From KYC/KYB to KYA

Building an Identifiable, Verifiable, Traceable, and Assessable Agent Trust Framework



Jointly developed by

ANT GROUP | DIGITAL TECHNOLOGIES
ZOLOZ
GLEIF
IIFAA
NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE
SMU SINGAPORE MANAGEMENT UNIVERSITY

Contents

Introduction

About This White Paper

01 Industry Background

- 1.1 Agents Move toward Delegated Action and Business Collaboration
- 1.2 Continuity and Extension of Identity Governance
- 1.3 Four Core Questions for Continuous Trust Management
- 1.4 Structure of the White Paper

02 Identity Characterization

- 2.1 Three Perspectives on Identity Characterization
- 2.2 Six Dimensions of Identity Attributes
- 2.3 Modeling Four Types of Relationships
- 2.4 Trustworthy Representation
- 2.5 Chapter Summary

03 Continuous Assessment

- 3.1 Trustworthiness Assessment
- 3.2 Cross-domain interoperation
- 3.3 Continuous Monitoring
- 3.4 Operational Feedback
- 3.5 Chapter Summary

04 Verifiable Evidence and Recordkeeping

- 4.1 Value of Verifiable Evidence and Recordkeeping

— 4.2 Scope and Linkage
— 4.3 Registration and Custody
— 4.4 Sharing and Verification
— 4.5 Audit and Capability Boundaries
— 4.6 Chapter Summary

05 Capability and Security Certification

— 5.1 Business Value of Certification
— 5.2 Objects under Evaluation and Verification Process
— 5.3 Business Capability Certification
— 5.4 Security Certification
— 5.5 Retesting and Maintenance of Certification Decisions
— 5.6 Chapter Summary

06 Practical Cases

— 6.1 Case One: Agentic Commerce and Payments
— 6.2 Case Two: Internal Enterprise Agent Governance

07 Conclusions and Outlook

— 7.1 Report Conclusions
— 7.2 Outlook

References

Introduction

When an AI agent books flights or initiates payments on behalf of a user, or invokes tools and modifies production configurations on behalf of an employee, the parties involved need to understand more than whether the system can complete the task. They also need to know which runtime instance performs it, whom it represents, what it is authorized to do, and whether the outcome can be verified. The more systems and parties a task involves, the more clearly this information needs to be expressed and linked.

Existing identity systems provide an important foundation. Know Your Customer (KYC) and Know Your Business (KYB) support due diligence on individuals and organizations, including identity verification and ongoing risk management; identity and access management, credentials, and permission policies identify callers and constrain resource access; and audit records support subsequent review. Introducing agents does not require these capabilities to be rebuilt in their entirety. However, agents may represent different parties in different tasks, change tools and execution steps, and operate under changing environments and delegation status. A service account, a single authentication event, or a set of standing permissions alone cannot fully establish whether a particular action still meets its execution conditions.

This report addresses that issue through continuous trust management for agent identities. Using Know Your Agent (KYA) as a research framework, it builds on identity verification for individuals and organizations to link an agent's product provenance, runtime instance, representation relationships, and task delegation. It combines these with current status, action records, and verification results to inform business decisions. Trust extends beyond a single authentication event: judgments must be updated as tasks, authorizations, and operating conditions change.

The report develops four areas. Identity characterization establishes who the agent is, which parties it is associated with, and the delegation under which it acts. Continuous assessment encompasses trustworthiness assessment, cross-domain interoperation, continuous monitoring, and operational feedback, allowing judgments to be updated with operational information. The verifiable evidence and recordkeeping framework links key actions, their contemporaneous basis, and actual outcomes to support

verification and dispute resolution. Capability and security certification provides supporting evidence on the verification of claimed capabilities and applicable security requirements, with explicit conditions of applicability. Formal certification additionally requires a decision under an applicable certification scheme. These forms of evidence complement one another; they do not replace valid authorization or each party's own business rules.

To illustrate their application to specific tasks, the report examines agentic commerce and payments, and internal enterprise development and operations. It considers how delegation establishment, operational judgment, controlled execution, and outcome verification connect. Positioned as foundational industry research and a conceptual framework, the report advocates necessary extensions to existing authentication, access control, and zero-trust security capabilities to address agents' autonomy, dynamism, and cross-domain collaboration, providing a reference for industry practice and standardization efforts.

This work involves not only technical capabilities, but also information representation, allocation of responsibilities, and rules for collaboration among participants. How identities are identified, delegations conveyed, status updated, evidence verified, and certification results accepted all require a shared understanding and practical arrangements. Progress depends on participation across the industry to advance technology development, business applications, and standardization together.

The progression from KYC and KYB to KYA extends identity governance beyond verifying the authenticity of parties to examining the basis for agents' delegated actions and whether that basis remains valid throughout operation and collaboration. This helps connect agents' ability to complete tasks with the authorization, control, and accountability required for collaboration. We look forward to working with industry participants to move from a single authentication event toward continuous trust management and establish a foundation of trust for broader agent participation in business collaboration.

About This White Paper

This report was jointly developed and published by Ant Group, Ant Digital Technologies, ZOLOZ, the Global Legal Entity Identifier Foundation (GLEIF), IIFAA, Nanyang Technological University, Singapore Management University, and other organizations. Trustworthy management of agent identities remains a rapidly evolving field, and the exploration presented here may not cover every aspect. We welcome comments and corrections from experts and industry colleagues.

Working group:

Li Shiwei, Tao Ye, Yang Jingwei, Jing Kexin, Sun Xi, Xin Shuang, Li Jianshu, Li Xu, Wang Yang, Chen Zhichao, Peng Jin, Shao Xiaodong, Sun Rungeng, Wu Bin, Miao Changtao, Yang Wenlong, Gu Lidong, Liu Qin, Zheng Xiaofu, Li Zhe, Li Shubo, Chen Shupeng, Wen Jun, Wu Sijie, He Junlin, Chen Yuhong, Clare Rowley, Ivan Mortimer-Schutts, Ines Gensinger, Jorge Vega Sánchez, Zhang Yanzhao, Joey Tianyi Zhou, Liu Yang, Jia Xiaojun.

01

Industry Background



CHAPTER 1

Industry Background

1.1 Agents Move toward Delegated Action and Business Collaboration

1.1.1 From “Software Executes Instructions” to “Agents Understand Goals and Act Autonomously”

Agents powered by large language models are changing how the digital world operates: from “software executes instructions” to “agents understand goals and act autonomously.” Traditional software generally follows predefined rules and workflows. Agents can take a user's goal, decompose it into tasks, select tools according to the circumstances, and adjust subsequent steps in response to execution results. For example, a user may simply ask, “Arrange a business trip for me,” and an agent may check itineraries, compare options, and invoke booking services without requiring the user to specify every operation.

This change goes beyond a higher degree of automation: agents are beginning to participate directly in business execution. They handle tickets and initiate approvals in enterprise workflows, invoke build tools and participate in deployments in development and operations, and request quotations, place orders, or request payments on users' behalf. The closer an operation comes to sensitive data, funds, or critical systems, the more explicit its authorization and controls must be. The ability to look up products does not confer permission to place an order; permission to place an order does not confer permission to pay.

Governance therefore needs to look beyond “what the model can do” or “which account initiated the request” and also answer: who is acting, on whose behalf, under what authorization, whether the conditions for execution still hold, and how actions can be traced and responsibility assessed if a problem occurs.

1.1.2 Behavioral Characteristics of Agents

The following characteristics are particularly relevant to identity governance when agents perform tasks and participate in business collaboration:

- **Autonomous decision–making.** Agents can select steps, tools, and execution sequences within specified goals and constraints. Governance needs to examine not only whether a particular API call is permitted, but also whether a combination of calls is appropriate for the task and whether any operation requires additional confirmation.
- **Dynamic delegation.** A single product may serve different individuals and organizations, and the same instance may perform tasks delegated by different parties at different times. Who provides the product, who operates the instance, and whom it currently represents are related but distinct questions that require separate representation and verification.
- **Cross–domain collaboration.** A task may span enterprises, platforms, cloud environments, and business systems. Participants maintain different account, permission, and data systems. They need to understand externally supplied identity, authorization, and status information while retaining their own governance boundaries.
- **Continuous evolution.** Models, prompt configurations, tools, dependencies, deployment environments, and task conditions may all change. Whether initial checks and existing evaluation results remain applicable must be reassessed in light of the nature and impact of those changes.

During these processes, identity confusion, privilege expansion, credential compromise, or context poisoning may cause a legitimate task to deviate from its intended purpose or even lead to unauthorized operations. OWASP's risk framework for agentic applications [1] likewise identifies agent goal hijacking, tool misuse, identity and privilege abuse, and supply chain risks as key concerns.

1.1.3 From KYC and KYB to KYA

Know Your Customer (KYC) and Know Your Business (KYB) provide a foundation for customer and business due diligence, including identity verification and ongoing risk management in business relationships. When agents begin performing tasks on behalf of individuals and organizations, verifying those underlying parties is no longer sufficient. The agents themselves must also be understood: who provides and operates them, what capabilities they have, whom they currently represent, and the authorization under which they act. This extends identity governance to Know Your Agent (KYA).

Knowing an agent requires information to be connected across provision, deployment and operation, delegation, and service onboarding. Providers describe product provenance and capabilities; deployers and operators supply relevant configuration and operational information; and delegators specify the matters and scope in which agents may act on their behalf. Together, these inputs help service providers decide whether to accept an agent's business request.

The value of KYA therefore lies not merely in adding another object of verification, but in establishing a shared information foundation for agent participation in business: enabling providers to describe their products clearly, users to select and delegate appropriately, and partners to understand onboarding conditions and collaboration boundaries.

1.2 Continuity and Extension of Identity Governance

1.2.1 Existing Identity Capabilities and Agent Adaptation Challenges

Existing enterprise identity systems already provide foundational capabilities for identifying entities, managing credentials, controlling access, and recording operations [2]. Employee identities identify people; service accounts and workload identities identify applications and runtime entities; tokens and certificates support identity and access checks; permission policies constrain resources and operations; and audit logs record access and execution. These capabilities remain applicable to agents and do not need to be rebuilt simply because agents are introduced. Agent identity governance should prioritize reuse of existing identity and access management (IAM), public key infrastructure (PKI), and authorization mechanisms, and align with the organization's security management framework.

However, these foundations do not necessarily cover agents' governance needs in full. Agents may perform tasks for different parties, adjust tools and steps during execution, and collaborate across multiple systems. Managing them primarily as fixed applications, service accounts, and predefined permissions can oversimplify the relationships among identity, authorization, and action. Five principal challenges are illustrated in Figure 1–1.

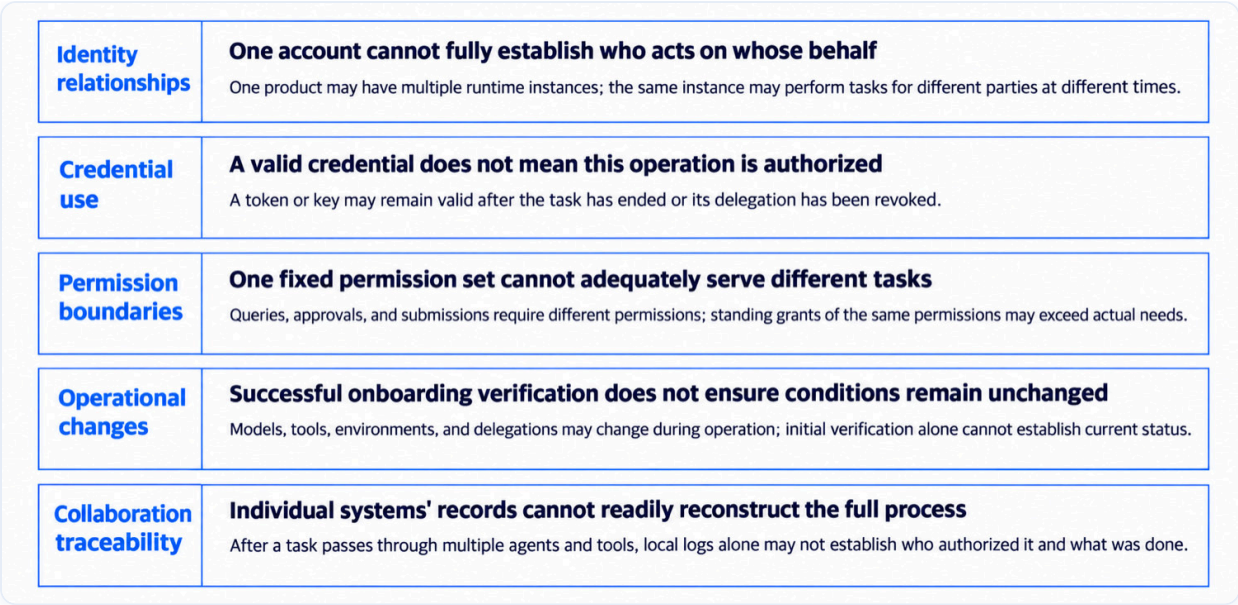

Figure 1–1 Agent Adaptation Challenges for Existing Identity Systems

- **A single service account cannot fully express multiple identity relationships.** An agent product may have multiple deployed instances, and the same instance may perform tasks for different parties over time. Service providers need to identify the specific acting instance and understand its product provenance, operator, and current delegation. Collapsing this information into one service account makes it difficult to distinguish “which agent, on whose behalf, for which task.” Incident response may then affect the entire account rather than the relevant instance or task.
- **Credential management must adapt to changing tasks and risks.** API keys, long–lived tokens, and shared secrets can enable system access, but they generally do not establish the current task, delegation source, or operational boundaries on their own. Access eligibility needs timely adjustment when a delegation is revoked, a task ends, or operational risks change. Agent scenarios therefore require checks not only of credential validity, but also of scope, conditions of use, and revocation status to determine whether the current request remains eligible for execution.
- **Fixed role permissions cannot fully satisfy task–level least privilege.** Role–based access control can provide a permission baseline, but agents' permission needs also depend on the specific task. A procurement assistant requires different resources, operation permissions, and confirmation conditions when querying a catalog, requesting approval, and submitting an order. Broad standing permissions can accumulate unnecessarily; applying the same fixed set to every task may

compromise either security or execution efficiency. Permission use therefore needs to be constrained by task, context, and risk on top of the role baseline.

- **Authentication at onboarding cannot reflect changes throughout ongoing operation.** Models, prompt configurations, tools, dependencies, and deployment environments may change while an agent runs, and delegations may cease to be valid. Successful authentication at startup does not ensure that subsequent operations continue to meet the original conditions. Governance must connect identity with operational status and reassess when key conditions change, rather than continuing to allow access solely on the basis of initial authentication.
- **Multi-agent collaboration complicates delegation, authorization, and action tracing.** In a "user—primary agent—subagent—tool—external service" chain, identifying each caller is not enough. Participants must also understand the delegation under which each acts, whether downstream permissions exceed upstream scope, and how revocation affects subsequent execution [3]. If systems retain only local accounts and call records, the final operation may be difficult to connect to the original task and authorization basis, or to reconstruct when a problem occurs.

These challenges do not invalidate existing identity technologies. They indicate that their application must adapt to the objects being governed. The key question is whether existing capabilities can be connected effectively with agents' tasks, delegations, operation, and collaboration, rather than whether an entirely new technology is used.

1.2.2 Toward Continuous Trust Management

To address these needs, identity governance must extend from managing accounts and their permissions to continuously verifying specific actions. It must look beyond "is it registered and does it hold a valid credential?" to assess "do current conditions support this operation?" and be able to explain that assessment afterward.

This requires connecting identity verification with subsequent operation. Identity information establishes who is acting, on whose behalf, and under what authorization. Operational observations provide facts such as environment status, call records, and fulfillment results. Assessment combines these inputs with the specific scenario to determine whether the requirements for an operation continue to be met. These three types of information are connected and updated as conditions change, extending

identity governance from onboarding verification to continuous trust management for identities throughout operation and collaboration.

This approach does not require the same monitoring intensity for every action, nor a complete reassessment for every operation. Resource sensitivity, operational impact, and collaboration requirements should determine when and what to verify. Low-impact queries may require only necessary identity and authorization checks. Payments, sensitive information access, and production changes may require status updates, additional confirmation, or supplementary verification as appropriate.

Accordingly, this white paper understands continuous trust management for agent identities as a governance capability that establishes understandable, verifiable identity context for an agent and links it to current status, evidence of key actions, and evidence of foundational security capabilities to support participating parties' business decisions. It extends beyond a one-time authentication event and must be maintained throughout operation and collaboration.

1.3 Four Core Questions for Continuous Trust Management

Moving from identity and permission management to the governance of specific actions during operation and collaboration requires answers to four related questions: who is it, is it trustworthy now, are its actions traceable, and have its claimed capabilities and minimum security requirements been verified? Identity characterization, continuous assessment, verifiable evidence and recordkeeping, and capability and security certification address these questions respectively, providing verifiable, updatable grounds for agent participation in business.

- **Q1: Who is it? — Identity characterization.** Knowing an agent requires more than an account or identifier. Participants must establish its product provenance, runtime instance, represented party, and delegation basis. A procurement assistant, for example, may have multiple instances serving different departments. A service provider must identify the instance, party, and task associated with the current request, rather than merely confirm that an account is valid. Identity characterization organizes these objects, attributes, and relationships into verifiable information, making provenance, capability claims, and authorization understandable and providing a basis for identifying the objects and relationships relevant to an action.

- **Q2: Is it trustworthy now? — Continuous assessment.** Once identity and delegation are established, participants must still determine whether current operating conditions support the operation. Tool changes, environment anomalies, behavioral deviations, or revocation of delegation may affect an earlier judgment. The same agent may also require different checks and controls across tasks and times. Continuous assessment updates trust status using identity, environment, behavior, and context information, providing a basis for understanding current status. Participants combine this with valid authorization and local rules to allow an operation, require additional confirmation, restrict permissions, or suspend it.
- **Q3: Are its actions traceable? — Verifiable evidence and recordkeeping.** Identity, authorization, and operational status may change, but historical actions must be checked against the conditions that applied at the time. Retaining only the latest status makes it difficult to establish the authorization, judgments, and outcomes of a past operation. An action may also span multiple participants: an agent's request, a tool's acceptance of a call, and a business system's completion of an operation are recorded separately. The verifiable evidence and recordkeeping framework links these key facts and their supporting basis into verifiable evidence that can be retrieved as needed. It provides a historical factual basis for action verification and supports business reconciliation, audit, correction, and dispute resolution.
- **Q4: Have the claimed capabilities and minimum security requirements been verified? — Capability and security certification.** Business collaboration requires an understanding of whether an agent can perform its claimed tasks under agreed conditions and has necessary security controls against attacks, anomalies, and failures. Can it deliver results that meet business requirements, constrain unauthorized tool use, obtain appropriate confirmation for sensitive operations, and stop or recover after an anomaly? Targeted testing, evidence collection, and evaluation establish which capabilities have been verified, which security requirements are met, what deficiencies remain, and the products and configurations to which conclusions apply. These provide capability and security certification evidence. Appropriate independent third-party evaluation can also help partners understand and use results across organizations, reducing repeated verification.

The four capabilities address different needs and connect throughout business processes. Identity characterization identifies the objects of assessment and recording; continuous assessment updates judgments using operational information;

the verifiable evidence and recordkeeping framework retains key facts and their basis; and capability and security certification supplies verification results for business capabilities and security controls as foundational support. Figure 1–2 illustrates the framework.

