Resource Rights:
Authority as a Managed Resource for Autonomous Systems

A discussion document based on conversations with many
Comments to foster@anl.gov

July 2, 2026

Abstract

Scientific cyberinfrastructure is undergoing a fundamental transformation as autonomous AI agents become active participants in scientific discovery. Rather than serving solely as interactive assistants, these agents increasingly design simulations, analyze experimental results, orchestrate workflows, control laboratory instruments, and coordinate complex computational campaigns spanning high-performance computing systems, cloud platforms, scientific repositories, and experimental facilities. Existing authorization mechanisms were designed for direct human interaction with individual resources and do not scale naturally to this emerging model of machine-mediated science.

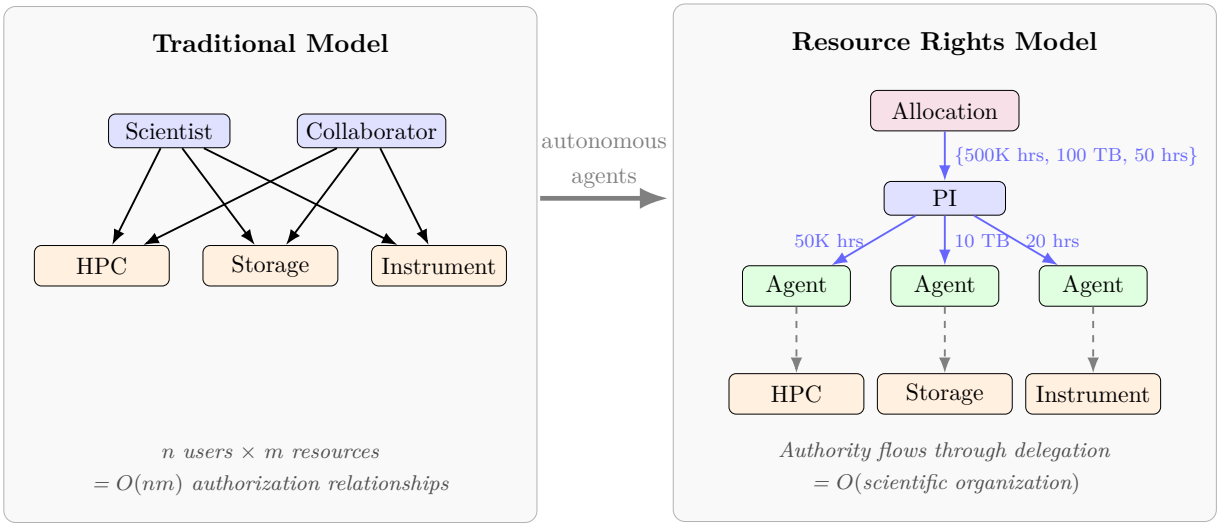
We argue that the underlying limitation is conceptual rather than technological. Current authorization systems attach permissions to identities through resource-specific access control mechanisms, making delegation, composition, and large-scale management of authority increasingly difficult as the numbers of agents, resources, and collaborations grow. We propose instead a new abstraction—*resource rights*—that treats authority as an explicit, measurable, delegable, and revocable object independent of the identity currently exercising it. Resource rights represent the authority to consume or manipulate scientific resources, including computational allocations, storage capacity, instrument time, repository access, workflow execution, and AI services.

Building on this abstraction, we present the *Resource Rights Architecture*, a federated authorization model in which allocation authorities issue resource rights, scientists delegate bounded subsets of those rights to autonomous agents, and resource providers authorize requests by validating presented rights against local policy. The resulting architecture separates identity, authority, policy, and accounting into independent concerns while supporting fine-grained delegation, dynamic revocation, comprehensive auditing, and scalable federation across independently administered scientific facilities.

We show how this architecture naturally supports autonomous scientific workflows while remaining compatible with existing identity and authorization technologies. Although motivated by the requirements of large-scale AI-enabled scientific cyberinfrastructure, the proposed model provides a general architectural foundation for authorization in distributed scientific ecosystems and offers a path toward scalable governance of autonomous scientific agents.

Contents

1 Introduction 3
2 The Rise of Autonomous Science 5
3 The Scaling Limits of Existing Authorization Models 7
4 The Resource Rights Model 8
5 A Resource Rights Architecture 12
6 Runtime Authorization 15
7 Example: An End-to-End Autonomous Scientific Campaign 19
8 Implementing Resource Rights 22
9 Properties of the Resource Rights Model 23
10 Related Work 26
11 Future Directions 29
12 Conclusion 32
Appendix A Threat Model and Security Analysis 37
Appendix B Implementation Sketch: HPC Facility 40



The authorization challenge for autonomous science. ***Left:*** *Traditional models grant users direct access to resources; authorization relationships scale as users times resources, and users exercise authority without explicit bounds.* ***Right:*** *The resource rights model enables authority to flow through delegation. Allocation authorities issue rights to PIs, who delegate bounded subsets to autonomous agents. Each agent receives only the authority it needs, with explicit limits on capacity, scope, and lifetime. Complexity scales with scientific organization rather than with the number of agents.*

1 Introduction

Scientific discovery is entering a period in which autonomous computational agents will become active participants in the research process rather than passive tools used by individual scientists. Large language models, planning algorithms, workflow systems, and specialized AI services are already capable of generating hypotheses, designing simulations, constructing workflows, analyzing experimental results, controlling laboratory equipment, and coordinating complex computational campaigns. As these capabilities mature, scientific investigations will increasingly be conducted by teams of humans and AI agents operating together across geographically distributed computing facilities, data repositories, instruments, laboratories, and cloud services.

This transformation has profound implications for scientific cyberinfrastructure. Existing research infrastructures were designed around a simple assumption: a human user authenticates to a resource and exercises authority directly. Authentication establishes identity, authorization determines whether that identity is permitted to perform a requested action, and accounting records resource consumption. This model has served high-performance computing, scientific data repositories, and research facilities well for several decades because the number of human users has remained manageable and authority has largely been exercised directly by those users.

Autonomous scientific agents fundamentally change this model. A single principal investigator may soon supervise dozens or hundreds of specialized agents, each responsible for different aspects of an investigation. One agent may search the literature, another construct machine learning models, another perform large-scale simulations, another schedule beamline experiments, and another coordinate laboratory robotics. Each agent must independently acquire data, submit jobs, invoke remote services, reserve instruments, update repositories, and communicate with other agents. Over the lifetime of a large scientific campaign, millions of authorization decisions may be made on behalf of a single scientist.

Current authorization mechanisms do not scale naturally to this environment. Access control lists, role-based authorization, and group memberships all assume that administrators assign permissions directly to users or service identities on individual resources. As the number of agents, resources, collaborations, and institutions increases, the number of authorization relationships grows combinatorially. Every new project requires administrators to provision accounts, configure permissions, synchronize policies across independent facilities, and eventually revoke those permissions when projects conclude. These administrative processes are already a significant burden in today's cyberinfrastructure; they become untenable when each scientist routinely employs large populations of autonomous agents.

The problem is not merely one of scale. Autonomous agents require authority that is fundamentally different from that granted to their human supervisors. Scientists rarely wish to give an agent unrestricted access to every resource available to them. Instead, they wish to delegate carefully bounded authority: a fixed allocation of GPU hours, permission to execute approved workflows,

read access to particular datasets, authority to operate specific laboratory instruments, or the ability to spend a limited computational budget before requesting further approval. Existing authorization systems can approximate such constraints through combinations of accounts, groups, quotas, and policy engines, but these mechanisms were developed independently, differ across resources, and provide no common conceptual framework for delegation.

This paper argues that these difficulties arise because current cyberinfrastructure lacks an appropriate abstraction for authority. Distributed systems evolved dramatically when files, virtual memory, remote procedure calls, objects, containers, and workflows became explicit abstractions with well-defined semantics. We argue that autonomous science requires a similar conceptual advance. Rather than viewing authorization as permissions attached to identities, we propose treating authority as a collection of *resource rights*: explicit, measurable, delegable, revocable authorizations to consume or manipulate scientific resources.

A resource right represents authority independently of the identity that currently exercises it. Examples include allocations of GPU-hours, storage capacity, beam time, laboratory operations, repository access, AI inference credits, workflow execution privileges, and software licenses. Resource rights may be quantified, partitioned, delegated to other principals or autonomous agents, constrained by policy, audited during execution, and revoked when circumstances change. Scientific resources thus become associated with corresponding rights that may be managed throughout their lifecycle in much the same way that physical resources are allocated and consumed.

Viewing authorization through the lens of resource rights produces a fundamentally different architecture for scientific cyberinfrastructure. Instead of maintaining large access control lists describing which users may access which resources, resource providers validate rights presented by requesting principals. Allocation authorities create rights, project leaders partition them among research activities, scientists delegate bounded subsets to autonomous agents, and resource providers verify those delegations before authorizing execution. Administrative responsibility is distributed naturally along existing scientific management structures while individual resource providers remain responsible only for enforcing local policy and validating presented rights.

The proposed architecture separates four concerns that are often conflated in existing authorization systems: identity, authority, policy, and accounting. Identity establishes who or what is making a request. Resource rights describe the authority that has been delegated to that principal. Local resource policy determines whether the requested operation is permissible under current operational constraints. Accounting records the consumption of delegated rights and provides the provenance necessary for reproducibility, reporting, and audit. This separation simplifies both implementation and reasoning while allowing each concern to evolve independently.

Although motivated by the needs of autonomous science, the proposed model applies broadly across modern scientific cyberinfrastructure. High-performance computing centers, cloud platforms, autonomous laboratories, telescope networks, synchrotron facilities, scientific repositories, and AI

services all allocate scarce resources to projects and collaborations. Each already manages implicit resource rights through allocations, quotas, reservations, or licenses. Our contribution is not to invent new forms of authority but to unify these diverse mechanisms within a common architectural framework that supports delegation, composition, accounting, and federation.

Large-scale AI-driven scientific initiatives provide an especially compelling motivating example. Emerging visions of autonomous scientific discovery require thousands of autonomous agents operating across a federation of national laboratories, supercomputing centers, cloud providers, data repositories, experimental facilities, and autonomous laboratories. Such an ecosystem cannot rely on manually administered access-control relationships among millions of principals and thousands of resources. Instead, it requires an authorization architecture whose complexity grows with scientific organization rather than with the Cartesian product of users and resources.

The remainder of this paper is organized as follows. Section 3 characterizes the scaling limitations of existing authorization models for autonomous scientific systems. Section 4 introduces the resource rights abstraction and defines its semantics. Section 5 presents a delegation architecture for scalable management of authority across federated organizations. Section 6 describes runtime authorization, accounting, and auditing. Section 7 illustrates the model through a representative scientific workflow. Section 8 discusses implementation strategies. Section 9 identifies the safety and scalability properties that emerge from the model. Section 10 positions resource rights relative to prior work in distributed authorization, capability systems, and scientific cyberinfrastructure. Section 11 outlines directions for future research, while Section 12 concludes with implications for autonomous scientific infrastructure.

2 The Rise of Autonomous Science

Scientific computing has historically followed a remarkably stable operational model. A scientist develops a computational experiment, authenticates to one or more computing facilities, submits jobs, retrieves results, and iteratively refines the investigation based on those results. Although workflows have become increasingly sophisticated, humans have remained the primary actors responsible for initiating computational activity and making scientific decisions. Cyberinfrastructure has therefore evolved around the assumption that authenticated users are the fundamental units of authority.

Recent advances in artificial intelligence challenge this assumption. Large language models coupled with planning algorithms, workflow engines, domain-specific reasoning systems, and scientific software are rapidly acquiring the ability to perform substantial portions of the scientific process with limited human intervention. Rather than merely answering questions or generating code, these systems increasingly formulate hypotheses, design computational experiments, orchestrate simulation campaigns, analyze observational and experimental data, identify inconsistencies, generate new experimental plans, and adapt future actions based on previous results.

The consequence is a transition from *human-directed computation* to *agent-mediated science*. In this

emerging model, scientists increasingly define objectives, constraints, and evaluation criteria, while autonomous computational agents determine how those objectives are achieved. Human expertise remains essential, but shifts from performing individual computational tasks toward supervising, validating, and directing populations of intelligent agents.

