Standards Body · Foundation paper, public edition · Released July 17, 2026
Canonical record: https://standardsbody.ai/library/foundation-paper/held-out-evaluations/

FOUNDATION_02_HELD_OUT_EVALUATIONS.md

Foundation 2: Held-Out Evaluations

Series: Foundations for Frontier AI Evaluation Infrastructure
Version: 1.0
Status: Canonical working white paper
Project: Standards Body
Primary domain: standardsbody.ai
Core line: Foundations for Frontier AI
Research basis reviewed through: July 16, 2026
Document owner: Standards Body
Review cycle: Annual review, with event-triggered revision after material leakage, compromise, methodological change, or new evidence

---

Document Purpose

This paper defines the Standards Body position on held-out evaluations for frontier artificial intelligence.

It is intended to serve as:

- A first-principles explanation of why some evaluation material should remain unavailable during system development
- A design framework for creating, securing, administering, and governing held-out evaluations
- A bridge between scientific reproducibility and necessary confidentiality
- A reference for future evaluation standards, auditor requirements, research programs, and institutional partnerships
- A durable source document from which shorter articles, technical specifications, and implementation guides can be derived

This paper is not a universal security standard, legal rule, or certification scheme.

It does not argue that every evaluation should be secret.

It does not claim that held-out tasks automatically produce valid results.

It establishes the principles and institutional conditions required for held-out evaluation to improve measurement rather than merely hide it.

Executive Summary

Public benchmarks have played a central role in artificial intelligence research.

They enable comparison, reproducibility, open criticism, and rapid experimentation. Public tasks often help researchers understand what systems can do, identify weaknesses, and coordinate around shared measurement.

But public visibility creates pressure on the measurement itself.

Once benchmark items, formats, answer keys, scoring rules, and common solution strategies are broadly available, they can enter:

- Pretraining corpora
- Fine-tuning datasets
- Reinforcement-learning data
- Prompt libraries
- Evaluation harnesses
- Synthetic data pipelines
- Developer workflows
- Public leaderboards
- Model-generated training material

A system may then perform well because it has encountered the evaluation, close variants, or its underlying template. Even when exact exposure cannot be proven, repeated optimization against a known benchmark can weaken the connection between benchmark performance and genuine generalization.

Held-out evaluations address this problem by reserving some evaluation material from normal development access.

The reserved material may include:

- Tasks
- Datasets
- prompts
- solutions
- scoring rules
- attack methods
- environment configurations
- hidden state
- evaluator interventions
- sampling procedures
- task generators

- capability thresholds
- combinations of these components

The objective is not secrecy for its own sake.

The objective is to preserve the informational value of the evaluation.

A held-out evaluation should help answer questions such as:

- Can the system generalize to tasks it was not optimized against?
- Does performance persist under unfamiliar but valid conditions?
- Can safeguards resist attacks not seen during development?
- Has a capability crossed a threshold that public tests can no longer measure reliably?
- Does an independent evaluator reproduce or challenge the developer's internal findings?
- Does the system behave differently when it cannot predict the test?

Held-out evaluation is not a complete solution.

A private test can still be:

- Poorly designed
- unrepresentative
- statistically weak
- insecure
- biased
- unfair
- irreproducible
- captured by the organization administering it
- disconnected from deployment
- overinterpreted

Confidentiality can protect measurement integrity, but it can also shield weak methods from scrutiny.

The central institutional challenge is therefore to combine protected content with inspectable process.

Standards Body adopts the following core position:

> **Some frontier AI evaluations should contain held-out components when public exposure would materially reduce their validity. Those components should be governed through proportionate security, independent oversight, documented provenance, controlled access, reproducible administration, transparent methodology, explicit compromise response, and clear limits on interpretation.**

Held-out evaluation should be understood as a spectrum, not a binary category.

A protocol may keep private:

- Only the final test items

- The task bank but not the construct
- The attack library but not the scoring method
- The sampling seed but not the generator
- The thresholds but not the evaluation family
- The full evaluation until administration, followed by delayed release
- The evaluation indefinitely because disclosure would create misuse risk

Different choices create different scientific and institutional tradeoffs.

A mature held-out evaluation ecosystem should distinguish at least five questions:

1. **What is being held out?**
2. **From whom is it held out?**
3. **For how long is it held out?**
4. **What threat is the holdout intended to mitigate?**
5. **Who can independently verify that the process remains valid?**

The strongest held-out systems do not rely on obscurity alone.

They use defense in depth:

- Task renewal
- provenance records
- access controls
- compartmentalization
- secure execution
- logging
- independent review
- statistical sampling
- cryptographic commitments where useful
- compromise detection
- result expiration
- formal retirement

The purpose of this foundation is to make held-out evaluation a credible scientific and institutional practice rather than an informal collection of secret test questions.

---

1. Foundational Proposition

1.1 Core Thesis

> **When public exposure would materially weaken an evaluation, some content should remain unavailable during system development and should be administered under controlled conditions.**

This thesis depends on four qualifications.

First, secrecy must serve a defined measurement purpose.

Second, the protected evaluation must still satisfy scientific and institutional standards.

Third, the process must be accountable even when the content is restricted.

Fourth, held-out evidence should complement other evidence rather than become the sole basis for broad claims.