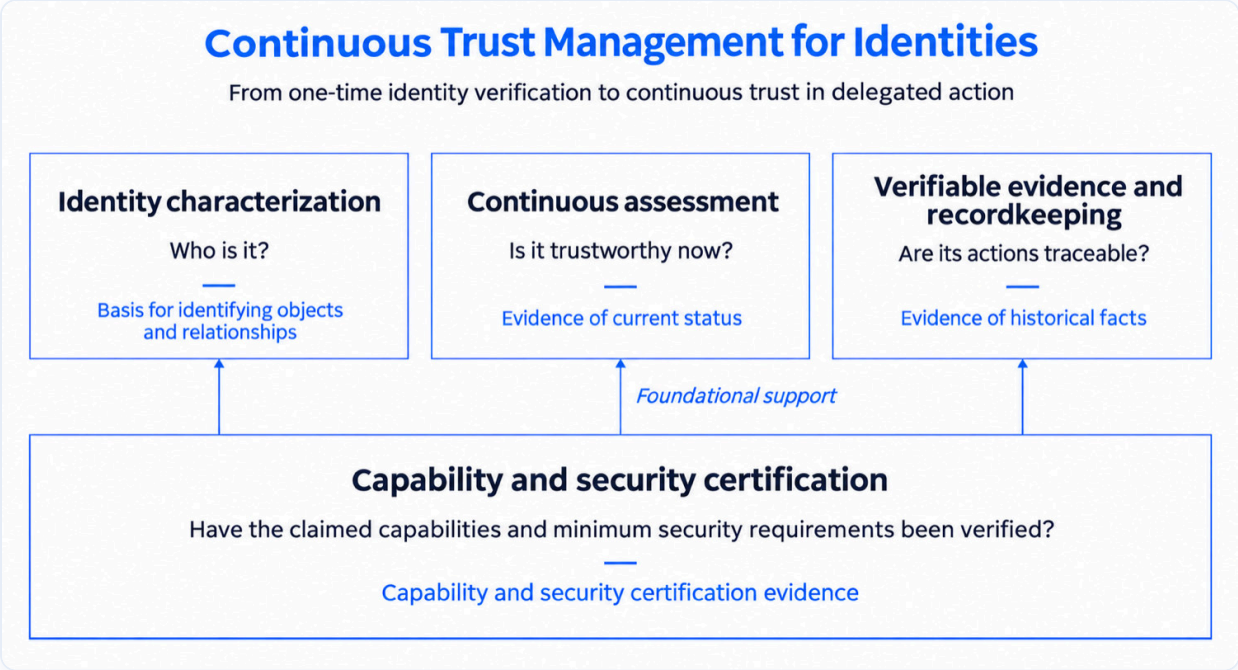

Figure 1–2 Overall Framework for Continuous Trust Management for Agent Identities

1.4 Structure of the White Paper

The following chapters address these four questions. Chapter 2 examines how to characterize agents, including their products, specific instances, and associated parties. Chapter 3 addresses how to judge an agent's current trustworthiness through trustworthiness assessment, cross–domain interoperation, continuous monitoring, and operational feedback. Chapter 4 examines how to preserve the basis for actions and support cross–domain verification: which records to retain, how to protect and link them, and how to verify them when disagreements arise. Chapter 5 addresses verification of agents' business capabilities and minimum security requirements, including how verification is organized, which configurations results apply to, and how judgments are updated after configuration changes. Chapter 6 illustrates continuous trust management for agent identities through two practical cases. Chapter 7 concludes the report and identifies further research and practice priorities.

02

Identity Characterization



CHAPTER 2

Identity Characterization

The first step in characterizing an agent's identity is to establish an identity attribute framework that business systems, security systems, and external partners can understand consistently. It must translate an abstract agent request into verifiable identity facts: which agent and instance acted, on whose behalf, under what authorization, in which environment, and what operation was performed.

This distinguishes an agent identity from a standalone account. Accounts and credentials indicate the account in whose name a request is made, but do not explain the underlying product, instance, or delegation relationships. An agent identity attribute framework must also describe product provenance, runtime instances, responsible parties, task delegation, capability scope, operating environment, and relevant governance status.

2.1 Three Perspectives on Identity Characterization

Agent identity needs to be characterized from three perspectives—product, runtime instance, and associated parties—corresponding respectively to base agent identity, runtime agent identity, and responsible-party identity, as illustrated in Figure 2–1.

- **Base agent identity describes provenance and claimed capabilities.** It describes who provides the agent, the version under which it is registered, its claimed capabilities, and the tools supported by the product. For example, an “enterprise travel assistant” may be registered as an agent product whose base agent identity links its provider, product version, functional scope, and tool claims. Base agent identity is relatively stable and supports product registration, supply chain management, version management, and capability discovery. The tools a product claims to support are expressed separately from those actually permitted in a deployment or task.
- **Runtime agent identity establishes the current instance.** One agent product may run simultaneously across multiple regions, containers, or cloud environments, and undergo model upgrades, tool changes, container migration, and security status fluctuations. Service providers therefore need to verify not only base agent identity, but also “which instance made this request, whether it runs in an approved

environment, and whether it uses approved versions." Runtime agent identity and its associated status must reflect the current instance. Startup, migration, or restart may change instance identity or operational status according to the defined instance lifecycle. Credential rotation updates verification material and does not, by itself, create a new agent instance.

- **Responsible-party identity establishes the parties associated with an action.** An agent product may be developed by a vendor, hosted by a platform, deployed by an enterprise, and act for an employee or customer in a specific task, or accept work passed down by an upstream agent. Responsible-party identity must therefore distinguish enduring affiliations from current task delegation. The former establishes responsibility for operation and foundational governance; the latter establishes whom the agent currently represents, its task scope and usable permissions, and when the delegation expires. It can further distinguish the direct delegator—the party initiating the current task delegation—from an associated responsible party with an operational or governance role, such as the agent provider, deployer, or operator. The identity model records parties and relationships as a factual basis for investigating responsibility; specific responsibility is determined in light of actions, contractual arrangements, and applicable rules.

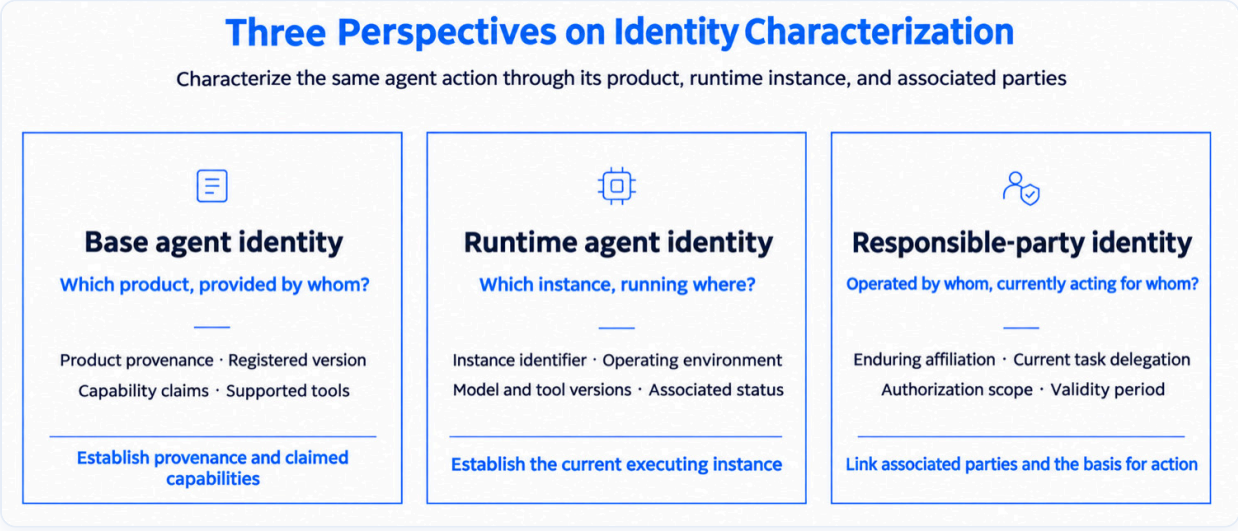

Figure 2-1: Three Perspectives on Agent Identity Characterization

Base agent identity, runtime agent identity, and responsible-party identity are not three independent accounts. They describe the same agent action from different perspectives. In enterprise travel, a vendor-registered "travel assistant" represents base agent identity; the specific instance on an employee device or in an enterprise

cloud, using a specified model version and flight-booking tool, represents runtime agent identity; and its current representation of an employee under a travel-booking authorization capped at RMB 5,000 reflects responsible-party identity. Expressing them separately enables service providers to assess product trustworthiness, instance compliance, and the party's authorization independently, rather than compressing all information into one service account.

Responsible-party identity is grounded in information about real individuals and organizations. For individuals, identity verification within KYC may combine identity document recognition, liveness detection, facial comparison, and checks against authoritative data sources to establish the delegator's identity. Verification results need to be anchored to responsible-party identity as verifiable facts so that subsequent calls can verify delegation provenance. For enterprises, KYB verifies the authenticity, legal status, and relevant qualifications of providers, hosting providers, and deployers separately, and confirms product provenance and deployment authorization at product registration and deployment activation. Enterprise creditworthiness assessment further considers business status, relationships, and compliance records as inputs to subsequent judgments. KYC and KYB include onboarding checks and ongoing due diligence under applicable requirements; creditworthiness assessment supplies additional information about business and compliance changes. Together, they establish a trust baseline for responsible parties from registration through operation and connect with continuous assessment in Chapter 3. As a supra-national, not-for-profit organization, The Global Legal Entity Identifier Foundation (GLEIF) recently also proposed a Agent Mandate Credential (AMC), which could bind an agent's cryptographic identifier to authority delegated by an authorized person acting for a verified organization. The framework is currently under discussion and will drive future development. [4],

Extending these verification capabilities internationally entails differences in identity document systems, fragmented data sources, divergent privacy and compliance requirements, and evolving AI-enabled fraud. Some industry solutions already address these challenges. For example, ZOLOZ [5] provides document recognition and forgery detection covering more than 200 countries and regions, cross-checks against authoritative identity data sources in multiple countries, and develops industry-leading liveness and deepfake detection capabilities through adversarial exercises and live operational experience. It has also obtained international certifications and assurance reports covering ISO/IEC 27001, ISO/IEC 27701, PCI-DSS, and SOC2 Type II. Such

practices, demonstrated at scale, indicate that cross-border verification of responsible-party identities is technically feasible.

2.2 Six Dimensions of Identity Attributes

The preceding section explains how to understand an agent through its product, runtime instance, and associated parties. This section identifies the identity information to record. It can be organized into six dimensions, as shown in Figure 2-2. Each dimension may relate to different identity perspectives; these dimensions are not subdivisions of the three perspectives.

- **Product provenance:** product name, unique identifier, provider, and registered version, supporting product identification and version linkage.
- **Runtime evidence:** instance identifier, operating environment, model and tool versions, and relevant environment evidence, including runtime environment attestation evidence where such a mechanism is used, supporting verification of current operation.
- **Associated parties and roles:** providers, hosting providers, deployers, operators, and delegators, their roles, and references to necessary verification results.
- **Capability claims:** the functions, tools, and task scope supported by the product, for capability discovery and onboarding decisions.
- **Authorization references:** the delegation source, permitted operations, resource scope, limits, and validity period for the current task.
- **Sources of trust status:** references to the services or records from which relevant assessment results and status can be obtained, with update times, for subsequent decisions.

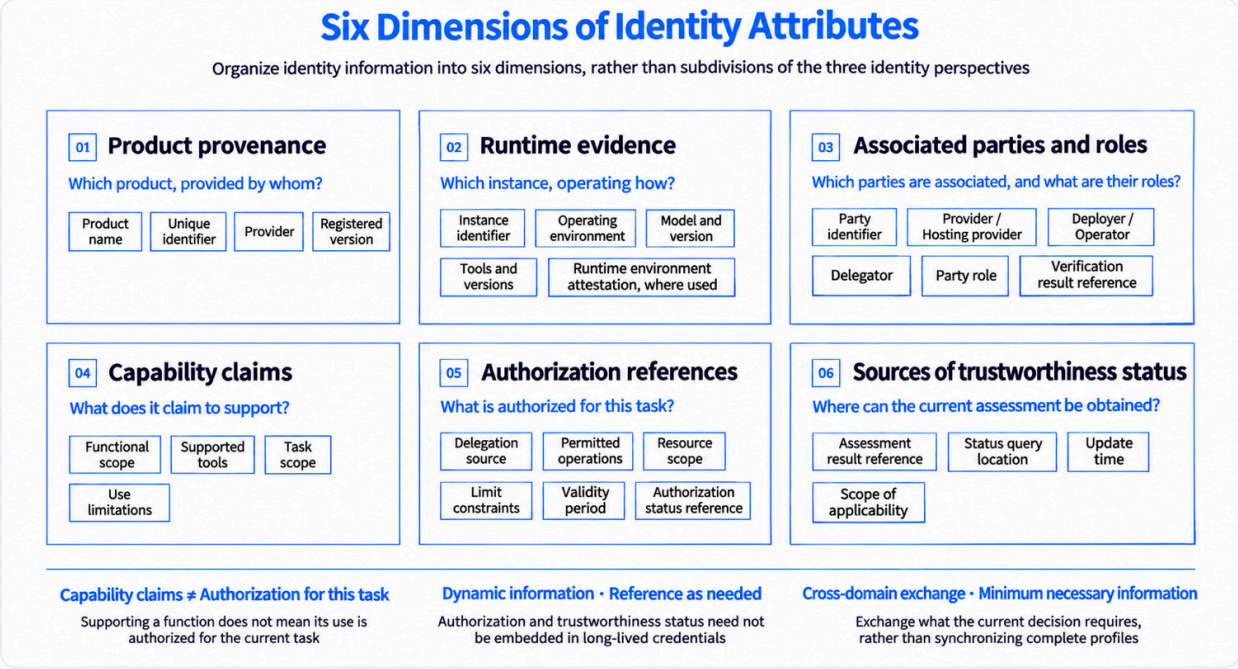

Figure 2–2 Six Dimensions of Agent Identity Attributes and Typical Content

2.3 Modeling Four Types of Relationships

Listing base agent identity, runtime agent identity, and responsible–party identity alone cannot reconstruct an agent action. Identity characterization must describe relationships as well as objects. Otherwise, a system may know “which identities exist” without determining “whether this request was initiated by the correct entity under the correct authorization.” Four relationship types address this need: instantiation, binding, delegation and authorization, and action attribution, as shown in Figure 2–3.

These relationships link products, instances, parties, tasks, and actions. Instantiation relationships connect a runtime instance with its product, version, and deployment records to check whether it corresponds to the claimed product and version. Binding relationships associate the instance with deployers, hosting providers, and operators, distinguishing product provision from actual governance responsibilities. Delegation and authorization relationships link the current task with its delegation source, delegated instance, permitted operations, resources, limits, and validity period. Action attribution relationships connect specific requests, tool calls, and outcomes to the executing instance, task, and identity, authorization, and environment information valid at the time. Together, they explain an action's provenance and course; they are not four sequential steps.

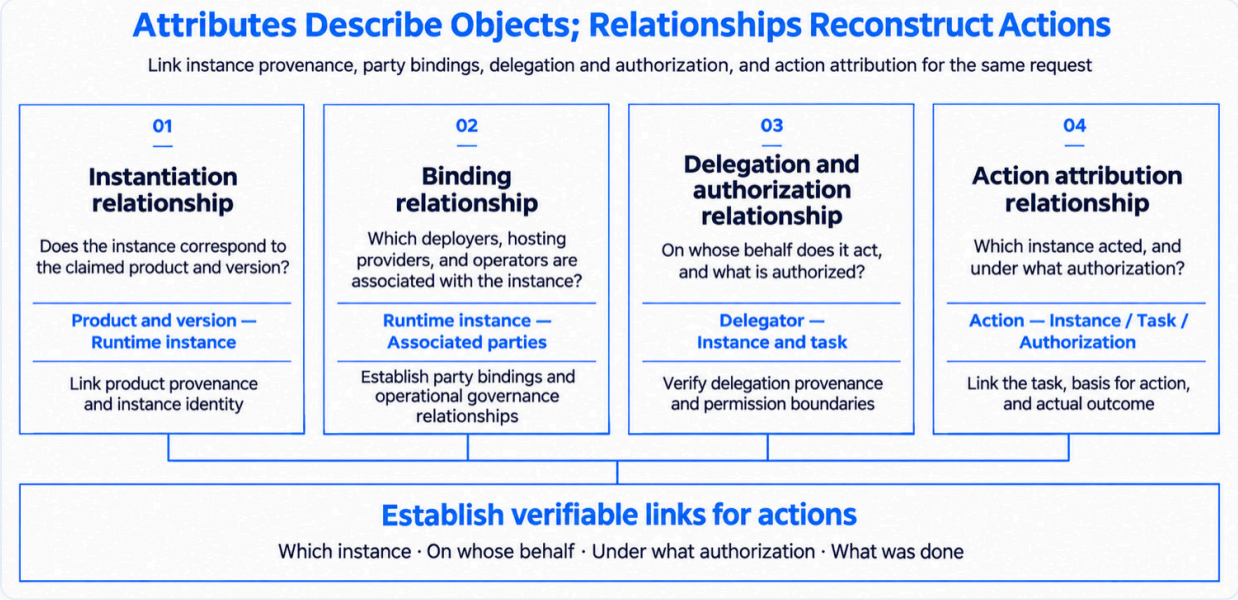

Figure 2–3 Four Types of Relationships in Agent Identity Characterization

For enterprise travel booking, an explainable and auditable action requires more than a record that "a travel assistant used the correct API key to book a flight." It should link the following facts: the request came from a vendor–registered travel assistant; a particular instance on an employee device or in the enterprise cloud initiated it; the instance used approved model and flight–booking tool versions; it currently represented an employee; authorization was based on corporate travel policy and the employee's confirmation of this trip; and the booking amount, dates, and destination fell within valid authorization boundaries. Verifiable relationships help the service provider distinguish a genuine, authorized agent request from one made using misappropriated credentials. They also support reconstruction of the relevant actors, authorizations, and actions when investigating responsibility in a dispute; the identity model does not itself determine legal liability.

2.4 Trustworthy Representation

The value of identity attributes depends not only on what they describe, but also on whether the information can be verified, updated as conditions change, and used to reconstruct historical status. Each key claim needs an explicit source, scope, and validity boundary. As shown in Figure 2–4, trustworthy representation has two complementary requirements: arranging updates and historical retention according to how attributes change, and defining how attributes are claimed, verified, and exchanged. Both apply across identity attributes; they are not sequential steps.

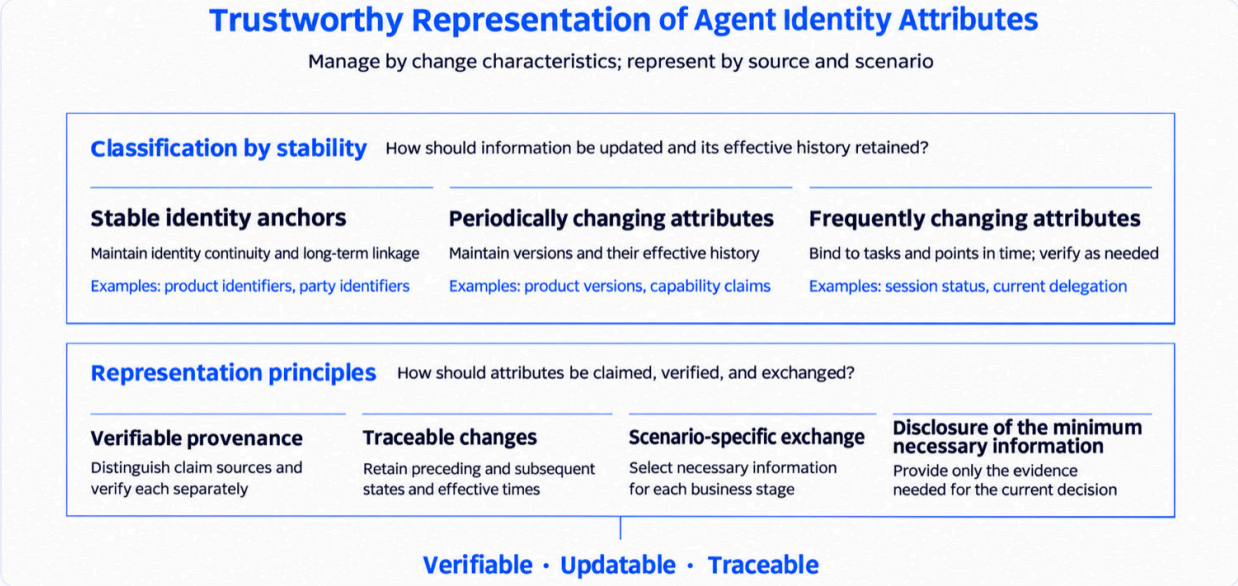

Figure 2–4 Trustworthy Representation of Agent Identity Attributes

Attributes change at different rates. Identifiers are usually stable; model and tool versions change with deployment upgrades; current delegation and credential status may change during task execution. Fixing everything in registration data risks overlooking dynamic changes; creating a new identity for every change breaks long-term linkage. Attributes can therefore be managed in three stability classes.

