [5], [6], [9], [10], [12], [14], [19], [29]; 3.9: [2], [9], [13], [29], [33].

4 Atlas sessions and user interaction

Mara clicks Submit, watches the progress indicator, and loses her connection. When the screen returns, the question she most wants answered is simple: did the system receive the work? A careless interface invites her to click again. A careful one preserves the identity of the original submission and asks the service responsible for it what happened.

This chapter follows that recovery. The same design also governs a less visible problem: a response prepared for an earlier session may arrive after the user signs out or changes institutional context. Atlas must keep the person oriented while the server determines which work and information remain available. We begin with the session, then follow a request into durable storage and back to the screen.

Learning objectives

- Trace typed user content through request admission into a durable workflow.
- Distinguish a submission identity from its content and from the resulting workflow identity.
- Explain how current session context and source revisions constrain a displayed response.
- Design truthful reconnect, review and cancellation behavior.

4.1 Establishing the working context

Atlas presents the institution, workspace and available actions for the authenticated person. Access supplies the server-side session and resolves the person's institutional context. A tenant identifies an isolation boundary; the organization and environment further locate the work. Mara may select among choices the server has made available, but typing another tenant into a request cannot move her into that tenant. The receiver compares the request with trusted state.

The browser carries a protected session cookie over verified HTTPS. Requests that change state or begin a disclosure also carry evidence linking them to the current session and the approved site. These protections address cross-site request forgery, in which another site tries to induce an unwanted request using an existing login. Once the request arrives safely, the responsible service evaluates the operation itself. The configuration details are collected in Technical reference 4A.

Institutional sign-in adds a redirect through an approved identity provider. The service must recognize the returning person as part of the same sign-in transaction, then establish the local session Atlas will use. That redirect needs its own tested handling because it deliberately crosses between sites. Chapter 5 follows the resulting identity into institutional membership and access decisions. [68]

An asynchronous screen needs a way to recognize that its context has changed. The adapter derives a non-secret `context_epoch` from the session and credential generations, authentication epoch, institutional domain and resolved workspace. Think of it as the version of the screen's security context. Each request captures that value and an increasing UI request sequence. Before applying a response, Atlas checks the current epoch, route, original operation and non-regressing source revision. An aborted network request may still produce a late response, so aborting it alone is insufficient.

Suppose Mara signs out while a result is returning. The interface masks sensitive content, clears its in-memory projection and discards the late response. If the server has not confirmed sign-out, the workspace remains locked with confirmation pending. A notification to another tab helps it clear its view, but server-side admission continues to protect requests even if the tab misses that notification. On focus or restoration from browser history, actions stay locked until current context is established.

4.2 Turning written intent into admitted content

Mara's question begins as user content. The expanded protocol first sends a typed user-content-submission to the content route. Request admission checks the real session, the allowed payload schema and the role and purpose required to create that content. It then produces a source_blob under the USER_INPUT-scoped ingress key. Chapter 3 explained what that wrapper contributes: exact content identity and an attributed producer. Here its role is narrowly defined as admitted user input.

That route cannot mint a grant, a review decision or a model result. A user may submit text saying “approved,” but it remains user content. Provider configuration likewise references a protected credential slot; the secret belongs in the separate credential-management interface. Keeping these routes distinct prevents a convenient upload mechanism from becoming a way to impersonate another component.

The admitted payload reference becomes part of task-submission. Its fields name the client submission, selected intent, payload reference, requested recipient and requested purpose. The receiving service resolves those requests against current institutional context. The expanded selection supports council research, mail drafting and bounded model tasks. An ambiguous request requires clarification before a privileged interpretation is selected.

Figure 4.1 follows the exchanges. Content admission supplies a reference that a task can use. SubmitTask establishes the durable request and workflow. Status reads then consult the original work. Each return has a limited meaning: recording a submission establishes that the system has accepted work, while later evidence establishes its progress and outcome.

A submission becomes recoverable work

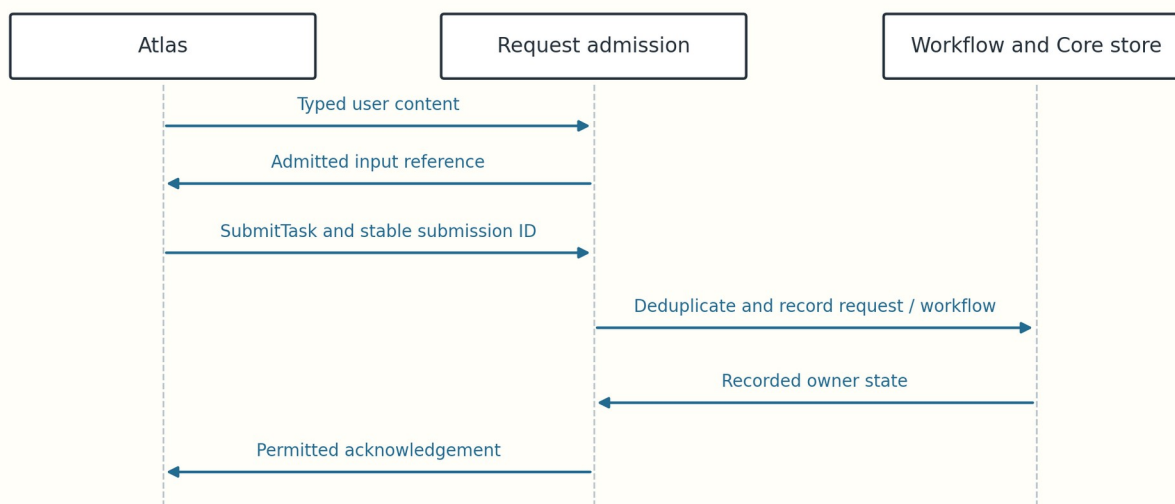


Figure 4.1 Content admission and durable submission. Arrows show logical requests and returns. The durable owner boundary deduplicates the original semantic submission; an acknowledgement records accepted work, not a completed research or effect outcome.

4.3 One deliberate submission through a lost reply

Atlas allocates client_submission_id before the first network call and keeps it attached to that deliberate submission. The backend deduplicates within the authenticated actor and institutional domain,

comparing the canonical semantic content. At the command interface, the server also checks which binding, caller and operation the command belongs to. The content identity excludes the authentication token, allowing a refreshed login token to accompany the same work.

Consider submission S17 asking for the council count over a fixed interval. The owner records workflow W42 but the acknowledgement is lost. Mara's browser enters OUTCOME_UNKNOWN. On reconnect it looks up the original identity under current entitlement. If a duplicate S17 with the same semantic content reaches the owner, deduplication resolves it to the recorded work. Atlas does not automatically send the command again merely because its first response disappeared.

Now change the request from council minutes to phone logs while retaining S17. The owner returns a conflict. Updating the old work would destroy the meaning of its earlier evidence. A genuinely new request needs its own governed admission, and a new identifier cannot bypass an unresolved protected operation or manufacture evidence that the old one never happened.

Table 4.1 Resolving the original submission

Incoming situation	Owner interpretation	Permitted next step
S17 and the same semantic content	Duplicate of the original work	Resolve the recorded disposition under current access.
S17 with changed content	Conflicting meaning under one identity	Return CONFLICT; retain the original history.
Reply lost after submission	Acceptance is uncertain to the browser	Look up S17; preserve OUTCOME_UNKNOWN until supported.
Original identity lost locally	Browser cannot establish safe replacement	Resolve pending work through the backend.

If the local identity has been lost, Atlas helps Mara resolve her pending work through the backend before starting a replacement. A lookup that finds nothing does not establish that no operation occurred: the query may be incomplete, unavailable or directed at state that cannot settle the question. A definite pre-admission refusal allows editing only when owner evidence establishes that no intent was accepted.

The retained initial browser profile keeps draft text in volatile memory. It has no offline queue, automatic startup replay or background synchronization of intent. Those capabilities would introduce further storage, disclosure and replay obligations. An interface can warn about losing unsaved text during navigation without promising to intercept every browser closure.

4.4 Showing progress without inventing an outcome

A projection is an authorized view of owner state, prepared for a particular recipient. Atlas obtains STATUS or RESULT through ReadWorkflow; additional bytes pass through Authorized Context. The projection preserves the material values, scope, source revision and time coverage needed to understand the result. Backend journal handles and private reference closures stay outside ordinary responses.

The display separates request acceptance, review, institutional authorization, dispatch, executor report, observation and reconciled outcome. A single green “Done” label would conceal which of these has actually occurred. For example, “Accepted for processing” tells Mara that the task exists. “Awaiting

review” tells her what dependency remains. “Outcome unknown” preserves an unsettled operation. These are proposed teaching labels for distinct evidence states, rather than new protocol enums.

Transport status also needs interpretation. The command facade can return HTTP 202 for accepted asynchronous work. An authorized status read can return HTTP 200 carrying OUTCOME_UNKNOWN. The HTTP code reports how the request to the API was handled; the payload describes the business operation. A browser must read both.

Time belongs to the evidence as well. The source's own timestamp, the instant at which state was observed, and the time a response arrived answer different questions. Missing source time remains missing. The browser may mark a view stale as time elapses, but only current owner evidence can restore a claim of freshness. A renewed session cannot extend the validity of an older review or control fact.

Reconnect asks about the original work

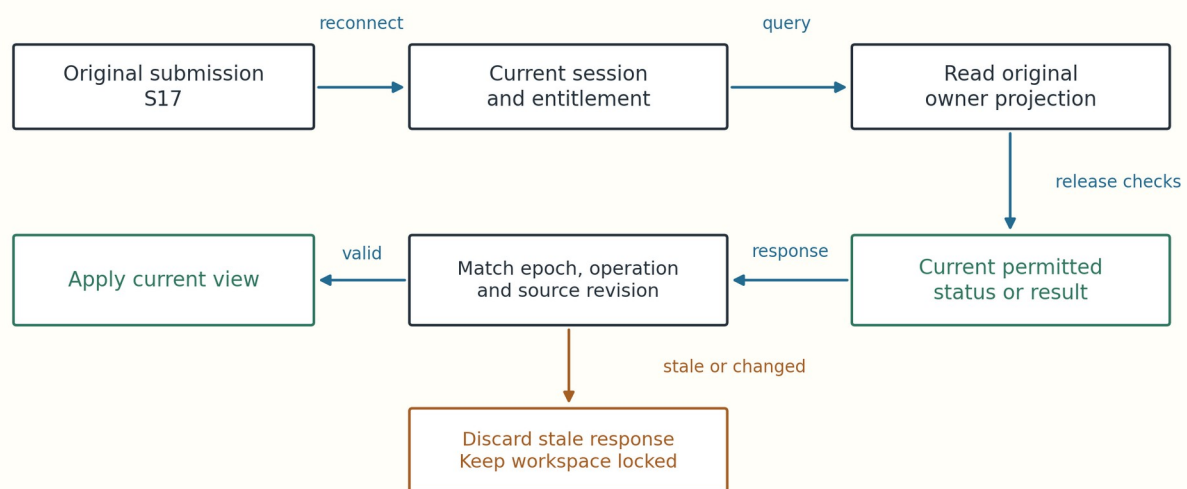


Figure 4.2 Reconnect through authorized status reads. The browser retains original identity and renders only a current compatible projection. A stale response is discarded. Reconnection does not create a replacement command or authorize another external attempt.

4.5 Review presents the action that will be considered

Return to the email from Chapter 1. The reviewer needs the exact recipient, subject, body, purpose, target and other material conditions of the proposal. A screen that hides the recipient prevents meaningful inspection of the action. The display therefore uses the released canonical review model and preserves the fields that determine the effect. If Atlas cannot display them faithfully, it disables approval and explains the problem.

The review response goes to Human Oversight, which authenticates the actual reviewer and checks the original view, challenge, eligibility and round. The browser contributes a deliberate response to that view; it does not supply a self-signed approval record. The expanded route accepts a review-command and produces the applicable owner decision after those checks. The retained screen requirements also distinguish approval, rejection and abstention, without a preselected approval or an Enter key shortcut that silently votes yes.

A changed recipient requires a changed proposal and the applicable new review. The same applies to material content, context or rendering changes that invalidate what was displayed. A reload also has to respect an uncertain earlier disclosure of the review view. Giving it a new view identifier does not make the earlier release disappear. These rules protect continuity between what the person saw and what the institution later evaluates.

The selected single-operator laboratory profile permits same-person confirmation for synthetic content and a non-relaying synthetic mailbox. Its record identifies that actual arrangement. External mail and the retained distinct-person profiles keep their own eligibility requirements. Two accounts used by one person never become two people.

4.6 Cancellation is a request with an answer still to come

Mara can request cancellation through the authorized workflow route. The owner records that request and blocks further work at the applicable boundaries. If information has already left for a provider, cancellation cannot retrieve it. If a mail send may have begun, the original operation stays under its execution and reconciliation controls.

Atlas should therefore show “Cancellation requested” while the owners determine what stopped. A stronger pre-effect cancellation outcome needs affirmative evidence and an enforced fence excluding an attempt. Closing a tab, aborting a fetch or reaching a timeout leaves that question open. Mara needs the screen to tell her what is known, what remains uncertain, and which original operation the system is checking.

Technical reference 4A Browser session settings

The selected local session cookie uses the `_Host-` prefix, Secure and HttpOnly attributes, SameSite=Strict, Path=/ and no Domain attribute. State-changing and disclosure-start routes require a session-bound CSRF token and exact origin checks.

The institutional sign-in redirect needs a separately tested correlation mechanism tied to the initiating transaction. Apply the local cookie rule within that supported flow; a cross-site callback needs its own treatment. [68]

What to retain

A reconnect should resolve the original submission and show the strongest supported state. The screen cannot turn a missing acknowledgement into a safe replacement send.

- Submission identity — the key joining a retry to accepted work
- Outcome unknown — a result that still needs evidence
- Projection — the permitted view of owner state

Exercises

Q4.1 The owner records S17, but Atlas loses the acknowledgement. Explain the next permitted recovery step and the evidence needed to display the recorded task.

Q4.2 Two tabs submit the same identifier with different canonical content. What should the owner do, and why would “last write wins” damage the history?