1.2 Epistemic Thesis

> **A held-out evaluation is valuable only to the extent that it provides information not already available through public optimization.**

The information gain may come from:

- Unseen tasks
- new distributions
- hidden adversarial strategies
- unpredictable sampling
- unfamiliar environments
- independent administration
- protected system configurations
- reduced opportunity for benchmark-specific tuning

If a held-out test closely reproduces public material, its secrecy may add little.

1.3 Institutional Thesis

> **Maintaining a credible holdout is an institutional function, not merely a data-storage function.**

It requires:

- Governance
- security
- funding
- task development
- quality assurance
- evaluator competence
- conflict management
- access procedures
- incident response
- archival systems
- versioning
- renewal
- appeals
- public accountability

1.4 Fairness Thesis

> **The more consequential the result, the stronger the obligation to provide fair notice, consistent administration, reviewable process, and a path to challenge material error.**

A developer should not need access to exact tasks in order to understand:

- The construct
- the domain
- the permitted tools
- the scoring principles
- the consequences
- the review process
- the grounds for appeal

---

2. Scope and Boundaries

2.1 What This Foundation Covers

This paper covers held-out components used in:

- Capability evaluation
- safeguard evaluation
- dangerous-capability evaluation
- red teaming
- agent evaluation
- model-behavior evaluation
- robustness testing
- deployment-readiness review
- third-party auditing
- standards conformity assessment
- incident-driven evaluation
- threshold evaluation
- post-deployment re-evaluation

2.2 Evaluated Objects

Held-out evaluations may apply to:

- Base models
- fine-tuned models
- deployed products
- agentic systems
- multimodal systems

- models with tools
- models with retrieval
- models with memory
- safety layers
- monitoring systems
- full sociotechnical deployments

2.3 What This Foundation Does Not Claim

This paper does not claim that:

- Public benchmarks are inherently inferior
- Confidentiality guarantees uncontaminated data
- A private evaluator is automatically independent
- A hidden test is automatically fair
- Every result should remain private
- Every laboratory should receive identical access in every case
- Cryptography can replace governance
- A passing result proves safety
- Exact reproducibility is always possible
- Open-source communities should be excluded
- One institution should control the global evaluation supply

2.4 Relationship to Dynamic Evaluation

Held-out evaluation is closely connected to `FOUNDATION_01_DYNAMIC_EVALUATION_PROTOCOLS.md`.

A static private test can become stale, leak, saturate, or lose relevance.

A strong holdout therefore requires:

- Rotation
- renewal
- versioning
- contamination monitoring
- expiration
- retirement

2.5 Relationship to Independent Review

Held-out systems concentrate information in a smaller number of institutions.

That makes independent review more important, not less.

2.6 Relationship to Security

Security protects:

- The integrity of the test
- the confidentiality of sensitive content
- the safety of evaluation environments
- developer intellectual property
- personal information
- national-security-relevant material
- result integrity

Security should not be used to avoid methodological scrutiny.

---

3. Canonical Definitions

3.1 Held-Out Evaluation

A held-out evaluation is an evaluation in which one or more material components are intentionally unavailable to the evaluated system's developer, training process, tuning process, or ordinary users before administration.

3.2 Holdout Set

A holdout set is a reserved collection of tasks, examples, environments, or observations not used for development or routine validation.

3.3 Private Benchmark

A private benchmark is a benchmark whose content is not generally available.

A private benchmark may or may not be well governed.

3.4 Confidential Evaluation

A confidential evaluation is an evaluation whose content, process, results, or combination is restricted to authorized parties.

Confidentiality can serve security, privacy, commercial, or measurement purposes.

3.5 Secret Evaluation

"Secret evaluation" is not preferred Standards Body terminology because it does not identify who is authorized, what is restricted, or why.

Use more precise terms such as:

- Held-out
- controlled access
- confidential
- restricted
- embargoed
- security-sensitive

3.6 Blind Evaluation

A blind evaluation restricts information from one or more participants to reduce bias or gaming.

Examples include:

- Developer does not see test items
- rater does not know model identity
- evaluator does not know treatment condition
- model receives no indication that it is being evaluated

3.7 Double-Blind Evaluation

A double-blind design restricts relevant information from both the evaluated party and the assessor where feasible.

In frontier AI evaluation, perfect double blinding is often difficult because system interfaces, behavior, or infrastructure can reveal identity.

3.8 Embargoed Evaluation

An embargoed evaluation remains restricted until a defined time or event.

3.9 Retired Holdout

A retired holdout is no longer used for active decision-making.

It may be:

- Published
- archived under control
- partially disclosed
- destroyed under policy
- preserved for historical replication

3.10 Evaluation Contamination

Evaluation contamination occurs when information about the evaluation enters the development or optimization process in a way that weakens the intended separation between training and testing.

3.11 Leakage

Leakage is unauthorized or unintended disclosure of protected evaluation information.

3.12 Compromise

An evaluation is compromised when its integrity, confidentiality, validity, or fairness has been materially weakened.

3.13 Chain of Custody

Chain of custody is the documented history of who created, accessed, transferred, administered, modified, or stored evaluation materials.