- **Stable identity anchors** maintain continuity and help determine whether an object remains the same product, instance, or party. Examples include product and party identifiers, and instance identifiers that remain continuous within a defined lifecycle. Stability does not mean immutability. Identifier migration, instance reconstruction, and key rotation should record changes to object relationships separately from changes to verification material, avoiding the assumption that a credential update necessarily changes identity.
- **Periodically changing attributes** can be maintained by version or validity period. Examples include product versions, capability claims, model and tool versions, deployment configurations, and organizational relationships. Updates should retain previous and subsequent states, effective times, and provenance rather than overwrite history, enabling later reviews to identify the version applicable when an action occurred.
- **Frequently changing attributes** are generally associated with sessions, tasks, or individual requests. Examples include session status, current delegation, credential validity, runtime environment attestation status where used, and authorization revocation status. Their use must account for generation time, task, resource

scope, and validity period, rather than whether the value was ever true. These classes are management choices, not fixed categories: update strategies should reflect how the information actually changes.

In addition to update management, four principles should govern the assertion and use of attributes.

- **First, verifiable provenance.** Product provenance, capability scope, party relationships, and delegated authorization may be asserted by providers, deployers and operators, enterprises, users, or third parties. Each key claim should identify its maker, supporting evidence, and verification method, and support independent verification, updating, or invalidation under the chosen format and mechanism. Verifying one claim does not establish that the others also hold.
- **Second, traceable changes.** Product updates, tool changes, instance migration, changes in party relationships, and authorization revocation should retain effective times, prior and subsequent states, and associated objects, and link to relevant action records. Historical review uses information valid when the action occurred, rather than inferring the past from the present. To determine whether a payment was authorized, for example, reviewers need the contemporaneous delegation, limits, validity period, and records. Later revocation alone should not invalidate the authorization basis for an already completed operation.
- **Third, scenario-specific exchange.** Onboarding primarily verifies product provenance, registration status, and the current instance. Authorization adds delegation provenance, task scope, and permission boundaries. Operational governance focuses on configuration, credentials, and behavioral status. Audit links identity, authorization, and execution records as needed. The scenario determines the information required for an interaction; a complete identity profile need not be submitted every time.
- **Fourth, disclosure of the minimum necessary information.** Once the necessary information is identified, its detail and disclosure scope should also be limited. A travel assistant booking through a hotel platform generally does not need to disclose an employee's internal staff number, complete organizational structure, or full travel history. The recipient should verify agent provenance, the current instance, necessary representation relationships, and proof of authorization for the relevant dates, amount, and service scope. Unrelated source data should not be disclosed when necessary claims and proofs suffice.

Combining stability classification with these principles enables identity attributes to support current decisions, preserve historical links, and allow verification as needed across domains. The objective is to give participants necessary evidence with clear provenance, scope, and timeliness, rather than identical copies of a complete profile.

2.5 Chapter Summary

An agent identity attribute framework is more than a field list: it is an information model for expressing objects, relationships, provenance, status, and responsibilities in a verifiable form. Its core value is to extend agent identity from “a number used to locate an object” to “a verifiable set of information supporting onboarding decisions, dynamic authorization, operational governance, cross-domain collaboration, and accountability.” It identifies a connected set of identity facts that can be understood, verified, governed, and traced, rather than an isolated agent. Once objects are identifiable, determining whether they may still act under current conditions becomes the subject of continuous assessment in the next chapter.

03

Continuous Assessment



CHAPTER 3

Continuous Assessment

Once an agent identity characterization model has been established, judgments still need to be updated as operation and collaboration change to ensure that an agent's trustworthiness remains verifiable. Continuous assessment, as used in this chapter, is an overarching mechanism encompassing trustworthiness assessment, cross-domain interoperation, continuous monitoring, and operational feedback. Trustworthiness assessment produces and updates trust status; cross-domain interoperation supports verification and use of relevant identity and status information across domains; continuous monitoring detects operational changes; and operational feedback incorporates response outcomes and business facts into subsequent judgments. These four elements work together to support continuous trust management.

3.1 Trustworthiness Assessment

3.1.1 Basic Concept

An agent's actual risk depends not only on identity information at registration, but also on its current operational status and observed behavior. If an agent that passed registration review operates in a tampered environment, changes model version, or exhibits an anomalous calling pattern, an earlier judgment may no longer apply and should be reassessed according to the nature of the change. Agents may run for hours or days, during which model upgrades, container migration, revocation of delegation, or context contamination may occur. Successful authentication at startup does not establish that every subsequent step remains trustworthy.

Authentication establishes that the claimant controls an authenticator bound to the claimed identity or account; authorization determines whether the current request satisfies applicable permissions and policy conditions. For autonomous agents, however, many risks lie between “fully trustworthy” and “fully untrustworthy”: identity and authorization may both hold while environment, behavior, or task risk has changed.

Trustworthiness assessment supplies this dynamic judgment: how much trust should be placed in the agent under the current environment, task, and risk conditions, and what corresponding level of control is appropriate. This is consistent

with zero trust architecture [6], in which access decisions combine identity, environment status, behavior, request context, and other signals, and are reassessed when conditions change. Figure 3–1 illustrates the concept.

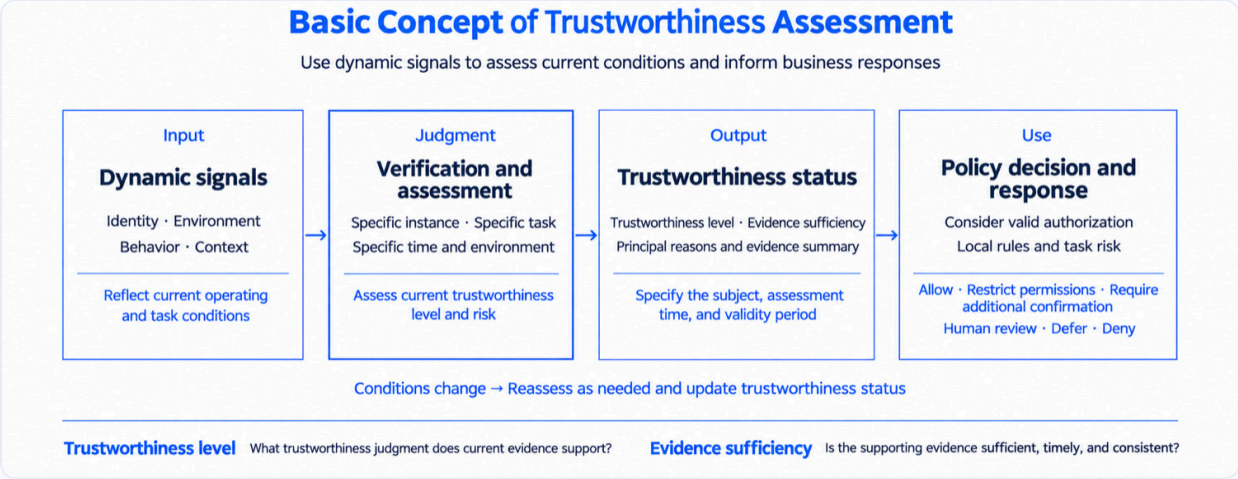

Figure 3–1: Basic Concept of Trustworthiness Assessment

KEY POINTS

Trustworthiness assessment uses dynamic signals, including identity, environment, behavior, and context, to verify and judge an agent's trustworthiness level for a particular task, time, and environment. It outputs a trust status with a timestamp, validity period, and explanation of evidence to inform policy decisions and risk responses.

Trustworthiness assessment does not replace authentication or authorization. Responsibilities are as follows:

- **Authentication** establishes "whether the claimant controls an authenticator bound to the claimed identity or account."
- **Authorization** establishes "whether the current request is permitted under applicable permissions and policy conditions."
- **Trustworthiness assessment** determines "the degree of trust and level of control appropriate for the request under current conditions."
- **Policy decision and response** combines authorization rules, trust status, task risk, and non–negotiable policy constraints to allow, restrict permissions, require additional confirmation, refer for human review, defer, or deny.

trust status should not be treated as a permanent agent rating. The same agent may have different trust statuses across instances, tasks, and times. For example, a collaborating agent with verified responsible-party identity and a stable operating history may be assessed at a high trustworthiness level for information queries. If it moves to a new container environment or invokes an unregistered tool, the existing evidence may no longer be sufficient or applicable, requiring reassessment and, where appropriate, tighter controls.

A complete trustworthiness assessment may include:

- **Assessment subject:** identifies whom the result applies to—base agent identity, runtime agent identity, responsible-party identity, and task or session identifier—so that the conclusion refers to a specific instance and execution.
- **Overall result:** provides the overall judgment, including trustworthiness level, expressed as a tier, score, or combination; risk judgment; assessment time; and validity period.
- **Dimension-level results:** explains the factors underlying the result, each factor's performance, principal contributions, and weaknesses.
- **Evidence summary:** identifies the assessment basis and its sources, including authentication results, binding relationships, runtime environment attestation evidence where available, call logs, and collaborative task records.
- **Evidence sufficiency:** describes how well the conclusion is supported, including data coverage, timeliness, consistency, and missing or conflicting evidence.
- **Response recommendations:** translates results into actionable recommendations, such as allowing, restricting permissions, applying additional verification, deferring, referring for human review, or denying.

Trustworthiness level and evidence sufficiency must be distinguished. The former describes the agent's trust status on current evidence; the latter asks whether the evidence supporting that conclusion is sufficient, timely, and consistent. A newly registered agent may have no adverse record, but should not automatically receive a high trustworthiness level if fulfillment history, required environment evidence, or key behavioral data are missing. A more appropriate approach is a cautious initial status that explicitly identifies insufficient evidence, followed by limited trials, additional verification, and gradual authorization to accumulate supporting evidence.

3.1.2 Design Principles

Trustworthiness assessment should satisfy four design principles.

- **Explainability.** Each result should be traceable to specific input factors, dimension-level performance, and calculation logic. Rather than merely outputting "high risk," the system should explain whether the principal cause is an unverified party, environment tampering, anomalous behavior, revoked authorization, or materially increased task risk. Results should be presented progressively as "overall conclusion—dimension-level performance—evidence summary," enabling business, security, and audit personnel to understand and review them.
- **Updatability.** trust status should change with runtime information and key events, rather than remain fixed after a registration-time calculation. This does not require real-time recalculation for every operation. Low-risk, infrequent operations may use event-triggered or batch updates; operations involving sensitive data, critical resources, production changes, or cross-domain delegation should be reassessed in real or near real time at session establishment, before sensitive operations, or after key status changes.
- **Auditability.** Each assessment should record its subject, input snapshot, metric and rule versions, dimension-level results, overall conclusion, and response actions. In a dispute, the system should reconstruct the assessment basis using identity, environment, authorization, and model versions valid when the action occurred, rather than infer historical results from current data.
- **Interoperability.** trust status formats should support coordination with cross-domain interoperation and continuous monitoring. Section 3.2 describes what may be exchanged across domains and its usage boundaries. Recipients must still make final judgments under local policies and must not equate an external trust status with local authorization.

3.1.3 Assessment Mechanism

Assessment follows "dynamic signal input—verification and assessment—trust status output—policy decision and response." The four input factor categories focus on:

- **Identity-related factors.** Building on Chapter 2's objects and relationships, verify the authenticity, validity, and completeness of provenance, credentials, party bindings, and delegated authorization. Relevant signals include frequent unbinding,

calls after delegation expiry, and anomalous expansion of authorization scope. Existing IAM, PKI, workload identity, and related proof capabilities can be reused.

- **Environment-related factors.** Check model and tool versions, device or container security status, process integrity, request signatures, and replay protection; where runtime environment attestation is used, check its evidence and results against the applicable appraisal policy.
- **Behavior-related factors.** Examine call patterns, historical fulfillment, and behavioral deviations, including anomalous scale, frequency, or counterparties; probing of sensitive APIs; unauthorized calls; task deviation; repeated submissions; and splitting operations to evade thresholds.
- **Contextual factors.** Consider resource sensitivity, irreversibility, high-risk periods, and collaboration chain complexity to assess potential impact. For example, file reading and document deletion require different verification and control intensities even when requested by the same agent.

Metrics within each category should be converted into computable, explainable observations with specified data sources, observation windows, sample sizes, and update times. For example, "completion rate of authorized collaborative tasks over the past 30 days" must be interpreted in light of sample size. "No risk detected following checks" and "relevant data not yet collected" must be expressed separately. Historical behavior metrics may combine near-real-time and longer-term windows to distinguish abrupt changes from sustained performance.

Aggregating dimensions into a trustworthiness level requires three constraints:

- Normalize before aggregation. Numerical methods should address differences in units and scales first, preventing dimensions with more metrics from dominating or an individual metric from exerting disproportionate influence because of its scale. Metrics sharing an event or data source should be marked as related to avoid double weighting.
- Hard-stop events take precedence over aggregate judgments. Confirmed credential revocation, forged signatures, and clear authorization violations must not be diluted by an aggregate score. Detection signals, such as frequent indicators of attack intent, should trigger further verification or restrictions according to evidence and local rules.
- Missing data must not default to a favorable status. Missing information should be reflected explicitly in evidence sufficiency and, when necessary, trigger additional

verification or a cautious default policy.

Service providers or platforms combine trust status, established authorization rules, task risk, resource sensitivity, and non-negotiable policy constraints to decide the response. They may allow execution when authorization holds, trustworthiness is high, and local conditions are met; restrict permissions, perform additional verification, defer, or require human review when trustworthiness is moderate or evidence is insufficient; and deny requests or suspend or revoke the relevant capabilities when trustworthiness falls materially or a hard-stop condition applies. Risk-adaptive responses can balance security and business continuity more effectively than a purely binary allow/deny decision.

trust status is updated after key events, with timing and frequency determined by event impact and operational risk. Typical arrangements include:

- **Identity events** , such as registration, party verification, credential renewal, party changes, and binding or unbinding: event-triggered or batch updates; changes affecting current execution eligibility are verified and enforced within agreed time requirements.
- **Environment events** , such as instance startup, device changes, model or tool upgrades, and failed runtime environment attestation, where used: updates at session establishment, before sensitive operations, or when the event occurs.
- **Task events** , such as task creation, authorization changes, unauthorized calls, and task completion: updates at key task stages.
- **Collaboration events** , such as task assignment, cross-domain calls, execution completion, result feedback, disputes, or revocation due to an anomaly: updates in response to the relevant event.
- **Governance events** , such as policy upgrades, risk-list changes, and credential suspension or revocation: near-real-time updates within the affected scope, with hard-stop events directly triggering restrictions or revocation.

Observation windows and buffers for tier changes can reduce frequent fluctuations around thresholds, but must not delay hard-stop responses. Assessment recommendations, policy decisions, and actual responses are recorded separately so that both decision grounds and execution outcomes can be reviewed.

3.2 Cross-domain interoperation

Identity characterization establishes who an agent is; trustworthiness assessment establishes its current status. If these are usable only within one platform, they cannot adequately support collaboration across enterprises, clouds, and services. Enterprise orchestration agents interact with external specialist agents; risk analysis agents use third-party data services; and agents on different clouds may jointly perform a task. These scenarios require identity, authorization, and relevant status information to be understandable and verifiable externally, rather than recognized only within the originating domain. Cross-domain interoperation is therefore an important foundation for collaboration beyond platform boundaries.

Within a domain, participants can use established trust configurations to verify identity, credentials, party bindings, and delegations, and combine these with operational information to form trust judgments, reusing IAM, PKI, OAuth [7], and workload identity [8]. Across domains, participants maintain separate account systems, trust configurations, risk policies, and business data. Identity and authorization accepted internally by one party are not sufficient grounds for another to accept a request. The recipient must also establish who supplies the external claims, whether the issuer is qualified to make them, how attributes should be interpreted, and whether credentials, delegations, and status information remain valid.

Cross-domain interoperation does not require a unified account system or synchronization of complete internal profiles. Its core is a shared arrangement for verification and acceptance, enabling recipients to verify an agent's identity, representation relationships, and authorization basis, then decide how to handle requests under local rules. Participants need to address four issues jointly:

- **Identity provenance and claim verification.** Establish the claim's source and completeness, whether its issuer is accepted and qualified to make the relevant claim, and its association with the current instance. A valid signature alone does not establish the truth of all claims or business qualifications.
- **Shared understanding of key attributes.** Establish sufficient common semantics for responsible parties, capability boundaries, delegation relationships, and authorization scope to support accurate understanding of provenance, party relationships, and permission boundaries.

- **Minimum necessary information sharing.** Supply the attributes and proofs required by the current task, resources, and risk, without exposing complete organizational relationships, internal account structures, or unrelated business data.
- **Status validity and lifecycle management.** Agree on updates, invalidation, revocation, and query timeliness for identities, keys, party bindings, delegations, and authorizations to reduce divergence between externally held information and the source's current status.

Implementation should proceed by "first recognizing verifiable identity and authorization facts, then exploring recognition of assessment conclusions." Verifiable external information does not oblige a domain to accept its conclusions or grant permissions. Authentic identity does not establish that a current delegation remains valid; intact credentials do not establish that an actual operation falls within authorization. Recipients must still decide according to resource sensitivity, task risk, and local policy.

Once participants have sufficient agreement on assessment semantics, key dimensions, evidence requirements, and usage boundaries, they may exchange trust status summaries as needed. Summaries should identify the assessment subject and scope and include trustworthiness level, key dimension results, principal risk reasons, assessment time, validity period, evidence sufficiency, and the result provider. This prevents scores or tiers detached from their basis from being used in unsuitable scenarios. Recipients verify provenance, timeliness, and applicability before deciding how to incorporate them into local assessment and decision-making.

Chapter 4 explains how credentials, registration, and status verification support trustworthy cross-domain representation of identity and authorization facts.

3.3 Continuous Monitoring

Traditional monitoring generally emphasizes system metrics, API availability, and security events. For agents, these observations must also be connected to tasks, delegations, and actual actions. Agents autonomously decompose tasks, combine tools dynamically, and adjust steps based on feedback; risks may arise in a single call or accumulate across multiple stages.

Environment migration, tool changes, or context contamination may cause claims to diverge from actual operation. Knowing that an API was invoked does not establish whether the call served the task goal, fell within valid authorization, or produced a particular result. Continuous monitoring must therefore link signals with identity, tasks, and authorization, supporting verification, decisions, and responses, and feed verified results into subsequent judgments. This creates an operational loop capable of sensing, judging, responding, and feeding back results, as shown in Figure 3–2.

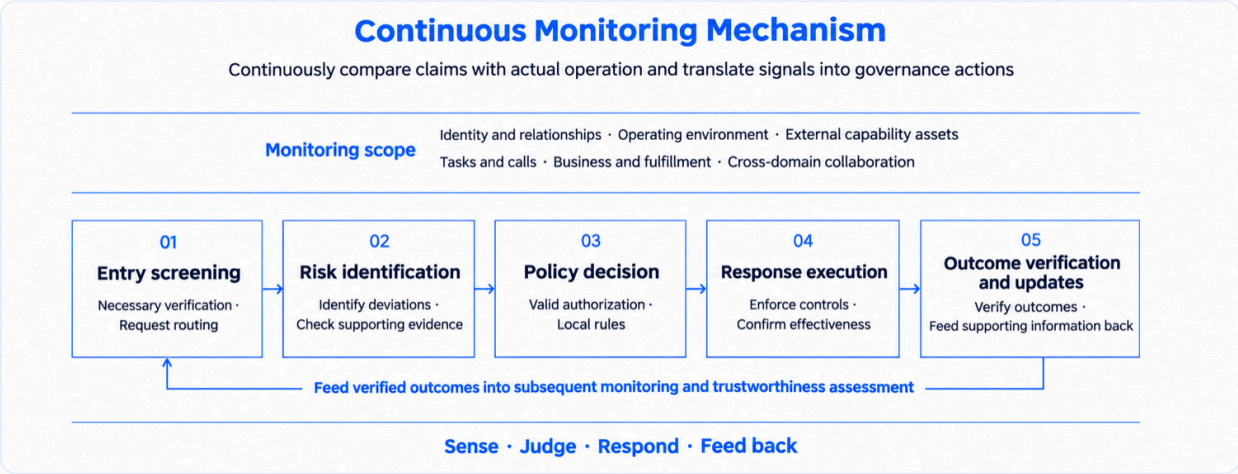

Figure 3–2 Overview of the Continuous Monitoring Mechanism

3.3.1 Monitoring Scope

Continuous monitoring primarily examines whether claims remain consistent with reality: whether the product's claimed capabilities, the instance's associated environment, and current delegation and authorization hold during actual execution. It must also consider external component risks, threat intelligence, and signals correlated across parties to identify changes not covered by existing claims, avoiding the assumption that “registered and authorized” means “always secure.” Monitoring covers six categories, as shown in Table 3–1.