Q4.3 A result requested under context epoch E7 arrives after sign-out and a new session at E8. List the conditions Atlas checks before rendering it.

Q4.4 An HTTP 200 response contains OUTCOME_UNKNOWN. Write one suitable screen sentence and identify the owner evidence needed for a stronger outcome.

Q4.5 A review screen hides the recipient in a tooltip and labels its primary button Send now. Explain how you would revise the screen and why.

Q4.6 A cancellation request is acknowledged after a model payload may have left. What can the interface truthfully say about future work, prior disclosure and recovery?

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 4.1: [8], [43], [47]; 4.2: [14], [30], [42], [44], [62]; 4.3: [42], [44], [49]; 4.4: [42], [43], [44], [45], [65]; 4.5: [42], [46], [47], [61]; 4.6: [6], [42], [43], [44], [47], [52].

5 Access, tenancy, and current trust

Mara starts a research task at 10:00. At 10:02 her assignment changes and the institution removes her access to the relevant records. At 10:03 the calculation finishes. The task has a valid history, but the answer is waiting at a boundary where permission must be checked again.

This case brings time into access control. Aetheria carries a stable account of who requested the work while consulting current evidence about what that person may do. The distinction allows an institution to preserve accountability through offboarding, credential changes and continuing operations. It also explains why a role printed on a screen cannot settle a later access decision.

Learning objectives

- Identify the person, service and institutional scope behind a request.
- Explain how local enrollment and institutional sign-in produce trusted mappings.
- Trace revocation through source paging, model disclosure and requester output.
- Apply conservative time bounds without promising instantaneous remote revocation.

5.1 Identity establishes the subject of the decision

Authentication supplies evidence about the caller. Aetheria uses that evidence to resolve a stable principal and the caller's current institutional membership. For a human, the principal maps to an enrolled person. For a workload, it maps to a configured software identity with a restricted role and purpose. Both require accountable identity, but their responsibilities differ. A source-reading service cannot take the place of a human reviewer.

A prompt saying “I am the investigator” is a claim made by the caller. Access obtains the relevant assignment from the institution's trusted mappings. Likewise, an email suffix or a familiar domain name cannot establish institutional authority. The enrollment and control-plane process binds the tenant to an institution-approved authority record.

The distinction becomes concrete during role changes. Suppose Mara gains a temporary council-research assignment that ends Friday. The role can support access within that assignment's scope and validity. It does not automatically grant access to personnel files or authority to approve outgoing mail. The requested operation, source classification and intended purpose still matter.

NIST's account of attribute-based access control provides useful terminology: a decision can consider attributes of the subject, resource, operation and environment against policy. That vocabulary helps explain why “clerk” alone is too little information. The following discussion applies those general concepts to Aetheria's selected records and owner checks. [66]

5.2 Scope travels with the work

The expanded command and content profiles distinguish `tenant_id`, `organization_id`, `environment_id`, `principal_id` and `purpose_id`. These fields express different constraints. The tenant separates institutional data boundaries. Organization locates responsibility within that boundary. Environment distinguishes a laboratory from an operational deployment. Principal attributes the work, and purpose identifies the admitted use.

Imagine two fictional municipalities with a clerk named Mara and an identically worded council question. A service that looks up data only by display name or prompt text can confuse the two. Binding

the complete scope and checking it at each receiving owner prevents the ambiguity from being resolved by convenience. An apparent match in one field cannot repair a mismatch in another.

Table 5.1 Scope in a fictional council request

Field	Illustrative value	Question it answers
tenant_id	town-lab-a	Which isolated institutional boundary contains the work?
organization_id	clerk-office	Which organizational context is involved?
environment_id	synthetic-lab	Which deployment environment is admitted?
principal_id	person-mara	Which stable caller is accountable?
purpose_id	council-research	Which admitted use is requested?

The values supplied by a client are requests to be checked. They become usable only when they agree with authenticated state and the installed profile. A source query, workflow result or content read carries that established context into its own boundary. The purpose is not free-form justification that a model can broaden when it encounters a useful document.

Administrative responsibility also has scope. An operator may be able to rotate a connection or manage an installed application while lacking permission to read a resident's private account. The municipal profile explicitly separates platform administration from business-source access. In a classroom review, ask which operation the person may perform and which records that operation permits them to see.

Scope stays attached to the request

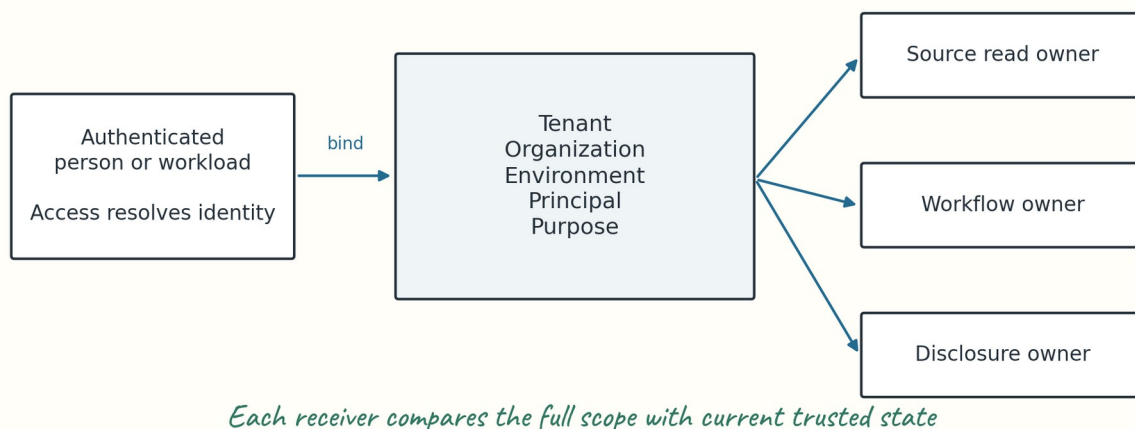


Figure 5.1 Identity scope and receiving owner checks. The arrows carry a bound scope into separate owner decisions. Fields supplied by a caller are compared with authenticated institutional mappings; no branch widens the scope simply because it needs more data.

5.3 Enrollment and institutional sign-in

The retained synthetic profile bootstraps its first administrator through a protected setup path with a one-use, origin-bound invitation. Subsequent enrollment requires an authorized operator and fresh

step-up authentication. Private security state retains protected password material, MFA seeds and hashed session or recovery secrets. Those secrets stay outside audit exports and ordinary logs.

Step-up asks the person to authenticate again before a sensitive operation. Several clocks may constrain that operation: time since activity, time since the session began, and time since the additional check. The earliest applicable expiry controls. Reauthentication refreshes the step-up evidence while the original session continues toward its absolute limit. Technical reference 5A gives the candidate laboratory settings.

The supported institutional adapter uses an institution-approved OpenID Connect provider and an Authorization Code flow with PKCE S256. It binds the initiating transaction, issuer, callback and session, then validates the returned identity evidence before creating the local session. RFC 9700 supplies the broader security guidance for transaction-bound PKCE and defenses against redirect-flow attacks. An installation must still select and test the exact provider configuration. [67]

OpenID Connect identifies a user through the issuer and subject pair. Its specification warns against treating email or display name as a stable unique identity. Aetheria maps that pair to a local person and separately maintains approved membership and role mappings. A change in the person's email therefore need not create a new institutional person; two aliases must also be reconciled when a review requires distinct people. [68]

This division gives offboarding a definite job. A provider logout or an expiring token cannot substitute for prompt knowledge that an employee's membership has ended. The adapter needs authenticated deprovisioning information or a bounded status refresh compatible with the selected profile. When freshness cannot be established, sensitive operations close. An unfamiliar provider group claim cannot appoint someone to a delegation or approve a proposed effect.

5.4 Trust attaches a key to a permitted role

Chapter 3 established that a signature binds bytes to a key. The receiving owner also needs an installed record of what that key may attest. A source-ingress key can identify admitted data under its purpose. A grant-signing key serves a different function. Accepting every signed object as interchangeable would allow a low-privilege producer to make statements reserved for an authorizing owner.

The trust registry therefore supplies the key's role, scope and permitted use from an authenticated installation or control-plane process. An incoming key identifier is a lookup hint, not an invitation to trust a new key. Evidence bundles and model output cannot install their own trust roots. Changes to trust and revocation advance protected state that survives the ordinary application's restore boundary.

This rule is particularly useful when reading a source document that describes policy. The document may be evidence for an authorized policy-maintenance process. Its prose does not change the current evaluator merely because it contains an instruction to "allow all requests." The owner of policy installation and the runtime policy decision retain their separate duties.

5.5 Revocation during a running task

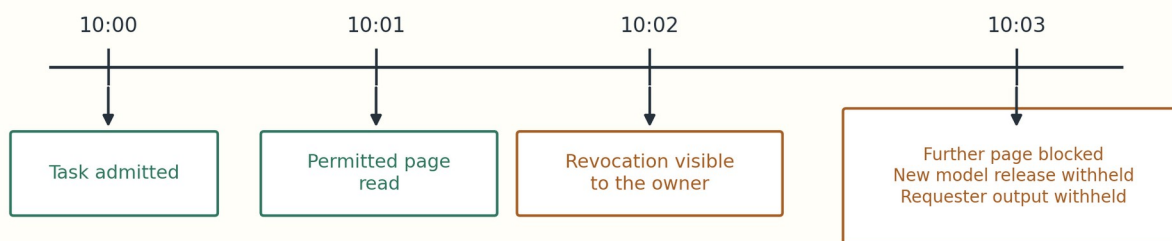
An epoch is a version of authoritative security state. When relevant membership, credential, source ACL or policy state changes, earlier evidence can become stale. Owners compare the generation or epoch tied to the request with current state at the protected use. The check makes a task's historical attribution compatible with changing rights.

Return to Mara at 10:02. If she loses source access after the first page, the old snapshot cursor cannot obtain another page. The expanded research contract invalidates continuation and unreleased results on an ACL or policy epoch change, including a change that appears more permissive. A newly admitted task and snapshot are required. Keeping the old snapshot in protected history preserves what was previously examined without authorizing its continued use.

Suppose all pages were already staged but no model payload had left. The provider boundary checks its actual recipient, purpose, exact bytes and current disclosure permission. It withholds an unauthorized release. A model that has already received bytes presents a different history: the local system can stop future releases and record what occurred, while the earlier disclosure remains part of the operation's evidence.

The final answer has its own requester-output boundary. Completed computation may remain protected even when Mara can no longer receive it. A separately authorized service may continue an applicable reconciliation task under its own current rights, while disclosure to Mara is checked against hers. Treating the original request as a perpetual credential would collapse these distinct responsibilities.

Historical work and current permission



Earlier evidence remains attributable; bytes already released cannot be recalled

Figure 5.2 Revocation across the task lifetime. This teaching timeline assumes the new revocation state is visible at the next owner check. Later uses refuse or withhold under that state. Remote propagation and already released bytes retain the limits described in Section 5.5.

A sensitive operation needs fresh evidence of current status at the moment of use. The selected foundation limits that evidence to a five-second use-time window. A remote revocation can occur during that short interval, after the status was issued. Local holds stop dispatch when installed before the relevant boundary; the deployment must also measure how quickly status and revocation changes propagate. Any stronger synchronization guarantee needs corresponding design and evidence.

5.6 Time is an interval the owner can defend

Distributed clocks have uncertainty. The selected admission rule represents current time as a lower and upper bound. An item valid from not-before until expiry can be used only when not-before is at or before the lower bound and the upper bound is strictly before expiry. The entire interval must fit.

Take a classroom credential valid from 10:00:00 until 10:05:00 UTC. At an estimated interval of 10:04:58.2 to 10:04:58.8, it is inside the validity window. At 10:04:59.6 to 10:05:00.2, the upper bound has reached beyond expiry, so admission refuses. A midpoint before 10:05 does not make every possible

current instant valid. These examples isolate the time condition; all identity and permission conditions still apply.

The same reasoning applies at the opening boundary. An interval of 09:59:59.8 to 10:00:00.2 straddles not-before and therefore fails. If the time source disappears, rolls backward or diverges from the persisted clock floor beyond the supported bound, the system holds rather than substituting a convenient browser time. The selected foundation caps clock uncertainty at one second; status freshness is the separate five-second condition introduced in Section 5.5.

A task may run longer than the authentication evidence used to begin it. Each page and sensitive use therefore obtains current context as required. After a restart, the owner combines persisted UTC bounds with fresh trusted time to determine what remains valid. The remaining time belongs to the original task.

5.7 Refusal without revealing the protected resource

An ordinary error can itself disclose information. “The seven mayoral records exist, but you cannot read them” reveals membership and a count. The selected facade therefore returns the same UNAVAILABLE response class and body shape for guessed or inaccessible resources, with detailed diagnostics confined to separately authorized views. A user can receive a conflict or stale-context response about their own known object without exposing another person's work.

Mara's departure leaves a record of the work she requested and the permissions under which it began. Later readers can attribute those events to her while checking any new use against the institution's current state. Protected grant consumption also remains spent across session replacement, preserving the operation's history.

Technical reference 5A Candidate laboratory session limits

Idle limit: 15 minutes (900 seconds). Absolute session limit: 8 hours (28,800 seconds). Maximum time since step-up authentication: 5 minutes (300 seconds). The earliest applicable limit governs the operation. These are candidate settings for the synthetic profile.

What to retain

Mara's completed computation can remain protected after her access changes. Permission is checked again at each later read or disclosure.

- Tenant — the institution boundary for a principal and data
- Epoch — a version of current access state
- Step-up — a recent additional authentication check

Exercises

Q5.1 A request supplies the correct principal but names another tenant's environment. Explain which evidence resolves the scope and the appropriate result of a mismatch.

Q5.2 Why does an issuer and subject pair make a better institutional identity input than an email address? What separate local decisions remain necessary?

Q5.3 Mara loses access after page one. Trace the consequences for the old cursor, staged model payload, completed answer and retained history.

Q5.4 Apply the conservative time rule to the three intervals in Section 5.6. Then evaluate an interval whose upper bound equals expiry exactly.

Q5.5 A source-ingress key signs text declaring a new delegation. Explain what the signature establishes and what additional owner process would be needed to establish authority.