3.14 Access Tier

An access tier is a defined level of authorization governing which evaluation components a party may see or use.

3.15 Controlled Evaluation Environment

A controlled evaluation environment is a technical and procedural setting designed to limit unauthorized access, data extraction, harmful action, and unlogged modification.

3.16 Attestation

Attestation is evidence used to determine whether a computing environment, code package, or security state is what it claims to be.

3.17 Cryptographic Commitment

A cryptographic commitment allows a party to commit to data or a decision before disclosure while later enabling verification that it was not altered.

3.18 Retro-Holdout

A retro-holdout is a newly created private evaluation designed to approximate the construct or distribution of an older public benchmark after the original benchmark may have become contaminated.[^retro-holdout]

4. Why Public Benchmarks Become Insufficient

4.1 Public Exposure Changes the Measurement Environment

A public benchmark is not only a neutral test.

Once it becomes influential, it becomes part of the environment that researchers and developers optimize within.

This can improve systems.

It can also reduce the benchmark's value as independent evidence.

4.2 Exact Contamination

Exact contamination occurs when evaluation items or solutions appear in training or tuning data.

The resulting score may partly reflect recall.

4.3 Semantic Contamination

A model may encounter paraphrases, transformed versions, translations, explanations, or synthetic variants.

Exact-string matching may therefore underestimate exposure.

4.4 Template Contamination

A model may learn the recurring structure of an evaluation even when individual items are new.

Examples:

- Prompt format
- answer schema
- common distractors
- scoring conventions
- reasoning templates

4.5 Development Feedback Contamination

Repeated testing on the same benchmark can influence:

- Checkpoint selection
- model architecture
- data curation
- fine-tuning

- system prompts
- inference settings
- safety tuning
- tool configuration

The evaluation then becomes part of development.

4.6 Public Solution Ecosystems

Popular benchmarks often generate:

- Tutorials
- answer repositories
- evaluation transcripts
- leaderboards
- benchmark-specific prompts
- derivative datasets
- synthetic training corpora

4.7 Benchmark Inflation

Research on retro-holdouts has reported that performance on newly constructed private analogues can reveal gaps not visible on established public benchmarks.[^retro-holdout]

This does not prove that all public benchmark scores are inflated by the same amount.

It supports the narrower conclusion that private comparison sets can reveal measurement weaknesses that public scores alone may miss.

4.8 Unknown Training Data

For closed or extremely large training pipelines, external evaluators may not know whether evaluation content was included.

A holdout created after the relevant training cutoff can reduce some forms of contamination, but it does not eliminate:

- Generalization from related material
- post-training exposure
- developer access
- template familiarity
- model-generated leakage

4.9 Benchmark-Specific Optimization

Even with clean training data, developers can optimize around a known benchmark through repeated evaluation.

The resulting performance may be real, but the benchmark becomes less independent as evidence.

4.10 High-Consequence Thresholds

When a result influences:

- Deployment
- safeguards
- access controls
- insurance
- procurement
- regulation
- national security
- public claims

the cost of a misleading score increases.

Held-out components become more valuable when the consequence of false confidence is high.

---

5. What Held-Out Evaluation Can and Cannot Establish

5.1 What It Can Improve

A well-designed holdout can improve evidence about:

- Generalization
- resistance to direct benchmark optimization
- unseen task performance
- robustness to unfamiliar conditions
- safeguard resilience
- threshold crossing
- independent replication
- capability under controlled conditions

5.2 What It Cannot Prove

A held-out result cannot by itself prove:

- Absence of dangerous capability
- safe deployment
- harmless intent
- robustness across all environments
- absence of contamination
- absence of model awareness

- future performance
- compliance outside the test
- institutional trustworthiness

5.3 Holdout Integrity Is Not Construct Validity

A perfectly secure evaluation can measure the wrong thing.

Security preserves the test as designed.

It does not establish that the design is meaningful.

5.4 Holdout Performance Is Configuration-Specific

The result depends on:

- Model version
- system prompt
- tools
- context
- retries
- compute
- scaffold
- safety layers
- administrator
- date

5.5 Negative Results Require Elicitation Analysis

Failure on a held-out task may mean:

- Capability is absent
- capability is present but poorly elicited
- the task is invalid
- the environment failed
- the scoring is wrong
- the system refused
- the scaffold is inadequate
- the model detected evaluation conditions
- stochasticity affected the run

5.6 Positive Results Require Deployment Analysis

Success under a privileged evaluator configuration may not imply the same capability is available in normal use.

5.7 Complementarity

Held-out evaluation should be combined with:

- Public benchmarks
- dynamic protocols
- expert review
- operational evidence
- incident analysis
- mechanistic evidence
- post-deployment monitoring
- developer disclosure

---

6. Taxonomy of Held-Out Designs

6.1 Fully Private Fixed Test Set

A fixed set remains confidential.

Advantages:

- Simple administration
- stable comparison
- clear chain of custody

Risks:

- Leakage
- saturation
- stale content
- insider optimization
- small task supply

6.2 Rotating Private Test Set

Tasks are replaced periodically.

Advantages:

- Reduced long-term exposure
- adaptation
- compromise recovery

Risks:

- comparability
- cost

- governance burden

6.3 Private Item Bank with Random Sampling

The administrator samples from a larger protected bank.