Table 3–1 Monitoring Objects, Observation Focus, and Typical Risk Signals

Monitoring object	Observation focus	Typical deviations and risk signals
Identity and relationship status	Credential validity, party bindings, delegation, and authorization status	Continued use of revoked credentials; frequent changes of represented party; calls after delegation expiry
Operating environment status	Device or container, process integrity, model and tool versions, network location	Migration to an unknown environment; process injection; unauthorized model or tool replacement

Monitoring object	Observation focus	Typical deviations and risk signals
External capability assets	Provenance, versions, permissions, and known risks of skill packages, MCP servers [9] and their tools, plugins, and dependencies	Unknown component provenance or poisoned versions; anomalous permission expansion; critical dependency vulnerabilities
Tasks and call processes	Task goals, observable plans, tool call order, parameters, and returned results	Task deviation; anomalous tool combinations; unauthorized outbound transfer following sensitive data access
Business and fulfillment behavior	Orders, payments, fulfillment, refunds, disputes, and human review results	Sudden increases in amount or frequency; splitting operations to evade limits; outcomes inconsistent with confirmation conditions
Cross-domain collaboration status	External identities and credentials, upstream delegation, downstream permissions, and execution feedback	Downstream execution after upstream authorization revocation; expired external proofs; privilege expansion across delegation hops

These are observation and review leads; one anomaly alone should not establish a violation or malicious intent. Party changes, environment migration, and external calls may be legitimate and should be interpreted against valid authorization, task context, and actual outcomes. Available evidence varies by runtime form, so monitoring scope should match observability and business risk.

Two monitoring areas merit particular attention.

- **First, external capability asset monitoring.** Actual capability depends on skill packages, MCP servers and their tools, plugins, external services, and dependencies as well as the model. Component poisoning, permission expansion, or critical vulnerabilities may affect multiple agents. Risk information from distribution sources must therefore be linked to actual runtime use: source-side monitoring examines maintainers, versions, code changes, and advisories; runtime monitoring identifies which instances installed, enabled, or invoked the component. Together, these establish which agents a component risk affects and which risks a given agent is exposed to. A maintainer change or new version is not itself a risk event, but may trigger provenance and configuration review.
- **Second, consistency between task goals and behavioral outcomes.** Some deviations arise from combinations of steps rather than an individually unauthorized call. Each call may appear permitted while the combined sequence departs from the user's goal or exceeds the task's permitted data use or operational scope. Analysis should link user-confirmed goals, authorization

constraints, observable plans, tool calls, resource changes, and business outcomes to identify task deviation, unconfirmed high-impact actions, and anomalous data flows. “Plans” here means explicitly recorded, observable task arrangements; it does not require collection of internal model reasoning or substitution of agent self-reports for actual execution records.

3.3.2 Attribute Changes and Operational Status Management

Initial registration and claims establish a basis for identity and authorization verification, but attributes, party relationships, and environment status may change during operation. Product updates, tool replacement, instance migration, changes of represented party, permission adjustments, or authorization revocation may alter the conditions under which earlier judgments apply.

Attribute changes must therefore connect registration updates with credential status, trustworthiness assessment, and runtime control. According to the nature and impact of a change, systems should determine which claims, judgments, and execution conditions need updating. Changes affecting execution eligibility or permission boundaries require timely verification of the current basis and, if necessary, reassessment, permission restriction, or suspension. Version mismatches, temporarily unavailable status, and expired evidence should be distinguished and addressed through risk-appropriate supplementary verification or restrictions. Unchanged base agent identity is not sufficient reason to retain an inapplicable runtime judgment.

Change handling should retain effective times, preceding and subsequent states, and related objects, allowing both timely current-status updates and historical review against information valid at the time.

3.3.3 Monitoring Loop

The value of continuous monitoring lies in translating signals into appropriate governance actions and using actual outcomes to update subsequent judgments, rather than in data volume. Five stages connect signals, responses, and feedback:

- **Entry screening.** Route requests according to current identity and authorization, operating environment, resource sensitivity, and business rules. Low-risk requests may follow baseline rules after necessary checks; those requiring further judgment proceed to risk identification.

- **Risk identification.** Perform rapid checks using credential status, trusted lists, and deterministic rules. For unresolved requests, combine real-time signals, task chains, and historical records to analyze anomalous combinations, task deviation, and supply chain risks.
- **Policy decision.** Combine identification results, trust status, valid authorization, and local rules to allow, require additional confirmation, restrict permissions, defer, deny, or refer for human review.
- **Response execution.** Apply controls, confirm whether they took effect, and record execution failures, timeouts, and downstream feedback. Issuing a response instruction must not be equated with effective enforcement.
- **Outcome verification and updates.** Verify actual outcomes before feeding information into operational observation records and trustworthiness assessment inputs. Confirmed unbinding, credential revocation, or component disablement updates affected availability status; environment anomalies, call deviations, and fulfillment outcomes inform subsequent assessments; human review and false-positive corrections calibrate rules, features, and models.

Not every request must pass through all five stages sequentially. Low-impact requests may use a fast path after necessary identity, authorization, and resource checks. Confirmed hard-stop events—such as credential revocation, forged signatures, or clear authorization violations—may trigger immediate blocking without waiting for an aggregate score.

For a high-risk business operation, record the goal and authorization boundaries when the task is created, then verify the responsible party, credentials, trust status, and current authorization before execution. Insufficient or uncertain evidence may require additional confirmation or human review. A clear authorization violation must be blocked and reconsidered only after new valid authorization is obtained; rescoring or additional authentication alone cannot justify allowing it. Verified response and fulfillment outcomes continue to update operational records and assessment inputs.

3.3.4 Implementation Principles

Continuous monitoring should cover necessary risks while controlling collection, computation, and response costs, avoiding excessive performance, privacy, or operational burdens. Implementation should follow these principles:

- **Minimum necessary collection.** Define collection scope according to governance purpose, operational impact, and resource sensitivity. Low-risk queries retain necessary identity and authorization references, tools, operation types, and result summaries. High-risk operations may additionally require task goals, authorization scope, and call chains. Sensitive content in conversations, user information, and tool outputs should preferentially be represented through redaction, summaries, or structured features, rather than retained in full by default.
- **Event-based collection and state-based management.** Events describe what happened; state describes the current situation. Link them to support current decisions and historical review. Monitoring events should have clear identifiers and, as needed, identity, task, action, risk, and response links so that systems can reconcile the same process.
- **Separation of synchronous decisions and asynchronous analysis.** Identity, authorization, and risk conditions that must hold before execution require synchronous checks or valid approval evidence; execution cannot precede the necessary decision. Supply chain inspections, historical behavior aggregation, and quality checks may run asynchronously over appropriate windows. Synchronous paths should carry only necessary context to control latency and data volume.
- **Consistent representation across runtime forms.** PCs, mobile devices, and cloud environments differ in collection capabilities and evidence sources. Secure clients, platform security capabilities, gateways, and workload identity may be used accordingly, but should map to consistent identity, task, tool, risk, and response semantics. Unavailable information must be identified honestly rather than inferred to complete a field set.
- **Coordination of rules and models with clear responsibilities.** Rules handle credential revocation, clear authorization violations, risk lists, and deterministic thresholds. Models help detect indirect prompt injection, concealed data exfiltration, and complex anomalies. Model outputs should identify their principal basis and uncertainty, without bypassing identity, authorization, or non-negotiable rules. High-impact automated responses should retain appropriate human review, appeal, and feasible rollback mechanisms.
- **Data lifecycle management.** Set separate retention periods, access permissions, and expiry procedures for raw data, structured events, and assessment results. Retention or retrieval of original conversations and tool outputs must have an explicit purpose and remain within necessary limits. Expired data should be

deleted, anonymized, or otherwise handled under applicable requirements; evidence that must still be retained by law or agreement remains in controlled custody.

The objective is to connect identity, authorization, trustworthiness assessment, and actual operation into an operational loop that can sense, judge, respond, and feed back, rather than continually expand collection or alert volume.

3.4 Operational Feedback

Operational feedback connects actual outcomes with earlier judgments, extending monitoring beyond alerts and responses to assessment updates and correction. It should identify what happened, what has been verified, what remains unresolved, and whether conditions for reinstatement are met.

- **Outcome feedback.** Responsible parties record task outcomes, downstream execution results, human review, disputes, and anomaly response conclusions separately, identifying provenance, time, applicable rule versions, and the relevant instance, task, or session. Information fed into observation records and assessment inputs should distinguish confirmed facts, signals awaiting review, and corrections. Unverified external ratings and agent self-reports must not be treated as confirmed facts. Corrections should retain their basis and link to the original record and, where necessary, update affected assessments.
- **Reinstatement of status and capabilities.** Before restoring an agent restricted or suspended because of an environment anomaly, insufficient evidence, or risk signals, rectify the anomaly, obtain required evidence, or complete human review; reverify current identity, credential, delegation, and authorization validity; and confirm operating conditions. Restore only the capabilities supported by the checks under local policy, rather than all permissions by default. Correct or invalidate false positives and outdated judgments promptly. Revoked or expired authorization must be obtained anew; lower risk or successful supplementary testing does not restore it automatically.
- **Coordination of synchronous and asynchronous processes.** Feedback should retain event times, result production times, and relevant versions. On arrival, asynchronous results must be checked against their subject and current status to prevent older conclusions from overwriting updated identity, authorization, or risk information. Older results may supplement historical facts but cannot directly justify current execution or restoration. Cross-domain changes are conveyed and

acknowledged through agreed mechanisms, with recipients deciding subsequent action under local rules.

- **Feedback updates do not overwrite history.** Original judgments, response grounds, outcomes, and corrections remain linked so that participants can understand both current status and how it formed and changed.

3.5 Chapter Summary

This chapter addresses dynamic operational judgments through trustworthiness assessment, cross-domain interoperation, continuous monitoring, and operational feedback. Trustworthiness assessment combines identity, environment, behavior, and context to establish current status. Cross-domain interoperation enables verification and conditional acceptance of external information. Continuous monitoring senses changes and connects decisions with execution. Operational feedback updates judgments using verified outcomes, supporting correction and controlled restoration.

Together, these form a loop from signal detection and assessment to response and feedback. The objective is to enable substantiated, adjustable, reviewable decisions based on current evidence, valid authorization, and local rules, rather than assign a permanent trustworthy label. Chapter 4 examines how inputs, rules, responses, and outcomes become verifiable evidence that endures over time and can be used across domains.

04

Verifiable Evidence and Recordkeeping



CHAPTER 4

Verifiable Evidence and Recordkeeping

Continuous assessment and runtime access control provide judgments and constraints for current actions, but after a task ends, participants still need to verify: the basis for decisions at the time, what was actually executed, and what outcomes were produced. Identity, delegation, and authorization information used across domains also needs traceable provenance and historical status so that different participants can reconcile facts about the same action.

This chapter examines how distributed key records can be organized into verifiable, traceable evidence, connecting record creation, presentation of materials, layered verification, and historical review, as illustrated in Figure 4–1. Verification conclusions, responses, and corrections are also preserved as evidence, progressively supplementing the factual basis throughout the business process rather than producing only a final log entry.

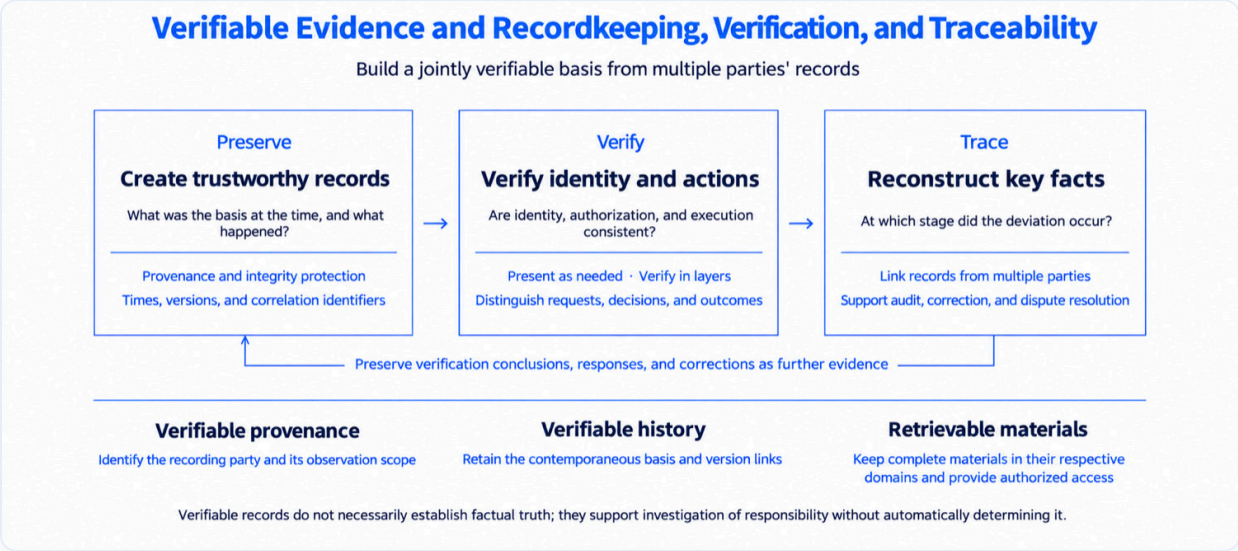

Figure 4–1 Verifiable Evidence and Recordkeeping, Verification, and Traceability

KEY POINTS

Verifiable evidence and recordkeeping means creating and retaining records of key events involving identity, authorization, decisions, execution, and responses, with verifiable provenance and content integrity, linkable relationships, and traceable history, to support business reconciliation, authorization conformity checks, audit, and dispute resolution.

Implementation mechanisms may be selected according to collaboration and custody requirements. A single domain or a setting with established trusted infrastructure may reuse protected logs, signed records, and trusted directories. Where multiple parties need to maintain a shared history and reduce dependence on one custodian, a blockchain may register necessary cryptographic digests, versions, and linkage information, while complete materials remain under controlled custody with their holders or designated custodians. Decentralized identifiers (DIDs), verifiable credentials (VCs), and verifiable presentations (VPs) may be used as needed for party association, claim representation, and presentation during interactions; they are not mandatory prerequisites for every scenario.

The chapter follows "why preservation is needed, what to preserve, how records are formed, how they are used, and how they are traced," and supports verification and reuse of the capability and security verification results discussed in Chapter 5.

4.1 Value of Verifiable Evidence and Recordkeeping

An agent task often involves a delegator, runtime platform, tool or resource services, and other participants. Each holds records of authorization, calls, processing, or outcomes; one log generally captures only part of the process. Records in different systems may not correspond, and tasks spanning organizations require jointly verifiable materials without centrally collecting all raw data.

Historical grounds cannot depend solely on later queries. Authorization may be revoked and status or rules updated; current information cannot fully establish past conditions. Requests, decisions, and outcomes are also distinct facts: allowing an operation does not mean it was executed, and an agent's completion claim does not mean the resource service confirmed success. Retaining the contemporaneous basis

and expressing these records separately while linking them enables checks of decision appropriateness, control enforcement, and the stage at which deviation occurred.

For enterprise agent data analysis, for example, the task owner confirms the purpose and data scope, the runtime platform records calls, the data service records actual access and usage, and the business system retains deliverables. Linking task authorization, access receipts, and outcome records enables participants to compare actual access with authorization and locate authorization changes, execution deviations, or missing records. External data services also require necessary materials and proofs to be shared under agreed arrangements for comparison.

The principal benefits are:

- **A basis for joint verification.** Provenance evidence, signatures, digests, and version links establish who created records, whether content is consistent, and whether it changed, reducing dependence on unilateral log assertions. Protecting a record does not automatically establish the underlying fact; provenance, observation scope, and multiple parties' materials must still be considered.
- **Recordkeeping within respective trust domains and sharing as needed.** Holders or designated custodians retain complete materials and provide necessary content and proofs under access controls, reducing the need to centrally collect identity profiles and business plaintext. Custody responsibilities, retention periods, and retrieval methods must be explicit so that verification does not depend on digests whose underlying materials are unavailable.
- **A connection between current verification and historical traceability.** Linking task authorization, verification decisions, individual operations, cumulative usage, and outcomes supports both runtime conformity checks and subsequent audit, correction, and dispute resolution. Authorization services and enforcement points still impose real-time restrictions; evidence preservation does not replace business decisions.

The verifiable evidence and recordkeeping framework therefore connects key facts and their basis throughout business processes and subsequent verification, response, and correction, rather than merely storing a final result. It supplies evidence for investigation and judgment without automatically determining responsibility.

4.2 Scope and Linkage

Evidence preservation should cover key lifecycle changes in identities, credentials, and related status, and connect records from delegation and verification through execution and outcomes. The former traces information changes; the latter reconstructs the basis and course of a particular action.

Scope should follow the facts requiring joint verification. Not all data needs to be recorded, but coverage, uncollected or unobservable information, and unconfirmed facts should be identified so that partial records are not mistaken for the complete process.

4.2.1 Evidence Objects and Participants' Responsibilities

Records should originate from participants able to observe the relevant facts and responsible for the corresponding function, with explicit provenance and scope. Table 4–1 lists the principal objects and sources.

Table 4–1 Evidence Objects, Record Content, and Principal Sources

Evidence object	Principal record content	Principal record sources
Identity and credential lifecycles	Product and instance registration, party associations, and updates, suspension, and revocation of verification material and credentials	Identity registration services, runtime platforms, credential issuers, and status services
Delegation and authorization	Basis, scope, duration, cumulative constraints, changes, and revocation of delegations and onward delegations	Delegators, authorization services, and delegatees
Verification and policy decisions	Presented materials, status queries, verification conclusions, and decisions to allow, restrict permissions, or deny	Verifiers, status services, and policy decision makers
Task execution and outcomes	Requests, actual operations, protective actions, business receipts, failures, retries, compensating actions, and human intervention	Runtime platforms, tool or resource services, and business systems
Trustworthiness assessment and risk response	Assessment inputs and references, conclusions, rule or model versions, alerts, responses, and subsequent status	Assessment services, monitoring services, runtime managers, and response executors

Evidence object	Principal record content	Principal record sources
Capability and security verification	Objects and configurations under evaluation, criteria, evidence references, results by criterion, scope, and conclusion changes	Evaluation bodies or organizers; for formal certification, certification bodies and parties maintaining certification status

One party may perform multiple roles, but records should still distinguish responsibilities and observation scopes. A platform forwarding authorization materials is not thereby the authorization issuer; an evidence service receiving logs has not thereby observed execution. The collector's protection of acquired materials and the original source's confirmation of business facts should be expressed separately.

Exchanges should also identify collection gaps, custody locations, and access conditions so that recipients can understand what conclusions the materials support and whom to approach for further evidence.

4.2.2 Linking Identity, Authorization, and Execution Records

Using Chapter 2's three identity perspectives, associate the product, specific instance, and relevant operators or delegators with the task. Link individual operations by task identifier, match requests, processing, and receipts by operation identifier, and reference the authorization, status, and policy versions actually used.

As shown in Figure 4–2, an execution record should link four information categories: which instance initiated it and whom it represented; the delegation and authorization applicable at the time; the checks and decisions made; and what was actually executed and with what outcome. Verification conclusions and policy decisions reference their supporting materials; execution and outcome records retain independent provenance, enabling reviewers to trace conclusions to their basis or outcomes back to original requests.