This transition fundamentally changes the scale at which scientific cyberinfrastructure must operate. A principal investigator who today submits tens of computational jobs during a week-long investigation may soon supervise dozens of specialized agents collectively submitting thousands of simulations, querying hundreds of repositories, coordinating laboratory equipment, invoking machine learning services, and communicating continuously with one another. Scientific campaigns become persistent distributed systems rather than collections of isolated user interactions.

Importantly, these agents are not homogeneous. Different agents perform different scientific functions and therefore require different forms of authority. A literature analysis agent requires access to publications and repositories but no authority to control laboratory instruments. A simulation agent requires access to supercomputing resources but no permission to modify experimental protocols. A laboratory execution agent may control robotics systems while possessing no authority to allocate additional computational resources. Treating every agent as a surrogate for its human creator violates the principle of least privilege and substantially increases operational risk.

Furthermore, scientific investigations increasingly span organizational boundaries. A single autonomous campaign may consume resources provided by multiple national laboratories, university computing centers, commercial cloud providers, scientific repositories, experimental facilities, and autonomous laboratories. Each organization retains responsibility for governing its own resources while participating in larger scientific collaborations. Authorization therefore becomes a federated systems problem rather than a local administrative function.

These developments suggest that authority itself must become a managed scientific resource. Just as computational campaigns consume processor time, storage capacity, network bandwidth, instrument time, and experimental materials, autonomous agents consume delegated authority while carrying out scientific work. That authority must be allocated, constrained, transferred, audited, renewed, and eventually revoked. Existing cyberinfrastructure already performs many of these functions implicitly through project allocations, quotas, reservations, and access policies, but does so using independent mechanisms that lack a common conceptual foundation.

The central observation motivating this paper is therefore straightforward: autonomous science changes not only how scientific computation is performed but also how authority must be represented and managed. Once authority becomes something that is routinely delegated among humans and autonomous agents, rather than exercised directly by authenticated users, existing authorization models cease to provide an adequate abstraction. The remainder of this paper develops an alternative model based on explicit, delegable resource rights that scales naturally to this new mode of scientific discovery.

3 The Scaling Limits of Existing Authorization Models

Modern scientific cyberinfrastructure employs a variety of mature authorization mechanisms, including access control lists (ACLs), role-based access control (RBAC), attribute-based access control (ABAC), capability systems, project allocations, quotas, and token-based authorization frameworks. These technologies have enabled secure operation of high-performance computing facilities, scientific repositories, cloud platforms, and experimental laboratories for many years. The objective of this paper is not to replace these mechanisms. Rather, we argue that they lack a common abstraction for representing and managing authority in the emerging setting of autonomous scientific discovery.

The common characteristic of existing authorization systems is that they associate permissions, directly or indirectly, with authenticated identities. A resource provider evaluates an incoming request by first establishing the identity of the requester and then determining whether that identity satisfies the local authorization policy. The policy may be represented by an ACL, a role assignment, a collection of attributes, or a capability token, but the underlying computational model remains similar: authorization is fundamentally an identity-centric operation.

This model is entirely appropriate when humans are the primary actors. Individual scientists authenticate to relatively small numbers of resources, permissions change infrequently, and administrative effort remains manageable. Resource providers maintain local policies while institutional identity providers establish user identities. The resulting architecture has proved remarkably successful for federated scientific computing over the past several decades.

Autonomous science introduces a qualitatively different workload. Instead of a relatively small population of human users directly interacting with resources, scientific investigations increasingly involve large populations of autonomous agents acting on behalf of those users. Each agent performs different scientific functions, possesses different operational constraints, and requires different subsets of the authority held by its supervising scientist. Furthermore, agent populations are dynamic: new agents are created as investigations evolve, existing agents are retired, and responsibilities are continually redistributed among cooperating computational processes.

Under an identity-centric authorization model, every new agent potentially introduces additional authorization relationships that must be established, maintained, synchronized across independently administered facilities, and eventually revoked. If a scientist supervises A autonomous agents interacting with R independent resource providers, the administrative state associated with authorization grows approximately with the number of agent-resource relationships rather than with the number of scientists. As both A and R increase, this state rapidly dominates the management effort required to operate the underlying cyberinfrastructure.

The challenge is amplified by the federated nature of modern scientific collaborations. Large scientific campaigns routinely involve multiple institutions, each maintaining independent administrative domains, security policies, accounting systems, and operational practices. Authorization decisions

therefore cannot rely on centralized administrative control. Instead, authority must propagate across organizational boundaries while preserving local autonomy and institutional governance. Existing federated identity infrastructures address the problem of establishing identity across organizations, but they provide comparatively limited support for representing the dynamic delegation of bounded authority among humans, projects, and autonomous agents.

Least-privilege operation presents an additional challenge. Scientists rarely wish to grant an autonomous agent unrestricted access to every resource available to them. Instead, authority is naturally scoped according to scientific function. A simulation agent may require authority to consume a limited allocation of GPU-hours while possessing no authority to modify experimental data. A laboratory agent may control robotic equipment while lacking permission to allocate additional computational resources. A literature analysis agent may require repository access but no authority to execute workflows. Constructing these differentiated authority profiles using conventional administrative mechanisms is possible but increasingly cumbersome as the number of agents grows.

Another limitation concerns resource accounting. Scientific facilities already allocate scarce resources such as processor time, storage capacity, instrument time, and cloud credits through project allocation processes. These allocations implicitly define rights to consume resources, yet they are generally represented independently from authorization policies. Authorization determines whether an operation is permitted, while accounting determines whether sufficient allocation remains. The separation of these mechanisms obscures the fact that both are concerned with different aspects of the same underlying concept: the authority to consume a finite scientific resource.

The central observation emerging from these considerations is that existing authorization mechanisms manage permissions, identities, and allocations as distinct administrative constructs even though they jointly describe a single operational reality. Scientists possess authority to consume scientific resources, delegate portions of that authority to collaborators and computational processes, and remain accountable for its use. Autonomous scientific systems merely expose this implicit structure at much larger scale.

The difficulty therefore lies not primarily in existing technologies but in the abstraction they implement. Identity remains essential for authentication, local policy remains essential for governance, and existing authorization protocols remain valuable implementation mechanisms. What is missing is an explicit representation of scientific authority that can be measured, partitioned, delegated, transferred where appropriate, constrained by policy, audited, and revoked independently of the identities that temporarily exercise it. The following section introduces such an abstraction.

4 The Resource Rights Model

The principal contribution of this paper is the introduction of the *resource right* as a first-class abstraction for representing authority within scientific cyberinfrastructure. Existing authorization

systems typically embed authority within access control policies, project allocations, role assignments, quotas, or capability tokens. Collectively, these mechanisms determine whether a requested operation may proceed, but they do not provide an explicit representation of authority that can be reasoned about independently of a particular resource or authorization mechanism.

We instead regard authority itself as a managed object. A resource right represents the authority to perform specified operations on a scientific resource, subject to quantitative limits and policy constraints. Like data objects or workflows, resource rights have an independent existence: they may be created, delegated, partitioned, constrained, audited, and revoked without modifying the underlying resource or the identity of the principal currently exercising them. The resulting abstraction separates the question of *who* is making a request from the question of *what authority* has been delegated to that requester.

The model begins with the notion of a resource. A resource is any entity whose use is governed by policy. Resources may be physical, such as laboratory instruments or robotic platforms; computational, such as GPU clusters, workflow engines, or storage systems; informational, such as scientific repositories; or logical, such as software licenses or AI inference services. Although these resources differ substantially in their implementation, they share an important common property: each exports operations whose execution must be governed according to institutional policy. The proposed model intentionally makes no distinction among these categories, treating them all as instances of the same underlying concept.

A resource right captures authority over such a resource. Conceptually, it specifies the resource to which it applies, the operations that are authorized, the amount of authority that has been delegated, the conditions under which that authority may be exercised, the period during which it remains valid, the authority from which it originated, and sufficient provenance to establish its chain of delegation. Practical implementations may represent these fields using signed certificates, capability tokens, JSON Web Tokens (JWTs) [16], or other cryptographically protected structures, but the semantics of a resource right are independent of its concrete representation.

Resource rights may represent either quantitative or qualitative authority. Quantitative rights describe consumable resources whose use may be measured, such as GPU-hours, storage capacity, instrument time, workflow executions, or API invocations. Qualitative rights describe authority that is naturally expressed as permission to perform an operation, such as reading a repository, operating a microscope, modifying metadata, or approving publication of a dataset. Although these forms of authority appear different operationally, they share the same semantic structure: each represents bounded authority over a governed resource and may therefore be manipulated using the same conceptual framework.

The defining characteristic of a resource right is that it is delegable. Delegation creates a new resource right whose authority is derived from an existing one while remaining subject to the constraints imposed by its issuer. A principal holding an allocation of one hundred thousand GPU-hours, for

example, may partition that allocation among several autonomous agents, each receiving only the authority necessary for its assigned task. Similarly, authority to operate a laboratory instrument may be delegated to a workflow responsible for a particular experimental campaign without granting that workflow unrestricted access to the remainder of the facility. Delegation therefore distributes authority according to scientific responsibility rather than according to administrative convenience.

Delegation also preserves accountability. Every delegated right carries with it the provenance of the authority from which it was derived, allowing resource providers to reconstruct the chain of responsibility associated with any authorized action. This provenance is essential for scientific audit, reproducibility, and governance. A resource provider need not understand the internal organizational structure of a project; it need only verify that the presented right forms a valid chain originating from an authority recognized by local policy.

An important property of the model is that delegation is monotonic. A delegated right may restrict the authority contained in its parent by reducing available capacity, shortening its lifetime, or imposing additional policy constraints. It may not, however, increase the authority that it inherits. Consequently, authority cannot be amplified through repeated delegation, regardless of the depth or complexity of the delegation hierarchy. This property substantially simplifies reasoning about security while permitting authority to be distributed dynamically among cooperating humans and autonomous agents.

For quantitative resources, delegation additionally satisfies a conservation principle. The total capacity delegated from a resource right cannot exceed the capacity possessed by the delegating principal. A scientist allocated one hundred thousand GPU-hours may distribute that allocation among collaborators or autonomous agents in any manner, but the aggregate delegated allocation cannot exceed the original authority. Resource accounting therefore emerges naturally from the structure of delegated rights rather than from an independent administrative mechanism.

The distinction between identity and authority is fundamental to the proposed model. Authentication establishes the identity of the requesting principal. Resource rights establish the authority that has been delegated to that principal. Local policy determines whether the requested operation is permissible under current operational conditions. By treating these three concerns as logically independent, the model permits each to evolve separately while preserving clear semantics for authorization decisions. The result is an authorization framework whose complexity reflects the scientific organization of authority rather than the administrative configuration of individual resources.

4.1 Conceptual Structure

The preceding paragraphs describe resource rights informally. Before proceeding to formal definitions, we summarize the key conceptual elements. A resource right binds together six concerns: the *resource* over which authority is granted, the *operations* that may be performed, the *quantity* of

authority available, the *constraints* that must be satisfied, the *validity period* during which the right may be exercised, and the *provenance* linking the right to its issuing authority. These elements are independent: the same resource may be governed by rights with different operations, quantities, or constraints; the same constraints may apply to rights over different resources; and the same provenance chain may underlie rights with different validity periods.

This structure supports several important use cases. Quantitative rights enable computational allocations to be partitioned among agents without over-commitment. Policy constraints allow institutional requirements—such as data use agreements, safety classifications, or access restrictions—to propagate through delegation chains. Validity periods support both short-lived tokens for individual operations and long-lived allocations spanning entire scientific campaigns. Provenance enables audit and accountability without requiring resource providers to understand the organizational structure of every collaboration.