Advantages:

- Reduced predictability
- repeated administration
- statistical sampling

Risks:

- bank compromise
- uneven forms
- calibration requirements

6.4 Procedurally Generated Holdout

Tasks are generated from protected rules, seeds, or environments.

Advantages:

- Scale
- renewal
- reduced item reuse
- adjustable difficulty

Risks:

- generator leakage
- artificial tasks
- exploitable generator regularities
- validation burden

6.5 Post-Cutoff Evaluation

Tasks are based on information created after the presumed training cutoff.

Advantages:

- Reduces exact pretraining contamination
- supports current-knowledge testing

Risks:

- training cutoff uncertainty
- retrieval access

- recent-data noise
- nonrepresentative sampling

LiveBench and related dynamic approaches illustrate this family of contamination-limited evaluation.[^livebench]

6.6 Event-Sourced Holdout

Tasks are derived from recent:

- Incidents
- competitions
- software changes
- scientific findings
- professional cases
- legal developments

Advantages:

- Current relevance
- authenticity

Risks:

- uneven difficulty
- event bias
- incomplete coverage

6.7 Expert-Authored Confidential Evaluation

Domain experts create protected tasks.

Advantages:

- Domain realism
- nuanced scoring
- high-stakes relevance

Risks:

- cost
- expert bias
- limited scale
- insider leakage
- consistency challenges

The UK AI Security Institute has described structured processes for developing frontier question-answer evaluations, including expert involvement, question design, quality review, and controlled administration.[^aisi-qa]

6.8 Adversarial Holdout

Evaluators create or select attacks intended to expose weaknesses.

Useful for:

- Safeguards
- cyber capability
- model control
- policy compliance
- deception
- robustness

Risk:

- Worst-case findings may be difficult to translate into prevalence or ordinary-use risk.

6.9 Hidden Environment Evaluation

The system operates in an environment with hidden:

- State
- objectives
- tests
- adversaries
- monitoring
- success criteria

Useful for agents.

Risk:

- Environment artifacts can dominate performance.

6.10 Delayed-Release Evaluation

Tasks remain private during active use and become public later.

Advantages:

- Eventual reproducibility
- scientific learning
- public accountability

Risks:

- Future contamination
- sensitive content exposure
- limited value for repeated testing

6.11 Split-Knowledge Evaluation

No single party holds all sensitive components.

For example:

- One party holds tasks
- one holds model access
- one administers infrastructure
- one scores outputs
- one audits logs

Advantages:

- Reduced unilateral control
- stronger compromise resistance

Risks:

- Coordination
- inconsistent responsibility
- more attack surfaces

6.12 Secure Remote Evaluation

The model is evaluated in a controlled environment without transferring model weights or test data broadly.

Advantages:

- Protects both developer IP and evaluator content

Risks:

- Integration difficulty
- limited observability
- infrastructure trust
- attestation complexity

6.13 Developer-Blind, Evaluator-Known

The evaluator knows the tasks, while the developer does not.

Common and practical.

Risk:

- Evaluator compromise.

6.14 Developer-Known, Model-Blind

The developer knows the broad evaluation but the model cannot access protected task metadata or external leakage channels.

Useful in agent sandboxes.

Risk:

- The model may infer the evaluation context.

6.15 Double-Blind Model Comparison

Raters do not know which system produced outputs.

Useful for reducing brand and expectation bias.

It does not address training contamination.

6.16 Hybrid Holdout

A mature protocol may combine:

- Public construct definition
- stable public anchors
- rotating private tasks
- protected attack library
- independent administration
- delayed task release
- post-deployment monitoring

---

7. The Holdout Threat Model

A credible holdout begins with a threat model.

7.1 Assets to Protect

Assets may include:

- Task text
- solutions
- scoring rubrics
- generator code
- sampling seeds
- item difficulty
- attack strategies

- system prompts
- evaluator identities
- thresholds
- model outputs
- vulnerability findings
- access credentials
- environment images
- result timing

7.2 Adversaries and Failure Sources

Threats may come from:

- Malicious outsiders
- insider actors
- compromised vendors
- developers seeking an advantage
- evaluators seeking influence
- accidental disclosure
- model extraction behavior
- insecure tooling
- public repositories
- logs
- backups
- collaboration platforms
- generated reports
- legal discovery
- supply-chain compromise

A threat model should not assume malicious developers by default.

Many failures arise from ordinary operational mistakes.

7.3 Threat Goals

A threat actor may seek to:

- Obtain test content
- infer task families
- alter scoring
- substitute a model
- change configuration
- remove unfavorable runs
- identify thresholds
- manipulate judges
- exfiltrate secrets

- delay publication
- falsely claim a pass
- discredit the evaluation
- expose harmful content

7.4 Attack Surfaces

Potential surfaces include:

- Task-author devices
- email
- cloud storage
- source control
- evaluation harness
- API traffic
- model logs
- sandbox
- human raters
- annotation platform
- report drafting
- backups
- third-party vendors
- remote access
- conference presentations
- retired-task archives

7.5 Model as an Active Participant

Advanced systems may:

- Probe the environment
- inspect files
- infer evaluation context
- exploit tools
- search for hidden data
- manipulate raters
- encode information in outputs
- attempt sandbox escape
- identify scoring vulnerabilities

AISI research has examined what sandboxed agents can infer about evaluation environments and how container isolation should be assessed.[^aisi-sandbox-learning][^aisi-breakout]

7.6 Security Objectives

A held-out system may seek:

- Confidentiality
- integrity
- availability
- provenance
- non-repudiation
- controlled disclosure
- recoverability
- auditability
- least privilege

7.7 Residual Risk

No holdout should be described as perfectly secure.