Q5.6 A product claim says revocation takes effect instantly everywhere. Revise it to reflect the specified currentness window, local holds and deployment evidence.

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 5.1: [8], [47], [59]; 5.2: [8], [42], [53], [59], [65]; 5.3: [47]; 5.4: [8], [9], [14], [48], [53]; 5.5: [4], [17], [18], [42], [48], [53], [55]; 5.6: [42], [48], [52]; 5.7: [42], [47], [48].

6 Workflow state, scheduling, and concurrency

Two workers wake up with the same task on their screens. Both believe the source pages are ready. Each prepares to advance the task to computation. If both can write a new state without checking what the other has done, the system may schedule the next stage twice or attach a result to the wrong inputs.

Workflow coordinates that shared history. It records what has committed, schedules bounded work and admits the resulting evidence. The design must remain intelligible when a process stops between those steps. We will follow one research workflow, then introduce a competing writer and a lost commit acknowledgement.

Learning objectives

- Read the selected council workflow as guarded transitions between exact phases.
- Separate phase, control state, mutation posture and disclosure state.
- Explain how expected revisions and atomic commits prevent conflicting progress.
- Distinguish recovery of local work from permission to repeat an external release.

6.1 A task needs several identities

Suppose Mara's research task is recorded as W42. She reconnects after a lost reply and asks what happened to that task. W42 is its durable `workflow_id`: it joins the stages into one continuing history. A conversation can contain several such tasks, so the conversation's identity would be too broad for this lookup.

Now place the lost reply inside the coordinator. A worker asks to advance W42 from source collection to computation. Did that update commit before the connection failed? The owner needs an identity for the attempted update as well as for the whole task. A `command_id` supplies it. In our example, command C18 names that one semantic update; recovery asks for C18's recorded disposition.

If C18 commits, the next worker receives a particular computation to perform on particular inputs. Call it I6. Its `invocation_id` distinguishes that scheduled call from other calls in W42 and binds its exact input and budget. When a result returns, the owner can determine whether it belongs to I6. W42, C18 and I6 are illustrative labels for three different questions: which task, which update, and which scheduled call?

These identities stay with the work during recovery. Grants and execution outbox identifiers have their own issuing owners; Workflow retains those identifiers when they apply. Giving an uncertain operation a fresh label would lose the connection to the evidence needed to settle it.

The installed template defines the legal transitions and their guards. Its digest pins the exact template artifact. A task profile separately identifies the admitted cognition behavior. In the selected council workflow, these choices permit research and optional synthesis under explicit bounds. Check the profile identity before attributing those capabilities to a particular running service.

6.2 Reading the actual council transition table

The selected council template starts at ADMITTED and ends at CLOSED. Workflow first moves it into CONTEXT. Source pages accumulate within that phase, each under the same snapshot and current rights. Once the input is complete, or an explicitly bounded partial input is ready, the work moves to COMPUTATION.

Computation can lead directly to DISCLOSURE when the deterministic result is ready. If optional synthesis is selected and its connection, budget and disclosure conditions hold, it instead enters MODEL. After the model is settled or synthesis is unavailable, the saved deterministic result continues to DISCLOSURE. The final transition records the release decision and actual disclosure state before closing. Figure 6.1 and Table 6.1 use the template's exact phase and event names. The owner column spells out the components: Workflow schedules progress, Authorized Context handles source and release boundaries, and the Aggregator coordinates computation and optional synthesis. In the source paths these packages are C09, C10 and C11 respectively.

The selected council research workflow

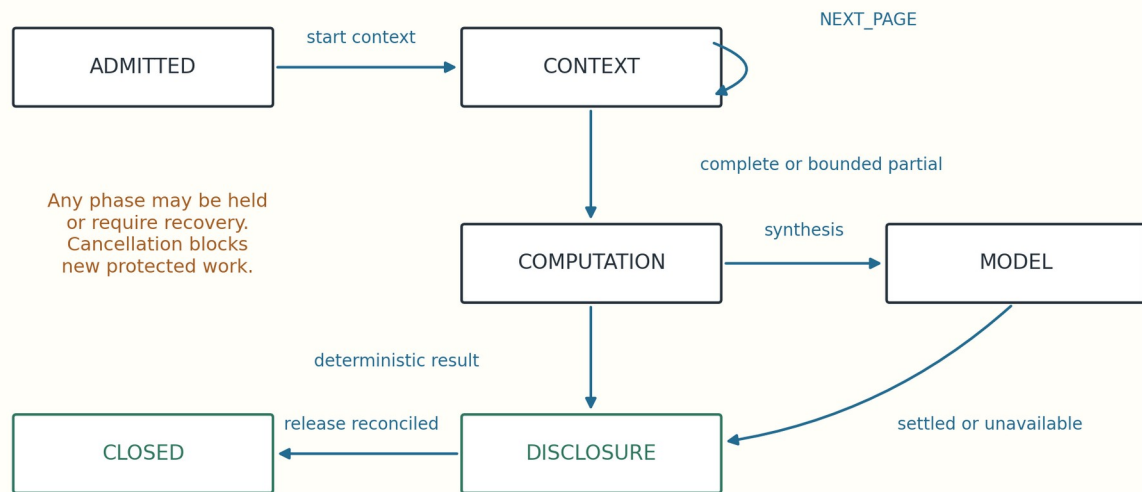


Figure 6.1 Selected workflow state transitions. Boxes and edges follow the exact council template. Arrow labels are shortened for reading; Table 6.1 gives exact events and owners. Optional synthesis has a separate release ledger. This template admits zero protected effect dispatches.

Table 6.1 Exact council template transitions

From	Event	To	Owner
ADMITTED	START_CONTEXT	CONTEXT	Workflow
CONTEXT	NEXT_PAGE	CONTEXT	Authorized Context
CONTEXT	INPUT_COMPLETE_OR_BOUNDED_PARTIAL	COMPUTATION	Aggregator
COMPUTATION	SYNTHESIS_SELECTED	MODEL	Aggregator
COMPUTATION	DETERMINISTIC_RESULT_READY	DISCLOSURE	Aggregator
MODEL	MODEL_SETTLED_OR_SYNTHESIS_UNAVAILABLE	DISCLOSURE	Aggregator
DISCLOSURE	RELEASE_RECONCILED_OR_WITHHELD	CLOSED	Authorized Context

The source and result descriptions use another vocabulary: READING and COMPUTING describe research activity, while ANSWER_READY, PARTIAL and NO_ANSWER describe the research result. These labels help explain behavior, but they are not substitutes for the workflow phase enum. A screen

or diagram that merges the two can accidentally create transitions the installed template never admitted.

The template labels every listed edge `external_effect=NONE` and permits zero effect dispatches. That means this research template does not schedule a protected target mutation. Optional model synthesis still crosses a consequential information-release boundary. Its protected commitment and uncertainty history remain necessary even though the research transition itself creates no mail or other target effect.

6.3 Four state questions remain separate

Phase locates progress in the template. Control state records whether the workflow is active, paused, held, requires recovery or is closed. Mutation posture holds an admitted reconciliation outcome when the task has an effect. Disclosure state records what happened to the outward release. A single field cannot faithfully represent all four.

Consider a research task that finishes its calculation after Mara loses access. The calculation may be complete while the result is withheld. Its mutation posture remains null because research created no protected mutation. Conversely, a separately governed mail operation can have an established mutation outcome while its evidence is withheld from a particular viewer. The owner's projection preserves these distinctions.

Table 6.2 Separate dimensions of workflow state

Dimension	Selected values or meaning	Interpretation
phase	CONTEXT, COMPUTATION, MODEL and other template phases	Where the workflow is in its admitted progression.
control_state	ACTIVE, PAUSED, HELD, RECOVERY_REQUIRED, CLOSED	Whether and how progress may continue.
mutation_posture	null or an admitted native reconciliation posture	What the evidence supports about a protected mutation.
disclosure_state	NOT_READY, PENDING, WITHHELD, RELEASE_STARTED, RELEASE_REPORTED, UNKNOWN	What is known about the outward information release.

The distinction also helps with unfamiliar input. An unknown phase or a changed duplicate is rejected or quarantined instead of coerced into a convenient default. Reaching CLOSED does not establish that an external action occurred, and the arrival of a late fact does not authorize reopening dispatch. Historical evidence follows its own admission and reconciliation path.

6.4 An expected revision protects the next commit

Return to the two workers at the chapter opening. A and B both read revision 7 in CONTEXT. Each proposes `INPUT_COMPLETE_OR_BOUNDED_PARTIAL`. A reaches the owner first and commits revision 8 in COMPUTATION. B's proposal was prepared against the earlier state. Accepting it unchanged could schedule the same next stage twice.

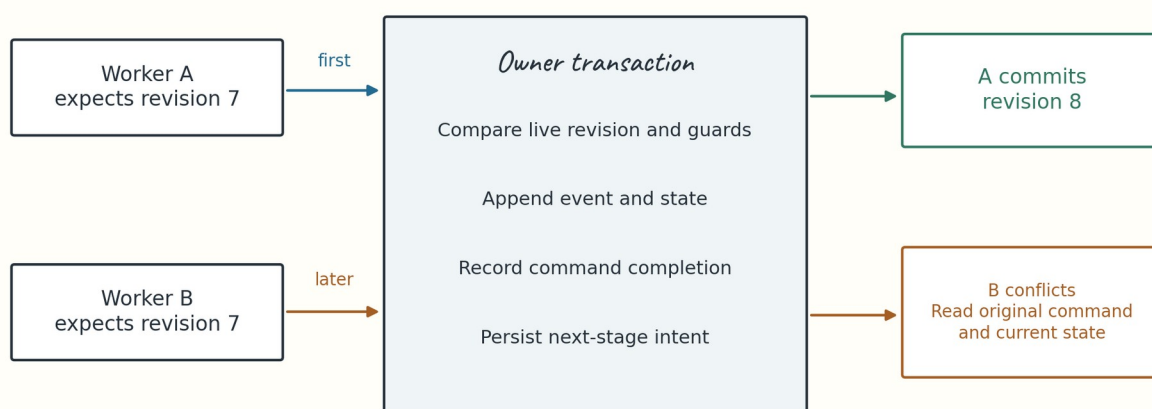
The command therefore carries the revision on which it was based, in a field named `expected_revision`. The owner commits only if that value still matches the live revision and the transition's other conditions

hold. This conditional update is called compare-and-swap, or CAS. B supplies 7, finds 8, and fails the comparison.

B then reads the owner state and its original command's disposition. Simply replacing 7 with 8 would conceal the stale basis of its proposal. The next step depends on what actually committed and which transition remains legal.

The successful transaction contains more than the revised row. It appends the event, retains the state and command completion, and records the next-stage intent together. If only the row changed, a crash could lose the instruction to schedule computation. If only the scheduling message appeared, a worker could run from a state that never committed. The atomic boundary joins the durable facts needed to advance.

One commit joins progress to its next stage



A lost acknowledgement is recovered by lookup of the same command

Figure 6.2 Atomic state event and next stage intent. The shaded boundary groups facts committed together. A wins the illustrated revision race; B fails its stale comparison. Preparation and external calls occur outside the transaction. The picture illustrates the contract rather than a running database test.

Aetheria prepares the private candidate and role-specific signatures outside database locks, then rechecks the live revision, generation, current holds, cancellation fence and applicable owner predicates inside the committing transaction. Network calls, model requests and human waits stay outside those locks. This arrangement makes the final commit conditional on current state while keeping slow external work from holding the database transaction open.

An in-memory mutex cannot preserve this rule through a restart or coordinate writers in separate processes. The selected storage design calls for unique constraints and serializable transactions or a verified equivalent CAS boundary. Berenson and colleagues' analysis of database isolation is useful background: familiar isolation labels describe different concurrency behavior. The implementation must demonstrate the invariant required here, rather than rely on the name of a database setting. [71]

6.5 Admitting a stage result

Only a committed, eligible invocation runs. The worker records its start under the exact lease and invokes the installed endpoint using the original invocation identity and input references. The returning

result must match that invocation, stage, workflow, request, domain and immutable template version. It also has to pass the producer and record checks appropriate to its role.

A duplicate with the same result identity and digest can resolve to the recorded disposition. A duplicate carrying changed bytes is an integrity conflict. A result calculated from stale pre-effect state cannot advance a task that has already changed. These rules stop an apparently useful answer from being attached to a different question simply because its text looks plausible.

Late post-effect evidence receives different treatment. If an external attempt occurred and its receipt arrives after a deadline, the system still needs that fact to establish what happened. Matching evidence can enter the immutable history and a linked reconciliation revision. It cannot reset the budget, rewind the phase or trigger another send. The distinction concerns the use of the evidence: historical attribution can remain valid after fresh authority has expired.

6.6 Budgets belong to the original run

Suppose W42 reads 80 source pages and uses three Aggregator pivots to organize its work. The page counter advances by 80; the pivot counter advances by three. Reading another page is one operation inside acquisition, rather than another round of orchestration. Separate counters let the system bound both the size of the evidence collection and the amount of coordination.

The task also has a deadline, a provider-send allowance and bounds on individual attempts. Technical reference 6A collects the selected numerical limits. Together they determine how much useful work remains possible. A large nominal page allowance cannot overcome a byte limit or an expiring task deadline. When a limit is reached, the result must describe the evidence actually obtained.

A restart resumes those durable counters. Changing a worker or command identifier leaves the original run's expenditure in place. Otherwise a recovery loop could turn bounded work into an unlimited series of fresh allowances.

Current permission constrains every continuation as well. The snapshot, cursor, task deadline and authentication each have to remain valid for another page read. A snapshot can still be within its nominal lifetime when its cursor expires; that page request must stop.

6.7 Locating the crash before deciding what to repeat

Start with a crash before a purely local state transaction commits. If the owner establishes that it aborted, the same logical work may be retried within the applicable bounded policy. Next place the crash after commit but before acknowledgement. The caller must query the original command, because a successful commit may already have advanced the state and created the next invocation.

Now move the crash past a possible provider release. A new invocation would risk a second disclosure and another charge. The protected model ledger retains `RELEASE_COMMITTED` and any `UNKNOWN` result while evidence settles the original operation. Repeating source acquisition and repeating a potentially sent model request have different consequences and therefore different recovery rules.