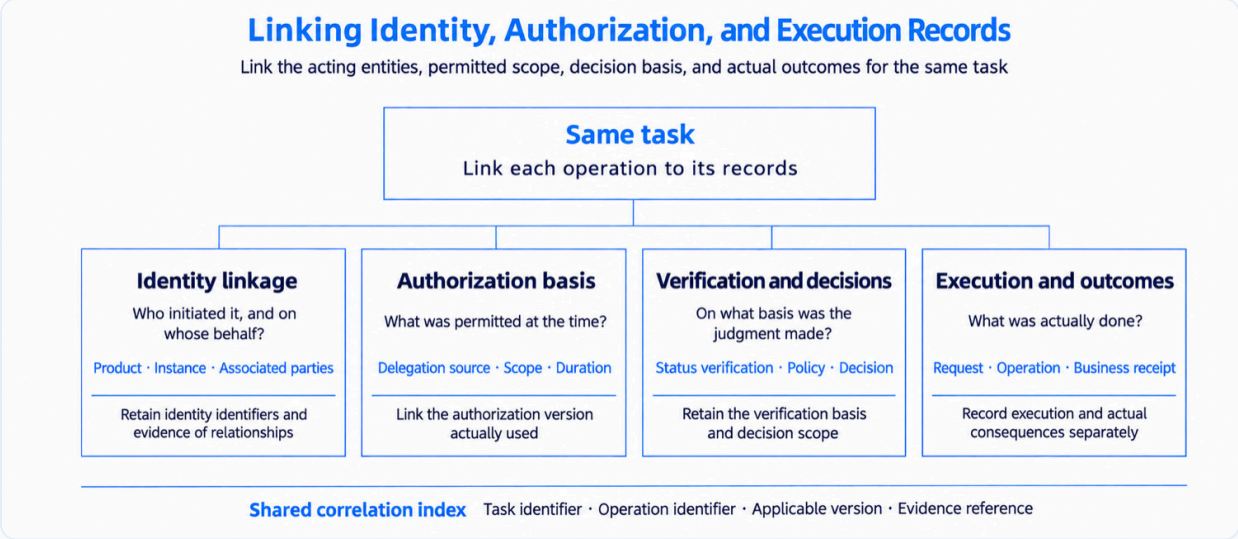

Figure 4–2 Linking Identity, Authorization, and Execution Records

Mappings between identity systems should retain provenance, applicability, and validity times. Identity binding, key possession, and business delegation must also be expressed separately: possession or custody of an authentication key cannot replace task authorization.

Onward Delegation records should reference the upstream basis and specify downstream task scope and permission constraints. Cumulative counts, data volumes, or budgets require explicit measurement rules and distinctions among retries, duplicate execution, and compensating actions to avoid double counting or omitted usage.

Capability and security verification materials should use report or result identifiers to link objects under evaluation, configurations, criteria, evidence, and conclusion versions. Supplementary testing, revision, and status changes should preserve predecessor and successor links so that users can determine whether the referenced conclusion applies to an action.

4.2.3 Time, Versions, and Historical Status

Records should distinguish event time, collection time, status query time, and registration confirmation time. Blockchain use also requires on–chain confirmation time. Asynchronous collection, registration, and cross–domain transmission may be delayed; registration order cannot replace business event order. Reconstruction should consider causal relationships among requests, responses, and preceding and subsequent events.

Changes to identity, verification material, authorization, policies, or evaluation results should link old and new versions and retain effective times and reasons. Verification records preserve the versions, sources, status responses, and timeliness actually used. Where sources lack historical queries, necessary snapshots and their provenance, time, and verification basis should be retained. A copy obtained afterward alone does not establish validity at a past time.

Current requests should use authoritative status that satisfies freshness requirements. Historical reviews should reconstruct applicable versions and the information that executors actually obtained and were required to check under the agreed arrangements. For revocation, distinguish effectiveness, propagation, query, and execution times. Later status should not directly invalidate earlier operations, nor should "unknown at the time" replace examination of status-checking obligations. Uncertain times and states remain unconfirmed facts, with their effect on conclusions identified.

4.3 Registration and Custody

Key records should be collected where the relevant facts can be observed, packaged and integrity-protected, then registered and placed in controlled custody under agreed arrangements. Digests, identifiers, and version references connect registration information with original materials so that subsequent reviews can locate materials, verify content, and understand provenance and applicability.

Existing trusted infrastructure can support registration and custody. Where blockchain is used, necessary digests and linkage information are stored on-chain while complete materials remain in controlled off-chain custody.

4.3.1 Key Event Collection and Record Packaging

Records come from their corresponding sources: delegation and authorization from delegators or authorization services; verification records from verifiers; actual access and business outcomes from enforcement points, resource services, or business systems, reconciled with runtime platform call records. Records should identify the producer, event type, object and task identifiers, applicable versions, relevant times, and references to preceding and subsequent records and original materials.

Sources capable of signing may sign their own claims. A collection service packaging existing logs should identify the original source, collector, and protection method separately. A source signature establishes what the source asserted; a collection signature establishes which materials the collector acquired and protected. The latter cannot replace the original source's confirmation of business facts. Both parties to an interaction may record sending, receipt, and execution separately and link them through the same request or operation identifier.

Events, receipts, and conclusions requiring independent verification or reuse may be expressed as claims with explicit responsibilities: executors confirm processing results, verifiers explain verification grounds, and evaluators state evaluation scope and results, each referencing original records and observation scope.

Denials, failures, retries, compensating actions, human intervention, and collection gaps also require records. Later additions should identify the original event time, addition time, and reason to avoid appearing contemporaneous. Scope should follow key facts and verification needs, without requiring complete collection of internal model reasoning.

4.3.2 On-Chain Registration and Integrity Protection

Where participants need a jointly maintained history and reduced dependence on one custodian, blockchain can retain necessary registration information and reduce unilateral retrospective rewriting. However, on-chain confirmation and the trustworthiness of a business claim's source are separate matters. Confirmation does not establish an issuer's qualification to make a claim. Participants must still agree on accepted issuers, claim scopes, and verification grounds.

Table 4-2 Principal On-Chain Registration Objects, Necessary Information, and Verification Uses

Registration object	Necessary on-chain information	Verification use
Issuers and verification information	Issuer identifiers, verification material references, key versions, and status changes	Check issuance provenance and historical verification grounds
Identity versions	Profile digests, versions, verification material references, and predecessor/successor relationships	Check identity information and changes

Registration object	Necessary on-chain information	Verification use
Credentials and key events	Material digests, necessary metadata, versions, and controlled-access references	Check consistency between materials and registration records
Verification, evaluation, and response results	Digests, versions, and linked references for conclusions and status changes	Trace judgment grounds and subsequent handling

Issuer verification information should be updated after key rotation, deactivation, or compromise response, retaining effective times and version relationships to support historical signature checks. Key compromise also requires reevaluation of the affected record scope; the existence of an old version alone does not establish historical validity.

When events or conclusions are represented as VCs, claims may reference original-material digests, business identifiers, and applicability. Their digests and versions may be registered as needed to connect "original materials—issued claims—registration records," without placing all documents or complete credentials on-chain.

Digest verification requires agreed material objects, encodings, and versions so that participants apply the same rules. Batch registration requires proof that an individual item belongs to the batch. Collection completion, integrity protection, registration submission, and confirmation should be marked separately. Before on-chain confirmation, registration must not be described as complete; this does not negate the existence and protection of records already created locally.

On-chain registration helps verify material existence, versions, and content consistency. It cannot alone establish the truth of underlying facts, current authorization validity, execution conformity, or complete collection of all required events. Requests, receipts, and multiple parties' records must still be checked for gaps in critical paths.

4.3.3 Off-Chain Custody and Privacy Protection

Holders, issuers, or designated custodians retain complete authorization texts, execution details, original business documents, and evaluation reports. Custody arrangements should specify responsibilities, retention periods, access authorization, encryption, access logging, availability requirements, and procedures when materials cannot be obtained. Derived materials should identify provenance, processing methods, versions, and purposes to distinguish them from original records.

Presentation as needed may use separately issued necessary claims, controlled access, or selective disclosure. Selective disclosure requires suitable credential design and cryptographic mechanisms: deleting fields from an ordinary signed credential does not preserve the original signature's validity. Partial content must retain a verifiable connection to the original claim or complete material.

Digests are not inherently anonymous. Stable identifiers, times, frequencies, and reference relationships may reveal parties or business connections without business plaintext; digests alone may also inadequately protect guessable sensitive content. Registration metadata and linkage scope therefore require controls, with access restrictions and privacy protections as appropriate.

At expiry, off-chain originals, access permissions, and indexes should be handled separately, with clarity on information retained in shared registration and its boundaries. Service migration or custodian exit requires appropriate handover of materials and verification grounds. If original or verification materials cannot be retrieved, identify which facts remain verifiable and which conclusions lack sufficient support, avoiding the assumption that registration guarantees perpetual evidence availability.

4.4 Sharing and Verification

Using preserved evidence requires more than matching signatures or digests. It also requires checking provenance, versions, applicable subjects, and validity status, then comparing authorization and business receipts to determine whether an action meets requirements. Verification supplies the basis, the local domain decides, enforcement points apply controls, and each stage retains its own evidence, as shown in Figure 4–3.

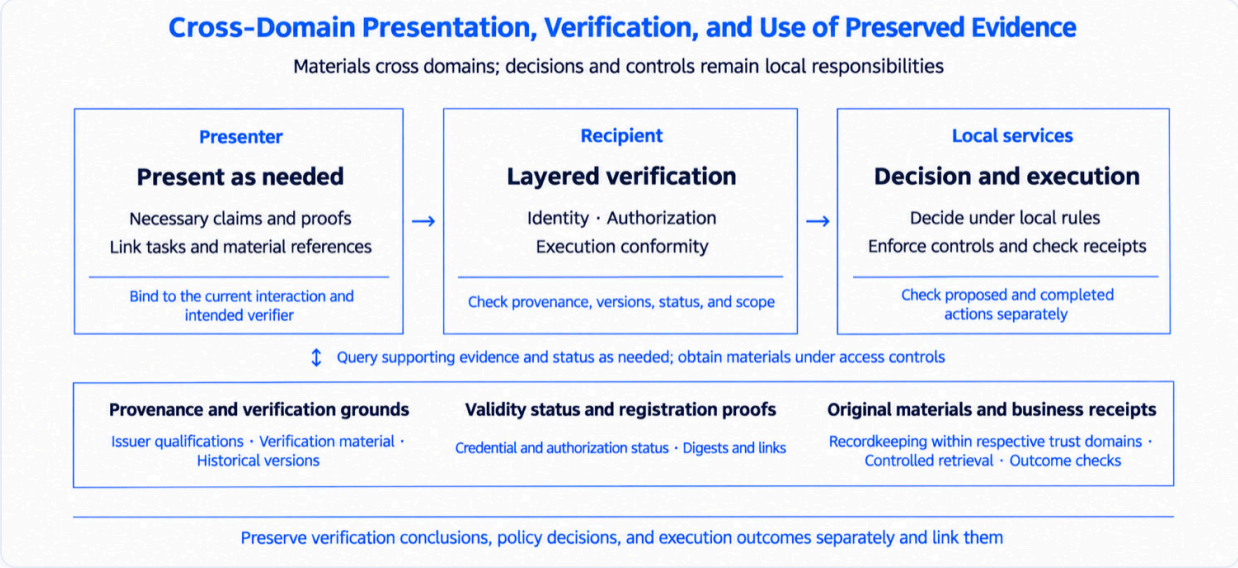

Figure 4–3 Cross–Domain Presentation, Verification, and Use of Preserved Evidence

4.4.1 Presentation and Transfer of Preserved Evidence

Presenters should select necessary claims, provenance proofs, and material references according to the verification purpose. These fall into three main categories:

- **Identity and authorization claims:** product provenance, instance–to–party associations, delegation relationships, and permission scope.
- **Key event and operational verification claims:** actions initiated, outcomes produced, or materials used for verification, linked to tasks, events, times, and applicability.
- **Capability and security verification claims:** objects under evaluation, configurations, verification scope, evaluation results, and evidence and status sources; formal certification additionally identifies certification criteria, the certification decision, and the responsible certification body.

Issuers, claim subjects, and presenters should be identified separately. Possession or forwarding of materials does not confer authority to make the relevant claims. Claims [10] may be represented as VCs; VPs organize the credentials and proofs needed for an interaction; and DID resolution [11] may retrieve corresponding verification methods. Existing identity systems remain supported, without requiring universal adoption of DIDs.

Presentation should use an appropriate one–time challenge, intended verifier, and validity constraints to bind proofs to the current interaction, and verify the presenter's relationship to the claim subject and authority to act on its behalf. Recipients obtain

necessary details under access controls. Further transfer should preserve provenance, applicable subjects, purpose, and time boundaries; a proof created for one interaction cannot simply be treated as valid in another context.

4.4.2 Layered Verification of Identity, Authorization, and Execution

Verification first checks provenance, signature scope, content integrity, and version links, then distinguishes identity, authorization, and execution conformity. It should also examine what source and collection signatures each establish, and the time to which status information applies.

Table 4–3 Layered Verification of Identity, Authorization, and Execution Conformity

Verification layer	Core question	Principal evidence
Identity verification	Does the request correspond to the claimed instance, and do provenance and party associations hold?	Identifiers and mappings, verification material, binding claims, presentation proofs, and relevant status
Authorization verification	Does this operation comply with delegation, resource permissions, time limits, and cumulative constraints?	Delegation chains, authorization records, resource rules, current status, and usage records
Execution conformity verification	Did actual requests, operations, and outcomes meet contemporaneous requirements, and did protections take effect?	Call records, protection records, business receipts, and authorization and policy versions valid at the time

Verifiable materials mean that the supporting basis can be examined, not that a particular business conclusion has already been established. Results should distinguish invalid proofs, insufficient authorization, execution deviations, missing materials, and unconfirmed status, rather than conceal different causes behind a generic pass or fail.

Before execution, check the identity, authorization, and conditions for the proposed operation. During and after execution, use actual records to check conformity. Task delegation and the target institution's resource permissions are verified separately; cumulative constraints require usage records. Authorization services and enforcement points continue to impose real–time restrictions.

For example, suppose outbound calls require confirmation, but an instance submits a request without it. If the server blocks the request and a receipt confirms no outbound transfer occurred, record separately that “the request violated requirements,” “the

protection worked," and "no outbound transfer occurred." Without outcome evidence, the consequence remains unconfirmed; a blocking instruction alone cannot establish its effect.

Reuse of verification or evaluation conclusions also requires checks of the provider's responsibilities, observation scope, object configuration, and current applicability. Such conclusions cannot replace the current business authorization. Unconfirmed status must not default to valid. High-risk operations, such as production changes, permission adjustments, or sensitive data access, should be suspended, subjected to supplementary verification, or referred for human review under local rules.

4.4.3 Cross-Domain Verification and Retrieval of Registration Evidence

Cross-domain refers to trust boundaries among organizations, platforms, or identity systems. Cross-chain refers to one means of obtaining and verifying evidence across blockchain registration systems. Cross-domain collaboration neither requires cross-chain mechanisms nor assumes identical registration systems.

Participants need agreements on acceptable issuers and claim scopes, identifiers and verification methods, necessary mappings, attribute semantics, status sources, timeliness, and disclosure. Claims express subjects and facts; verification material supports provenance checks; registration records retain versions and links; and status services provide information with explicit freshness. Recipients use these with local rules to decide.

Different registration systems may supply evidence through source queries, registration proofs, or agreed verification services. Cross-chain mechanisms should be used only where cross-chain evidence retrieval is actually needed, with explicit verification methods and trust boundaries.

For example, Enterprise A delegates an agent to access Institution B's data service. A creates task authorization defining data scope, purpose, duration, and cumulative constraints, retains the original, and registers its digest and version as needed. The agent presents necessary identity, party-association, and authorization proofs to B, bound to the interaction. B checks provenance, applicable versions, current status, and its own resource permissions before deciding. The data service executes, records actual access and usage, and supplies receipts. Both parties link materials by task, operation, and authorization version, retrieving details under access controls when review is needed.

External trustworthiness assessment summaries may supplement local judgment, but do not constitute local authorization. Unverifiable status remains unconfirmed, with continuation determined by risk.

4.4.4 Preserving Verification Results

Verification conclusions, policy decisions, and execution outcomes should be recorded separately and cross-referenced:

- Verification conclusions identify which checks passed, failed, or remained unconfirmed, and the materials and status evidence used.
- Policy decisions specify the scope, conditions, and reasons for allowing, restricting, or denying.
- Execution outcomes describe actual actions, protective effects, and business consequences.

All records should link tasks, operations, material versions, status times, and applicable rules, distinguishing "verification passed," "execution allowed," and "execution succeeded."

When errors are found, append review, correction, and impact records, retain their relationship to the original conclusion, and state whether it remains usable. Formal revisions to evaluation results or certification decisions should link new results and status issued by authorized parties, preserving historical verification facts, revision reasons, and effective scope. Where necessary, notify participants relying on the original conclusion so that affected judgments can be reexamined.

4.5 Audit and Capability Boundaries

An audit starts with a task or dispute, verifies material provenance, integrity, and links, reconstructs the identity, authorization, and policy versions applicable at the time, then compares requests, decisions, operations, and outcomes to locate deviations in authorization, status propagation, verification, decision-making, or execution. It must identify missing or conflicting materials as well as examine available records, rather than reconstruct the process from one party's conclusion alone.

In the example of Enterprise A's agent accessing Institution B's data service, if A believes access exceeded authorization, the parties retrieve records using retained references, digests, and versions. Task and operation identifiers link the datasets,

purposes, time ranges, and cumulative usage accessed. Review then establishes B's authorization basis, decisions, and the service's actual execution.

Historical review and current verification use different temporal grounds, as illustrated in Figure 4–4. Suppose authorization takes effect at T1, an operation executes at T2, authorization is revoked at T3, and review occurs at T4. Reviewing T2 requires the identity, authorization, and policy versions applicable then, compared with status responses, actual receipts, and cumulative usage. Later revocation alone cannot negate the authorization basis of an earlier operation. A new request at T4 requires current validity checks: revoked authorization cannot continue to be used, and historical records do not automatically extend execution rights.

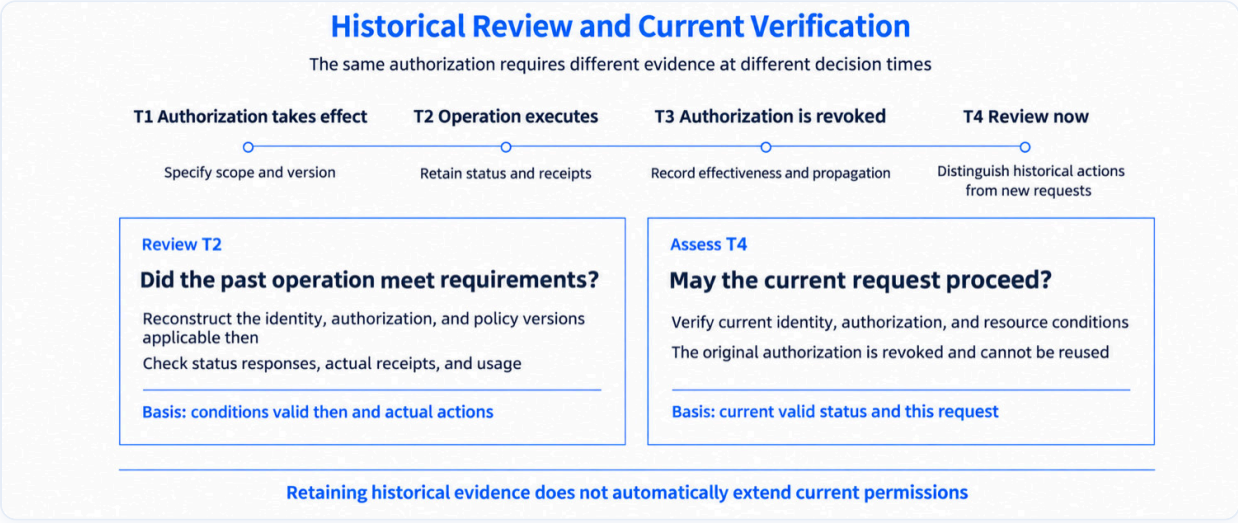

Figure 4–4: Temporal Grounds for Historical Review and Current Verification

Revocation may occur during task execution, and effectiveness, propagation, query, and execution may not be synchronous. Review should establish when revocation took effect, reached the executor, was queried or learned, and when operations occurred, examining both the evidence actually obtained and compliance with agreed status-checking obligations. Uncertain times and states remain unconfirmed rather than being inferred from the final state.