The threat model should state residual risk and assumptions.

---

8. Design Principles

8.1 Purpose-Limited Confidentiality

Restrict only what must be restricted to preserve validity, safety, privacy, or legitimate proprietary interests.

8.2 Transparent Construct

The capability, risk, or behavior being evaluated should normally be public or reviewable.

8.3 Fair Notice

Participants should know the general domain, administration rules, and decision consequences.

8.4 Independent Verification

A qualified party not controlled by the primary developer should be able to verify material aspects of the process.

8.5 Defense in Depth

Do not rely on one control.

Combine:

- People
- process
- technology
- governance
- renewal

8.6 Least Privilege

Each participant receives only the access needed for their role.

8.7 Separation of Duties

Task creation, administration, scoring, approval, and appeals should be separated where consequence warrants it.

8.8 Provenance

Every task should have a documented origin and modification history.

8.9 Reproducible Administration

The same protocol should be administrable by qualified evaluators under controlled conditions.

8.10 Calibrated Disclosure

Disclosure should be sufficient for accountability without destroying the test.

8.11 Time-Bounded Authority

Held-out results should expire, be renewed, or be retired.

8.12 Compromise Readiness

The protocol should assume that some protected content will eventually leak.

8.13 Accessibility

Security should not unnecessarily exclude smaller qualified evaluators, researchers, or open-source communities.

8.14 Non-Exclusivity

No single private test should become the only accepted measure of a broad capability.

8.15 Contestability

Material errors and procedural failures should be appealable.

8.16 Proportionate Security

Security investment should reflect:

- Consequence
- sensitivity
- replacement cost
- misuse potential
- likelihood of attack
- duration of intended use

8.17 No Security Theater

Complex controls should not substitute for sound evaluation design.

---

9. What Should Be Held Out

A protocol should specify the minimum necessary protected surface.

9.1 Test Items

Protect when exact exposure would enable memorization or targeted tuning.

9.2 Solutions and Rubrics

Protect when answers reveal task construction or enable reverse engineering.

9.3 Sampling Rules

Protect when predictability allows developers to narrow preparation to a small subset.

9.4 Task Generators

Protect when access would enable generation of near-identical training data.

9.5 Attack Libraries

Protect when disclosure would:

- Reduce safeguard-test value
- expose vulnerabilities
- enable misuse
- reveal monitoring gaps

9.6 Thresholds

Threshold confidentiality is controversial.

Potential rationale:

- Prevent cliff-edge gaming
- preserve blinded assessment

Potential harms:

- Reduced fairness
- weak accountability
- arbitrary decision-making

Standards Body position:

Decision thresholds should normally be disclosed at least in principle. Temporary restriction may be justified in narrow cases, but hidden thresholds should not become the default for consequential decisions.

9.7 Environment Details

Some hidden state can test generalization.

Core administration conditions should still be documented.

9.8 Model Identity

Raters may be blinded to model identity to reduce bias.

9.9 Results

Result embargo may be justified for:

- Developer verification
- security remediation
- coordinated disclosure
- legal review

- peer review

Indefinite suppression requires stronger justification.

9.10 Evaluator Identity

Evaluator identity may need protection in sensitive work.

Governance should still verify competence and conflicts.

---

10. Task Development and Provenance

10.1 Task Specification

Each task should include:

- Identifier
- construct mapping
- domain
- difficulty
- author
- creation date
- source
- expected solution
- scoring
- validation status
- security classification
- contamination risk
- retirement status

10.2 Source Categories

Tasks may come from:

- Original expert authorship
- public sources transformed after cutoff
- real incidents
- professional workflows
- simulations
- procedural generators
- controlled data partnerships
- adversarial discovery
- user research
- synthetic generation with expert validation

10.3 Provenance Risks

Risks include:

- Copyright
- personal data
- trade secrets
- dual-use content
- hidden public duplicates
- author conflict
- incorrect solutions
- unverifiable source
- contaminated generation model

10.4 Model-Generated Tasks

Models can help generate tasks, but the protocol should record:

- Generator model
- version
- prompts
- source material
- human review
- filtering
- contamination concerns
- validation

A model trained on public benchmarks may reproduce benchmark-like tasks or solutions.

10.5 Expert Validation

Expert review should assess:

- Correctness
- relevance
- ambiguity
- realism
- difficulty
- safety
- scoring
- construct coverage

10.6 Independent Replication of Solutions

High-consequence tasks should have more than one qualified solution review where feasible.

10.7 Task Families

Organize tasks into families so coverage can be evaluated.