Workflow also cannot create another execution handoff when an action's status is unclear. Authority owns the sole execution outbox created through issuance. A notification or projection wakeup only prompts an authorized consumer to inspect committed owner state. It carries no permission to call `Nullius`.

Table 6.3 Recovery follows the last durable boundary

Failure location	What remains uncertain	Required recovery
Pure local transaction proved aborted	No commit remains to recover	Retry the same logical work within its bounded policy.
Commit acknowledgement lost	The transaction may have committed	Query the original command and read owner state.
Provider release may have occurred	Disclosure and spend may have happened	Preserve commitment and invocation; settle from evidence.
Protected send may have occurred	Target effect may exist	Retain original execution and fence; reconcile.

6.8 Cancellation and later evidence

A user cancellation persists `cancel_requested` and stops new work at the appropriate owner boundaries. In the research template it blocks new reads, model releases and disclosure. An already released payload remains a historical fact. For a protected mutation, a pre-effect cancellation verdict additionally requires the native no-attempt fence and supporting owner evidence.

Suppose cancellation commits before a queued worker starts. The worker must recheck that fence before performing the consequential step. Suppose instead the send may have occurred before cancellation. Recovery preserves the original operation and seeks evidence. A deadline or session closure cannot convert silence into proof that nothing happened.

When a late receipt settles the operation, reconciliation appends the new supported conclusion while retaining the earlier unknown state as history. Mara can see the earlier uncertainty and the later evidence that resolved it. The history explains what the institution knew at each point, while the original operation remains the one being reconciled.

Technical reference 6A Council workflow resource limits

The expanded council profile allows at most eight Aggregator pivots, one provider send, 1,000 source pages and 100 records per page, within a 600-second task deadline. Page reads occur inside acquisition and have their own counter.

A source-page attempt has at most 10 seconds within the remaining task time. Optional model work has at most 120 seconds. A snapshot has a nominal 900-second lifetime, capped by the task deadline and current authority; its cursor must also remain valid. Byte bounds can reduce the number of records that fit on a page. These ceilings describe permitted work, not measured throughput.

What to retain

Two workers race on the same task; an expected revision lets only a valid state transition commit. External work stays outside that local transaction.

- Workflow revision — the version used for a state comparison
- Command — a request for one owner to advance work
- Invocation — one prepared external call with its own identity

Exercises

Q6.1 Trace the council workflow with and without optional synthesis, using the exact phases in Table 6.1. Where do repeated page reads occur?

Q6.2 Two workers read revision 7. A commits revision 8; B's old command arrives later. Explain the required comparison and why changing B's expected revision alone is unsafe.

Q6.3 Explain the failure created if next-stage intent commits without the matching state, and the failure created if the state commits without that intent.

Q6.4 A task reads 80 pages and uses three orchestration pivots. Which counters have advanced? Does a worker restart change their remaining budgets?

Q6.5 A response disappears after a provider release commitment. Compare the appropriate recovery with a proved-aborted local database transaction.

Q6.6 An effect receipt arrives after cancellation and administrative closure. Explain how it can change the evidentiary conclusion without authorizing another dispatch.

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 6.1: [36], [42], [49], [51], [61]; 6.2: [6], [42], [56], [60], [63]; 6.3: [50], [63]; 6.4: [6], [51], [60], [64]; 6.5: [50], [51]; 6.6: [4], [42], [52], [60]; 6.7: [6], [42], [51]; 6.8: [4], [6], [42], [50], [52], [60].

7 Authorized Context and source admission

Mara asks for public council minutes and adds a second request: compare them with the mayor's phone logs. The first source is within her research assignment. The second is restricted. If a program loads both into a model and removes the phone-log sentences from the final answer, the protected records have already crossed the model boundary.

Authorized Context controls the earlier decision: which records and fields can enter the work at all? It also governs the later release of admitted material for a particular recipient and purpose. This chapter follows those two responsibilities and the ingestion process that makes source material available for governed retrieval.

Learning objectives

- Trace a source request through access decisions, connector reading and content admission.
- Distinguish permission to read a source from permission to disclose its contents.
- Explain quarantine, provenance and indexing without treating parsed material as automatically trusted.
- Identify direct and indirect disclosures of inaccessible records.

7.1 The reader holds the credential

For the council minutes, the Aggregator requests the records needed for Mara's comparison from Authorized Context. The request names an approved source and a bounded query. Authorized Context resolves the installed connector and owns its source-reading credential. The source service independently enforces its own authenticated access rules. Atlas, models, the Factory and the Aggregator have no alternate SQL or file path to the corpus. This arrangement gives the institution an identifiable point at which source permission and provenance are checked.

The installed connector determines how that source can be read: which resource scope it covers, what queries and record format it supports, and how it identifies revisions. Those settings keep the request on the approved route. A natural-language question does not authorize the caller to substitute an arbitrary URL or SQL query.

Before touching source data, Authorized Context compares the current authenticated context with the request, domain, principal, purpose, intended recipient and use. It rejects unsupported queries, unknown connectors and excessive work. Access metadata used for policy comes from an authoritative catalog, rather than a classification label a model found inside a document.

The READ evaluation happens on the information path. A research task does not need a hypothetical future effect before it can ask for permitted minutes. When a known denial applies, the source request stops. Missing current facts or an unsupported obligation yields an unresolved decision. A genuinely empty, permitted search has a different meaning: it can have permission with zero authorized resources.

7.2 A permitted read names exact material

Take one minute in Mara's requested period. Permission must identify that stable resource, its revision and the fields she may use. A general permission to "read council material" would leave all three uncertain. For a collection, the reader establishes an entitlement-filtered snapshot or equivalent

evidence before exposing names, totals or contents. A count of inaccessible rows is already information about the source. Filtering after enumeration is too late for that metadata.

The connector checks the permit and its own workload identity before fetching the exact revision. If source ACLs change between decision and fetch, the staged material is discarded and the operation is reevaluated. A connector that exposes only an unversioned current value cannot support an operation requiring a verified exact prestate.

Suppose the source stores a public minute alongside the clerk's private contact details. Mara's permitted projection contains the minute and the fields needed for her task. Authorized Context parses and retains only the admitted projection for the caller's path, with restricted provenance linking it to the source. Forbidden raw fields and even hashes of hidden content stay out of caller-readable wrappers. A digest is an identifier that may itself reveal a relationship; it is not a permission token.

In Mara's mixed request, the system can supply the permitted council evidence if the task contract allows that useful result, while the unsupported phone-log request remains constrained by its own decision. It must not describe how many logs were excluded or manufacture a comparison from remembered restricted content. A failure to obtain the second source changes what analysis is supportable.

7.3 Admission preserves how the bytes were obtained

After the read, Authorized Context validates the returned identity, revision, encoding, structure and allowed fields. It retains the exact projected content and a source wrapper that identifies its provenance. This is an evidence-producing step: the system records which source and revision supplied the material, what projection was applied, and which admitted bytes continued downstream.

W3C PROV-DM distinguishes entities, activities and responsible agents, including relationships of derivation. Those concepts help explain the difference between an original minute, an extracted passage and a computed count. Aetheria uses its own typed records for the corresponding evidence. The conceptual comparison does not imply that every record implements a PROV serialization or that provenance alone establishes factual correctness. [41]

An original minute can contain a mistake. Its admitted revision remains evidence of what that source said at the snapshot. A later correction is a new revision with a traceable relationship. Retaining that distinction allows a reader to explain a changed result without rewriting the earlier source history.

The same reasoning applies to text extraction. If an admitted parser derives text from a supported document type, its output needs the original source identity, parser profile and coverage information. A missing page or unreadable segment must remain visible to the task that depends on it. The institution can then decide whether the remaining evidence supports a partial answer.

7.4 Reading and releasing are separate decisions

Mara now has an admitted set of minutes. The task can calculate a count from them, but sending passages to a model adds a recipient and a purpose of use. Exporting an answer adds another destination. Each release needs a decision about the material that will actually leave its current boundary.

The system first assembles the exact content and metadata for the proposed consumer. A handoff manifest binds that selection to its sources. Governance evaluates the manifest, recipient, purpose and destination. At the sending boundary, the sender obtains the current use check and consumes a one-use

challenge in protected state before sending. This final check connects the earlier decision to the actual release.

The selected design distinguishes `CONTEXT_HANDOFF`, `REQUESTER_OUTPUT`, `PROVIDER_EGRESS` and `ARTIFACT_EXPORT`. They cover release to cognition, to the requester, to a model provider and into an exported artifact. Each can involve different recipients, destinations and data-use conditions. Permission to read public minutes inside a task does not automatically authorize sending all related context to an external provider.

Read permission and release permission

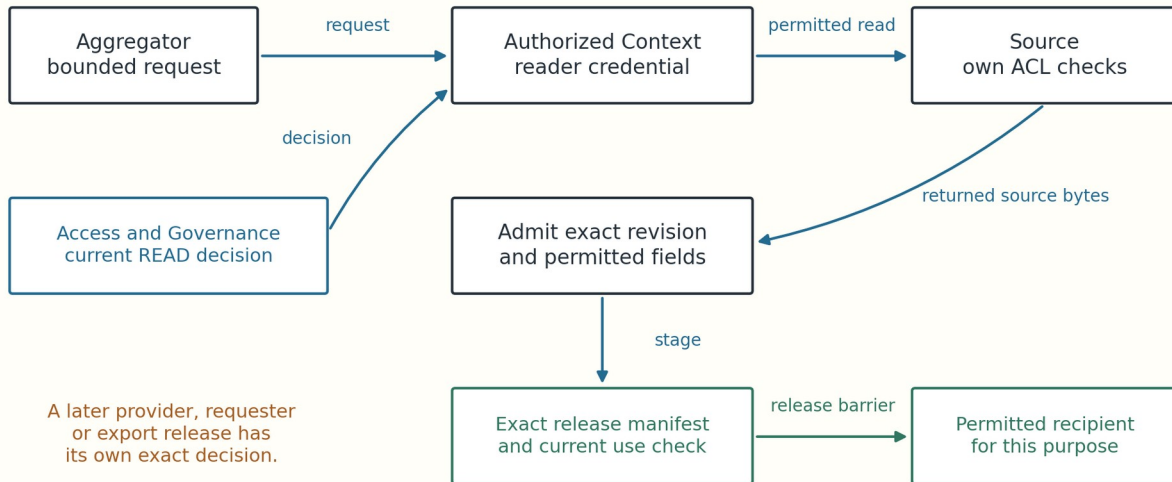


Figure 7.1 Governed source reading and release. Arrows distinguish the request, permitted fetch, returned bytes, admission and final release. Access and Governance supply the current READ decision. Disclosure is evaluated again for the exact handoff and actual recipient; the reader retains the source credential.

Suppose a permitted answer is assembled and then receives an additional quotation. The exact release input has changed. The sender evaluates the final bytes and metadata, including citations, attachments and relevant destination fields. This ensures that the object actually released is the object the disclosure decision considered.

Release has its own uncertainty window. The barrier records intent before network I/O, but a process may fail after bytes could have left and before a conclusive report is retained. The operation then preserves the original identity and unknown delivery state. A deliberate new disclosure attempt needs the applicable current authorization and a new challenge; the system does not silently replay the uncertain one.

The earlier retained profile disables provider egress. The expanded selection admits a bounded, authorized model call under its connection and release protocol. These are explicit profile choices. Neither allows a local model to escape institutional data-use rules merely because its process runs on premises.

7.5 Ingestion starts in quarantine

Retrieval uses an admitted corpus; ingestion creates that corpus. An authorized data custodian supplies a configured source list or entitled snapshot under a narrow ingestion purpose. The process identifies

source ownership, classification, access-metadata provenance, retention requirements, parser and expected revision before publication. Crawling everything with an administrator credential would bypass that admission problem rather than solve it.

The selected lifecycle is REGISTERED, STAGED, VALIDATED, ADMITTED and INDEXED. Staged and validated material remains private. A malformed batch, unknown classification, provenance conflict or revoked ingestion permission goes to QUARANTINED. The prior active corpus remains available under its existing access rules while the new batch is examined. Figure 7.2 makes the publication boundary explicit.

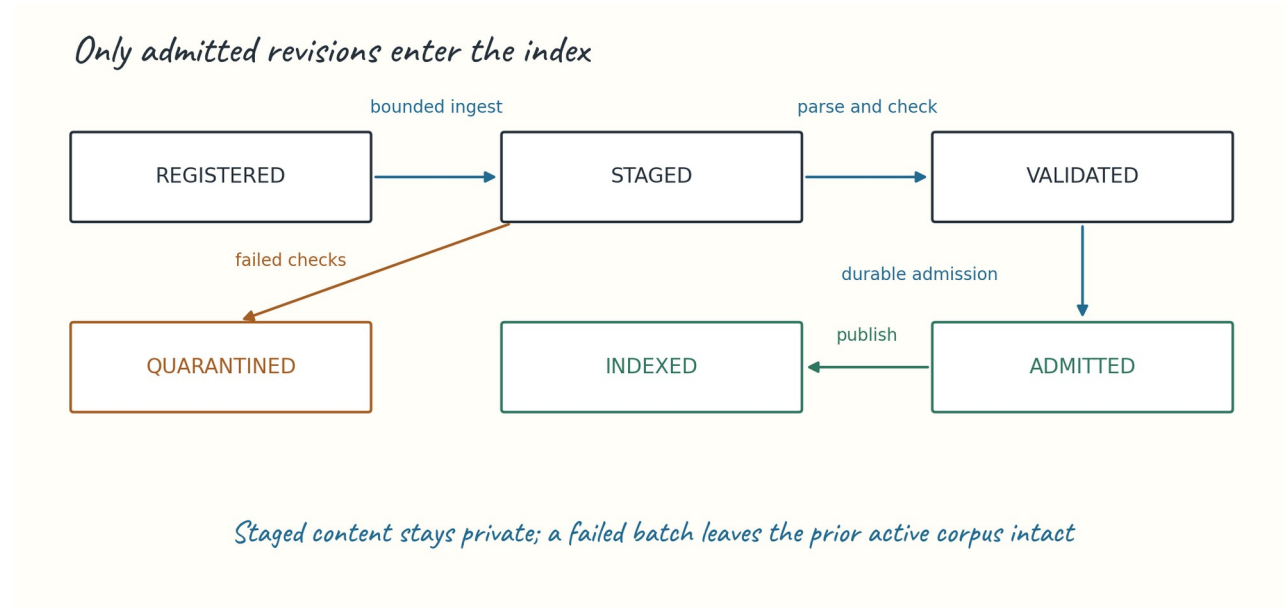


Figure 7.2 Ingestion from quarantine to the admitted index. The main path is the selected ingestion lifecycle. The quarantine arrow summarizes failed admission checks, including failures discovered during validation. A repair creates a new immutable job or revision. Searchability begins only after committed admission and index publication.