4.2 Formal Foundations

The preceding discussion may be made precise as follows. Let $\mathcal{R}$ denote a set of *resources*, $\mathcal{O}$ a set of *operations*, and $\mathcal{P}$ a set of *principals* (humans, organizations, or autonomous agents). A *resource right* is a tuple

$$r = (R, O, q, C, \tau, \pi)$$

where $R \in \mathcal{R}$ is the governed resource, $O \subseteq \mathcal{O}$ is the set of authorized operations, $q \in \mathbb{R}_{\geq 0} \cup \{\infty\}$ is the quantitative capacity (with $q = \infty$ denoting qualitative authority), C is a set of policy constraints, τ is a validity interval, and π is a provenance chain linking the right to its issuing authority.

The lifecycle of a resource right is governed by six operations:

- $\mathsf{Issue}(R, O, q, C, \tau) \to r$: An allocation authority creates a new right with specified capacity and constraints.
- $\mathsf{Delegate}(r, p, R', O', q', C', \tau') \to r'$: A principal holding $r = (R, O, q, C, \tau, \pi)$ creates a derived right $r' = (R', O', q', C', \tau', \pi')$ for principal p. The delegation must satisfy $R' \subseteq R$ (resources may only be narrowed), $O' \subseteq O$ (operations may only be restricted), $q' \leq q$ (capacity may only decrease), $C' \supseteq C$ (constraints may only be strengthened), and $\tau' \subseteq \tau$ (validity may only be shortened). The provenance chain π' extends π with a record of this delegation.
- $\mathsf{Restrict}(r, C'') \to r'$: Adds constraints C'' to produce a more restricted right.
- $\mathsf{Consume}(r, q'') \to r'$: For quantitative rights, reduces available capacity by q'', producing r' with capacity $q - q''$.
- $\mathsf{Expire}(r)$: The right becomes invalid when the current time exits τ.
- $\mathsf{Revoke}(r)$: The issuing authority invalidates r and all rights derived from it.

These operations satisfy three invariants that constitute the fundamental laws of the model:

Proposition 1 (Non-amplification)**.** *Let $r = (R, O, q, C, \tau, \pi)$ be a resource right. For any delegation* Delegate$(r, p, R', O', q', C', \tau') \to r'$ *where $r' = (R', O', q', C', \tau', \pi')$, the derived right must satisfy: $R' \subseteq R$ (resources may only be narrowed), $O' \subseteq O$ (operations may only be restricted), $q' \leq q$ (capacity may only decrease), $C' \supseteq C$ (constraints may only be added), and $\tau' \subseteq \tau$ (validity may only be shortened). Authority cannot increase through delegation.*

Proposition 2 (Conservation)**.** *Let $r = (R, O, q, C, \tau, \pi)$ be a quantitative resource right with capacity q. If a principal delegates rights $r_1, \ldots, r_n$ from r, where each r_i has capacity q_i, then $\sum_{i=1}^{n} q_i \leq q$. The total delegated capacity cannot exceed the available capacity. This invariant ensures that resource consumption remains bounded by the original allocation.*

Proposition 3 (Provenance preservation)**.** *Every right r' produced by* Delegate *or* Restrict *contains a provenance chain π' that extends the parent's chain π, such that the origin of authority is recoverable by traversing π' to a recognized allocation authority. Authorization requires validating this chain against locally recognized allocation authorities.*

These invariants ensure that the delegation graph forms a directed acyclic structure rooted at allocation authorities, with authority flowing monotonically from issuers through principals to autonomous agents. The security of the model rests on this structure: no sequence of operations can produce authority exceeding what was originally issued, and every exercised right is traceable to a recognized source. Figure 4 summarizes the lifecycle operations and the laws they satisfy.

The resource rights model is intentionally independent of any particular authorization technology. Existing systems for authentication, federation, token management, and policy evaluation remain essential components of scientific cyberinfrastructure and may be used to implement resource rights in different environments. The contribution of this paper is not a replacement for these mechanisms but a higher-level abstraction that provides a common conceptual foundation for representing, delegating, and governing authority in an era of autonomous scientific discovery.

5 A Resource Rights Architecture

The resource rights model naturally induces a distributed authorization architecture whose primary objective is to separate the management of scientific authority from the operation of scientific resources. Rather than requiring resource providers to maintain detailed knowledge of every user, project, collaboration, and autonomous agent that may access their services, authority is represented explicitly by resource rights that accompany each request. Resource providers are therefore responsible for enforcing local policy, while the creation and distribution of authority is managed elsewhere.

This separation mirrors the organization of scientific collaborations. Funding agencies allocate resources to projects, projects distribute those resources among principal investigators, investigators

assign portions of their allocations to collaborators, and collaborators increasingly delegate bounded authority to autonomous computational agents. Existing cyberinfrastructure already embodies this hierarchy administratively, but does so using independent allocation systems, local access control mechanisms, project databases, and manually administered accounts. The resource rights architecture unifies these processes by representing every stage of authority distribution using the same abstraction.

The architecture consists of four logical roles: allocation authorities, principals, resource providers, and autonomous agents. These roles describe responsibilities rather than software components, and multiple roles may be fulfilled by the same organization or service.

Allocation authorities create resource rights. They represent the origin of authority for particular classes of scientific resources and are responsible for determining the total quantity of authority available to a project or organization. An allocation authority might represent a funding agency assigning computational resources to a scientific project, a laboratory allocating instrument time, or a repository granting access to a protected data collection. Once issued, resource rights may be delegated without further involvement of the issuing authority, provided that all delegations remain consistent with the constraints encoded in the original right. Constraints propagate through delegation: if an allocation authority issues a right with a constraint restricting access to certain personnel categories (e.g., due to export control regulations), that constraint must appear in every derived right. The resource provider enforces constraints at runtime by checking both the presented right and the identity of the requester against the constraint set.

Principals receive resource rights and redistribute them according to scientific responsibility. A principal may represent an individual scientist, a research group, an institution, or another organizational entity. Importantly, principals are not merely consumers of authority; they also act as local allocation authorities within their own administrative domains. A principal investigator who receives an allocation of computational resources, for example, may partition that allocation among several research activities, each with distinct objectives, budgets, and operational constraints. The same mechanism may then be used to delegate bounded authority to the autonomous agents responsible for executing those activities.

Autonomous agents exercise delegated authority but do not possess inherent authority of their own. Every operation performed by an agent is therefore traceable to an explicit delegation originating from one or more human or organizational authorities. This property preserves accountability while allowing agents to operate independently over extended periods. It also naturally supports the principle of least privilege, since each agent receives only the authority required to perform its assigned scientific function.

Resource providers remain responsible for governing their own resources. Their role is not to understand the organizational structure of every collaborating project, but rather to determine whether a presented resource right authorizes a requested operation under local policy. Authorization

therefore becomes a process of validating delegated authority rather than consulting resource-specific administrative state. Resource providers retain complete autonomy over local operational policies while avoiding the need to maintain detailed authorization relationships for every potential user and agent.

Taken together, these roles define a distributed delegation graph that describes the current allocation of scientific authority. Vertices correspond to principals holding resource rights, while directed edges represent delegations from one principal to another. The graph evolves continuously as projects create new agents, redistribute computational budgets, revoke obsolete authority, or conclude scientific campaigns. Unlike conventional access control structures, however, the graph represents authority itself rather than administrative configuration. It therefore reflects the scientific organization of work rather than the implementation details of individual facilities.

Authorization becomes the evaluation of a simple question: does the requesting principal possess a valid resource right authorizing the requested operation on the target resource? Answering this question requires three logically independent steps. The identity of the requester must first be established through conventional authentication mechanisms. The presented resource rights must then be validated to ensure that they form a legitimate chain of delegation and remain within their stated quantitative and policy constraints. Finally, the resource provider evaluates the requested operation against local policy, taking into account operational considerations such as current availability, maintenance status, safety requirements, or institutional regulations. Only if all three conditions are satisfied is the operation authorized.

Figure 1 illustrates the separation between the authority path (left) and the decision path (right). Authority flows downward through delegation; authorization decisions combine identity, presented rights, and local policy. Figure 2 shows how the delegation graph distributes authority from allocation authorities through projects and investigators to autonomous agents, with resource providers validating presented rights without requiring global knowledge of the collaboration structure.

An important consequence of this architecture is that its complexity grows with the scientific organization of authority rather than with the number of participating resources. Each allocation authority manages only the authority that it issues. Each principal manages only the authority that it delegates. Each resource provider evaluates only the requests that it receives. No component requires global knowledge of all users, all resources, or all collaborations. As autonomous science continues to increase both the number of participating agents and the diversity of scientific resources, this separation of responsibilities becomes essential for maintaining manageable operational complexity.

The architecture described here is intentionally abstract. It specifies the flow of authority without prescribing particular protocols, token formats, trust frameworks, or implementation technologies. Section 6 describes how authorization decisions are made at runtime, Section 8 discusses how the model may be realized using existing technologies, and Appendix Appendix A analyzes the security

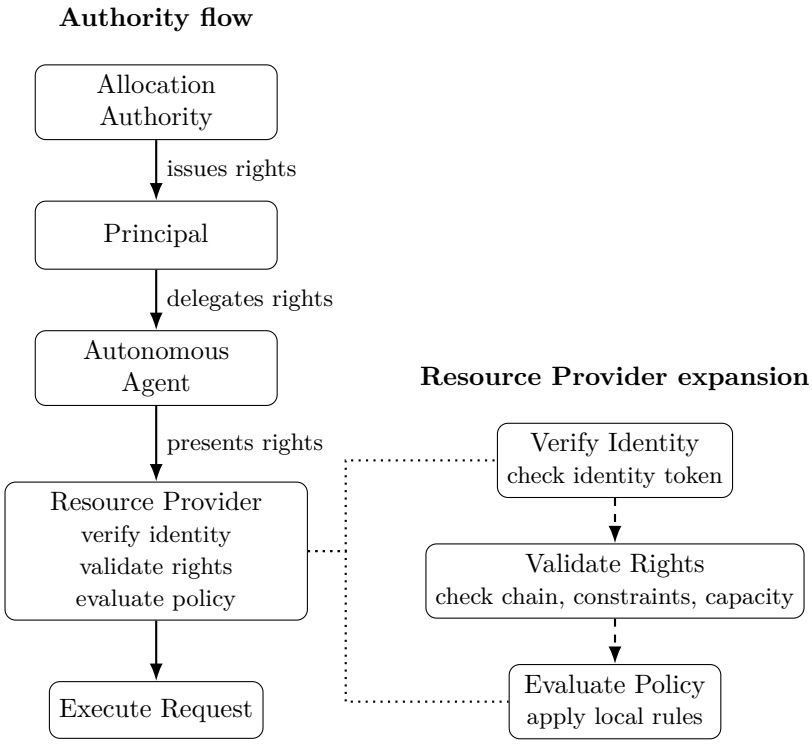


Figure 1: Conceptual architecture. Resource rights carry authority from allocation authorities through principals to agents. The resource provider (expanded at right) verifies the requester's identity, validates the presented rights, and evaluates local policy before permitting execution.

properties that result. By first identifying the architectural roles induced by the resource rights model, we obtain a framework that can be realized using a variety of existing authentication and authorization technologies while remaining independent of any particular implementation.

6 Runtime Authorization

A resource right is useful only if it can be evaluated efficiently at the moment a scientific operation is requested. The runtime authorization problem is therefore to determine whether a requesting principal may perform a specific operation on a specific resource under current conditions. In the resource rights architecture, this decision is not made by consulting a resource-local list of users and roles. It is made by validating the authority presented with the request and then applying local resource policy.

The runtime process has three distinct stages. First, the requester authenticates and establishes an identity. Second, the requester presents one or more resource rights that purport to authorize the requested operation. Third, the resource provider evaluates those rights against local policy and current resource state. These stages are logically independent. Authentication establishes who is making the request; the presented rights establish what authority has been delegated to that principal; local policy determines whether the requested operation may proceed at this time.