10.8 Item Exposure Accounting

Each administration should update:

- Exposure count
- parties with access
- models tested
- transcripts generated
- disclosure events
- estimated residual value

10.9 Retirement Conditions

Retire a task when:

- It leaks
- it saturates
- its solution becomes obsolete
- its environment changes
- its validity fails
- exposure becomes excessive
- safety risk changes
- it is superseded

---

11. Statistical Design and Sampling

11.1 The Task Population

A held-out set should represent a defined population, not merely a collection of difficult questions.

11.2 Sampling Plan

Specify whether sampling is:

- Random
- stratified
- adaptive
- risk-weighted
- difficulty-balanced
- scenario-based

- purposive
- exhaustive

11.3 Sample Size

Sample size should reflect:

- Expected variance
- desired precision
- threshold consequence
- task cost
- model stochasticity
- subgroup analysis
- failure rarity

11.4 Repeated Runs

Repeated runs may estimate:

- Reliability
- success probability
- stochastic variance
- sensitivity to prompts
- tail behavior

They also increase exposure.

11.5 Form Equivalence

If different participants receive different forms, the protocol should assess:

- Difficulty equivalence
- coverage
- scoring
- subgroup effects

11.6 Threshold Evaluation

Near a threshold:

- Increase sample size
- use independent replication
- report confidence
- avoid deterministic classification where evidence is weak
- consider conservative interim status

11.7 Human Baselines

Human comparison should specify:

- Expertise
- tools
- time
- compensation
- selection
- sample size
- scoring
- assistance

11.8 Missing Data

Define treatment of:

- Timeouts
- refusals
- infrastructure failures
- invalid tasks
- missing logs
- aborted runs

11.9 Item-Level Records

NIST has emphasized the value of richer item-level analysis and statistical modeling because aggregate benchmark scores can hide uncertainty and assumptions.[^nist-statistical]

Held-out systems should retain item-level evidence under appropriate controls.

11.10 Avoiding Unjustified Precision

A private score is not more precise merely because the tasks are secret.

---

12. Security Architecture

12.1 Security Classification

Each component should be classified by sensitivity.

Example levels:

- Public
- internal

- controlled research
- confidential evaluation
- restricted security
- highly restricted

12.2 Identity and Access Management

Controls may include:

- Named accounts
- multi-factor authentication
- role-based access
- time-limited credentials
- device requirements
- geographic restrictions
- approval workflows
- periodic access review

12.3 Data at Rest

Use:

- Encryption
- key management
- backup policy
- access logging
- retention controls
- secure deletion
- compartmentalization

12.4 Data in Transit

Use:

- Authenticated encrypted channels
- endpoint verification
- transfer logging
- restricted export

12.5 Data in Use

Conventional encryption does not protect data while actively processed.

Confidential computing uses hardware-based, attested trusted execution environments to protect data in use.[^ccc-technical]

Potential use cases include:

- Running protected tasks against proprietary models
- limiting administrator visibility
- producing attestable execution records
- supporting multi-party evaluation

Limitations include:

- Hardware trust assumptions
- side channels
- implementation vulnerabilities
- limited accelerator support
- operational complexity
- attestation interpretation
- cost

12.6 Sandboxing

Agentic evaluations may require isolated environments.

AISI has released sandboxing tools for Inspect and has emphasized that sandbox selection should match the threat model.[^aisi-sandbox]

A sandbox is not automatically secure.

Validate:

- Isolation boundary
- network policy
- filesystem
- credentials
- host access
- logging
- escape resistance
- cleanup

12.7 Secure Logging

Logs should support:

- Reproducibility
- investigation
- scoring
- anomaly detection
- appeals

Logs can themselves expose tasks or sensitive model behavior.

12.8 Endpoint Security

Task-author and evaluator devices are common weak points.

12.9 Vendor Risk

Third-party platforms should be assessed for:

- Data access
- subcontractors
- training use
- retention
- breach notification
- jurisdiction
- deletion
- model access

12.10 Backup and Recovery

A secure system should recover from:

- Data loss
- ransomware
- accidental deletion
- corrupted task bank
- key loss
- service outage

12.11 Destruction

Secure deletion may be appropriate for:

- Temporary exports
- expired access packages
- sensitive intermediate files

Permanent destruction should not eliminate necessary audit records.

---

13. Access-Control Models

13.1 Tier 0: Public Access

Available:

- Construct

- methodology
- governance
- summary results
- limitations
- version history

13.2 Tier 1: Registered Research Access

Available:

- Task taxonomy
- selected retired items
- validation materials
- limited reference data

13.3 Tier 2: Qualified Evaluator Access

Available:

- Active tasks
- scoring tools
- administration environment
- controlled logs

Requirements:

- Competence
- conflict disclosure
- security controls
- agreement
- monitoring

13.4 Tier 3: Domain-Sensitive Access

For:

- Cyber
- biology
- critical infrastructure
- other dual-use domains

Additional requirements:

- Domain qualification
- stronger security
- need-to-know
- incident obligations

- legal review

13.5 Tier 4: Custodian Access

Limited personnel with authority over:

- Full task bank
- keys
- generators
- compromise response
- archival systems

13.6 Developer Access

Developers may receive:

- Broad scope
- interface
- configuration requirements
- output format
- process rights
- limited debugging support

They should not receive active items unless the protocol explicitly requires collaborative testing.

13.7 Temporary Debug Access

Integration failures sometimes require developer visibility.