Long-term traceability also depends on continued availability of verification material. Following normal key rotation, trusted historical material and version relationships can support review. Key compromise or loss of source trustworthiness requires reexamination of affected times, claims, and records; successful verification of a historical signature alone does not establish reliability. Missing receipts, irretrievable originals, time conflicts, and unconfirmed facts should be reflected in evidence

coverage and its limits on conclusions, rather than using partial records to infer the complete execution process.

4.6 Chapter Summary

Addressing “are its actions traceable?”, this chapter examines preservation scope, record linkage, registration and custody, cross-domain verification, and audit boundaries. The verifiable evidence and recordkeeping framework organizes identity, authorization, judgments, execution, and outcomes into evidence with verifiable provenance, traceable history, and retrievable materials so that participants can reconcile the same action. It preserves the contemporaneous basis and distinguishes claims, decisions, and outcomes: historical review uses conditions at the time, while current requests require current validity checks. Existing trusted infrastructure can be reused, with shared registration and recordkeeping within respective trust domains as needed, without central collection of all data. Evidence supports audit, correction, and dispute resolution, but does not automatically establish underlying factual truth or replace authorization or determination of responsibility. Chapter 5 builds on this foundation to examine capability and security verification conclusions with explicit subjects, conditions, and evidence.

05

Capability and Security Certification



CHAPTER 5

Capability and Security Certification

Identity characterization establishes provenance and representation relationships; trustworthiness assessment supports dynamic operational judgments; and the verifiable evidence and recordkeeping framework makes key actions and their basis verifiable. Beyond these, business collaboration still needs to answer: whether a product's claimed capabilities have been verified and are assessable, whether minimum security requirements have been implemented, and under what conditions these conclusions apply.

This chapter examines how capability and security certification translates product claims and security requirements into verifiable tasks, produces evidence through testing, evidence collection, and evaluation, and supports procurement, deployment, and onboarding decisions. Testing produces execution evidence; evaluation applies specified criteria to that evidence. These verification activities support certification but do not, by themselves, constitute formal certification. Where an applicable certification scheme exists, a certification body makes the certification decision under its rules. Capability and security have separate criteria and results, each linked to the object, configuration, and evidence.

5.1 Business Value of Certification

Certification converts product claims into verifiable, usable evidence. Its principal functions are:

- **Moving from product claims to verifiable evidence.** Developers generally describe functions, service scope, and security measures, but users also need to know which capabilities have been verified, under what conditions, and on what evidence. System access or a successful demonstration alone cannot substantiate every business commitment. Certification translates claims into explicit requirements and uses specific tasks and evidence to form conclusions, reducing reliance on product descriptions. Differences between claims and measured performance should be recorded with their impact; partial success must not substantiate every commitment.

- **Testing critical capabilities and controls before use.** Normal daily operation does not establish that boundary and exceptional conditions have been tested. Verification should cover normal tasks, misleading inputs, insufficient permissions, service failures, and uncertain outcomes, examining task completion, effective restrictions, and the ability to stop, verify, or hand over when problems arise. Conclusions must correspond to actual deployment combinations: the same model may behave differently with different tools, permissions, data, and human processes. Component results can be reused but cannot directly replace assessment of the complete deployment conditions.
- **Informing procurement, deployment, and onboarding.** Users need results matched to their requirements, rather than an isolated overall score: verified tasks, satisfied security requirements, limitations, and necessary supporting controls. Purchasers compare products, deployers check configurations, and integrating parties define acceptable scope on this basis. Capability and security verification cannot replace one another, and certification does not directly grant business permissions. Results and conditions must be presented together to prevent partial success from being mistaken for universal applicability.
- **Reducing repeated verification and supporting interorganizational collaboration.** Existing results may be referenced when objects, configurations, and requirements match, concentrating additional verification on differences and gaps. Independent third-party evaluation provides evidence distinct from developers' own claims, but acceptance and use still depend on provenance, methods, evidence, and applicability. Reuse removes unnecessary repetition without omitting the recipient's required judgment.
- **Retaining the verification basis for subsequent review and governance.** Certification records establish what was verified, which requirements remain unmet, and how issues were remediated. Certification provides scoped, conditional conclusions rather than a permanent guarantee. Trustworthiness assessment and monitoring identify operational changes; changes affecting applicability trigger supplementary testing, reevaluation, or recertification under the relevant rules. Operational success rates and real-time risk judgments may prompt review but cannot replace documented verification conclusions.

5.2 Objects under Evaluation and Verification Process

Certification requires verification against explicit objects, conditions of applicability, and criteria so that conclusions correspond to particular claims and requirements and are traceable to evidence. Capability and security verification may share object descriptions, test organization, and evidence rules, but should define separate criteria and results. Figure 5–1 illustrates the basic process.

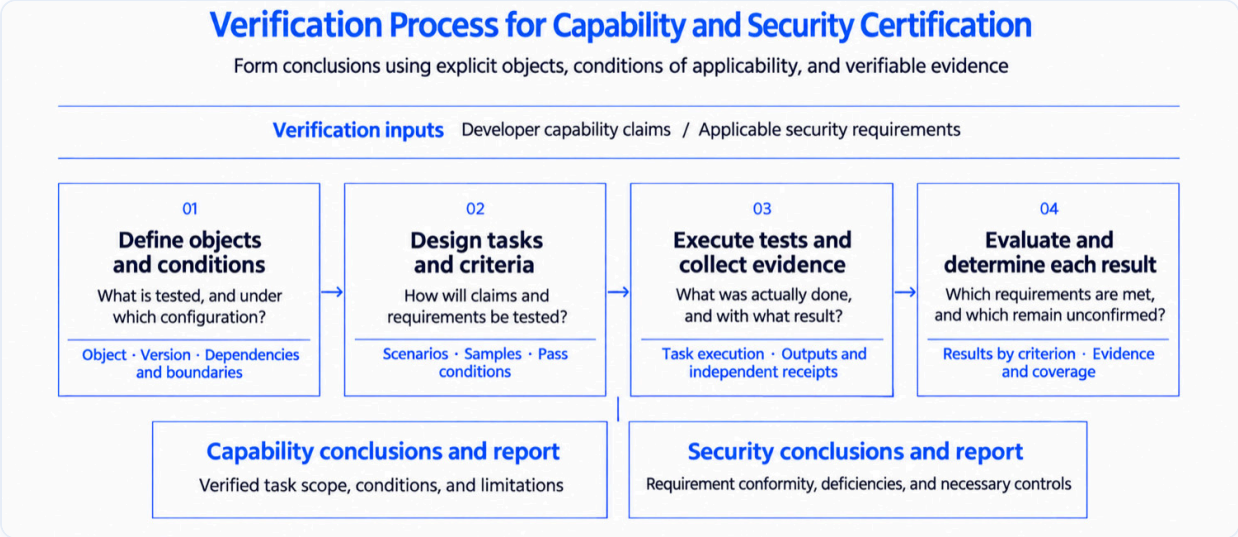

Figure 5–1 Verification Process for Capability and Security Certification

5.2.1 Object and Configuration under Evaluation

An evaluation result or certification decision must first identify its object, version, and configuration. Developers provide capability claims, dependencies, and use limitations, from which the evaluating organization determines verification scope; formal certification follows the applicable scheme. The focus varies by object:

- **Model.** Fix a reference execution framework, tools, and services and describe model performance within that combination, without attributing the entire combination's results to the model.
- **Execution framework.** Specify the model, tools, and environment while retaining the framework's own planning and execution logic, identifying the components and conditions on which conclusions depend.
- **Complete agent.** Verify end–to–end performance for the actual deployment combination, specifying task interfaces, data, permissions, human involvement, and evidence channels.

Verification may use real or simulated services and black-box testing according to observability. If internal information is unavailable, state the observation scope rather than assume internal processes can be checked. If external conditions cannot be fixed, explain their effect on repeatability and applicability.

Test tools organize inputs, observe execution, and collect evidence; they must not plan tasks, complete deliverables, or correct answers for the object under test. Necessary human assistance or additional tools should be recorded as test conditions so that testing-platform capabilities are not attributed to the object under evaluation.

5.2.2 Business Requirements and Verifiable Tasks

Capability verification primarily uses developers' claims, while security verification uses applicable baselines and scenario constraints. Each translates requirements into specific tasks and criteria. Tasks should specify inputs, available resources, expected deliverables, operating conditions, and evidence requirements, with coverage of normal, boundary, exceptional, and necessary adversarial scenarios.

Sample selection, execution budgets, repeated testing, and stopping conditions should be agreed beforehand. Incremental exploration also requires predefined selection rules and records of every attempt and untested area. Successful tasks alone must not be presented selectively, nor should success under one condition imply success across all combinations.

Before execution, confirm that tasks are achievable, environments ready, and evidence channels and evaluation methods available. If prerequisites are unmet, correct them or document the limitations. Unexecuted tasks remain “not tested”; failures of test infrastructure or evaluation procedures do not automatically count as capability failures of the object under evaluation.

5.2.3 Execution Evidence, Result Determination, and Reporting

Reliable verification requires appropriate separation of execution, evidence preservation, and evaluation. Materials visible to the object under test should be managed separately from reference results and evaluation rules. After execution, fix the deliverables and observation records to prevent later modification from affecting conclusions. Agent self-reports, call records, and independent business receipts must identify their sources and cannot substitute for one another.

Implementation includes preparation, task assignment, execution, completion confirmation, evidence collection, and handover. Independent tasks should be isolated as required; sequential tasks retain necessary state. Conditions must not be arbitrarily reset or changed to improve results. Missing observation channels, incomplete service responses, and unconfirmed outcomes should be recorded accurately.

Evaluation should distinguish “pass,” “fail,” “not applicable,” “not tested,” and “insufficient evidence” for each item, and identify test or evaluation procedure failures separately. Model-assisted evaluation requires explicit rubrics, evidence references, and review requirements; tone or fluency cannot establish missing facts. All attempts, key failures, and unresolved matters should be retained, rather than selecting only the best run.

Capability and security produce separate verification results with coverage, evidence, and limitations. Completed testing or generated scores do not automatically establish an overall pass; formal certification additionally requires a decision under the applicable scheme. Section 5.5 addresses retesting, reevaluation, and conclusion maintenance after remediation and material changes.

Capability and security evaluation reports should each identify the object and version, evaluating organization and date, test scope and methods, actual configuration, criteria and evidence references, results by criterion, conditions of applicability, and review requirements. A formal certification report should additionally identify the responsible certification body, applicable scheme, and certification decision. Interorganizational recipients should check alignment with their needs and retain supplementary checks for deployment differences, uncovered requirements, and new risks. Shared materials should be purpose-limited and protect personal information, commercial materials, and security-sensitive details.

5.3 Business Capability Certification

Business capability certification translates developers' capability claims into specific tasks and uses testing, evidence collection, and evaluation to form conclusions with explicit scope and applicability. It should establish not only whether tasks can be completed, but under what conditions, what remains unsubstantiated, and whether results support current needs, as illustrated in Figure 5-2.

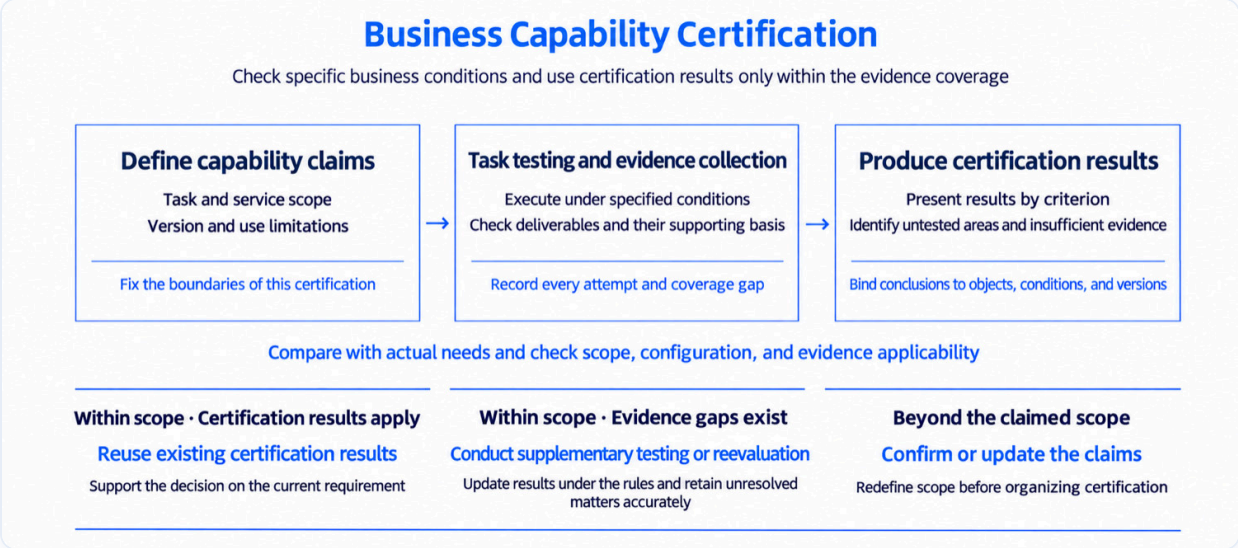

Figure 5–2 Scope and Use of Business Capability Certification Results

5.3.1 Capability Claims

Capability claims generally describe task types, service scope, and use limitations. Certification first fixes the claim and version, then decomposes it into observable requirements, such as geography, temporal granularity, data types, permitted operations, and delivery methods, identifying combinations to test and uncovered areas.

Complex claims may be organized by scope refinement, prerequisites, and combination requirements for staged testing. Passing separate dimensions does not establish every combination; success in daily statistics does not establish hourly statistics. Exploration allocates testing resources without establishing the presence or absence of untested capabilities. Selection rules, budgets, and stopping conditions should be predefined, with uncovered areas retained accurately.

Capabilities discovered outside existing claims may be reported to the developer, who decides whether to supplement or update the claims before a corresponding certification scope is established. A single successful task cannot expand the original claim or certification decision.

5.3.2 Checking Task Outcomes and Supporting Evidence

Certification metrics should follow business requirements and examine deliverables, supporting grounds, and implementation conditions. Applicable metrics may be selected by task type from:

- **Task achievement.** Check goals, service scope, required outputs, and delivery status. A successful query response does not establish completion if it concerns the wrong object or omits the required file.
- **Data quality.** Check accuracy against reliable references, completeness within an explicit scope, definitions, consistency, and source update times. Correct formatting does not establish factual accuracy.
- **Correspondence between sources and content.** For document or image processing, check links among sources, extraction locations, fields, and target objects. Correct partial extraction does not establish complete understanding or source trustworthiness.
- **Interaction and tool use.** Check necessary clarification, call parameters, and multi-step coordination without treating the reference execution path as the only correct method.
- **Stability and efficiency.** Observe completion, elapsed time, and call costs under explicit samples, resource budgets, and human involvement. Results under different loads and service conditions are not directly comparable.

Without reliable ground truth, format, scope, and delivery can still be checked, or related outputs compared using predefined methods, but unsupported conclusions should be identified. Agreement among results does not establish truth. Statistical definitions should be fixed in advance and tolerances agreed for approximations. An agent accurately reporting gaps in source data should be distinguished from one omitting available records.

5.3.3 Certification Results and Applicability

Business capability results should connect developers' claims, test conditions, execution evidence, and results by criterion, identifying supported capabilities, dependencies, failures, untested areas, and insufficient evidence. Results meeting the applicable criteria may support use within their documented scope, but omitting failed or untested areas must not create the impression of overall certification.

Results may be organized into a capability certification profile by task and business condition, displaying conclusions and boundaries. Developer claims and certification results should identify their sources separately, and each certification decision should be traceable to its report, object version, and evidence. Aggregation into a profile does not expand certification scope or establish real-time operational status.

Users should decompose actual needs into conditions such as geography, time range, data definitions, operation type, and deliverables, then check certification coverage and configuration alignment. Similar names or semantics may help discover candidates but cannot replace applicability checks.

Reuse existing certification when conditions match and results remain applicable. Where necessary evidence is missing within the claimed scope, conduct supplementary testing and evidence collection. Where existing evidence suffices but new criteria apply, reevaluate it. Where needs exceed the claimed scope, obtain developer confirmation or updated claims before defining a new certification scope. Comparisons should also check metric definitions, conditions, budgets, and unresolved matters, rather than rely solely on overall scores.

Capability certification establishes the scope supported by existing evidence under specified conditions. It does not guarantee completion of every request or replace request–specific authorization.

5.4 Security Certification

Security certification examines agent behavior under misleading inputs, anomalies, and failures, including compliance with authorization, information protection, and appropriate stopping or recovery. Scenario testing and evidence should reflect the actual deployment configuration and establish which requirements are met, which problems remain, and the conditions on which conclusions depend. Figure 5–3 outlines the principal areas.

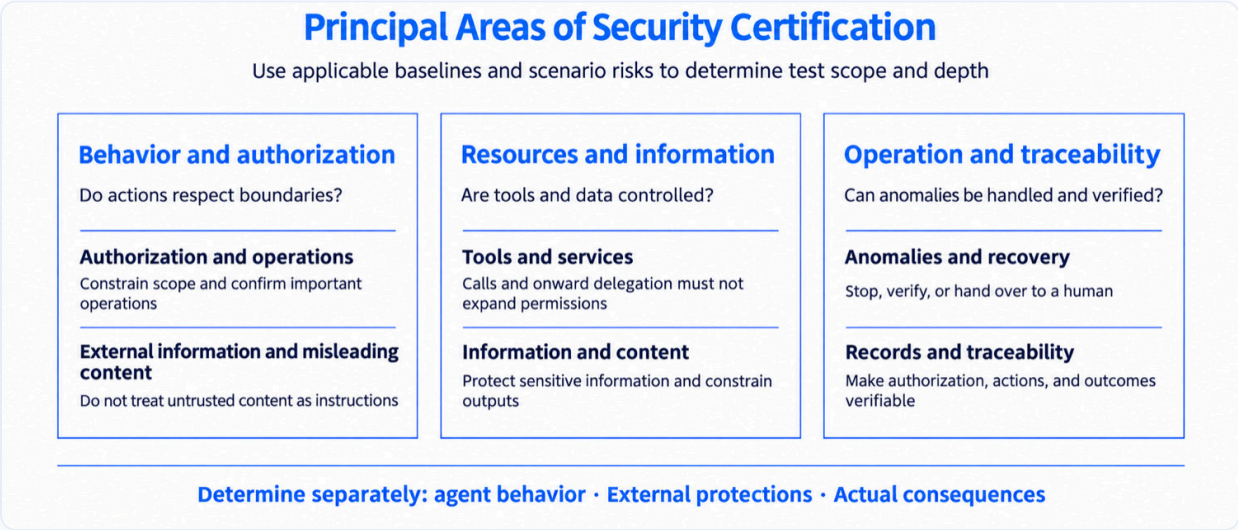

Figure 5–3 Principal Areas and Determination Boundaries of Security Certification

5.4.1 Scenario Risks and Security Requirements

Security requirements should derive from applicable regulatory requirements, industry standards, and scenario risk analysis. Existing requirements and methods should be reused and supplemented according to tasks, permissions, and potential impact, rather than applying one framework mechanically.

Certification first identifies whether the subject is the agent's own behavior, external protections, or the combined deployment. Testing takes place in controlled environments and within authorized scope; active attacks on production are not the default. Applicable requirements are determined principally across six areas:

- **Authorization and operations.** Are tasks, data, and operations within permitted scope, and are important operations subject to necessary confirmation?
- **Tools and services.** Do permissions remain constrained after tool calls or onward delegation, and are higher privileges being used to perform unapproved operations?
- **Information and content.** Do sensitive information handling and outputs meet requirements, and are data from other tasks or customers improperly mixed?
- **External information and misleading content.** Is untrusted content in web pages, files, or tool outputs mistaken for instructions, causing task deviation?
- **Anomalies and recovery.** When services fail, information is insufficient, or outcomes are uncertain, can the agent stop, verify, or hand over without duplicating operations or increasing impact?
- **Records and traceability.** Can important authorizations, actions, and outcomes be verified, and are their records appropriately protected?

Risk determines test scope and depth. Where conclusions depend on tool permissions, approvals, or human review, these should be listed as conditions of applicability. External protective effects must not be described as the agent's own capabilities.