Consider an uploaded archive that expands far beyond its apparent size. The parser must stop within its resource bounds while the batch remains private. Or suppose a file points outside the registered source directory: the adapter must establish that the read stays within the approved root. Technical reference 7A collects the implementation constraints behind these two checks.

Admission commits the exact projection, wrapper, provenance and owner state with the ledger event. The index publishes only that committed revision. A duplicate identical ingestion intent can return the recorded result, while changed bytes under the same intent conflict. A repaired batch enters as a new immutable job or revision; failed bytes are not relabeled trusted in place.

Parser support is an explicit capability. The retained baseline admits strict UTF-8 JSON records; broader PDF, OCR, Office and archive handling needs the selected parser and adversarial acceptance evidence. A realistic case-file example should therefore state which inputs were supported and which remained unavailable. A mention of “dossier analysis” cannot supply those missing capabilities.

7.6 The index and cache remain subject to current rights

An index accelerates lookup. It does not own the institution's access decision. When ACL or policy epochs change, the system invalidates the affected continuation, staged disclosure and cache reuse even

if reindexing is unfinished. Otherwise a removed permission could remain effective through an older index entry.

Conversation reuse presents the same problem in another form. An earlier answer may contain information that its recipient could read yesterday. Repeating it today requires current permission for the underlying provenance and intended release. Quoting, summarizing or asking for a derived relationship does not remove the restricted source from the information dependency.

Consider two source worlds that contain the same permitted council records but different hidden phone logs. The ordinary content, counts, citations and provider payloads must be invariant under that hidden change. This comparison can expose leaks that a direct-download test misses. A response such as “I excluded more records this time” fails even when it contains none of the protected text.

The wider assessment also examines response size, errors, cache behavior and timing. Equal visible counts do not demonstrate equal timing under load. The system's contract therefore separates the required disclosure behavior from measurement of residual channels. Protected administrative diagnostics have their own purpose and access controls; they cannot be passed through the clerk's ordinary result for convenience.

7.7 Applying the boundary to observation

After a protected action, the Observer needs evidence from the target. It requests that evidence through Authorized Context under the configured observation identity. The source reader retains the credential, and the returned revision and provenance remain subject to admission. The Observer then produces its own observation record for reconciliation.

Suppose Mara leaves the assignment while an earlier operation is being reconciled. The observation service may continue under its separately authorized rights, preserving the evidence the institution needs. Mara's access to the resulting record is evaluated independently. The institution can finish establishing what happened while respecting the change in who may see it.

Technical reference 7A Parser confinement

The parser runs without network access, secrets, executable macros, plug-ins or source-side writes. Bounds cover decompression, recursion, processor time, memory, file count and output. A platform adapter must confine access through file handles relative to the registered root and check for concurrent replacement. A path string alone does not establish that confinement.

What to retain

Restricted bytes are excluded before model use. Admitted material keeps its source identity through indexing, quotation and later disclosure checks.

- Source admission — a permitted intake of source material
- Provenance — the chain from output to source
- Disclosure — release to a particular recipient or destination

Exercises

Q7.1 Draw the permitted route from Aggregator to source and back. Place the source credential, the READ decision and the context-release decision at their owners.

Q7.2 A source read was permitted, but an ACL changes before the connector returns the bytes. What happens to the staged result and why?

Q7.3 A batch passes parsing but has unknown classification. Where may it be stored, can it enter the index, and how should a repair be represented?

Q7.4 Identify three ways a system could reveal restricted phone-log information without returning a phone-log document.

Q7.5 An engineer says that an on-premises model needs no disclosure check. Evaluate that claim using the recipient, purpose and exact-content requirements.

Q7.6 Explain how an Observer can continue permitted reconciliation after the requester is revoked while still withholding the result from that requester.

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 7.1: [4], [53], [54]; 7.2: [4], [53], [55]; 7.3: [4], [14], [16], [53], [54]; 7.4: [4], [18], [42], [53], [55], [61]; 7.5: [6], [54]; 7.6: [4], [6], [55]; 7.7: [17], [53], [54], [55].

8 Snapshots, coverage, and exact council research

Mara runs the council query twice and gets different totals. Between the two runs, the clerk's office corrected one set of minutes. Both calculations could be right for their respective sources. To understand the difference, Mara needs more than the final number: she needs the population, authoritative revisions, matching rule and evidence that each scan covered its stated population.

This chapter develops that chain with a small fictional corpus. The exercise is deliberately small enough to calculate by hand. It introduces duplicates, a corrected revision, punctuation and multibyte text, then shows exactly which facts support a complete result and which remain available after an interrupted scan.

Learning objectives

- Define the authorized meeting population and its authoritative minute revisions.
- Explain the different jobs of a snapshot, page identity and opaque cursor.
- Apply LEXICAL_COUNCIL_V1 and distinguish matching meetings from text occurrences.
- Check citation offsets against original UTF-8 bytes and report incomplete coverage honestly.

8.1 Define the population before counting

The question “How often did the council discuss AI?” can mean several things. It might ask how many meetings contained a phrase, how many phrases occurred, or how long substantive discussion lasted. The selected lexical method answers the first two. An analysis of discussion duration or meaning needs different evidence and a separately declared method.

Eligible meetings are final, noncancelled meetings in the requesting principal's entitled catalog whose start times fall in the query interval. Each contributes one authoritative final-minute revision designated by the source as of the snapshot. The denominator therefore counts meetings. Agenda copies, transcripts and superseded revisions do not increase it.

Identical duplicate documents collapse by stable identity, revision and hash. If two conflicting revisions both claim to be authoritative, the source requires repair or an explicit incomplete result. Choosing whichever copy produces the more convenient total would conceal a source conflict inside a calculation.

The catalog itself matters. A complete scan of an incomplete catalog cannot establish that every eligible meeting in the intended period has been examined. A source may have a known gap in a year for which Mara is entitled to see the catalog. Coverage must carry that gap. Hidden meetings remain outside her denominator and outside ordinary missing-interval disclosures.

8.2 The snapshot fixes a reproducible research basis

A snapshot freezes the authorized eligible catalog, query, selected revisions and source high-watermark at one consistent source transaction boundary. The high-watermark identifies the source position used to establish that view. A snapshot manifest records the basis so page one and page ten can be interpreted together.

Suppose meeting M108 has an earlier minute r1 mentioning AI and a corrected minute r2 that removes that reference. An old snapshot can remain tied to r1 if those immutable bytes remain available and their use remains entitled. A new snapshot selects r2 when it is the source's authoritative revision. The

task must not read half its meetings before the correction and silently switch the remainder to the new source basis.

Content revision and permission changes have different consequences. Revision drift may allow the old immutable snapshot to finish. An ACL or policy epoch change invalidates its cursor and unreleased result. A new task and snapshot establish the new basis even when the permission change appears to widen access. This preserves a clear account of what each result means.

Database snapshot isolation offers background for thinking about a consistent view, but the Aetheria contract adds source membership, revision selection, current permission and disclosure obligations. A database isolation setting alone cannot demonstrate all of those properties. [71]

8.3 Pages preserve the snapshot and the caller

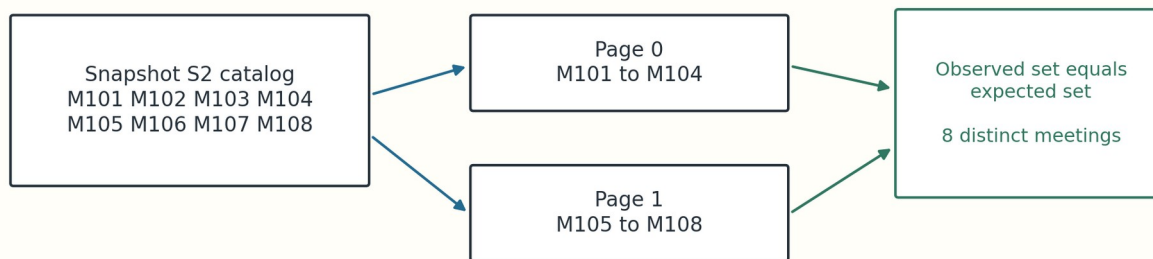
The source returns pages in stable meeting-time, meeting-ID and revision-ID order. Each page contains at most 100 records and must also fit the native byte bounds. The system can reduce a page's record count to satisfy those bounds without dropping records from the scan.

The cursor is a random handle to server-held continuation state. The expanded protocol selects 256 random bits represented by 43 unpadded base64url characters and stores a SHA-256 lookup key. That handle is bound to the caller, scope, query, snapshot, epochs and expiry. It does not encode a visible offset into a hidden dataset, and possessing it does not replace current authentication.

A repeated still-valid page request can reproduce the same page under the same snapshot. The calculation recognizes its identity and does not count its records again. Changing the query or scope while retaining the cursor refuses. A nonfinal empty page is invalid because it provides no progress while claiming that continuation remains.

The verifier checks the page digest, page index, query and snapshot identity, source admission, record and revision membership, and end-of-stream. The page digest covers the canonical page object with only `page_sha256` omitted. Cursor and scope fields therefore remain inside that identity. A client cannot alter one of them while preserving the same valid page digest.

A final page still needs complete membership



Also verify scope, query, revisions, digests, page order and end-of-stream

Figure 8.1 Snapshot page membership and citation closure. The two pages partition the authorized S2 catalog in this classroom fixture. Completeness requires matching membership and the accompanying integrity checks. `is_last` alone cannot establish that every entitled meeting was scanned.

End-of-stream is necessary but insufficient for completeness. A source could mark its second page final while omitting M106. The consumer must reconcile the expected authorized catalog membership with the observed set. If the sets differ, a reassuring completion flag cannot make the answer complete. This check also detects duplicate or contradictory membership that simple row counting misses.

8.4 Apply the named lexical method exactly

LEXICAL_COUNCIL_V1 fixes Unicode data version 15.0.0. It decodes strict UTF-8, applies NFKC normalization, performs Unicode case folding, and applies NFKC again. It then splits tokens at every character that is not a Unicode letter, mark or number. A token is a maximal uninterrupted run of those categories. [69, 70]

NFKC applies compatibility normalization as well as canonical composition. It can turn a width variant into its ordinary counterpart. Case folding supports case-insensitive comparison. These transformations make a matching representation; the original document bytes remain unchanged for citations. The selected sequence and Unicode version are part of the method's identity, so silently changing them would create a different method. [70]

Two patterns count: a token equal to `ai`, and an adjacent token pair equal to `artificial` followed by `intelligence`. Punctuation and line breaks can separate the two words. Thus `artificial-intelligence` and `artificial` followed by a newline and `intelligence` both match. `A.I.` produces separate `a` and `i` tokens and contributes no match. Neither `paid` nor `chair` contains a token equal to `ai`.

Each matching occurrence contributes to `text_matches`. A meeting containing one or several occurrences contributes exactly one to `meetings_with_matches`. The two selected patterns do not overlap. A semantic synonym such as `machine learning` contributes nothing under this method unless a selected lexical pattern also appears. That limitation is part of the meaning of the reported count.

The Unicode standard explains the normalization operations; Aetheria's source specification chooses the particular token categories and matching patterns. Keeping those sources distinct prevents an

implementation from substituting a library's default word-boundary rule for the specified algorithm. [69, 70]

8.5 Work through eight fictional meetings

Table 8.1 gives a complete entitled population of eight final, noncancelled meetings inside the frozen benchmark interval. Each row identifies the authoritative minute for snapshot S2. The corpus also includes an agenda for M101, an exact duplicate of its final-minute record, an older M108 revision and a cancelled meeting. Those extra records do not add meetings to this population. All data in this exercise are fictional.

Table 8.1 Authoritative final minutes in snapshot S2

Meeting	Rev	Original minute text or exact character description	Matches
M101	r1	AI procurement was discussed.	1
M102	r1	Artificial intelligence and AI tools were reviewed.	2
M103	r1	The chair paid the invoice.	0
M104	r1	A.I. policy was noted.	0
M105	r1	Artificial-[line break]intelligence training was approved.	1
M106	r1	Café + space + fullwidth A and I + space + tools.	1
M107	r1	Artificial intelligence. artificial intelligence.	2
M108	r2	The committee deferred the item.	0

M102 has two occurrences, one phrase and one ai token. M107 has two phrase occurrences. M101, M105 and M106 each contribute one. The total is seven text matches across five meetings. With eight eligible meetings, the proportion containing a match is 5 divided by 8, or 62.5 percent. A reader can verify every term of the calculation from the table.

M106 uses fullwidth Latin A and I, the code points U+FF21 and U+FF29. Its normalized token is ai. The table describes those characters by name so their identity remains clear across fonts. The accompanying UTF-8 fixture retains the actual characters and byte offsets. M104's dotted A.I. remains a non-match because punctuation separates its tokens.

Now compare snapshot S1, taken before the correction to M108. In that snapshot, r1 contains one AI occurrence. The eligible population remains eight meetings, while matching meetings increase to six and occurrences to eight. The proportion is 75 percent. The difference has an identifiable cause: the authoritative revision for M108 changed. Combining r1 and r2 into one run would destroy that explanation.

Table 8.2 Comparing complete snapshots

Snapshot	M108 revision	Eligible / matching	Occurrences	Matching share
S1	r1 mentions AI	8 / 6	8	75%
S2	r2 removes it	8 / 5	7	62.5%

The classroom files include a manually specified oracle for membership, counts and spans, followed by a separate computation checked against it. This validates the arithmetic of the exercise. It supplies no performance result for the full municipal workload, whose profile calls for at least 384 qualifying council meetings plus adverse cases and other domains.