Use:

- Synthetic tasks
- public shadow tasks
- retired items
- narrowly scoped traces
- supervised sessions

Avoid exposing active holdouts merely to solve ordinary integration problems.

13.8 External Researcher Access

Structured external access can improve critique and innovation.

Recent work on external access distinguishes dimensions such as model access, information access, and evaluation time as separate determinants of evaluator effectiveness.[^external-access]

---

14. Chain of Custody

14.1 Required Events

Record:

- Creation
- review
- approval
- classification
- storage
- transfer
- access
- modification
- administration
- scoring
- export
- publication
- retirement
- destruction

14.2 Minimum Record

Each event should include:

- Person or system
- date and time
- purpose
- material
- action
- authorization
- location
- integrity check
- anomaly

14.3 Task Packaging

Evaluation packages should have:

- Unique identifiers
- hashes
- version
- manifest
- expected files
- signature or authenticated provenance

14.4 Pre-Administration Verification

Confirm:

- Correct model
- correct protocol version
- correct environment
- intact task package
- approved access
- logging
- time synchronization
- no unauthorized files

14.5 Post-Administration Verification

Confirm:

- Complete logs
- task integrity
- scoring package
- deviations
- data transfer
- cleanup
- access revocation

14.6 Separation of Evidence

Preserve:

- Raw outputs
- scored outputs
- adjudicated results
- final report

Do not overwrite original evidence.

---

15. Administration Protocol

15.1 Pre-Registration

Before seeing results, record:

- Scope
- configuration
- tasks or sampling method

- scoring
- thresholds
- exclusion rules
- retry policy
- adjudication
- publication plan

The exact tasks may remain private.

15.2 Model Verification

Confirm the evaluated system identity through available methods.

15.3 Configuration Lock

Document and, where possible, lock:

- System prompt
- tools
- sampling
- context
- safety settings
- model endpoint
- scaffold
- compute budget

15.4 Dry Run

Use non-active tasks to test integration.

15.5 Active Run

Apply:

- Authorized personnel
- monitoring
- incident procedures
- deviation logging
- access controls

15.6 Transcript Review

NIST guidance on evaluation cheating emphasizes transcript review as a way to detect cheating, integration problems, tool issues, and task failures.[^nist-cheating]

Transcript review should combine:

- Automated screening
- targeted human review
- anomaly investigation
- documented adjudication

15.7 Scoring

Use the pre-specified method.

Any post-hoc scoring change should be recorded and justified.

15.8 Adjudication

Ambiguous cases may require:

- Second rater
- domain expert
- blind review
- appeals panel

15.9 Developer Notification

Before public release, the developer may be given a limited period to identify:

- Configuration errors
- factual errors
- security concerns
- proprietary disclosures

The developer should not receive veto authority over unfavorable findings.

15.10 Publication

Publish according to the disclosure plan.

---

16. Scoring, Uncertainty, and Reporting

16.1 Score Types

Possible outputs include:

- Accuracy
- success probability

- severity-weighted failure
- pass band
- capability tier
- safeguard bypass rate
- time-to-completion
- cost
- calibrated confidence
- qualitative finding

16.2 Hidden Scoring Rules

Hidden scoring rules can reduce gaming but weaken fairness.

Preferred approach:

- Public scoring principles
- protected item-specific answers
- protected adversarial details where justified

16.3 Model Judges

Model judges may reduce cost.

They require validation for:

- Bias
- consistency
- sensitivity
- model-family favoritism
- prompt injection
- manipulation
- domain competence

16.4 Human Raters

Human scoring should document:

- Qualification
- training
- rubric
- blinding
- agreement
- compensation
- conflicts

16.5 Uncertainty Sources

Include:

- Sampling
- stochasticity
- scoring
- task ambiguity
- configuration
- environment
- missing data
- contamination
- judge error

16.6 Result Bands

Use bands or ranges when point estimates imply unjustified certainty.

16.7 Public Reporting Minimum

A public report should include:

- Protocol name
- version
- date
- evaluator
- system
- configuration
- construct
- task categories
- sampling
- scoring
- uncertainty
- limitations
- deviations
- conflicts
- result status
- expiration
- disclosure restrictions

16.8 Private Technical Annex

A controlled annex may include:

- Item-level results
- transcripts

- vulnerabilities
- scorer details
- task provenance
- security incidents

16.9 Claims Boundary

State what the result cannot establish.

---

17. Reproducibility Without Full Disclosure

17.1 Exact Public Reproduction

Useful but not always compatible with a live holdout.

17.2 Controlled Reproduction

A qualified independent evaluator reruns the protocol under controlled access.

17.3 Method Reproduction

Researchers recreate the method using different tasks.

17.4 Result Verification

An auditor verifies:

- Model identity
- task integrity
- execution
- scoring
- report

without receiving unrestricted task access.

17.5 Delayed Reproduction

Tasks are released after retirement.

17.6 Cryptographic Verification

Potential tools include:

- Hash commitments

- signed manifests
- timestamping
- attestation
- reproducible build records
- tamper-evident logs

These can show that materials were not changed after commitment.

They do not establish:

- Task validity
- fairness
- correct interpretation
- absence of hidden selection bias

17.7 Multi-Party Verification

Several institutions can independently verify parts of the process.