5.4.2 Separately Assessing Behavior, Protections, and Business Consequences

Security certification should include legitimate tasks to confirm normal functionality and risk scenarios to test critical restrictions. Permitted, prohibited, and confirmation–

dependent behaviors should be defined in advance, with separate records for agent-initiated actions, external protection responses, and actual business consequences.

For example, a customer service assistant may read authorized tickets and prepare replies, but external sending requires human confirmation. If misleading ticket content induces it to request sending without confirmation and the sending system blocks the request, the findings are distinct: the assistant violated the confirmation requirement, the external protection worked, and the message was not sent. Successful blocking does not erase behavioral failure. Without actual status or receipts, absence of consequences cannot simply be assumed.

Blocking requirements and criteria should be set before testing according to operational impact, permissions, and deployment conditions. Bypassed confirmation, unauthorized important operations, or sensitive information disclosure cannot be offset by high scores elsewhere. Deployment acceptability must follow predefined requirements, without post-test adjustments to accommodate results.

Where impact can be isolated and the isolation has been tested, certification scope may be restricted under the rules. Shared control failures or uncertain impact require broader examination. Unimplemented remediation recommendations must not be represented as satisfied conditions. Retesting must cover the original issue and affected legitimate tasks so that a security fix does not prevent legitimate business completion.

5.4.3 Certification Results and Conditions of Applicability

Security results should link objects under evaluation, configurations, requirements, evidence, and results by criterion, identifying supported requirements, confirmed issues, uncovered scenarios, and insufficient evidence. Evaluation reports support certification decisions; completing a report does not automatically establish an overall security pass.

Conclusions should specify scope and necessary controls, including tool and data permissions, human confirmation, external protections, and other dependencies. Agent behavior, external protective effects, and actual consequences should be presented separately. A locally successful interception must not be generalized into overall security, nor should absence of loss replace checks of critical requirements.

Unmet requirements should identify impact, relevant blocking requirements, remediation, and retesting needs. Where use can be restricted, conditions should be

explicit. Insufficiently supported matters remain unresolved rather than receiving a determination merely to complete a report.

The certification body describes test facts and certification decisions, developers remediate, and users decide how to apply results in light of valid authorization and their business requirements. Cross-organizational recipients should also verify provenance, object versions, scope, and applicability. Shared materials should protect personal information, commercial materials, and security-sensitive details. Security certification supplies evidence under specified conditions without replacing onboarding decisions or business authorization.

5.5 Retesting and Maintenance of Certification Decisions

Certification decisions apply to specified objects, configurations, and requirements. Remediation, product changes, or new risk information may affect their applicability. Arrangements should define when to retest, which evidence may be reused, and how conclusions and usage scope are updated, while retaining old and new versions and their basis, as shown in Figure 5-4.

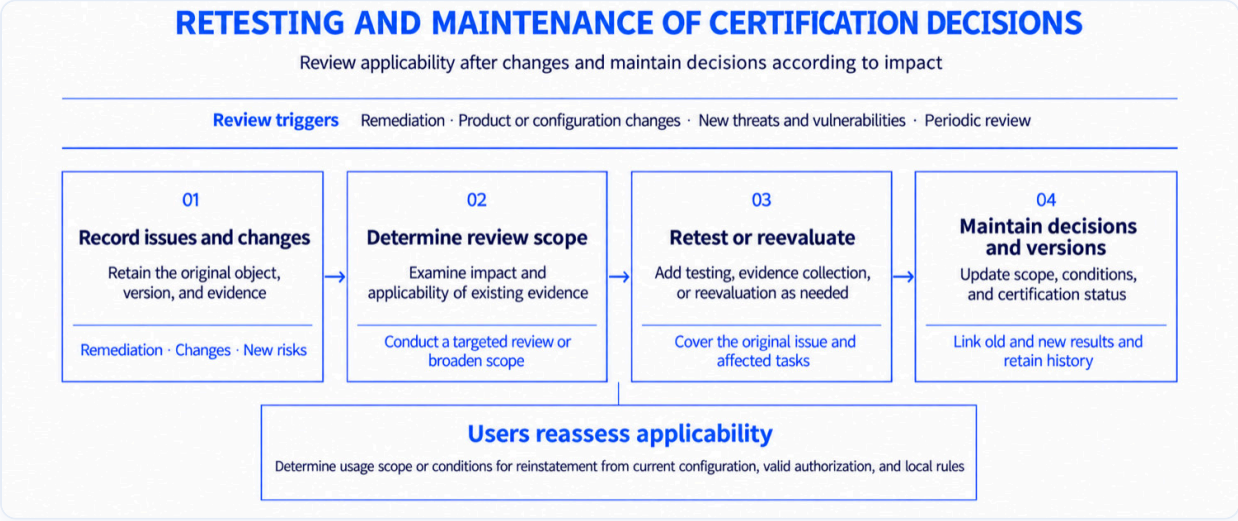

Figure 5-4 Mechanism for Retesting and Maintaining Certification Decisions

5.5.1 Retesting after Remediation

Developers record the original issue, object version, and remediation. The certification body determines retest scope according to impact. Retesting checks resolution of the original issue and covers related scenarios and affected normal tasks, avoiding new defects or loss of legitimate functionality after a local fix.

Clearly bounded, isolatable impacts may permit targeted retesting; shared controls or uncertain scope require broader testing. Major or disputed issues, and those requiring independent review under the rules, should be reviewed by personnel separate from remediation work.

Original failure evidence remains preserved. New results reference remediation, test conditions, and corresponding evidence; later success must not overwrite earlier facts. The certification body incorporates retest results into a revised certification decision or updated report under the applicable rules. Users then decide whether and to what extent to resume use based on current configuration, valid authorization, and local requirements.

5.5.2 Reevaluation and Recertification after Changes

Material changes to claims, models, tools, data definitions, permissions, or deployment conditions require examination of which conclusions remain applicable and which require additional support. New threats and vulnerabilities, critical dependency changes, and scheduled review periods may also trigger examination.

Follow-up activities depend on the change. If evaluation rules change and existing evidence suffices, reevaluation may use the original materials. New execution facts, or evidence that no longer covers current objects and conditions, require supplementary testing and collection. Certification rules then determine whether conclusions must be revised or recertification is needed. Report formatting or wording changes that do not alter results or scope may reference existing results in a revised report, without being described as newly completed certification.

Reused evidence and new tests should identify their respective scope and basis. Unisolatable impacts require broader checks; insufficiently supported areas remain unresolved. Operational monitoring and user feedback may prompt review but do not automatically create new certification decisions. Initiating review does not mean an issue is resolved or certification status restored.

5.5.3 Certification Versions and Usage Boundaries

Certification records should link versions of claims, objects, configurations, criteria, and evidence. Revisions retain predecessor/successor links, reasons, effective times, and impact. Authorized parties maintain current, suspended, or revoked status under the applicable rules. Historical reports preserve contemporaneous testing and certification facts; a historical pass alone does not establish current applicability.

An operator's restriction of current operations and a certification body's suspension or revocation of certification are decisions at different levels. Trustworthiness assessment and monitoring detect operational changes, enabling operators to control requests under local policies. Certification maintainers organize review and update formal certification decisions and status. Operators need not wait for certification status changes before taking necessary measures, and declining operational risk does not authorize them to restore an invalid certification decision themselves.

Records of issues, remediation, retesting, conclusion revisions, and status changes should be retained and cross-referenced using Chapter 4's mechanisms. Operational issues connect to certification maintenance through evidence and formal review requests, making current conclusions queryable and historical grounds traceable.

5.6 Chapter Summary

This chapter examines certification objects, testing and evidence collection, result determination, and conclusion maintenance for agents' business capabilities and minimum security requirements. Business capability certification establishes whether claimed tasks can be completed under specified conditions; security certification checks implementation of authorization, information protection, and anomaly handling requirements. The conclusions are separate and complementary, each requiring explicit objects, configurations, evidence, and applicability. Certification provides assessable, bounded, reusable evidence for procurement, deployment, and collaboration rather than a permanent qualification label. Remediation, configuration changes, or new risks trigger impact-appropriate retesting or reevaluation, preserving old and new results and their links. Certification does not replace current authorization or operational judgments; it connects with identity characterization, trustworthiness assessment, and verifiable evidence and recordkeeping to support delegated action and ongoing governance.

06

Practical Cases



CHAPTER 6

Practical Cases

6.1 Case One: Agentic Commerce and Payments

6.1.1 Scenario and Participants

A user delegates a task to a personal assistant agent: “Book me a flight to Singapore next Wednesday, with an aisle seat, for no more than USD 900 in total, and remind me to check in the day before departure.” This expresses a goal and preferences, but is not yet an executable order or payment instruction. The assistant must obtain necessary details such as origin and date, query flights and prices, check key conditions, obtain confirmation as agreed, and handle booking, payment, and reminders. The following is a reference design for the framework, rather than a claim that a publicly available product already supports the entire process.

The task involves parties with different information and different facts they can confirm. The user and device provide intent, confirmation, and environment information; the assistant organizes options and invokes services. Airlines or booking platforms supply quotations, change and refund conditions, and order and ticketing results. Payment and acquiring services, card issuers, and card networks perform their respective request verification, transaction authorization, and transmission functions. A user's delegation of ticket purchasing and transaction authorization within a payment network are distinct and cannot substitute for one another.

The business process establishes the delegation, queries and confirms options, processes orders and payments, verifies ticket issuance, and provides reminders and after-sales support. Order and payment sequencing, inventory holds, and price validity follow the actual service arrangements. In cross-border transactions, contracting, payment processing, and fulfillment parties may be in different jurisdictions. Their conditions and results require separate checks; one assistant message saying “completed” cannot represent the entire process. Figure 6–1 illustrates these relationships.

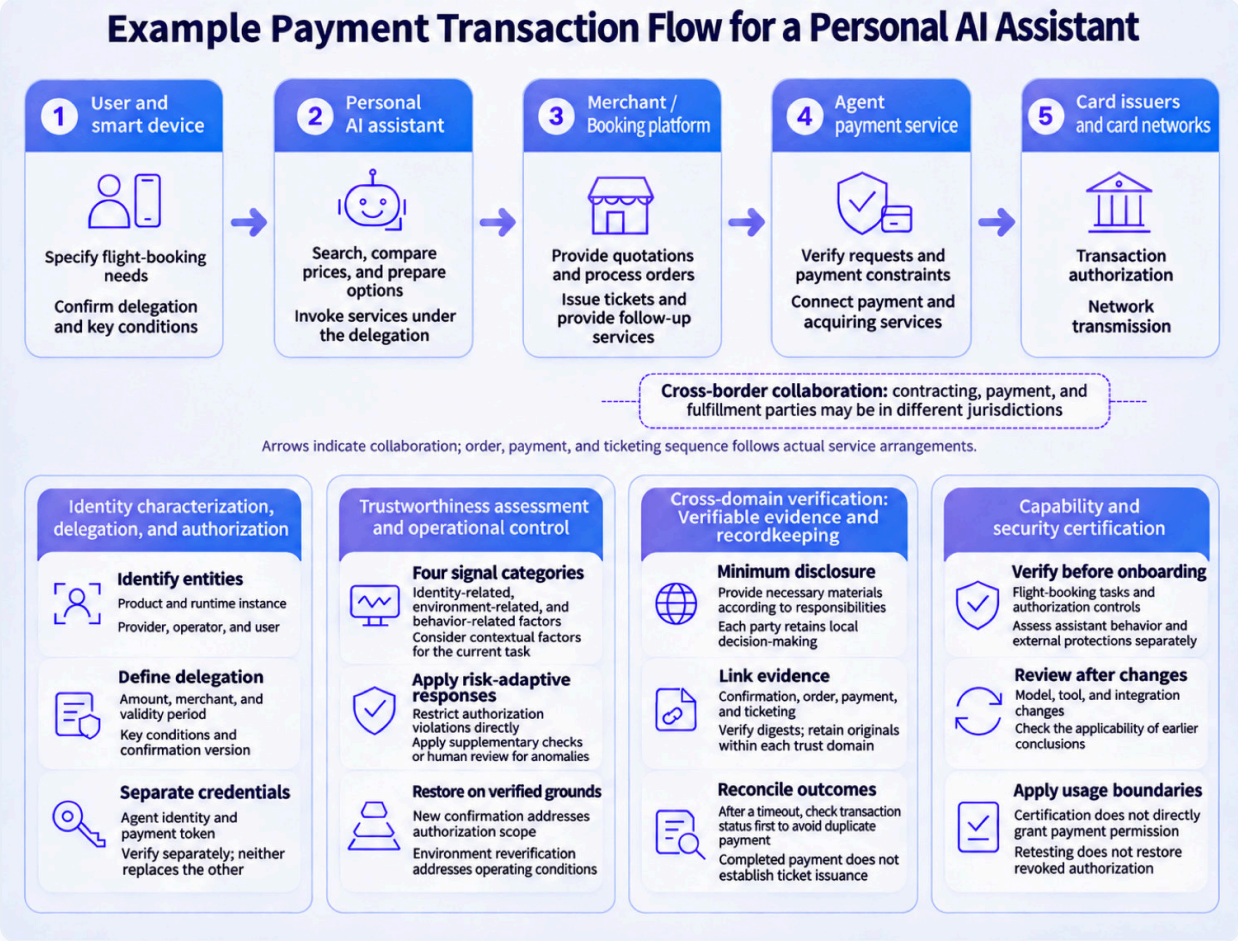

Figure 6–1 Example Payment Transaction Flow for a Personal AI Assistant

6.1.2 Identity Characterization, Delegation, and Authorization

First establish who executes the task and what it may do for the user. Base agent identity helps merchants and payment services understand provenance and claimed capabilities; runtime agent identity locates the requesting instance; responsible party relationships link the provider, operator, and represented user. An agent identifier alone cannot identify the instance using authorization, and user login alone cannot establish the operations delegated to that instance.

At task creation, link user confirmation to the specific instance, task, and valid scope, recording product category, spending cap, currency, merchant scope, validity period, and onward delegation restrictions. After the assistant proposes an option, the user may approve a particular order or permit automatic execution within clear constraints. Operational relationships and user delegation are expressed separately. Identity verification establishes relevant claims and relationships without predetermining responsibility in a dispute.

Delegation must distinguish mandatory conditions from preferences. The USD 900 cap should apply to the final tax-inclusive total within the agreed fee scope and cannot be bypassed by separating service fees. “Aisle seat” must be identified as either a preference or a mandatory condition, rather than inferred by the assistant. Quotes in other currencies require an explicit conversion basis and retention of key tax, fee, change, and refund terms and their confirmed version.

Whether subsequent changes require new confirmation depends on the original scope. Explicitly permitted price variation may proceed to further checks. An amount over the cap, a changed merchant, or material change or refund terms outside prior confirmation requires stopping the relevant submission and obtaining new confirmation. Revocation affects subsequent execution but does not automatically cancel an existing order; cancellation and refunds follow business rules.

Identity and payment credentials should also be verified separately. In this reference design, a controlled payment service links the user, instance, and payment token. Merchants receive necessary agent claims and payment materials rather than the user's original card number. Merchants check the agent's authority and order conditions; payment-side checks cover tokens and constraints on fund operations. A valid token does not mean the user approved the current order, and successful identity verification does not confer payment permission.

6.1.3 Continuous Assessment and Cross-Domain Collaboration

After identity and authorization are established, current conditions still require checks before key operations. Flight purchasing may involve lengthy searches, comparisons, and tool calls while quotations, delegation status, and environments change. Continuous assessment combines trustworthiness assessment, cross-domain interoperation, continuous monitoring, and operational feedback so that participants update judgments with operational information rather than permanently label the assistant product trustworthy. Its four trustworthiness input categories are:

- Identity-related factors: whether product, instance, user binding, and delegation remain valid.
- Environment-related factors: whether device status, runtime configuration, signatures, and runtime environment attestation, where used, meet requirements, and whether unapproved tools or confirmed tampering are present.

- Behavior–related factors: anomalous amounts, frequency, payees, or order parameters, and indications of split orders, task deviation, or injection effects.
- Contextual factors: whether currency conversion, operational impact, time constraints, or onward delegation complexity require stronger verification.

Responses must distinguish definite authorization boundaries from operational risk. A final total above USD 900 clearly exceeds authorization and cannot be paid because of favorable history. A price within scope combined with an anomalous instance environment instead requires supplementary checks, payment restrictions, or human review. Both may pause the task but have different causes and conditions for reinstatement: new user confirmation addresses authorization scope; environment reverification addresses operating conditions. They cannot substitute for one another.

Merchants also control requests according to their business status. Rapid creation of large numbers of unpaid orders, repeated inventory holds, or split orders to evade purchase limits may trigger rate limits, renewed confirmation, or order restrictions under applicable rules, with records retained. Verified agent identity helps identify provenance but does not waive inventory, purchase–limit, or anti–abuse rules.

Current information should be checked before accepting key order terms or submitting payment. Status changes and long–task reviews follow Chapter 3, with responses and business feedback informing subsequent judgments. Expired proofs, temporary proof unavailability, and confirmed environment compromise are handled separately. Trust scores may inform verification and response intensity, but do not create authorization. Credential revocation and clear authorization violations directly constrain relevant operations.

Because decisions are distributed, collaboration must exchange verifiable grounds without requiring identical complete records for every party. The assistant domain holds the user's task and delegation, the merchant domain holds order and commercial terms, and the payment domain holds credentials and transaction status. Each obtains necessary materials for its responsibilities while retaining local decisions and resource control.

Merchants focus on delegation, permitted purchases, and user–confirmed key order conditions. Payment services focus on instance, delegation, token, and payment constraints linked to the transaction. Handover to other services or specialist agents requires checking permitted onward delegation and narrowed downstream scope;

joining a service chain does not automatically confer additional permissions or access to complete spending histories.

A trust registry can help recipients identify acceptable issuers, claim scopes, and validity periods, but trusted provenance does not establish applicability to the current order. Recipients still check current validity, task links, and local rules. Verifiable identity and authorization materials do not guarantee request acceptance.

Disclosure is determined by recipient, purpose, and task; original identity documents, identity references, and order data are managed separately. Further disclosure requires checks of purpose and processing grounds and a record of scope. Cross-border data and transaction requirements must be addressed for specific participants and processing stages; common representation alone does not establish that every party may use the materials.

6.1.4 Verifiable Evidence and Recordkeeping for Outcome Verification

The verifiable evidence and recordkeeping framework links user confirmation, execution requests, and actual outcomes. This case uses evidence anchoring with recordkeeping within respective trust domains: relevant services retain delegation and confirmation, merchants retain quotation terms and orders, and payment participants retain their processing records. Task, order, and transaction identifiers link materials; digests and versions support joint checks, with authorized retrieval of originals.

Outcome checks separately establish whether the request was accepted, the order established, payment completed, and tickets issued. These states may update at different times. Completed payment does not establish ticket issuance, and issuance does not establish subsequent travel fulfillment. Assistant self-reports or a single API response cannot replace receipts from the corresponding business party.

For example, if a complete result does not arrive promptly after payment submission, a timeout alone must not trigger another payment. First query payment acceptance and completion, then reconcile merchant order and ticketing status before deciding to wait, retry, or involve a human. Record confirmed facts and unresolved areas separately. “Assistant execution stopped” must not be recorded as “order canceled” or “funds refunded.”

In a dispute, participants need to retrieve user confirmation, final order terms, request parameters, status checks, and business receipts. Custody responsibilities, access permissions, and evidence indexes should therefore be defined beforehand. Later

revocation also requires distinctions among effectiveness, propagation, and execution times. Records support investigation and substantiation without automatically assigning responsibility; digests alone do not establish business facts, which still require source and multiparty outcome checks.

6.1.5 Capability and Security Certification

Before onboarding, verification should check not only whether payment APIs work, but also whether the assistant can perform the claimed flight-booking task and satisfy confirmation, authorization, and anomaly handling requirements. Capability results identify verified task scope; security results identify supported controls and their dependencies. They are referenced separately under Chapter 5.

Security verification should focus on:

- Whether unauthorized payments are blocked.
- Whether old approvals are reused after changes to amounts, payees, or key order terms.
- Whether prompt injection can bypass authorization or confirmation requirements.
- Whether revocation is enforced for subsequent operations as agreed.
- Whether sensitive actions are allowed by default during service failures or uncertain outcomes.