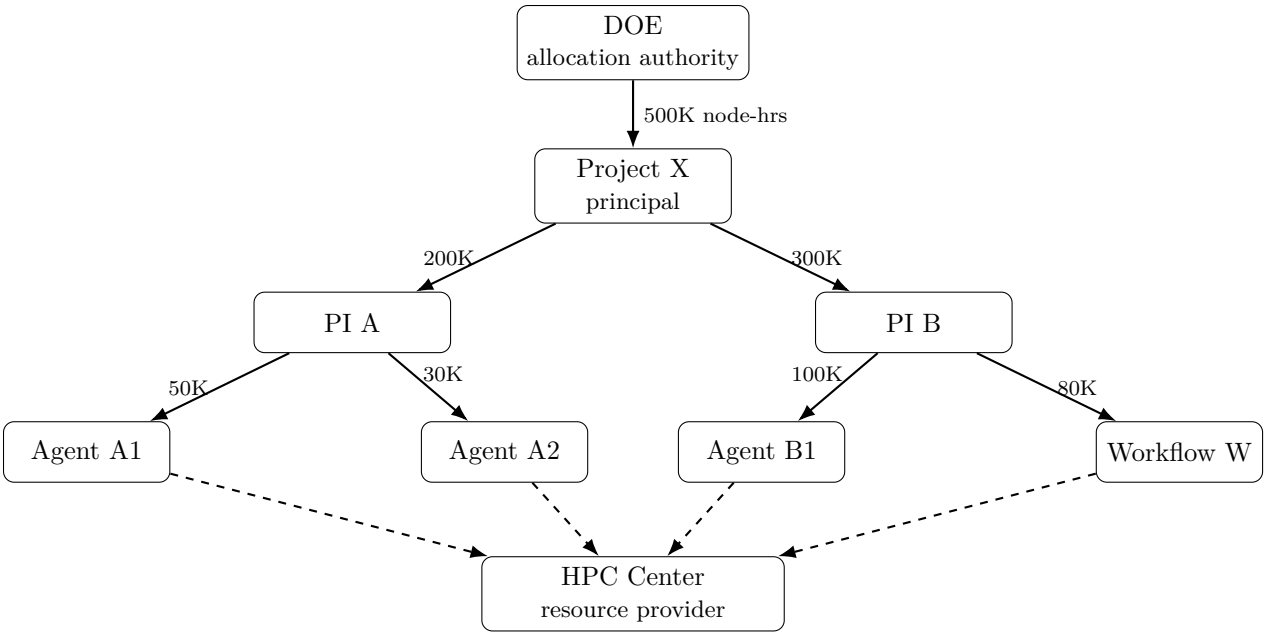


Figure 2: A delegation graph. Authority originates at an allocation authority and is progressively delegated through projects, investigators, agents, and workflows. Resource providers validate presented rights without needing global knowledge of the graph.

Figure 3 illustrates the interaction. An autonomous agent first obtains an identity token from an identity provider. It then presents that token, together with its delegated resource rights, to a resource provider. The resource provider verifies the identity of the requester, validates the delegation chain associated with the presented rights, checks that the requested operation lies within the scope of those rights, verifies that sufficient quantitative capacity remains where relevant, and finally evaluates local operational policy. Only after these checks succeed is the request dispatched to the underlying scheduler, repository, instrument controller, workflow engine, or service interface.

This structure deliberately avoids user impersonation. An agent does not act by borrowing the full authority of a scientist. It acts under its own identity while presenting specific delegated rights. This distinction is important for both security and accountability. If an agent is compromised, the exposed authority is limited to the rights that were delegated to that agent. If an operation later requires audit, the resource provider can identify both the immediate actor and the chain of authority under which that actor operated.

Runtime validation also provides the point at which resource-specific governance is preserved. A valid resource right is necessary but not sufficient for execution. A facility may still deny a request because the resource is unavailable, a maintenance window is active, a safety interlock has been triggered, a regulatory restriction applies, or the requested operation violates local policy. The resource rights model therefore does not displace institutional authority. It provides a common way to present delegated authority while preserving the right of each provider to make final execution decisions.

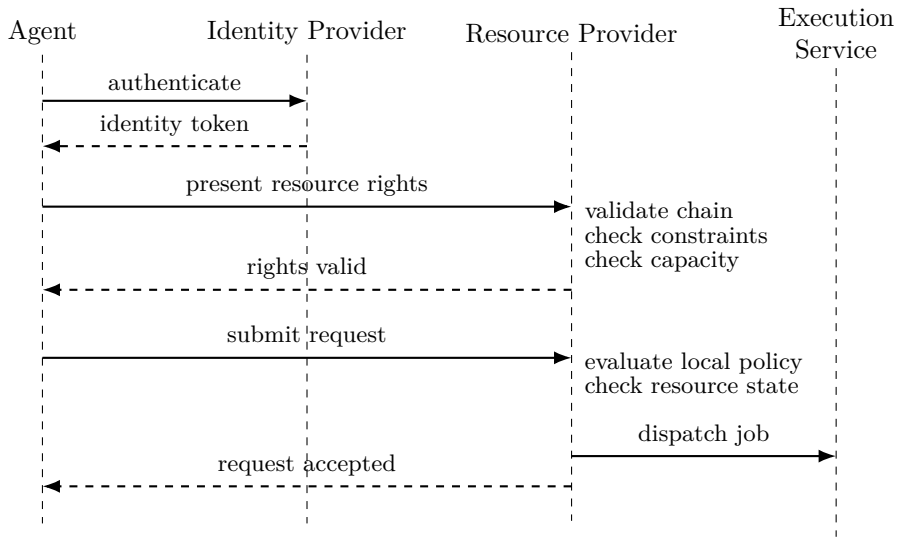


Figure 3: Runtime interaction. An agent authenticates, presents delegated rights, and submits a request. The resource provider validates the rights and local policy before dispatching to the underlying execution service (e.g., job scheduler, instrument controller, or workflow engine).

For quantitative resources, runtime authorization is coupled to consumption. A successful operation may debit GPU-hours, storage capacity, workflow executions, instrument time, or other measurable capacity from the presented right. This debit becomes part of the accounting record associated with the operation. The same record can include the requester identity, delegation chain, resource identifier, operation, time, consumed capacity, policy version, and outcome. Authorization and accounting are therefore no longer separate administrative processes but two views of a single event: the exercise of delegated authority.

This coupling is particularly important for autonomous agents because their actions may be numerous, rapid, and adaptive. A human user may submit a small number of jobs and monitor their progress manually. An agent may submit thousands of jobs, adjust parameters based on intermediate results, invoke multiple services, and continue operating for days or weeks. Runtime authorization must therefore be automatic, fine-grained, and auditable without requiring human intervention for every operation. Resource rights provide the necessary bounded authority; local policy supplies the operational guardrails.

The accounting infrastructure also enables principals to monitor the status of their delegated rights. A delegation service can provide real-time visibility into how much capacity remains for each right, how consumption is distributed across agents, and whether any delegation is approaching exhaustion. This monitoring supports both operational management—reallocating authority among agents as scientific priorities evolve—and anomaly detection, flagging unexpected consumption patterns that may indicate misconfigured agents or security incidents. Such visibility is essential when a principal supervises many autonomous agents operating across extended time periods.

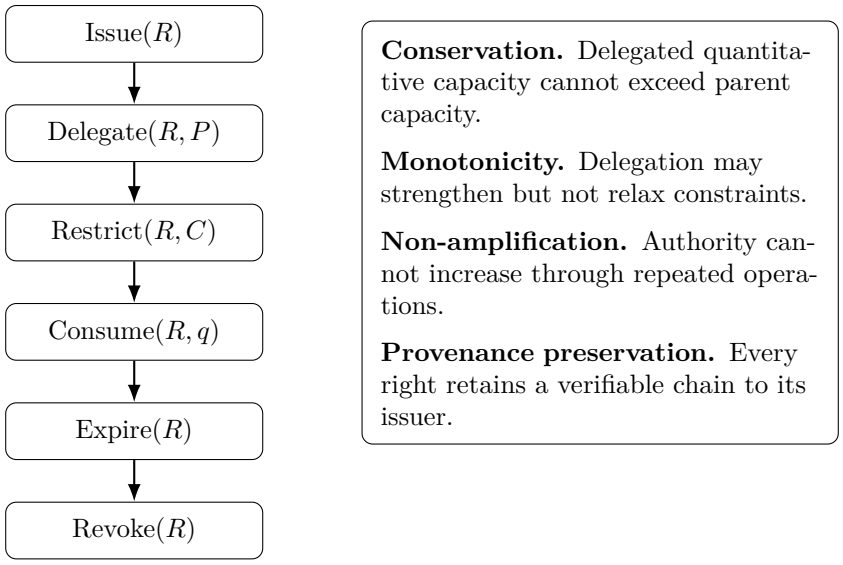


Figure 4: Resource rights algebra. A small set of operations governs the lifecycle of authority, while conservation, monotonicity, non-amplification, and provenance preservation provide safety properties.

The same mechanism supports human approval when required. A delegated right may authorize an agent to prepare a request but require explicit approval before execution. For example, an agent may be permitted to design a chemical synthesis workflow but not to execute it until a scientist confirms that safety constraints have been satisfied. In this case, the approval itself can be represented as an additional right or as a signed policy decision attached to the request. The runtime model therefore accommodates both fully autonomous execution and human-in-the-loop governance without changing the underlying abstraction.

The essential runtime invariant is that execution occurs only when identity, delegated authority, and local policy are simultaneously satisfied. This invariant can be summarized as

$$\text{Execute}(p, o, r) \Longleftarrow \text{Authenticate}(p) \wedge \text{ValidateRights}(p, o, r) \wedge \text{EvaluatePolicy}(o, r, s),$$

where p is the requesting principal, o is the requested operation, r is the target resource, and s is the current resource state. The equation is not intended as an implementation specification; it expresses the separation of concerns that the architecture enforces.

This separation is the central practical advantage of runtime authorization based on resource rights. Identity providers need not understand scientific allocations. Allocation authorities need not operate resources. Resource providers need not maintain global collaboration state. Autonomous agents need not possess broad user credentials. Each component performs a limited function, yet the resulting system supports accountable execution across federated scientific infrastructure.

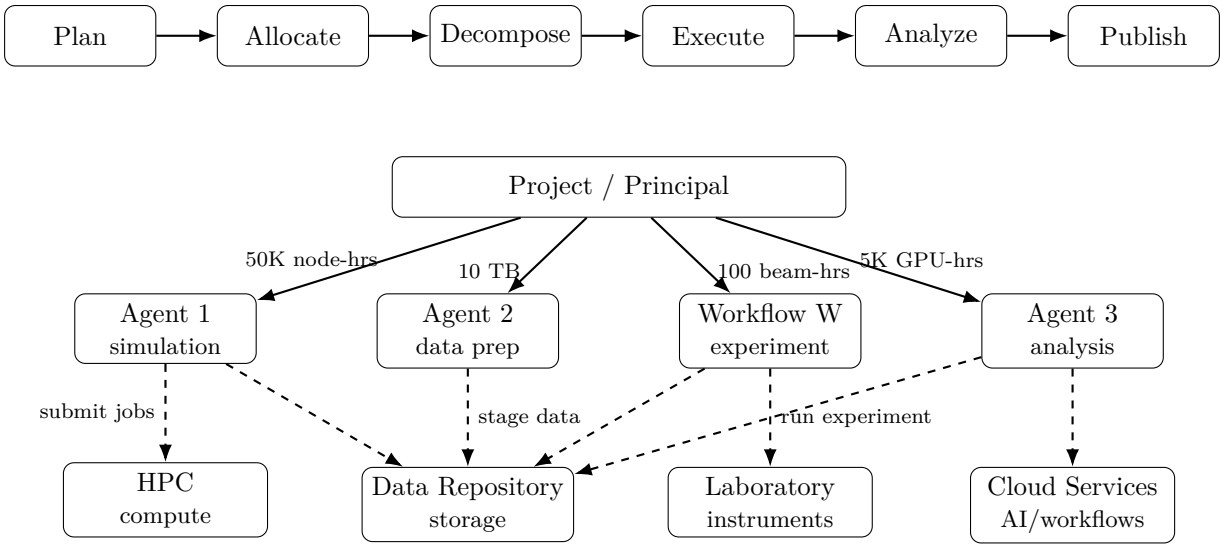


Figure 5: End-to-end scientific workflow. Resource rights support a complete campaign spanning allocation, delegation, execution across heterogeneous resources, analysis, and publication with verifiable provenance.