8.6 Cite original bytes after normalization

A citation identifies a span in the original immutable UTF-8 document. Its start is inclusive, its end exclusive, and both must lie on code-point boundaries. The quote digest hashes the exact bytes in that slice. It does not hash the normalized token or a prettified quotation.

Use M106's text, represented as Café followed by a space, fullwidth A, fullwidth I, a space and tools. The prefix Café plus its space occupies six UTF-8 bytes: C, a and f use one each, é uses two, and the space uses one. Each fullwidth letter occupies three bytes. The matching original span is therefore [6, 12), six bytes long, although its normalized form ai uses two ASCII bytes.

The match is normalized; the citation stays original

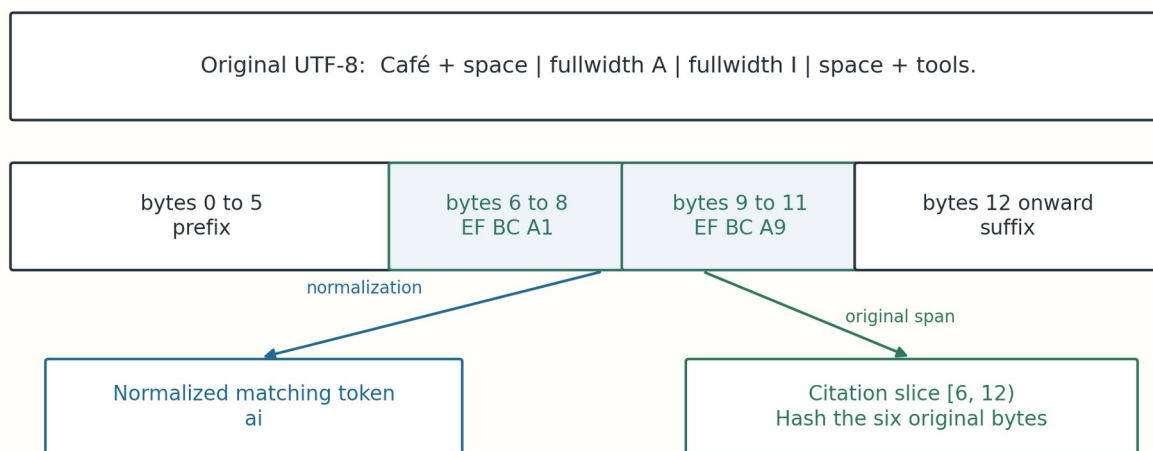


Figure 8.2 Normalized matching and original byte spans. The fullwidth letters occupy six bytes but normalize to the two-letter token ai. Both arrow branches begin at the same source occurrence: one supports matching and the other preserves the original citation bytes. Offsets are zero-based and end-exclusive. [70]

The original span's hex bytes are EF BC A1 EF BC A9. Hashing those six bytes verifies the quoted occurrence. Hashing 61 69, the ASCII bytes for normalized ai, verifies a different object. A character-indexed substring beginning at position five would also be an incorrect byte offset. The distinction becomes visible as soon as the text contains multibyte characters.

Normalization can expand or combine characters. The implementation therefore needs a mapping from normalized matches back to their containing original spans. It must preserve that mapping through the whole selected normalization sequence. An algorithm that normalizes each character in isolation is not a general substitute: combining sequences can interact across adjacent code points. The supplied classroom checker explicitly verifies its mapping assumptions for this small fixture and claims no general Unicode conformance. [70]

The result page can cite at most 100 spans and reference further authorized citation pages. That presentation limit does not reduce the total count. Each citation expansion remains a current disclosure, so an earlier result page cannot become a permanent access token for the underlying document.

8.7 Partial coverage changes the claim

Stop the S2 scan after its first four meetings. The observed subset contains two matching meetings and three occurrences. The authorized snapshot catalog still establishes eight eligible meetings, but only four have been scanned. A truthful partial result can report all those facts with coverage marked incomplete. It cannot call three the total number of occurrences across the eight meetings.

This is a coverage fraction of four out of eight, or 50 percent. It is not the final proportion of meetings containing a match. The two matching meetings among four scanned also happen to produce 50 percent, but those equal numbers describe different things. The full scan later gives 62.5 percent for matching-meeting prevalence. Labeling the measure prevents an accidental statistical claim.

If the eligible population itself is unavailable, `eligible_meetings` remains null. Missing counts stay null as well. `PARTIAL` may preserve observed counts under its contract; `NO_ANSWER` carries no fabricated scan or match totals. `DENIED`, `CANCELLED` and `HELD` are stricter: their result payload releases no membership, counts, snapshot or citations. The same failure word cannot be used interchangeably across these states.

Zero requires special care. A complete entitled scan with no matches can report zero matches. An interrupted scan cannot establish a complete zero. If the complete entitled population itself is empty, eligible and scanned counts may both be zero, while the proportion of matching meetings is undefined because the denominator is zero. Reporting “0 percent” would add a mathematical claim the data do not support.

Table 8.3 Coverage and result claims

Situation	Supported claim	Restriction
Complete S2 scan	8 eligible; 8 scanned; 5 matching; 7 occurrences	Current disclosure still applies.
First four S2 meetings only	8 eligible known; 4 scanned; 2 matching and 3 occurrences observed	<code>PARTIAL</code> ; observed matches are not complete totals.
Eligible catalog unavailable	Unavailable denominator is null	Do not infer hidden or missing membership.
Complete empty population	Zero eligible, scanned and matches	Matching proportion is undefined.
<code>DENIED</code> , <code>CANCELLED</code> or <code>HELD</code>	No membership or counts disclosed	Null counts and references; empty citations.

8.8 Calendar windows and optional synthesis

The frozen benchmark interval begins at 2011-10-08T00:00:00Z and ends at 2026-10-08T00:00:00Z. It includes its start and excludes its end. A meeting starting exactly at the ending instant is outside this benchmark. Original local time can be displayed separately, but interval comparison uses the specified UTC start time.

A live query derives its start from the actual trusted as-of time by subtracting fifteen calendar years. It does not subtract a fixed 5,475 days. For example, an as-of instant of 2024-02-29T12:00:00Z maps to 2009-02-28T12:00:00Z because that anniversary year has no February 29. For the frozen October interval, the elapsed span is 5,479 days because it crosses four leap days. Both calculations are checked in the classroom materials.

Optional model synthesis explains an already verified result. Its separate state can be AVAILABLE, UNAVAILABLE, WITHHELD or NOT_REQUESTED. If synthesis fails, a valid deterministic result can remain ANSWER_READY with synthesis unavailable, subject to current disclosure. The model cannot repair missing coverage by writing a convincing paragraph or alter seven occurrences into another authoritative count.

Mara can now explain why her two totals differ. She can point to the corrected minute, reproduce each calculation from its own snapshot, and open the original bytes behind every counted quotation. A colleague has enough information to check the answer and understand exactly what changed.

What to retain

The eight-meeting answer is tied to one entitled snapshot. A partial scan reports observed counts and incomplete coverage; its total remains unresolved.

- Snapshot — a fixed revision population for the calculation
- Coverage — the portion of eligible work actually scanned
- Byte span — the interval of original source bytes supporting a match

Exercises

Q8.1 Recalculate the S2 eligible meetings, matching meetings, occurrences and proportion. Explain the different treatment of M102, M104 and M106.

Q8.2 Replace M108 r2 with r1 under a separately admitted S1 snapshot. Recalculate the result and explain why mixing the two revisions in one run is invalid.

Q8.3 Stop after M101 through M104. State the observed counts, coverage fraction and quantities that cannot be presented as complete totals.

Q8.4 Reconstruct M106's [6, 12) byte span. Explain why hashing normalized ai or using a character offset verifies the wrong object.

Q8.5 A final page is validly hashed and marked is_last, but one entitled meeting is absent from the observed membership. Can the answer be ANSWER_READY? Explain the missing check.

Q8.6 Calculate the live fifteen-year start for 2024-02-29T12:00:00Z, identify whether a meeting at the frozen end instant is included, and explain how an empty complete population differs from an interrupted scan.

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 8.1: [4], [58]; 8.2: [4], [42], [58]; 8.3: [4], [42], [57], [58]; 8.4: [4]; 8.5: [4], [59]; 8.6: [4], [56]; 8.7: [4], [56]; 8.8: [4], [42], [56].

9 The Aggregator and bounded cognition

Mara has the council records in front of her. The calculation is finished: five of eight meetings contain a match, with seven occurrences in all. She asks for a short explanation to accompany the result. The returned draft says that the council discussed the subject in seven meetings. The sentence sounds plausible because seven is a real number in the evidence. It has acquired the wrong meaning.

This failure gives the Aggregator a concrete job. It coordinates the work that turns admitted evidence into a proposed answer, keeping the calculation, interpretation and supporting sources connected. It must preserve the distinction between a meeting and an occurrence through every stage, including an optional model call. This chapter follows that coordination and shows how a useful continuation can remain bounded by the original task.

Learning objectives

- Trace a cognition task from admitted intent through evidence acquisition and result validation.
- Count scheduled cognition pivots separately from source pages and provider sends.
- Explain how a compiled prompt preserves provenance, scope and coverage.
- Evaluate a proposed answer without treating its fluency or structure as evidence of authority.

9.1 Giving reasoning a defined task

Mara's request reaches the Aggregator after request admission has established who is asking, the institutional domain, the purpose and the selected kind of work. Workflow supplies the durable parent: the task that owns the deadline, cancellation state and permitted continuation. The Aggregator receives a typed cognition task referring to that request and its admitted input. It can then determine the next reasoning stage within an installed task profile.

A task profile describes the work an implementation has been accepted to perform. Council research, mail drafting and a bounded model task have different output contracts. Council research produces an answer with coverage and evidence. Mail drafting may produce a candidate that Binder will check before constructing an exact proposal. The generic model task produces bounded output without an institutional effect. Selecting the profile therefore establishes the shape of the work before any model interprets the request.

The term Aggregator names a logical responsibility. Its trusted coordination can run within Core, while model adapters and other untrusted cognition execute behind narrower interfaces. The separation matters when a generated sentence asks to fetch another database or change a record. A model worker has no institutional source-reader credential, grant-signing key or target-writer credential with which to carry out that request. It can return a candidate for the responsible component to evaluate.

Table 9.1 makes the ownership visible. The source archive uses package identifiers such as C09 for Workflow and C11 for the Aggregator; those identifiers locate files in the bibliography. Component names carry the explanation here.

Table 9.1 Responsibilities in the research path

Responsibility	Owner	Evidence passed onward
Admit the request	Request admission	Scoped request and selected intent
Schedule and recover work	Workflow	Durable stage and remaining task budget
Read permitted sources	Authorized Context	Source revisions, coverage and provenance
Compute and evaluate an answer	Aggregator	Result, claim support and advisory posture
Decide and perform release	Governance and Authorized Context	Current decision and actual release state

There are two useful readings of this table. Read downward to follow Mara's task. Read across to ask what entitles the next owner to proceed. A completed calculation gives the disclosure owner something precise to evaluate. The right to release it comes from that owner's current decision and checks.

9.2 Counting the work that can continue

Suppose the eight records arrive in two pages of four. The Aggregator does not need to invent a new research question after each page. Authorized Context advances the iterator under the same snapshot and query, checking current rights and the remaining page and time limits. The Aggregator resumes from the admitted page closure when the input is complete or when the permitted bounded partial result is ready.

A cognition pivot is an admitted step in the Aggregator's reasoning plan. Pagination has its own accounting. The selected council template allows at most eight cognition pivots in total and one provider send; repeated page reads consume the source-page allowance instead. The distinction lets a large but mechanical scan remain predictable. It also prevents a small-looking reasoning loop from concealing repeated external model calls.

For this teaching trace, use four admitted pivots: plan the research, acquire context through its owner, compute the result, and optionally synthesize and validate explanatory prose. These are descriptive stage names, rather than additional protocol state names. The optional synthesis and its local validation share the fourth pivot in this decomposition. The four-pivot split is illustrative, not a canonical execution sequence. An implementation records and counts the stages in its accepted task profile; a different accepted decomposition can use a different number of pivots within the same ceiling.

Table 9.2 One bounded teaching trace

Work completed	Cognition pivots used	Source pages read	Provider sends
Plan fixed	1	0	0
Context acquisition complete	2	2	0
Exact calculation saved	3	2	0
Optional synthesis checked	4	2	1
Requester release evaluated	4	2	1

After the fourth pivot, the unused allowance is $8 - 4 = 4$ pivots. The single provider send has already been used. Further work must fit every remaining constraint, including the deadline, source scope and permitted recipient; the pivot balance alone is insufficient to admit another stage.

When another cognition stage is needed, Workflow schedules it with explicit parent-stage and input references. The continuation belongs to the same run and consumes its remaining budget. A recovered worker obtains current owner state before proceeding. Cancellation, supersession or expiration can close the route even when a local function has more work to do.

The next stage returns through Workflow

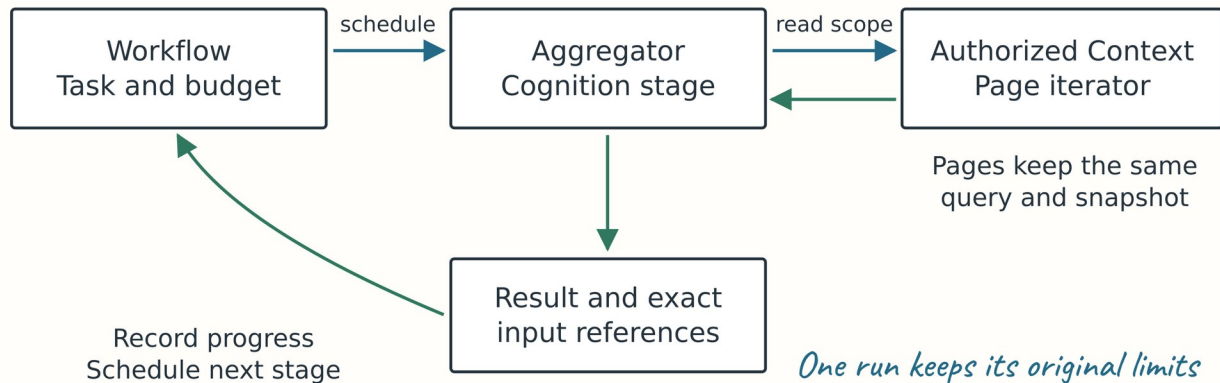


Figure 9.1 A bounded continuation. Solid arrows show scheduled work and returned artifacts. Source-page iteration stays inside Authorized Context; a subsequent Aggregator stage returns through Workflow with explicit input references. The drawing shows logical responsibilities, not required process placement.