Record assistant behavior, external protections, and actual consequences separately. If an assistant initiates payment without confirmation and the payment service blocks it, retain both the behavioral failure and successful protection. Successful blocking must not be described as assistant compliance. Human confirmation and external limits remain explicit deployment dependencies.

When citing results, check product version, tools, payment integration method, language, and market scope. After model, prompt, tool, or integration-scope changes, review whether existing evidence applies, then conduct impact-appropriate supplementary testing, reevaluation, or recertification. Certification informs onboarding and use without directly granting payment permissions. Successful retesting still requires current delegation checks and cannot restore revoked authorization automatically.

6.1.6 Case Summary

The key to agentic commerce and payments is connecting user delegation, transaction terms, payment requests, and actual outcomes. User goals become scoped delegations, under which the assistant searches and organizes options. Before accepting key terms or submitting payments, parties separately verify the current instance, valid authorization, and local requirements. Order, payment, and ticketing receipts then confirm actual outcomes, extending verification beyond initial login or binding.

Changed conditions require cause-specific adjustments and restoration checks. Amounts above the cap require stopping submissions and obtaining new confirmation. Environment anomalies require supplementary checks, tighter controls, or human review even if the amount is unchanged. Uncertain outcomes require checking orders and in-flight payments before retrying. Revocation stops subsequent actions, but existing orders and completed payments still follow business rules; stopping a task is not canceling a transaction.

Identity characterization, continuous assessment, and verifiable evidence and recordkeeping connect throughout, with capability and security certification results as foundational evidence. Parties retain responsibility for accepting and executing requests and update decisions as authorization, configuration, and business status change. Collaboration thereby advances from "the assistant says it is complete" to scoped delegation, constrained execution, and verifiable outcomes.

6.2 Case Two: Internal Enterprise Agent Governance

6.2.1 Scenario and Governance Questions

A developer asks an operations assistant: "Following the change request, verify the service configuration in the test environment. After the checks pass and approval is obtained, update production during the change window. If anomalies arise, stop or roll back under the predefined conditions." This involves planning, testing, approval, production execution, and outcome verification, rather than a directly executable configuration instruction. The following is a reference design for the framework, rather than a claim that a product already supports every stage.

The developer provides the change purpose and technical materials; the business or service owner confirms impact and approval scope; the assistant organizes work and

invokes tools; deployment platforms and resource systems execute actual changes; and security and audit personnel examine the supporting basis. The assistant may assign configuration checks and test analysis to subagents, but production execution eligibility cannot be determined solely by a task description or model judgment.

Internal enterprise inputs are not inherently trustworthy. Code repositories, tickets, documents, and tool outputs may contain misleading content, and legitimate accounts may be used beyond task scope. Checks must establish which instance acts, whom it represents, what it may change, and whether current operating conditions still satisfy requirements. After execution, actual production effects require confirmation; an assistant's "completed" message cannot replace resource-side results. Figure 6–2 shows the relationships.

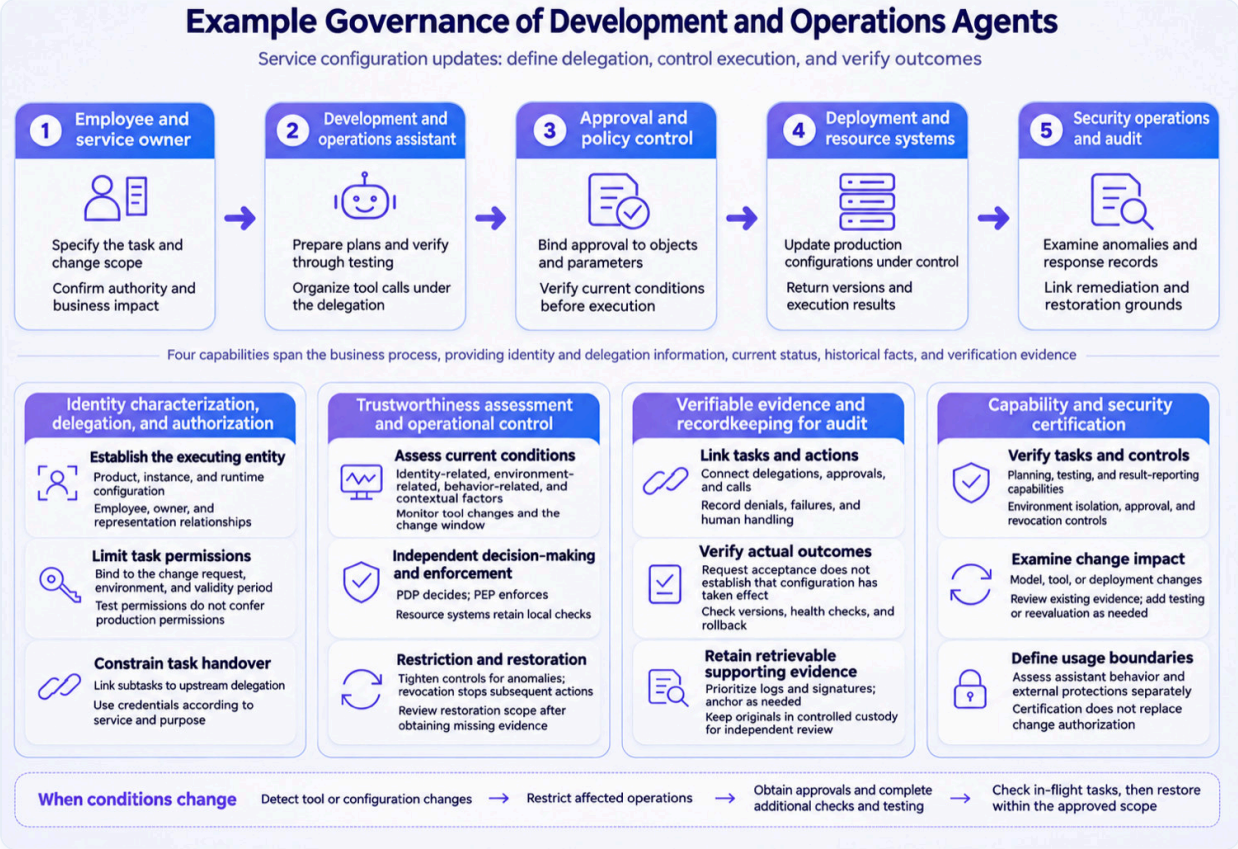

Figure 6–2 Example Governance of Development and Operations Agents

6.2.2 Identity Characterization and Enterprise IAM Integration

Enterprises can reuse existing IAM while adding links among agent products, runtime instances, and task delegations. Base agent identity records versions, providers, and claimed capabilities; runtime agent identity locates specific instances and deployment configurations; responsible party relationships identify developers, service owners, and

technical administrators. Employee identity establishes the source of authority; instance identity identifies the executor. The change task connects them, rather than substituting one shared service account or owner field for all relationships.

Task delegation should reference the change request and constrain target services, environments, permitted actions, configuration versions, and validity. Querying test results, generating configuration drafts, and updating production require separate authorizations; access to test environments does not imply production write access. A primary agent may only narrow permissions within permitted scope when handing tasks to subagents, retaining links among parent delegation, subtask, and executing instance.

Long-term identity identifiers and task-duration credentials are managed separately. Keys and tokens should be supplied through controlled mechanisms, not embedded in prompts, knowledge bases, or model context. Tool calls use credentials appropriate to the target service rather than indiscriminately forwarding upstream tokens. Staff departure, reassignment, or service ownership changes require review of responsibility transfer and residual authorization. Temporary permissions are withdrawn after task completion as agreed.

Valid identity materials do not directly establish that current runtime configuration matches registration. Tool replacement, model updates, and deployment migration require controlled release, instance binding, and necessary runtime evidence checks; original signatures and product names alone cannot justify continued production access.

6.2.3 Continuous Assessment and Action Control

Current execution must satisfy the employee's authority to delegate, task delegation, deployment approval, change approval, and resource-side rules. Current risk may justify tighter constraints. Trustworthiness assessment supplies operational status evidence without creating permissions or replacing change approval.

This design organizes controls by impact and reversibility, with specific levels determined by the enterprise:

- Restricted read-only operations. Query authorized service status, configurations, and logs with data scope limits, sensitive information filtering, and access records.
- Drafting and test operations. Generate plans, modify test configurations, or run approved tests within specified environments and resources, without automatic

extension to production.

- Important production changes. Bind production configuration updates and service parameter changes to explicit approval, pre-execution verification, and necessary restoration arrangements.
- High-impact operations. Autonomous privilege elevation, audit policy changes, or irreversible bulk operations are unavailable by default. Necessary exceptions use dedicated controlled channels and human review.

Continuous assessment updates judgments from identity, environment, behavior, and context. Revocation of delegation, changed tool provenance, anomalous bulk calls, target environment changes, or the end of a change window may affect subsequent operations. Expired proofs, temporary unavailability, and confirmed environment compromise require distinct treatment, including supplementary verification, restrictions, isolation, or suspension. Each restriction and restoration should state its basis rather than provide only an unexplained score.

Controls are enforced by mechanisms independent of the model. A policy decision point (PDP) considers authorization, approvals, non-negotiable constraints, and current risk; a policy enforcement point (PEP) enforces decisions at gateways or tool wrappers; deployment platforms and resource systems retain their own checks. Approvals should bind change requests, target environments, services, key parameters, and validity windows. Material target or parameter changes require renewed applicability checks. Resource systems should also verify current configuration versions to avoid execution under outdated assumptions after resources have changed since planning.

Stopping the assistant, revoking credentials, and terminating an actual deployment are distinct. Revocation requires stopping credential issuance, canceling queued tasks, and terminating operations that can be stopped. Accepted or partially completed changes require actual-status queries before deciding to continue, roll back, or involve a human. Rollback itself needs authorization and applicable conditions; not every failure warrants automatic rollback. Sensitive production operations must not be allowed by default when authorization or policy status cannot be verified.

6.2.4 Verifiable Evidence and Recordkeeping for Audit

Evidence should connect change purpose, approval grounds, requests, and outcomes. Registration and task creation record products, instances, and configurations;

delegation and approval record who permitted changes to which objects; execution links tools, parameters, target versions, and call identifiers; post–execution evidence includes deployment status, actual configuration, and necessary health checks. Denials, failures, timeouts, aborts, rollbacks, and human handling are retained alongside successful releases.

Parties record facts they can observe or confirm and identify sources separately from evaluation conclusions. Developers confirm task content; authorization services retain delegations and approvals; deployment platforms record execution; resource systems supply actual status; and verifiers and auditors explain their judgments. “Request accepted” does not mean configuration took effect, and passing tests does not establish production deployment completion. Missing receipts or conflicting materials leave status unresolved; query before deciding to retry.

Records primarily capture observable inputs, summaries of decision grounds, tool requests, and business outcomes, without requiring internal model reasoning. Tasks, change requests, decisions, calls, and deployment records should be linkable and identify collection times, gaps, and later additions so that auditors can reconstruct contemporaneous conditions rather than see only retrospective success summaries.

Evidence infrastructure should have independent identity and permission arrangements so that assistants cannot selectively retain records themselves. A single enterprise domain may prioritize trusted directories, protected logs, and signatures. Where independent evidence is needed across legal entities or organizations, digests may be anchored as needed while originals remain in their respective domains. Signatures and digests verify provenance and integrity without alone establishing authorized operations or successful changes. Registration, retrieval, and historical verification follow Chapter 4; synchronous on–chain recording of every call is not required.

6.2.5 Capability and Security Certification

Verification of development and operations assistants should separately establish task capabilities and security conditions. Capability checks examine understanding of change scope, compliant configuration generation, test organization, and accurate reporting. Security checks examine environment isolation, permission control, approval binding, tool use, revocation, and failure handling. The object under evaluation is the actual deployment combination, not only the model.

Scenario tests should focus on control interfaces: whether test credentials can access production; misleading document or tool content can alter targets; parameter changes after approval are detected; subtasks stop after revocation of the parent delegation; and timeout retries trigger duplicate deployments. Releases, stops, and rollbacks each require predefined expected behavior, observation points, and resource-side outcomes, beyond checking the assistant's answers.

Assistant compliance and external protection effectiveness should be assessed separately. If an assistant requests an unapproved production change and the deployment platform blocks it, retain both behavioral failure and successful protection. Controls depending on human approval, deployment platforms, or resource restrictions remain explicit use conditions rather than being attributed entirely to the assistant.

Enterprises organize four sets of verification activities under Chapter 5:

- Onboarding. Verify actual configurations and necessary controls; failure of a blocking requirement prevents enablement of the relevant capability.
- In-service review. Recheck key controls according to risk; unmet requirements trigger restrictions or suspension.
- Changes. Review applicability after changes to models, tools, prompts, purposes, or deployment conditions, with supplementary testing or reevaluation as needed.
- Incidents. Review identity, approvals, execution paths, revocation, and evidence in light of the incident, rather than only changing prompts.

Results should link tested versions, tools, environments, and conditions for use in onboarding, authorization policies, and continuous assessment. Test-environment results do not automatically cover production; an old deployment-tool report does not automatically extend to a new connector. Authorized maintainers handle formal certification status under the rules, while enterprise operators may restrict current operations first. Successful supplementary testing neither directly restores revoked permissions nor replaces approval for the current change.

6.2.6 Case Summary

Internal enterprise agent governance can reuse IAM, change approvals, deployment controls, and audit infrastructure, adding runtime instance linkage, task-duration delegation, configuration associations, and actual-outcome checks. Consider replacement of a deployment tool during a task. The new tool may change accessible

environments, credential use, and execution paths; this is more than a name update. When the deployment platform detects inconsistency with approved configuration, it passes the change to continuous assessment. The PDP restricts affected production writes, the PEP stops new submissions, and accepted deployment tasks are queried. Local rules determine whether query or diagnostic capabilities remain available.

Parties then supply their respective grounds. The service owner confirms whether the new tool serves the original purpose and whether prior approval covers current conditions. Technical administrators verify provenance, configuration, credentials, and target environments. The evaluation team identifies reusable report evidence and additional checks needed for release, stopping, and rollback. Business agreement to use the tool does not replace technical verification; successful supplementary testing does not replace production change approval.

Before restoration, check queued tasks, the currently effective version, and partially completed changes. If outcomes are unclear, the window has closed, or valid confirmation is unavailable, the responsible owner takes over and decides whether to defer, cancel, or address existing effects under the rules. The model must not infer missing prerequisites. Rollback requires separate confirmation of authority, target version, and current applicability.

Once required approvals, configuration checks, and verification are satisfied, the PDP rechecks current authorization and risk and restores the necessary scope. Support limited to testing must not be extended to production; revoked or expired delegation requires new authorization. Identity services, deployment platforms, resource systems, and security operations retain their records and link them to the original anomaly so that restrictions, responses, and restoration remain verifiable. One organization may perform several roles, but records should distinguish them.

The case focuses on reconciling who proposed and approved a change, which instance executed it, and what actually happened in production. Identity characterization establishes the executor and delegation; continuous assessment supports decisions and responses after changes; the verifiable evidence and recordkeeping framework links action grounds and outcomes; and capability and security certification supplies foundational verification evidence. Together, they clarify which operations to pause, what further evidence to obtain, and the conditions for reinstatement.

07

Conclusions and Outlook



CHAPTER 7

Conclusions and Outlook

7.1 Report Conclusions

Agent participation in business requires a progression from identity verification to continuous trust management. KYC and KYB establish a foundation for verifying individuals and organizations. When agents perform tasks on their behalf, participants also need to understand who provides and operates them, whom they currently represent, their capabilities, and the authorization under which they act. KYA establishes a shared information foundation for understanding acting entities and verifying their relationship to the current task, rather than merely adding another object of verification.

To this end, the report proposes four connected capabilities. Identity characterization establishes objects and representation relationships; continuous assessment updates judgments through trustworthiness assessment, cross-domain interoperation, continuous monitoring, and operational feedback; the verifiable evidence and recordkeeping framework retains key actions and their basis; and capability and security certification supplies verification results under specified conditions. These four forms of evidence support decisions without creating permanent trustworthy labels or replacing authorization and business rules. Participants still need to verify scope, versions, and timeliness and impose appropriate restrictions for clear authorization violations or revoked credentials.

Moving from framework to application requires connecting who may delegate, what an agent may do, and what it actually did. Commercial collaboration compares user delegation with order, payment, and fulfillment outcomes. Internal enterprise governance compares employee authority and change approvals with production actions and verifies actual execution results. Implementation can reuse IAM, approvals, resource controls, and audit systems, first covering critical activities such as sensitive data access and important operations, then expanding progressively. Measures should reflect business risk and identify verified and uncovered areas and controls dependent on external systems or human handling.

7.2 Outlook

Further work should first address how shared information is represented and used. Existing specifications and practical gaps [12] can be reviewed across identity relationships, authorization grounds, trust status, evidence linkage, and certification results. Pilots can then test whether information is linkable, status updates are timely, evidence is retrievable, and results can be reused appropriately. Shared semantics do not require identical systems or algorithms; they enable recipients to understand materials' provenance, subjects, and applicability and make local decisions.

Research should then focus on coordination across parties, time, and configuration changes. Multi-hop delegation requires traceable authorization without expanding downstream permissions. Identity migration, version updates, and key rotation must avoid relationship confusion and residual permissions. Revocation must take effect across caches, long-running tasks, and downstream calls. Further work should examine long-term verification of historical evidence, result reuse after configuration changes, and the balance among necessary disclosure, privacy protection, and jurisdictional requirements in cross-domain use.

These efforts require industry participation and progressive testing and verification in real business settings. Device manufacturers, large language model companies, platform providers, payment institutions, security technology companies, testing and certification bodies, and research institutions can reuse existing standards, protocols, and tools, pilot specific needs, and improve rules and collaboration based on results rather than seek a complete solution in a single step.

Agent identity provides value beyond identifying a request's origin. It helps participants understand who acts for whom, whether current action conditions are met, whether historical facts are verifiable, and which capabilities and security requirements have supporting verification evidence. This evidence must be clearly expressed, continuously maintained, and open to examination when disagreements arise. Collaboration that connects identity, authorization, operational control, and evidence within specific tasks can progressively establish a verifiable, maintainable foundation of trust.

[1] OWASP GenAI Security Project. OWASP Top 10 for Agentic Applications 2026 [online]. OWASP Foundation, 2026, https://genai.owasp.org/resource/owasp-top-10-for-agentic-applications-for-2026/

[2] National Cybersecurity Center of Excellence. Accelerating the Adoption of Software and AI Agent Identity and Authorization: Concept Paper [online]. NIST, 2026, https://www.nccoe.nist.gov/projects/software-and-ai-agent-identity-and-authorization

[3] Cloud Security Alliance. Securing the Swarm: Governance, Attack Surfaces, and Zero-Trust Architectures in Multi-Agent AI Environments [online]. CSA, 2026, https://cloudsecurityalliance.org/blog/2026/06/24/securing-the-swarm-governance-attack-surfaces-and-zero-trust-architectures-in-multi-agent-ai-environments

[4] Global Legal Entity Identifier Foundation. Agentic AI in Payments: Establishing Interoperable Trust and Control. GLEIF, 2026, https://www.gleif.org/media/pages/organizational-identity/research-publications/bd5cf9e83d-1788248740/2026-08-13_agentic_ai_in_payments_v1.0-1.pdf

[5] ZOLOZ, https://www.zoloz.com/

[6] NIST. Zero Trust Architecture (NIST Special Publication 800-207) [online]. NIST, 2020, https://csrc.nist.gov/pubs/sp/800/207/final

[7] IETF. The OAuth 2.0 Authorization Framework: RFC 6749 [online]. IETF, 2012, https://www.rfc-editor.org/info/rfc6749

[8] IETF. Workload Identity in Multi-System Environments (WIMSE) Working Group Charter [online]. IETF, 2024, https://datatracker.ietf.org/doc/charter-ietf-wimse/

[9] Anthropic. Model Context Protocol [online]. Anthropic, 2024, https://modelcontextprotocol.io/

[10] W3C. Verifiable Credentials Data Model 2.0 [online]. W3C, 2025, https://www.w3.org/TR/vc-data-model-2.0/

[11] W3C. Decentralized Identifiers (DIDs) v1.0 [online]. W3C, 2022, https://www.w3.org/TR/did-core/

[12] NIST Center for AI Standards and Innovation. AI Agent Standards Initiative [online]. NIST, 2026, https://www.nist.gov/caisi/ai-agent-standards-initiative