7 Example: An End-to-End Autonomous Scientific Campaign

To illustrate the operation of the resource rights architecture, consider an autonomous campaign to discover improved battery electrolytes. The campaign combines molecular simulation, machine learning, laboratory experimentation, and data analysis across multiple independent facilities. Although simplified, the example is representative of the distributed scientific workflows increasingly envisioned for large-scale autonomous discovery.

The campaign begins when a funding organization allocates computational, experimental, and data resources to a research project. Rather than creating user accounts or configuring permissions at every participating facility, the allocation authority issues a collection of resource rights describing the authority granted to the project. These rights include allocations of GPU-hours at a supercomputing center, storage capacity within a shared repository, laboratory execution time at an autonomous synthesis facility, and permission to access project data collections. Collectively, they define the computational budget available to the project without specifying how those resources will ultimately be used.

The principal investigator subsequently partitions these rights according to scientific objectives. One portion of the computational allocation is assigned to large-scale molecular dynamics simulations, another to training machine learning models, and a third to uncertainty quantification. Laboratory capacity is reserved for synthesis and characterization of promising candidates, while repository rights are distributed among the workflows responsible for storing intermediate and final results. At this stage no autonomous agents yet exist; authority has merely been organized according to the anticipated structure of the investigation.

Table 1: Comparison with traditional authorization models. The resource rights model aligns authorization state with the scientific organization of authority rather than with resource-local administrative configuration.

Aspect	**ACL/RBAC/Quota Models**	**Resource Rights Model**
What is managed?	Identities, roles, memberships, and quotas per resource	Authority objects that can be delegated, constrained, consumed, and revoked
Where is policy?	Embedded in each resource or access-control system	Carried with delegated rights and enforced against local policy
Scalability driver	Number of users multiplied by number of resources	Number of delegations reflecting scientific organization
Delegation	Indirect and ad hoc through accounts, groups, roles, or project allocations	Native, uniform, auditable, and composable
Auditing and provenance	Resource-specific logs	Delegation history attached to each exercised right
Agent support	Indirect, typically through service accounts or user impersonation	Direct, with bounded authority assigned to each agent
Cross-domain use	Difficult because mappings must be reconciled across domains	Natural because rights are independently verifiable

As the campaign begins, specialized autonomous agents are created to execute different components of the workflow. A simulation agent receives authority to consume a bounded quantity of GPU-hours together with permission to retrieve structural data from the project repository. A laboratory agent receives authority to schedule robotic synthesis workflows but only within predefined safety classifications and daily execution limits. An analysis agent receives authority to read simulation results, update derived datasets, and invoke AI inference services. None of these agents receives unrestricted access to the full project allocation. Each possesses only the authority necessary to perform its assigned scientific function.

During execution the agents operate independently. The simulation agent submits thousands of computational jobs to one or more supercomputing facilities, adapting simulation parameters as new information becomes available. The laboratory agent schedules synthesis and characterization experiments for compounds identified by the simulation campaign. The analysis agent continuously integrates computational and experimental results, updates predictive models, and recommends future directions for exploration. Throughout this process every interaction with a resource provider is accompanied by the presentation of delegated resource rights, allowing each provider to validate the authority under which the request is made.

Importantly, no resource provider requires knowledge of the internal organization of the project.

The supercomputing center need not understand the relationships among investigators, laboratory workflows, or machine learning services. It need only verify that the simulation agent presents a valid delegation authorizing the requested computation. Similarly, the laboratory controller need not understand the computational workflow that produced a candidate compound; it need only determine whether the presented laboratory rights authorize execution of the requested experiment. Each facility therefore remains administratively autonomous while participating in a coordinated scientific campaign.

As the investigation progresses, authority evolves together with scientific priorities. Early exploratory simulations may reveal promising regions of chemical space that justify increased computational investment. Additional GPU-hours can be delegated to simulation agents without modifying authorization policies at participating facilities. Conversely, if an agent begins consuming resources inefficiently or exhibits unexpected behavior, its delegated authority can be reduced or revoked immediately. Because authority is represented explicitly by resource rights rather than embedded within resource-local administrative state, these adjustments require no changes to user accounts, group memberships, or access control lists.

The same mechanism naturally supports human oversight. Consider a stage of the campaign in which an autonomous agent proposes synthesis of compounds requiring hazardous precursors. The laboratory resource right may specify that experiments involving particular chemical classes require explicit approval by a qualified scientist. The agent therefore prepares the experimental protocol, presents supporting evidence, and requests authorization. Approval results in the issuance of an additional constrained resource right permitting execution of the experiment. Human judgment is incorporated into the workflow without interrupting the overall delegation model or requiring exceptional authorization mechanisms.

Every authorized operation contributes to a comprehensive provenance record. The execution history associated with an experimental result includes not only the computational workflow that produced it, but also the chain of delegated authority under which each computation, data access, laboratory operation, and analytical step was performed. Resource accounting, authorization, and scientific provenance become closely aligned, providing a richer foundation for reproducibility, audit, reporting, and governance than is possible when these concerns are managed independently.

Figure 5 illustrates the overall structure: a campaign lifecycle at the top, delegation from project to specialized agents in the middle, and heterogeneous resource providers at the bottom. The dashed lines indicate that each agent interacts only with the resources relevant to its function, presenting appropriate rights at each interaction.

Although this example concerns battery discovery, the same pattern applies across many areas of computational and experimental science. Astronomical observing campaigns, climate model ensembles, genomics pipelines, autonomous materials laboratories, and distributed digital twin simulations all involve the coordinated use of heterogeneous resources by cooperating computational

processes. The particular resources differ, but the underlying problem is the same: authority must be allocated, delegated, exercised, audited, and eventually revoked as scientific investigations evolve. The resource rights architecture provides a common mechanism for managing that lifecycle across diverse scientific domains.

8 Implementing Resource Rights

The resource rights model deliberately separates semantics from implementation. A resource right defines what authority has been delegated and under what conditions that authority may be exercised. It does not prescribe how that authority is represented, transmitted, validated, or revoked. This separation is intentional. Scientific cyberinfrastructure already possesses mature technologies for authentication, federation, token management, policy evaluation, and secure communication. The objective of the proposed architecture is not to replace these technologies, but to provide a common semantic model within which they can operate.

Authentication remains unchanged. Users and autonomous agents continue to authenticate using established institutional identity providers, federated identity systems, or other trusted mechanisms. The responsibility of authentication is simply to establish the identity of the requesting principal. Nothing in the resource rights model requires modification of existing identity infrastructures.

Similarly, existing authorization technologies remain valuable implementation mechanisms. Resource rights may be represented using signed capability tokens, structured security assertions, JSON Web Tokens, macaroons, or other cryptographically protected credentials. The architecture requires only that a resource provider can verify the authenticity and integrity of a presented right, determine its delegation provenance, evaluate any embedded constraints, and establish whether sufficient authority remains to authorize the requested operation. The choice of representation is an engineering decision rather than an architectural one.

The same observation applies to policy evaluation. Resource providers already maintain sophisticated local policies governing operational safety, institutional regulations, scheduling priorities, maintenance windows, and other resource-specific concerns. The resource rights architecture does not attempt to centralize these policies or replace them with a global authorization framework. Instead, presented resource rights become one input to local policy evaluation. A valid right establishes that delegated authority exists; the resource provider remains responsible for determining whether execution is appropriate under current operational conditions.

Resource accounting likewise integrates naturally with existing allocation systems. Most scientific facilities already maintain accounting services that record resource consumption for reporting, quota management, and project administration. The proposed model extends these services by treating each accounting event as the exercise of a specific delegated right. Accounting records therefore acquire an explicit connection to the authority under which consumption occurred, simplifying both audit and scientific provenance.

Revocation presents a more subtle challenge. Since delegated rights may already have been distributed among autonomous agents operating across multiple administrative domains, revocation cannot generally rely upon modification of resource-local authorization state. Instead, implementations may combine finite validity periods, revocation services, signed status records, or online validation mechanisms appropriate to their operational environment. The architecture intentionally accommodates multiple revocation strategies because different scientific facilities have different requirements for latency, availability, and operational independence.

Although the architecture is technology-neutral, it aligns naturally with contemporary federated cyberinfrastructure. Existing identity federations provide authentication. Token-based authorization frameworks provide secure credential transport. Scientific workflow systems provide execution environments for autonomous agents. Resource schedulers, repositories, and laboratory controllers already implement local policy evaluation and accounting. The principal architectural change is therefore not the introduction of fundamentally new infrastructure, but the adoption of a common representation for delegated scientific authority.

This compatibility is an important practical property of the model. Scientific cyberinfrastructure evolves incrementally over long time scales, often spanning decades of investment and software development. Architectures that require wholesale replacement of existing identity systems, schedulers, repositories, or workflow engines face substantial barriers to adoption. By contrast, resource rights can be introduced progressively. Existing allocation systems may begin issuing structured rights while resource providers incrementally adopt validation services. Autonomous agents may initially exercise only a subset of available authority, with additional capabilities introduced as operational experience accumulates. The resulting migration path allows the architecture to coexist with conventional authorization mechanisms during a potentially extended transition period.

Viewed in this way, the resource rights model occupies a layer above existing security technologies. Authentication establishes identity, authorization technologies transport and validate delegated rights, local policy governs execution, and accounting records resource consumption. Resource rights provide the semantic glue connecting these independent functions into a coherent authorization architecture for autonomous scientific discovery. Their value lies not in replacing existing technologies, but in enabling those technologies to represent and manage authority in a form that scales naturally to large populations of cooperating autonomous agents. Appendix Appendix B illustrates how these integration points map to concrete infrastructure at an HPC facility; companion documents provide detailed designs for facility-specific deployment [2] and a facility-agnostic delegation service [1].

9 Properties of the Resource Rights Model

The preceding sections introduced the resource rights abstraction and showed how it induces a distributed authorization architecture for autonomous science. We now consider the properties that arise from the model. These properties are not additional design goals; rather, they emerge

naturally from the separation of authority from identity and from representing delegated authority explicitly.

9.1 Separation of Identity and Authority

A central feature of the model is the separation of authentication from authorization. Authentication establishes the identity of the requesting principal, while resource rights establish the authority that has been delegated to that principal. Neither concept depends on the other.

Proposition 4. *Authentication and authorization are logically independent.*

A resource right remains valid regardless of the authentication mechanism used to establish the identity of its holder. Conversely, authentication alone does not imply any authority to consume scientific resources. This separation permits identity infrastructures and authorization infrastructures to evolve independently while preserving consistent authorization semantics.

9.2 Non-Amplification of Authority

Delegation redistributes authority; it does not create new authority.

Proposition 5. *Delegation cannot increase authority.*

A delegated right may reduce available capacity, shorten its validity period, or introduce additional policy constraints. It may never relax inherited constraints or increase the quantity of delegated authority. Consequently, every delegation represents a monotonic reduction in authority, ensuring that complex delegation chains cannot accidentally amplify privilege.

This property substantially simplifies security analysis because the maximum authority possessed by any principal is bounded by the authority originally issued by an allocation authority.

9.3 Conservation of Quantitative Resources

Many scientific resources are consumable. Computational allocations, storage capacity, beam time, and laboratory operations all represent finite resources.

Proposition 6. *Delegation conserves quantitative authority.*

If a principal possesses an allocation of capacity C, the aggregate capacity delegated to subordinate principals cannot exceed C. Consumption reduces the remaining delegated capacity but does not alter the conservation property.

Resource accounting therefore follows naturally from the delegation structure rather than requiring an independent allocation mechanism.

9.4 Local Autonomy

Scientific cyberinfrastructure is inherently federated. Individual resource providers retain responsibility for operational policy, scheduling, safety, and institutional governance.

Proposition 7. *Authorization decisions require only local policy and presented authority.*

A resource provider need not understand the internal organization of a project, nor maintain resource-local authorization state for every collaborating institution or autonomous agent. It need only verify that the presented resource rights form a valid delegation chain recognized by local policy.

Consequently, authority may propagate across administrative domains without requiring centralized authorization management.