9.3 Compiling the evidence into a prompt

The optional explanation requires a smaller input than the entire institutional archive. A prompt compiler assembles the admitted question, task instructions, selected evidence and required output shape. It records the versions and references that explain how the input was made. Ordering, extraction and truncation can change what the model sees, so each belongs to the compilation record.

The compiler keeps installed instructions separate from user content and retrieved material. A council record might contain the words “ignore the previous instructions.” Within the task, those words are source text. A valid signature on the record establishes its producer and exact bytes; it leaves the words in their original role as evidence. The model may react to them, which is why the surrounding interfaces also enforce what it can read, release or change.

This approach has a useful connection to retrieval-augmented generation. Lewis and colleagues combine a language model with retrieved material to support knowledge-intensive generation. Retrieval makes selected external evidence available during generation. In Aetheria, the additional architectural question is which evidence this task may use and which recipient may receive it. The retrieval method supplies material for an answer; the institutional owners establish the permitted scope. [90]

Mara's bundle includes the exact counts and the definition of a match. If only four meetings had been scanned, it would also preserve that coverage. A compiler that removes the partial-coverage statement to save space changes the meaning of the evidence. Required coverage qualifications belong in the

compiled input. If they, the mandatory instructions or the output contract exceed the available space, the task defers or abstains under its profile.

The final provider-specific rendering happens before disclosure evaluation. The renderer produces the application bytes the adapter intends to send, including the allowed non-secret request metadata. A later change to the prompt or destination requires a new release input. Chapter 11 follows that boundary in detail; here the consequence is simple: the Aggregator must finish assembling its proposed input before asking whether that input may leave.

9.4 Preserving a calculation through synthesis

Return to the fictional S2 corpus from Chapter 8. To make this chapter independently checkable, Table 9.3 repeats the occurrence totals after the chapter's declared matching procedure. Each meeting contributes either zero or one to the matching-meeting count, however many occurrences its text contains.

Table 9.3 The S2 counts carried into the explanation

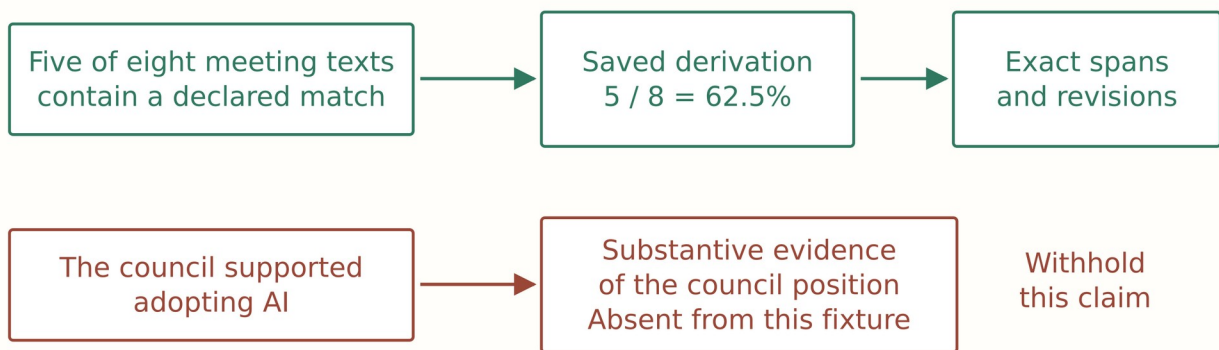
Meeting	Occurrences	Matching meeting
M101	1	1
M102	2	1
M103	0	0
M104	0	0
M105	1	1
M106	1	1
M107	2	1
M108 revision 2	0	0
Total	7	5

The first page, M101 through M104, contains three occurrences across two matching meetings. The second contributes four occurrences across three matching meetings. Together they give seven occurrences and five matching meetings in a population of eight: 62.5 percent. These quantities are saved before optional synthesis. The selected workflow requires the model path to preserve them.

The proposed sentence “seven meetings discussed AI” fails twice. Its meeting count conflicts with the stored derivation. Its verb also interprets a lexical match as evidence of discussion. A literal appearance in an agenda, quotation or title can satisfy the declared match rule without establishing how the meeting treated the subject. A defensible explanation is narrower: “The declared terms appear seven times across five of the eight meeting texts, or 62.5 percent of meetings.”

The claim graph makes the check inspectable. The percentage points to the five-meeting numerator and eight-meeting denominator. The numerator points to the five matching records. Each occurrence points to its exact original source span and revision. An interpretive sentence needs support suitable for that interpretation, beyond the mere existence of a citation.

A citation must support the claim being made



Numeric support follows the saved derivation

Figure 9.2 Two claims and their support. The numeric claim follows the saved derivation. The proposed claim about council support would need substantive source evidence that this lexical fixture does not supply. Arrows mean “requires support from.” The unsupported branch is withheld from the answer.

Checking structure comes first. Structured output must parse as the specified document under the installed decoder. Duplicate keys, unknown fields or a fragment extracted from explanatory prose cannot silently become an accepted candidate. Semantic checking follows: the accepted structure may contain a false count, an unsupported interpretation or a citation to the wrong revision. A model's own second opinion remains advisory; independent verification requires checks grounded in the underlying evidence.

9.5 Returning a useful result

The Aggregator records an advisory posture with the output. Recommend indicates that the applicable evidence and task checks support the candidate. Defer leaves a material dependency unresolved. Refuse rejects a candidate under the applicable epistemic rules. Abstain records the absence of a supported answer. The responsible admission logic derives these outcomes from attributed inputs and the applicable checks.

The output's job determines what happens next. A council result can proceed to requester disclosure with no effect candidate. A mail draft can become eligible for Binder's independent checks, which construct the exact proposal and preserve the origin of its material fields. Governance, Authority and the execution path retain their own decisions. The cognition-result schema makes this explicit with an `authority_conferred` value fixed to false.

If the optional model times out, the council template can continue with the checked deterministic result when requester disclosure remains permitted. The unfinished model invocation retains its separate uncertainty and budget obligation. Mara can receive a valid count while the system continues to account for a possibly completed provider call. The answer should state that explanatory synthesis was unavailable where that distinction affects her understanding.

The same restraint applies to specialist panels retained in the broader design. Several models repeating one source share an evidentiary dependency. A future accepted panel must define its task, dependence rules, evidence checks and total budget. Running that panel would require an accepted extension to the selected one-send route.

Mara's corrected answer now has a clear shape: the declared search, the population actually scanned, the exact counts, and links to the supporting records. Any additional interpretation carries its own support. She can inspect where the answer came from without accepting the model's confidence as a substitute for the calculation.

9.6 Working a specialist disagreement

Mara now asks a different question: “Did the council authorize permanent deployment?” Counting appearances of AI will not answer it. A specialist task must distinguish an agenda, an adopted motion and a staff recommendation. The broader Aggregator design retains specialist profiles, dependence accounting, typed conflicts, bounded challenges and calibration for work of this kind. The following panel is an illustrative extension profile, P9. It would require separate admission before use; the selected one-send council route in Section 9.2 remains unchanged.

P9 admits three evidence items for a fixed question. S901 is an agenda saying, “Discussion: AI-assisted records search. No vote scheduled.” S902 is approved minutes saying, “Motion to fund a six-week records-search pilot passed 4–1. Production deployment requires a separate vote.” S903 is a staff memorandum saying, “Staff recommends automatic deployment following the pilot.” These are fictional teaching texts. Their document classes and revisions are material inputs.

The installed profile, rather than a model, chooses the lanes and their contracts. A policy-reading lane identifies the kind of decision each document records. A second interpretation lane checks the proposed answer. A required evidence-verification lane checks source attribution, the recorded vote and the distinction between a recommendation and an adopted decision. Computational role names describe tasks; they do not appoint a person to a legal office.

Table 9.4 A proposed panel has dependencies and required roles

Lane	Initial output	Dependence and admission fact
Policy reading A	Permanent deployment was approved	Uses a truncated excerpt of S902
Interpretation B	Agrees with A	Uses the same excerpt; agreement shares an evidence dependency
Evidence verifier V	Pilot approved; production needs a separate vote	Uses complete S902 and checks its document class

A and B may use different models and still repeat the same error. P9 records model/deployment dependencies separately from source dependencies. The complete S902 is one primary record, regardless of how many models quote it. The profile can accept that one record as adequate evidence for what its motion says; it cannot relabel repeated readings as independent corroborating sources.

For this example, the recommendation rule requires all three valid lane outputs, at least two admitted model-dependence groups, completion of V, and no unresolved material conflict. Its conflict rule treats a contradiction about the authorized action as blocking. The numerical agreement of A and B therefore leaves the proposed permanent-deployment claim blocked when V supplies the omitted sentence. A different institution may select another accepted rule, but the rule is fixed before this task begins.

First consider a missing output. If V times out, the panel has two of three required outputs. Its denominator remains three. The missing verifier is recorded as unavailable; it is not an abstention that

lowers a threshold. The task defers or returns an explicitly permitted partial result according to P9. It cannot publish the permanent-deployment claim under the complete-panel recommendation rule.

Now let V return on time. A bounded challenge names the exact disputed claim, cites the omitted sentence and requests reconsideration using the complete admitted passage. It stays within the original purpose. New source reads pass through Authorized Context, and the new provider disclosure receives its own decision. The challenged lane cannot choose a new recipient, add a tool or expand the task.

Table 9.5 The panel spends one shared allowance

Stage	Provider sends consumed	Remaining sends	Material result
Three initial lanes	3	1	Conflict is preserved
One admitted challenge to A	4	0	A corrects its interpretation
A request for another model vote	4	0	Further send is unavailable

P9's four-send ceiling is a teaching assumption for the proposed extension. Its lanes also share the task's deadline, cost and compute reservations. A required challenge that cannot fit those bounds leads to a bounded incomplete outcome. It does not acquire an additional allowance by changing the stage name.

Suppose A now agrees with V, while B retains its original claim. The record preserves B's dissent, the disputed statement and its source dependency. P9 may recommend the narrower factual sentence only if its installed conflict-resolution rule explicitly classifies the permanent-deployment claim as unsupported and permits that narrowing. Otherwise it defers. The supported sentence is: "The minutes record approval of a six-week pilot by a 4–1 vote and require a separate vote for production deployment." No vote on permanent deployment is supplied by these three texts.

Calibration belongs to this exact kind of work. A result from a benchmark on another model revision, prompt, domain or evidence class cannot supply P9's calibration reference. If the relevant calibration validity expires or a required deployment changes, the affected profile must obtain the required new assessment before making the associated recommendation claim. Chapter 25.7 works a small evaluation example; its eight cases illustrate arithmetic rather than establish statistical calibration.

A panel result remains advisory. If Mara later asks to send it, Binder constructs an exact new proposal. A challenge that changes a material field of an already-bound action requires a new Binder revision and the applicable review. The specialist machinery improves the quality and inspectability of the proposal while preserving each downstream owner's responsibility.

What to retain

The Aggregator can explain a saved count and ask bounded specialists for advice. Neither a fluent explanation nor panel agreement changes the review and authority for a later send.

- Deterministic result — the saved calculation to explain
- Specialist profile — the declared lane and dependency rules
- Challenge — one bounded request to examine a material conflict

Exercises

- Q9.1 In Table 9.2, replace the two source pages with twenty pages under the same admitted context stage. Which counters change? What evidence would establish that the stage stayed within its original task?
- Q9.2 A proposed answer says, “Five meetings supported adopting AI.” Identify the part supported by Table 9.3 and the additional evidence the whole sentence requires.
- Q9.3 The model returns a well-formed result with a correct percentage and a citation to an inaccessible revision. Describe the checks needed before the result can be released.
- Q9.4 A continuation has four pivots remaining and zero provider sends remaining. It proposes a second model “for verification.” Explain how Workflow should evaluate that proposal.
- Q9.5 A retrieved document includes a signed instruction to change the task's recipient. Explain the role of the signature and the owner that decides the actual recipient.
- Q9.6 The optional synthesis becomes uncertain after sending. Explain how the deterministic research result and the provider invocation can have different completion states.

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 9.1: [36], [42], [60], [72], [73], [88], [89]; 9.2: [42], [60], [73], [77], [88]; 9.3: [53], [74], [75]; 9.4: [60], [76]; 9.5: [60], [73], [76], [78], [79], [89]; 9.6: [78].

10 Model connections, adapters, and credential custody

The institution rotates the credential used for Mara's model connection. One task is waiting to send. Another has already left the system, but its reply has not arrived. Both tasks refer to the same connection name. If that name now means “use the new key,” the system risks losing which account and credential generation carried the earlier disclosure.

A model connection gives that changing relationship a durable structure. It records the owner, approved route and capability profile while protected custody keeps the secret itself out of the task. This chapter follows a connection from provisioning through activation and rotation, then returns to the two tasks to decide what each may do.

Learning objectives

- Distinguish institutional identity, provider authentication and task authorization.
- Trace connection metadata separately from the credential it references.
- Explain why a capability claim applies to an exact model and task profile.
- Handle rotation differently for unsent work and a possibly sent invocation.

10.1 Separating the identities involved

Mara signs into Aetheria through the institution. Access establishes her principal and the domain in which she is working. A provider credential establishes a separate relationship with the service that will perform inference. The institution then decides whether this principal may use that connection for the present purpose and disclose the selected content to its destination. Each relationship answers a different operational question.

The distinction is easy to see with billing. A provider account may have sufficient credit and a valid credential while institutional policy withholds permission to send a personnel record. Conversely, Mara may be entitled to read that record while the selected provider account lacks an eligible route. The task needs both the institutional permissions and a usable connection.