9.5 Least Privilege

Autonomous agents rarely require all of the authority possessed by the humans that supervise them. Instead, they require carefully bounded authority matched to their scientific responsibilities.

Proposition 8. *Every agent may operate with only the authority required for its assigned scientific function.*

This property follows directly from explicit delegation. Rather than impersonating users or sharing long-lived credentials, agents receive resource rights describing only the authority necessary to perform their assigned tasks. Compromise of an individual agent therefore exposes only the subset of authority explicitly delegated to that agent.

9.6 Scalability

Perhaps the most important property of the model concerns administrative complexity.

Traditional authorization systems maintain relationships between principals and resources. As the number of autonomous agents grows, the amount of authorization state grows approximately with the number of principal-resource relationships that must be administered.

By contrast, the resource rights model manages only the delegation of authority. Resource providers evaluate presented rights without maintaining knowledge of the global collaboration.

Although the precise complexity depends upon implementation, the conceptual difference is significant. Administrative effort shifts from managing resource-local authorization state toward managing scientific delegation relationships. These delegation relationships already exist within every scientific collaboration; the proposed model merely represents them explicitly.

9.7 Accountability

Every authorized action is associated with three independent pieces of information: the authenticated identity of the requester, the delegated resource rights under which the request was made, and the local policy that authorized execution.

Proposition 9. *Every authorized operation possesses a verifiable chain of responsibility.*

Because delegated rights preserve provenance, the authority under which an operation was performed remains recoverable long after execution has completed. Resource accounting, scientific provenance, and security auditing therefore become closely aligned, simplifying reproducibility, reporting, and governance.

Taken together, these properties distinguish the resource rights model from conventional authorization architectures. Rather than introducing a new security mechanism, the model provides a new representation of scientific authority. The resulting architecture scales naturally to autonomous scientific campaigns because it mirrors the way scientific responsibility is already distributed within research collaborations.

10 Related Work

The resource rights model draws upon several established research traditions, including distributed authorization, capability systems, scientific cyberinfrastructure, resource allocation, and autonomous agents. While each has addressed important aspects of authorization, none provides a unified model for representing, delegating, accounting for, and governing authority over scientific resources.

10.1 Distributed Authorization

The foundations of distributed authorization were established through early work on protection domains, authorization, and secure delegation. Lampson's protection model emphasized the importance of representing protection relationships explicitly [17], while Saltzer and Schroeder articulated design principles that continue to guide secure system design [22]. The Digital Distributed System Security Architecture introduced practical mechanisms for delegation in large-scale distributed systems [12]. Authorization logics subsequently provided formal frameworks for reasoning about distributed trust and delegation [3]. Public-key authorization systems such as SPKI/SDSI further demonstrated how authority could be delegated across organizational boundaries without centralized administration [10].

Early work on certificate-based authorization for scientific computing anticipated many of the challenges addressed by resource rights. The Grid Security Infrastructure [11] established a security architecture for large-scale distributed computations, introducing proxy credentials that enabled limited delegation across administrative domains. The Akenti system developed by Johnston and

colleagues [24] demonstrated how digitally signed certificates could represent and delegate access rights across distributed scientific resources, with policy enforcement at resource gateways. Akenti's use of attribute certificates to express conditions on authority foreshadowed the policy constraints central to the resource rights model.

These systems established many of the principles underlying modern distributed authorization. However, they were primarily concerned with secure delegation of permissions rather than with representing consumable scientific authority, resource accounting, or long-lived autonomous computational campaigns.

10.2 Capability Systems

Capability-based security has a long history, beginning with Dennis and Van Horn's capability architecture [9] and continuing through systems such as Hydra [31], KeyKOS [15], EROS [23], Capsicum [26], and, more recently, CHERI [27]. These systems demonstrated that unforgeable capabilities provide an elegant mechanism for expressing authority while avoiding many of the limitations of identity-based access control.

Resource rights share several characteristics with capabilities, particularly their explicit representation of authority and support for delegation. However, the focus of capability systems has traditionally been protection within operating systems and distributed computing environments. The resource rights model instead addresses authority over scientific resources distributed across independently administered organizations, incorporating quantitative allocations, scientific accounting, provenance, and policy-constrained delegation.

Table 2 summarizes the key distinctions. Traditional capabilities represent qualitative permission to access an object or invoke an operation. Resource rights extend this concept to include quantitative authority that is consumed during use, conservation invariants that prevent over-allocation, explicit accounting integrated with authorization, delegation provenance for scientific audit, and policy constraints that propagate through delegation chains. These extensions address requirements arising from scientific resource management that fall outside the traditional scope of capability-based protection.

10.3 Federated Scientific Cyberinfrastructure

Modern scientific cyberinfrastructure relies extensively on federated identity and authorization infrastructures. Systems such as Globus [25], OAuth 2.0 [14], OpenID Connect [21], CILogon [6], SciTokens [28], and the Worldwide LHC Computing Grid authorization infrastructure [7] have enabled secure access to distributed scientific resources spanning multiple institutions.

These systems solve the critical problems of authentication, identity federation, credential management, and secure token exchange. The resource rights model is intended to complement rather than replace these technologies. Its contribution lies in providing a common semantic model for

Table 2: Comparison of capabilities and resource rights.

Aspect	Capabilities	Resource Rights
Authority type	Qualitative (permit/deny)	Qualitative and quantitative (measurable capacity)
Consumption	Not modeled	Explicit; capacity decreases with use
Conservation	Not applicable	Delegated capacity $\leq$ available capacity
Accounting	External to model	Integrated; consumption recorded per right
Provenance	Typically absent	Required; full delegation chain preserved
Constraints	Implicit in capability	Explicit; propagate and strengthen through delegation
Primary domain	OS/distributed systems protection	Scientific resource governance

representing scientific authority independent of the mechanisms used to authenticate users or transport credentials.

10.4 Resource Allocation

Scientific computing facilities have long employed allocation mechanisms to manage scarce computational and experimental resources. High-performance computing centers allocate processor time through peer-reviewed allocation processes, schedulers such as Slurm manage execution against project allocations [32], cloud providers manage resource quotas and service limits, while synchrotrons, telescopes, and other large scientific facilities allocate observation or instrument time through proposal review.

These systems implicitly manage resource rights by controlling who may consume finite scientific resources. However, allocations, accounting, authorization, and delegation are generally represented using separate administrative mechanisms. The resource rights model proposes a common abstraction spanning all of these domains.

10.5 Autonomous Scientific Agents

Recent advances in large language models have led to increasing interest in autonomous software agents capable of planning and executing complex scientific workflows. Representative systems include AutoGen [30], LangGraph [18], and a growing ecosystem of agent frameworks supporting tool use, workflow composition, and collaborative reasoning. Protocols such as the Model Context Protocol (MCP) [4] enable these agents to interact with external tools and data sources.

These systems largely assume the existence of credentials permitting access to external services. Comparatively little attention has been devoted to the representation and governance of delegated authority itself. As autonomous agents become long-lived participants in scientific discovery, explicit models for allocating, constraining, auditing, and revoking authority become increasingly important.

10.6 Commercial Agent Identity

The rapid deployment of autonomous agents has prompted commercial identity providers to extend their platforms with agent-specific capabilities. Auth0 for AI Agents [5] provides machine-to-machine authentication, a token vault managing OAuth lifecycles across third-party integrations, and asynchronous authorization flows enabling human approval of agent-initiated actions. Okta for AI Agents [20] emphasizes enterprise governance: discovering unregistered "shadow agents," enforcing human sponsorship for each agent, and maintaining audit trails across agent lifecycles. Microsoft Entra Agent ID [19] introduces a first-class agent identity type distinct from both human users and traditional service accounts, with enforced sponsorship, lifecycle governance, and specialized OAuth flows. Google Cloud's Agent Identity [13], built on the SPIFFE standard [8], provides cryptographically attested identities tied to agent lifecycles. WorkOS has proposed auth.md [29], an open protocol for agent registration that extends OAuth with explicit agent identity in access tokens.

These commercial systems address important operational challenges: establishing agent identity, managing credential lifecycles, preventing credential sprawl, and maintaining audit logs. However, they remain fundamentally permission-based. An agent either has access to a resource or it does not; there is no representation of quantitative authority that is consumed during use. Delegation is implicit—a user grants an agent access—rather than explicit with preserved provenance. Accounting for resource consumption remains external to the authorization model. The resource rights abstraction proposed in this paper is complementary: commercial agent identity systems can serve as the authentication and token management layer, while resource rights provide the semantic model for representing bounded, delegable, consumable authority over scientific resources.

Taken together, these research traditions provide many of the building blocks required for autonomous scientific cyberinfrastructure. Distributed authorization explains how authority may be delegated, capability systems show how authority may be represented explicitly, federated cyberinfrastructure provides mechanisms for authentication and credential management, resource allocation systems manage scarce scientific resources, and modern agent frameworks demonstrate increasingly capable autonomous behavior. Table 1 summarizes how the resource rights model differs from traditional ACL and RBAC approaches across several dimensions. To our knowledge, however, no prior work proposes a unified abstraction that treats scientific authority itself as an explicit, measurable, delegable, and consumable object independent of identity and implementation technology. The resource rights model proposed in this paper is intended to provide that missing conceptual layer.

11 Future Directions

The resource rights model provides a foundation for governing autonomous systems, but its full implications extend beyond the authorization architecture presented in this paper. We conclude by identifying several directions that merit further investigation.

11.1 Authority Economics

Resource rights share structural similarities with economic objects. They are scarce, quantifiable, delegable, and consumable. This observation suggests potential connections to resource markets, pricing mechanisms, and economic allocation theory. Scientific facilities already implicitly price resources through allocation committees and proposal reviews; resource rights could make these valuations explicit and enable more dynamic allocation mechanisms. Whether authority should be tradable, lendable, or subject to market forces raises both technical and governance questions that deserve careful study.

11.2 Rights Composition and Decomposition

Many scientific workflows require simultaneous access to multiple resources: computation, storage, instruments, and data repositories. The current model treats each resource right independently, but complex workflows may benefit from composite rights that bundle authority over multiple resources into a single delegable unit. Conversely, decomposition mechanisms could automatically partition high-level campaign authority into the specific rights required by individual workflow steps. The algebra of such compositions—and the constraints under which they preserve the model's safety properties—remains to be developed.

11.3 Dynamic Agent Hierarchies

The architecture assumes that humans create autonomous agents and delegate authority to them. As agent capabilities mature, agents may increasingly create subordinate agents to handle specialized subtasks. Such dynamic hierarchies raise questions about delegation depth, authority propagation, and accountability. Under what conditions should an agent be permitted to delegate authority it has received? How should the provenance chain represent multi-level agent delegation? What governance mechanisms prevent runaway delegation while preserving operational flexibility?

11.4 Cross-Domain Federation

Although motivated by scientific cyberinfrastructure, the resource rights model applies wherever autonomous agents require bounded, accountable authority: enterprise operations, manufacturing, logistics, healthcare, and autonomous vehicles. Federation across these domains—where a scientific agent might invoke commercial cloud services, or an industrial agent might access research data—requires mechanisms for translating authority across trust boundaries. Standards for interoperable resource rights could enable new forms of cross-sector collaboration.

11.5 Build Artifacts as Managed Resources

Software environments present a resource management challenge distinct from computation or storage. An agent porting code across facilities repeatedly rebuilds the same dependencies—a venv on one system, a container on another—because there is no first-class concept of a build artifact that persists across execution contexts and transfers across facilities. A *build-artifact right* could authorize creation, versioning, and distribution of immutable software bundles (container images, lockfiles with cached packages) through the same delegation and transfer mechanisms used for data. The artifact becomes a managed resource: content-addressed, Globus-transferable, and scoped by the delegation that created it.

11.6 Scoped Privileged Capabilities

Some operational tasks require elevated privileges: resetting a GPU after a crash, installing a system library, or modifying network configuration on a testbed. The current model assumes unprivileged execution contexts, forcing such tasks back to human administrators. A *privileged-capability right* could authorize specific elevated actions—not a root shell, but a bounded set of operations ("may reset GPU 0–3," "may write to /opt/project")—with the same delegation, constraint, and audit semantics as other resource rights. This extends the model into system administration without abandoning the principle of least privilege.