Connection ownership is scoped to the tenant, organization, environment, principal and permitted purpose. An institution may assign a controlled service connection under its rules. The design does not supply a shared founder credential that silently pays for and authenticates every user's inference. The owner and custody arrangement must be explicit enough to survive staff changes and account revocation.

The preferred starting route in the selected plan is a user API credential or an assigned institutional service connection. Provider-supported delegated authentication can also be considered when the actual integration permits inference and satisfies the required transport and disclosure controls. Validation must establish those capabilities for the intended inference route. Consumer passwords and browser-session scraping are outside this credential path.

10.2 Keeping metadata useful and secrets protected

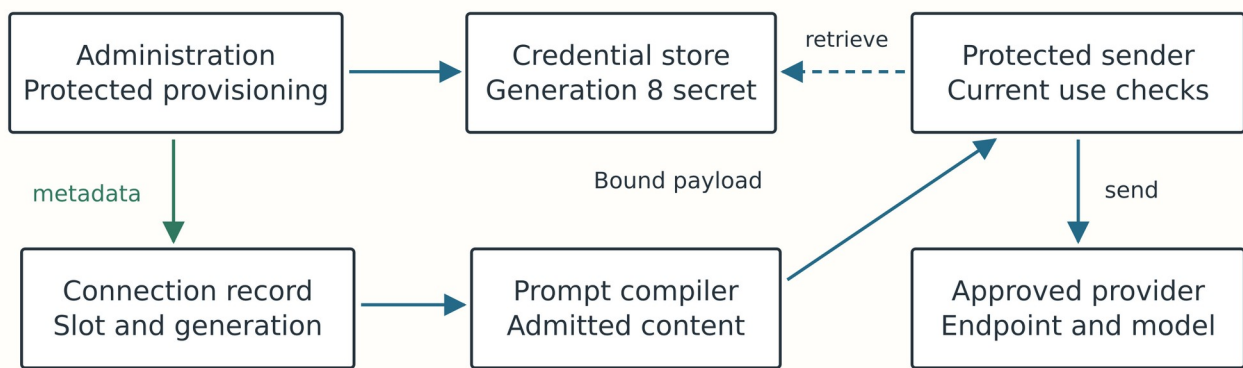
Administration owns the connection-management path. The user enters secret material through a protected provisioning interface with body logging disabled. The credential store retains that material

and returns a protected reference. Ordinary connection records carry the reference, generation and approved configuration, allowing the rest of the system to reason about a connection without copying its secret.

A generation identifies one immutable version of the credential binding. It answers a practical question that the connection name cannot: which credential version did this task prepare to use? A connection might be named “Mara council research” through several rotations. Generation 7 and generation 8 preserve the difference between its old and new bindings.

The schema records provider family, authentication mode, endpoint profile, model profile, capabilities and lifecycle state. It also references the applicable disclosure and budget policies. These fields describe the route. The responsible owner checks the evidence and performs the transition that makes the connection ACTIVE.

Connection metadata points to protected custody



The prompt carries content and references

Figure 10.1 Metadata and credential custody. Solid arrows carry admitted references or configuration. The dashed arrow shows the protected sender retrieving the exact credential generation at use. The credential never passes through the prompt compiler or the ordinary connection record. This is a logical custody diagram.

The distinction continues at send time. The content manifest binds the application payload and approved non-secret metadata. Authentication secrets stay outside caller-visible artifacts and hashes. Instead, the release binds the credential slot, generation and account identity, plus the permitted authentication-header names. The sender fills those designated fields under current checks immediately before using the approved transport.

This arrangement also determines what operators can inspect. A diagnostic view can report that a connection is suspended, which profile it selected and which generation an invocation used. Secrets stay in protected custody, and restricted prompt or response evidence has its own accountable access and disclosure path.

10.3 Establishing what a connection can do

A connection begins PENDING while its claims are checked. VALIDATED records successful validation of the relevant endpoint, model, account and capability evidence. ACTIVE additionally requires the applicable disclosure and budget bindings. Suspension, revocation or expiration prevent ordinary new use under the selected lifecycle rules. A revoked generation remains revoked.

Validation has a precise object. “Supports a model provider” is too broad to describe a working capability. The acceptance cell binds the provider, authentication route, exact model or deployment, task, modality and adapter profile. A text-generation route says little about whether the same deployment can produce the required embedding shape or classification labels. Each advertised capability needs its own meaningful conformance evidence.

Table 10.1 anchors the credential discussion in provider documentation checked on 9 October 2026. It deliberately records only the credential fact used in this chapter. The table is not an Aetheria acceptance report, and it makes no claim that every provider feature fits the selected transport profile.

Table 10.1 Documented credential routes

Provider family	Credential fact used here	Source
OpenAI	API keys authenticate API requests and require protected server-side custody	[91]
Anthropic Claude	The API documents API-key authentication	[92]
Google Gemini	API keys authenticate requests and are associated with a Cloud project	[93]
xAI Grok	The API quickstart uses a key to authenticate requests	[94]

The adapter turns the institution's approved request into that provider's exact protocol and decodes the response under an installed profile. Similar-looking APIs can differ in request defaults, retention options, finish reasons, token accounting or model identification. Replacing a base URL in an SDK is therefore a configuration change that needs validation, even when the client code continues to run.

The selected initial general-text route is one bounded, non-streaming request. Provider-hosted tools, remote files, provider-side conversation persistence and silent fallback are outside that route. A provider may offer them; an accepted extension must define their additional disclosures and controls before Aetheria uses them. Keeping the initial route narrow makes the evidence needed for acceptance manageable.

Local and specialty connections receive the same kind of scrutiny. A local route can use a no-secret authentication mode, but it still has a model profile, permitted input classes and positive compute accounting. The runtime remains isolated from institutional readers, signing keys, control interfaces and target writers. A specialty classifier needs an accepted task and output contract; the presence of a loaded model is only one part of that evidence.

Readiness checks begin offline. They can establish that configuration, schemas and test artifacts are present without sending institutional data anywhere. A live validation probe is a separate, authorized invocation with its own bounded input, disclosure and budget. Until the required evidence exists, the capability remains unverified.

10.4 Binding the destination

Imagine a council attachment containing a URL labelled “preferred inference endpoint.” If an adapter treats that source text as its next destination, the attachment has chosen where institutional content will go. The connection instead selects an installed endpoint profile whose scheme, host, port and path are constrained. Caller text and model output cannot replace those settings.

Transport checks connect the approved name to the actual network destination. Public-provider routes reject destinations such as loopback, link-local and private addresses; permitted local inference uses its own admitted network profile. Automatic cross-origin redirects and uncontrolled proxy overrides

would undermine the destination binding, so the selected route does not rely on them. TLS verification remains part of the accepted transport.

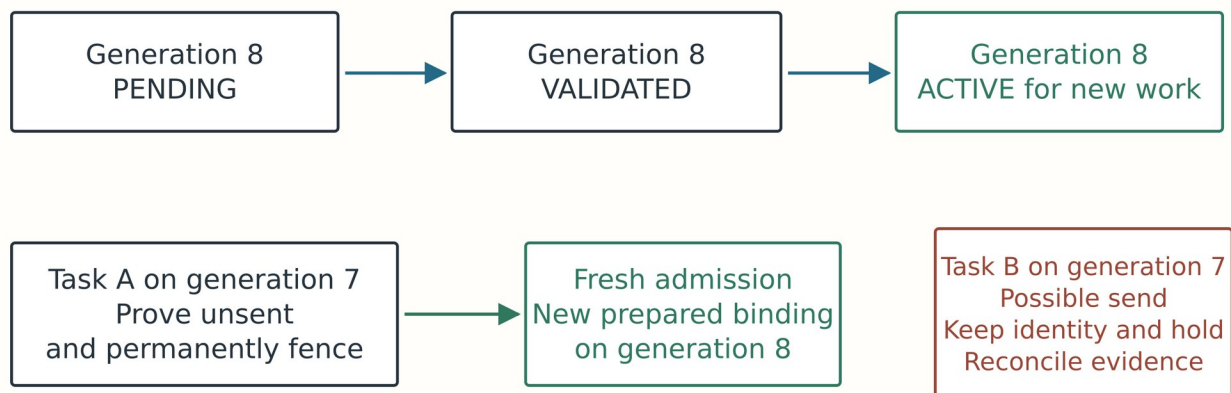
The endpoint also carries data-use consequences. The release decision needs the applicable retention, training-use and residency constraints, together with evidence that the selected profile can enforce what it promises. A contractual commitment establishes the agreed obligation. Claims about technical enforcement or a particular remote deletion require evidence of those events and controls. A route with missing controls can remain ineligible even when authentication and inference both work.

Model identity belongs in this binding too. When a task requires a pinned model revision, an alias whose actual revision cannot be established fails that requirement. This protects the meaning of evaluation results: tests of one model, prompt and adapter combination do not automatically qualify a changed combination.

10.5 Rotating a credential with work in flight

Return to generation 7. Task A has prepared an invocation but has not crossed the release barrier. Task B has a recorded release commitment and an uncertain reply. The administrator provisions generation 8 into a separate protected slot, validates its account and route, and activates the future-use binding through the controlled transition. In this example, generation 7 is then revoked for new sends.

Rotation preserves the earlier invocation



Each release keeps its original credential binding

Figure 10.2 Rotation creates a new generation. The upper path establishes generation 8 for future work. The lower branches show the different treatment of an unsent generation-7 task and a generation-7 invocation with a possible external outcome. Arrows show allowed preparation or reconciliation, never a rewrite of the old release.

Task A needs current admission under generation 8 before it can proceed. Its old prepared invocation is immutable. If a replacement is necessary, the sender first establishes that the old attempt is permanently fenced and unsent, then prepares the newly bound invocation under the applicable policy. Updating an old row to say generation 8 would erase the relationship the earlier admission actually evaluated.

Task B keeps generation 7 attached to its original invocation. Evidence from the earlier attempt must establish whether the provider received its bytes or incurred a charge. The uncertain invocation retains its release identity and reservation while reconciliation looks for that evidence. Any later authorized work remains separately accountable, alongside the earlier liability.

This is why a task identity and an invocation identity both matter. The task expresses Mara's continuing goal. An invocation identifies one exact prepared input, destination and credential generation, with its own release and budget history. One user goal can outlive a credential rotation, while each possible provider send keeps the identity needed to account for it.

The sender and rotation path must also agree about the instant at which new use is permitted. The current-generation check belongs inside the same protected release barrier that serializes against revocation and local holds. Otherwise a worker could check generation 7, pause while rotation completes, and send using its stale conclusion.

If generation 8 fails validation, it stays unavailable and generation 7 stays revoked. Recovery follows the recorded lifecycle and current policy, preserving the operator's revocation decision.

10.6 Making connection status understandable

Mara needs a status that explains what she can do next. "Connection active" describes eligibility for new admitted work through its current binding. "Previous invocation outcome unknown" describes the unresolved generation-7 call. Both statements can be true at once. Combining them into a single green indicator would conceal the question she is trying to answer.

The same care applies when a connection is disabled after suspected exposure. Disabling the route prevents new sends under its controls. Previously released content and unresolved usage remain part of the evidence and reconciliation problem. Operational recovery starts from those recorded facts.

With that account of the two tasks, rotation becomes understandable. New work uses the newly validated binding after current admission. Earlier work retains the binding under which it was prepared or released. The connection can change without changing the history of what the institution may already have disclosed.

10.7 Reading the complete model portfolio

The integrated roadmap's model requirement extends beyond demonstrating a single successful provider call. The engineering completion matrix requires real permitted inference in six model families. This requirement belongs to the portfolio; a particular task can select only the families and routes its installed profile admits.

Table 10.2 Model families in the engineering completion requirement

Family	Required demonstration subject	Evidence that stays distinct
OpenAI	An admitted provider deployment and actual permitted inference	Account binding, task profile, release and result
Claude	An admitted provider deployment and actual permitted inference	Its own credentials, endpoint and result
Gemini	An admitted provider deployment and actual permitted inference	Its own credentials, endpoint and result
Grok	An admitted provider deployment and actual permitted inference	Its own credentials, endpoint and result

Family	Required demonstration subject	Evidence that stays distinct
Local model	Actual local language-model inference	Installed model identity and local execution boundary
Specialty model	Actual task-specific inference such as classification or reranking	Its own task contract and measured behavior

A local language model and a specialty classifier satisfy different rows even when they share a host. Provider names do not specify an interchangeable model version, modality or authentication route. The accepted matrix binds provider, account, deployment, task, modality and profile. A consumer chat subscription also supplies no assumed API entitlement.

Each demonstration needs the associated denial and recovery cases, including rotation, revocation, destination mismatch and uncertain outcomes. A stub result establishes a harness path only. Evidence of an actual accepted model call establishes that call's scope. Appendix G places this obligation in DONE06 and keeps it separate from full portfolio, independent-assurance and customer acceptance.

What to retain

A connection can be active for new work while an invocation under an older credential generation remains unresolved. Those states require separate records.

- Connection — an institution's approved provider route
- Generation — one immutable credential binding
- Rotation — replacement for future eligible work

Exercises

Q10.1 Mara can read a document and her provider account has credit. What additional decisions are needed before that document can enter a model request?

Q10.2 A developer stores a key beside the connection's public metadata to simplify debugging. Trace which readers would gain access and describe the protected-reference alternative.

Q10.3 An adapter passes its text-generation test. What evidence is missing before it can advertise an embedding capability?

Q10.4 Generation 8 fails validation after generation 7 was revoked. What should the active-route selection do, and which fact prevents automatic restoration of generation 7?

Q10.5 A worker checks generation 7, pauses, and resumes after revocation. Locate the check that must occur before release and explain why the earlier check cannot carry the send.

Q10.6 Compare the recovery of Task A and Task B. Which needs proof of no send, and which needs evidence about a possible external outcome?

Sources for this chapter

Internal design sources by section. External literature remains cited in the discussion. 10.1: [18], [80]; 10.2: [18], [42], [74], [75], [79], [81]; 10.3: [18], [75], [80], [81], [83]; 10.4: [18], [55], [75], [80]; 10.5: [6], [18], [75], [77]; 10.6: [6], [79]; 10.7: [80], [266].