11.7 Agent-Accessible State Services

Agents operating across multiple facilities need to track progress, coordinate work, and maintain state that survives individual job completions. Filesystem-based solutions fail when facilities don't share storage. A token-gated *state service*—a simple key-value or document store accessible via HTTPS from any network-connected agent—would provide a natural complement to storage rights. The agent's resource right authorizes reads and writes to a scoped namespace; the state service handles replication and availability. This is structurally similar to what Globus provides for bulk data but optimized for small, structured, frequently-updated state.

11.8 Formal Verification

The safety properties identified in this paper—non-amplification, conservation, provenance preservation—are stated as invariants but not formally verified against a complete operational semantics. A rigorous formal treatment would strengthen confidence in the model and potentially reveal additional properties or edge cases. Such formalization could also support automated verification of delegation policies and runtime enforcement mechanisms.

11.9 Authority Lifecycle Management

Real deployments will require tooling for managing authority at scale: dashboards showing current delegations, alerts when authority is nearly exhausted, automated renewal workflows, and forensic tools for investigating authorization failures. The operational practices surrounding resource rights—how organizations structure their delegation hierarchies, how they handle exceptions, how they audit authority flows—will likely prove as important as the underlying model.

11.10 Governance and Policy

Resource rights make authority explicit, but they do not themselves determine policy. Questions of who should receive authority, under what conditions, and subject to what constraints remain organizational and governance decisions. The model provides a framework within which such policies can be expressed and enforced, but the policies themselves must emerge from scientific communities, institutions, and funding agencies. Understanding how resource rights interact with existing governance structures—and how they might reshape those structures—is an important area for future work.

The central claim of this paper is that authority should be treated as a first-class managed resource. If this abstraction proves useful, its implications will extend well beyond the authorization architecture presented here. We offer the resource rights model as a foundation for that broader investigation.

12 Conclusion

Autonomous systems are transforming how scientific discovery is conducted. As AI agents become active participants in research—designing experiments, executing workflows, controlling instruments, and coordinating across institutions—the question of how to govern their authority becomes unavoidable.

This paper has argued that the underlying challenge is conceptual rather than technological. Existing authorization systems manage permissions attached to identities; they do not provide an explicit representation of authority that can be measured, delegated, constrained, consumed, and revoked independently of who exercises it. We introduced *resource rights* as that missing abstraction: authority treated as a managed resource, subject to the same lifecycle as any other scarce scientific asset.

From this abstraction, an architecture follows naturally. Allocation authorities issue rights, principals delegate bounded subsets according to scientific responsibility, autonomous agents exercise only the authority explicitly granted to them, and resource providers validate presented rights against local policy. The resulting system scales with organizational structure rather than with the Cartesian product of users and resources.

The model satisfies important safety properties by construction. Authority cannot be amplified through delegation. Quantitative capacity is conserved. Every authorized action carries verifiable provenance linking it to a chain of human responsibility. These guarantees emerge from the abstraction itself, not from additional enforcement mechanisms.

We do not claim that resource rights solve every problem of autonomous system governance. Questions of policy—who should receive authority, under what conditions, subject to what oversight—remain organizational decisions that no technical model can answer. What resource rights provide is a foundation: a common language for expressing, distributing, and accounting for authority across federated systems.

The organizations that adopt explicit authority management early will be positioned to deploy autonomous systems confidently at scale. Those that continue to rely on ad-hoc combinations of service accounts, API keys, and manual approvals will find themselves constrained—either moving slowly or accepting unmanaged risk.

Authority is no longer something humans exercise directly. It is something they delegate. The infrastructure for that delegation should be as carefully designed as the infrastructure for computation, storage, and communication. Resource rights offer a starting point for that design.

References

[1] Resource rights delegation service: Design document. Technical report, Argonne National Laboratory, 2026. Companion document specifying a facility-agnostic delegation service for storage and compute resource rights.

[2] Resource rights implementation at ALCF. Technical report, Argonne National Laboratory, 2026. Companion document describing facility-specific deployment of resource rights at the Argonne Leadership Computing Facility.

[3] Martín Abadi, Michael Burrows, Butler W. Lampson, and Gordon D. Plotkin. A calculus for access control in distributed systems. *ACM Transactions on Programming Languages and Systems*, 15(4):706–734, 1993. doi:10.1145/155183.155225.

[4] Anthropic. Model Context Protocol specification, 2024. URL: https://modelcontextprotocol.io.

[5] Auth0. Auth0 for AI agents, 2025. URL: https://auth0.com/docs/get-started/auth0-for-ai-agents.

[6] Jim Basney, Terry Fleury, and Jeff Gaynor. CILogon: A federated X.509 certification authority for cyberinfrastructure logon. *Concurrency and Computation: Practice and Experience*, 26(13):2225–2239, 2014. doi:10.1002/cpe.3265.

[7] Ian Bird. Computing for the Large Hadron Collider. *Annual Review of Nuclear and Particle Science*, 61:99–118, 2011. doi:10.1146/annurev-nucl-102010-130059.

[8] Cloud Native Computing Foundation. SPIFFE: Secure production identity framework for everyone, 2024. URL: https://spiffe.io.

[9] Jack B. Dennis and Earl C. Van Horn. Programming semantics for multiprogrammed computations. *Communications of the ACM*, 9(3):143–155, 1966. doi:10.1145/365230.365252.

[10] Carl Ellison, Bill Frantz, Butler Lampson, Ron Rivest, Brian Thomas, and Tatu Ylonen. SPKI certificate theory. RFC 2693, 1999. URL: https://www.rfc-editor.org/rfc/rfc2693.

[11] Ian Foster, Carl Kesselman, Gene Tsudik, and Steven Tuecke. A security architecture for computational grids. In *Proceedings of the 5th ACM Conference on Computer and Communications Security (CCS)*, pages 83–92, 1998. doi:10.1145/288090.288111.

[12] Morrie Gasser, Andy Goldstein, Charlie Kaufman, and Butler W. Lampson. The digital distributed system security architecture. In *Proceedings of the 12th National Computer Security Conference*, pages 305–319, 1989.

[13] Google Cloud. Agent identity for Google Cloud, 2025. URL: https://cloud.google.com/blog/products/identity-security/whats-new-in-iam-security-governance-and-runtime-defense.

[14] Dick Hardt. The OAuth 2.0 authorization framework. RFC 6749, 2012. URL: https://www.rfc-editor.org/rfc/rfc6749.

[15] Norman Hardy. The KeyKOS architecture. *ACM SIGOPS Operating Systems Review*, 19(4):8–25, October 1985. doi:10.1145/858336.858337.

[16] Michael Jones, John Bradley, and Nat Sakimura. JSON Web Token (JWT). RFC 7519, 2015. URL: https://www.rfc-editor.org/rfc/rfc7519.

[17] Butler W. Lampson. Protection. In *Proceedings of the Fifth Princeton Symposium on Information Sciences and Systems*, pages 437–443, 1971.

[18] LangChain, Inc. LangGraph, 2024. URL: https://github.com/langchain-ai/langgraph.

[19] Microsoft. Microsoft Entra Agent ID, 2025. URL: https://learn.microsoft.com/en-us/entra/agent-id/what-is-microsoft-entra-agent-id.

[20] Okta. Okta for AI agents, 2026. URL: https://www.okta.com/products/govern-ai-agent-identity/.

[21] OpenID Foundation. OpenID Connect Core 1.0, 2014. URL: https://openid.net/specs/openid-connect-core-1_0.html.

[22] Jerome H. Saltzer and Michael D. Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, 1975. doi:10.1109/PROC.1975.9939.

[23] Jonathan S. Shapiro, Jonathan M. Smith, and David J. Farber. EROS: A fast capability system. In *Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP)*, pages 170–185, 1999. doi:10.1145/319151.319163.

[24] Mary R. Thompson, William Johnston, Srilekha Mudumbai, Gary Hoo, Keith Jackson, and Abdelilah Essiari. Certificate-based access control for widely distributed resources. In *Proceedings of the 8th USENIX Security Symposium*, 1999. URL: https://www.usenix.org/conference/8th-usenix-security-symposium/certificate-based-access-control-widely-distributed.

[25] Steven Tuecke, Rachana Ananthakrishnan, Kyle Chard, Mattias Lidman, Brendan McCollam, Stephen Rosen, and Ian Foster. Globus Auth: A research identity and access management platform. In *Proceedings of the IEEE 12th International Conference on e-Science*, pages 203–212, 2016. doi:10.1109/eScience.2016.7870901.

[26] Robert N. M. Watson, Jonathan Anderson, Ben Laurie, and Kris Kennaway. Capsicum: Practical capabilities for UNIX. In *Proceedings of the 19th USENIX Security Symposium*, pages 29–46, 2010. URL: https://www.usenix.org/conference/usenixsecurity10/capsicum-practical-capabilities-unix.

[27] Robert N. M. Watson, Jonathan Woodruff, Peter G. Neumann, Simon W. Moore, Jonathan Anderson, David Chisnall, Nirav Dave, Brooks Davis, Khilan Gudka, Ben Laurie, Steven J. Sheridan, Stacey Son, and Michael Sherr. CHERI: A hybrid capability-system architecture for scalable software compartmentalization. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 20–37, 2015. doi:10.1109/SP.2015.9.

[28] Alex Withers, Brian Bockelman, Derek Weitzel, Duncan Brown, Jeff Gaynor, Jim Basney, Todd Tannenbaum, and Zach Miller. SciTokens: Capability-based secure access to remote scientific data. In *Proceedings of the Practice and Experience in Advanced Research Computing (PEARC)*, pages 1–8, 2018. doi:10.1145/3219104.3219135.

[29] WorkOS. auth.md: Open protocol for agent registration, 2026. URL: https://workos.com/auth-md.

[30] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023. URL: https://arxiv.org/abs/2308.08155.

[31] William A. Wulf, Ellis Cohen, William Corwin, Anita Jones, Roy Levin, C. Pierson, and Fred Pollack. HYDRA: The kernel of a multiprocessor operating system. *Communications of the ACM*, 17(6):337–345, 1974. doi:10.1145/355616.364017.

[32] Andy B. Yoo, Morris A. Jette, and Mark Grondona. SLURM: Simple linux utility for resource management. In *Proceedings of the 9th International Workshop on Job Scheduling Strategies for Parallel Processing (JSSPP)*, pages 44–60, 2003. doi:10.1007/10968987_3.

Appendix A: Threat Model and Security Analysis

This appendix examines the security properties of the resource rights model under adversarial conditions. We identify representative threats, describe how the model's structural properties provide mitigation, and clarify assumptions and limitations.

A.1 Trust Assumptions

The resource rights model assumes the following trust relationships:

- **Allocation authorities are trusted roots.** They issue legitimate resource rights and do not collude with adversaries. Compromise of an allocation authority is catastrophic but out of scope for the model itself.
- **Cryptographic integrity is maintained.** Resource rights are protected by digital signatures or equivalent mechanisms that prevent forgery and detect tampering. The security of these mechanisms is assumed.
- **Authentication is reliable.** Identity providers correctly authenticate principals. The model builds on authentication but does not replace it.
- **Resource providers enforce local policy honestly.** They validate presented rights correctly and do not grant unauthorized access.
- **Principals may be malicious or compromised.** The model explicitly anticipates that individual principals, including autonomous agents, may attempt to exceed their authority or may be compromised by external attackers.

A.2 Threat Analysis

Table 3 summarizes representative threats and how the resource rights model addresses them.

A.3 Security Properties Derived from Model Invariants

The three formal invariants established in Section 4 directly address core security concerns:

Non-amplification ensures that no sequence of delegations can produce authority exceeding what was originally issued. An attacker who compromises a principal at any level of the delegation hierarchy gains access only to the authority that was explicitly delegated to that principal. This property bounds the blast radius of any single compromise.

Conservation ensures that quantitative authority cannot be created through delegation. A principal cannot delegate more capacity than they possess, preventing resource exhaustion attacks that attempt to over-commit allocations. Combined with consumption tracking, this property supports detection of anomalous behavior.

Provenance preservation ensures that every authorized action can be traced to its originating

Table 3: Threat analysis for the resource rights model.

Threat	**Description**	**Mitigation**	**Residual Risk**
Privilege escalation	Agent attempts to acquire authority beyond what was delegated	Non-amplification invariant: delegation can only restrict, never expand authority	None if invariant enforced
Compromised agent	Attacker gains control of an autonomous agent	Least privilege: agent holds only delegated rights; revocation invalidates all derived authority	Exposure limited to delegated scope
Credential theft	Attacker steals resource right tokens	Time-bounded validity (τ); revocation services; consumption tracking detects anomalies	Window of exposure until expiration or revocation
Unauthorized delegation	Principal delegates rights they do not possess	Conservation invariant: delegated capacity $\leq$ available capacity; provenance validation at runtime	None if validation enforced
Forged rights	Attacker fabricates resource rights	Cryptographic signatures; provenance chain validation to recognized allocation authority	Depends on cryptographic assumptions
Collusion	Multiple principals combine authority	Each right validated independently; no mechanism to merge authority from independent chains	Intended behavior: separate rights remain separate
Revocation delay	Revoked rights used before invalidation propagates	Short validity periods; online validation; revocation lists; local policy can require freshness checks	Trade-off with availability
Denial of service	Attacker exhausts legitimate principal's allocation	Consumption tracking; alerts on anomalous usage; sub-delegation with reserved capacity	Operational rather than architectural
Provenance forgery	Attacker falsifies delegation chain	Signed delegation records; each link verifiable to issuer	Depends on cryptographic assumptions

allocation authority through a verifiable chain. This property supports forensic analysis after incidents and provides accountability that deters malicious behavior.

A.4 Limitations and Out-of-Scope Threats

The resource rights model does not address:

- **Compromise of allocation authorities.** If an allocation authority is compromised, illegitimate rights may be issued. Organizational security practices and key management are required.
- **Side-channel attacks.** Information leakage through timing, resource consumption patterns, or other side channels is outside the model's scope.
- **Availability attacks.** The model does not prevent denial-of-service attacks against resource providers or authorization infrastructure.
- **Implementation vulnerabilities.** Bugs in software implementing resource rights validation could undermine security guarantees.
- **Social engineering.** Attacks that manipulate humans into issuing inappropriate delegations are outside the technical model.
- **Covert channels between agents.** Agents operating within their delegated authority might still coordinate maliciously.

A.5 Comparison with Identity-Based Authorization

Traditional identity-based authorization faces a different threat profile. Compromising a user's credentials typically grants access to everything that user can access. In contrast, compromising an agent's resource rights exposes only the bounded authority explicitly delegated to that agent.

Similarly, revocation in identity-based systems often requires modifying access control state at every resource the user could access. In the resource rights model, revoking a right (or its ancestor in the delegation chain) invalidates all derived authority without requiring changes at individual resource providers.

These differences do not make resource rights universally superior—identity-based systems have advantages in simplicity and are well-understood—but they illustrate the security trade-offs inherent in the two approaches.

Appendix B: Implementation Sketch: HPC Facility

This appendix illustrates how the resource rights model maps to existing infrastructure at a leadership-class computing facility. The goal is not a complete implementation specification but a demonstration that resource rights can be realized by extending current systems rather than replacing them. Companion documents provide detailed designs for facility-specific deployment at ALCF [2] and a facility-agnostic delegation service supporting both storage and compute resource rights [1].

B.1 Existing Infrastructure

A typical HPC facility already manages implicit resource rights through several mechanisms:

- **Allocations:** Peer-reviewed awards granting projects a bounded quantity of node-hours on specific systems for defined periods.
- **Identity:** Federated authentication via institutional credentials or services such as Globus Auth.
- **Scheduling:** Job schedulers (PBS, Slurm) that verify project membership and available allocation before accepting submissions.
- **Accounting:** Systems that track consumption against project allocations.

These components implement the core resource rights operations—issue, consume, expire—but lack explicit support for delegation to autonomous agents. An agent today must either impersonate the supervising researcher (using their credentials) or operate through manually configured service accounts with static permissions.

B.2 Resource Rights as Scoped Tokens

A resource right can be represented as a signed token (e.g., a JSON Web Token) containing the tuple elements defined in Section 4:

```
{
  "iss": "facility.example.org",
  "sub": "agent-uuid-12345",
  "aud": "facility-scheduler",
  "exp": 1738368000,
  "project": "ProjectX_2026",
  "delegator": "pi-username",
  "resources": {
    "system": "aurora",
    "node_hours": 10000,
```

```
    "max_nodes_per_job": 512,
    "queues": ["prod", "debug"]
  },
  "constraints": {
    "require_approval_above": 2048
  },
  "provenance": ["allocation-789", "delegation-456"]
}
```

The `resources` object represents (R, O, q): the target system, permitted operations (implicit in queue membership), and quantitative capacity. The `constraints` object represents C: policy restrictions that propagate through delegation. The `exp` field represents τ, and the `provenance` array represents π.

The token is signed by the facility's delegation service. The scheduler validates the signature, verifies that the delegator holds sufficient allocation, checks that the requested job falls within the token's constraints, and debits consumption from both the delegation and the underlying allocation.

B.3 Integration Points

Figure 6 illustrates how resource rights integrate with existing HPC infrastructure. Three extensions connect resource rights to existing systems:

Agent identity. Agents register as OAuth2 clients (e.g., via Globus Auth) and receive their own credentials distinct from their supervising researcher. This establishes the agent as a principal that can hold delegated rights.

Delegation service. A lightweight service allows researchers to create resource rights delegating portions of their allocation to registered agents. The service maintains delegation state, issues signed tokens, supports revocation, and provides a dashboard showing consumption per delegation. This service need not be in the critical path for job submission; tokens can be issued in advance.

Scheduler integration. The job scheduler (or a submission proxy) is extended to accept a resource right token alongside job submissions. Validation checks the signature, expiration, constraint satisfaction, and available capacity. To avoid placing the delegation service in the critical path, the resource provider maintains a local replica of delegation state: the delegation service pushes new delegations and revocations, while consumption updates the local state immediately and reports back asynchronously. The delegation chain is recorded in job metadata for accounting and audit.

B.4 Example: Autonomous Simulation Campaign

A researcher with a 500,000 node-hour allocation deploys an agent to explore parameter space through thousands of short simulations:

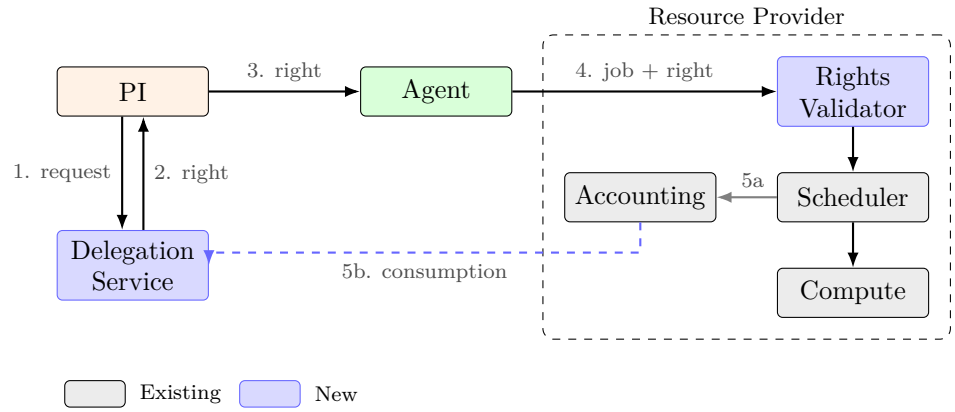


Figure 6: Integration architecture. The PI requests a resource right from the delegation service and receives it as a signed token (steps 1–2), then passes it to the agent (step 3). The agent presents the resource right when submitting jobs (step 4). After execution, consumption is recorded in both the facility's accounting system (step 5a) and reported back to the delegation service (step 5b) to debit the delegation.

1. The researcher registers "SimExplorer-Agent" as a Globus application linked to their project.
2. Using the delegation portal, they create a resource right granting 50,000 node-hours, limited to jobs of 128 nodes or fewer, valid for 30 days.
3. The delegation service issues the resource right as a signed token encoding the agent's identity as subject, the researcher as delegator, the quantitative limit (50,000 node-hours), the constraints (max 128 nodes), the validity period, and a provenance chain linking to the original allocation. The service returns this resource right to the researcher, who configures the agent with it—either by storing it in the agent's secure credential store or by providing an endpoint from which the agent can retrieve it.
4. The agent authenticates with its own credentials and includes the resource right in each job submission.
5. The scheduler validates the resource right and accepts jobs within constraints.
6. Consumption is tracked against both the delegation and the researcher's allocation.
7. If the agent exhausts its delegation, it cannot submit further jobs until the researcher issues additional authority.
8. If results warrant, the researcher increases the delegation without administrative intervention.
9. If the agent misbehaves, the researcher revokes the delegation immediately.

The researcher retains oversight while the agent operates autonomously within bounded authority. Figure 7 summarizes the runtime flow. The facility's existing accounting, scheduling, and security

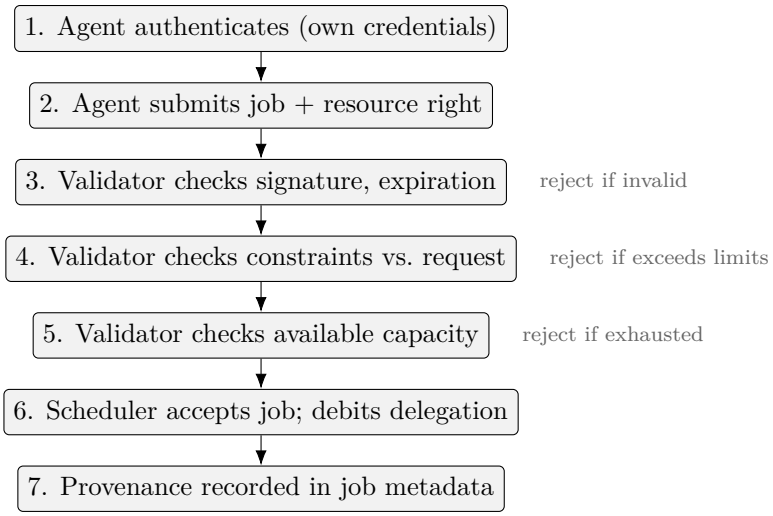


Figure 7: Job submission flow with resource rights validation. Each step can reject the submission if validation fails.

infrastructure remains intact; resource rights provide the semantic layer for explicit, delegable, auditable authority over scientific resources.

B.5 Multi-Facility Allocations

The example above assumes a single facility, but scientific allocations increasingly span multiple sites. A DOE allocation might grant 500,000 node-hours usable at ALCF, OLCF, or NERSC. The resource rights model supports this scenario, though the accounting infrastructure requires additional coordination.

The key challenge is maintaining the conservation invariant across facilities: if an agent holds a token authorizing 100,000 node-hours across all DOE facilities, no combination of submissions should exceed that limit. Two approaches are possible:

Centralized accounting. A DOE-level delegation service issues tokens and tracks consumption across all participating facilities. Each facility's rights validator queries this central service to check remaining capacity before accepting jobs and reports consumption after completion. The provenance chain in each token traces back to DOE as the root allocation authority. This approach maintains strict conservation but requires real-time coordination.

Pre-partitioned delegations. The researcher explicitly divides the allocation among facilities when creating delegations: 50,000 node-hours to an ALCF-specific token, 30,000 to OLCF, and so forth. Each facility manages its slice independently. This approach is simpler but less flexible—reallocating mid-campaign requires explicit action by the researcher.

The formal model accommodates both approaches. The delegation operation can restrict a right to a specific resource (single-facility token) or leave it valid across multiple resources (multi-facility

token with centralized accounting). The choice is an implementation decision that may vary based on the administrative relationships among facilities and the maturity of cross-site coordination infrastructure.