17.8 Replication Package

A controlled replication package can include:

- Protocol
- environment
- scoring code
- public sample tasks
- retired tasks
- access process
- validation report
- expected behavior

---

18. Transparency and Confidentiality

18.1 False Choice

The choice is not between total secrecy and total openness.

A protocol can protect test content while disclosing:

- Purpose
- construct
- governance
- funding
- methods

- access policy
- validation
- limitations
- change history
- result summary

18.2 Transparency Layers

Layer A: Constitutional Transparency

Why the holdout exists and what principles govern it.

Layer B: Methodological Transparency

How tasks are created, sampled, scored, and validated.

Layer C: Procedural Transparency

Who administers, reviews, approves, and hears appeals.

Layer D: Evidentiary Transparency

What evidence supports the result.

Layer E: Content Transparency

The exact tasks, solutions, and attack methods.

Content transparency may be delayed or restricted.

18.3 Justification Record

For every restricted category, record:

- What is restricted
- why
- authority
- duration
- who can review
- disclosure trigger
- appeal

18.4 Public Summary of Restricted Findings

When detailed disclosure is unsafe, publish:

- High-level finding
- evidence category
- confidence

- decision relevance
- mitigation status
- independent-review status

18.5 Transparency Failure

A protocol fails transparency when outsiders cannot distinguish:

- Strong evidence from assertion
- valid security restrictions from convenience
- independent review from internal approval
- current results from expired results

---

19. Fairness and Access

19.1 Procedural Fairness

Participants should receive consistent:

- Rules
- timelines
- configuration requirements
- scoring
- appeal rights
- confidentiality
- publication treatment

19.2 Unequal Resources

Large laboratories may have:

- Better elicitation teams
- more compute
- direct evaluator relationships
- faster integration
- specialized legal and security staff

The protocol should report developer assistance and resource differences.

19.3 Open-Source and Open-Weight Participation

Open communities may lack one legal entity or centralized developer.

Possible approaches:

- Community-nominated representative
- independent evaluator runs
- public configuration track
- controlled access without privileged developer support
- transparent asymmetry statement
- subsidized testing

Open-source representation should not be symbolic.

Experts should have meaningful participation in:

- Construct design
- task review
- access policy
- result interpretation
- governance

19.4 Small Evaluator Participation

Security requirements can unintentionally create a closed market.

Options:

- Shared secure facilities
- standardized environments
- grants
- tiered accreditation
- supervised access
- federated evaluation
- pooled infrastructure

19.5 Geographic Access

International participation may be limited by:

- Export controls
- privacy law
- security clearance
- sanctions
- data localization
- language
- funding

These constraints should be explicit.

19.6 Disability and Language

Task design should distinguish the intended construct from irrelevant accessibility barriers.

19.7 Fairness Does Not Mean Identical Treatment

Different systems may require different integration.

The reasons should be documented.

---

20. Developer and Evaluator Relationship

20.1 Necessary Cooperation

Developers may need to provide:

- Model access
- configuration
- safety information
- technical support
- system limitations
- training cutoff
- tool documentation
- risk analysis

20.2 Independence Risks

Dependence can arise through:

- Funding
- access
- infrastructure
- publication approval
- legal restrictions
- repeated contracts
- personnel relationships

20.3 Recommended Agreement Terms

Cover:

- Scope
- access
- confidentiality
- data handling

- evaluator independence
- publication
- developer review period
- no result veto
- incident handling
- IP
- liability
- termination
- audit
- dispute

20.4 Developer Challenge Process

Developers should be able to present evidence concerning:

- Configuration
- invalid tasks
- scoring
- environment failures
- interpretation
- security

20.5 No Benchmark Coaching

Evaluator support should solve integration problems without teaching active test content.

20.6 External Testing

OpenAI has described structured external testing and subject-matter-expert probing as complements to internal Preparedness Framework evaluations.[^openai-external]

Google DeepMind's early-warning proposal similarly envisioned structured access for external safety researchers and auditors.[^deepmind-warning]

Anthropic's 2026 Responsible Scaling Policy revisions formalized external review under specified governance conditions.[^anthropic-rsp]

These examples reflect a broader movement toward combining developer evaluation with external scrutiny. They do not yet establish a common external-access standard.

---

21. Governance

21.1 Governing Principle

The institution controlling the holdout should not possess unreviewable authority over:

- Task selection
- scoring
- result interpretation
- disclosure
- appeals
- compromise determination

21.2 Roles

A mature system may include:

- Protocol sponsor
- task-development lead
- domain panel
- security custodian
- evaluation administrator
- scoring team
- independent review panel
- appeals panel
- archive custodian
- public-interest representative
- open-source representative
- developer liaison

21.3 Separation of Duties

For high-consequence evaluations:

- Task author should not unilaterally score
- administrator should not unilaterally approve
- funder should not control publication
- developer should not control active tasks
- security custodian should not determine scientific validity alone

21.4 Conflict Disclosure

Disclose:

- Employment
- consulting

- funding
- equity
- research collaboration
- competitive interest
- policy advocacy
- prior disputes
- intellectual commitments

21.5 Recusal

Material conflicts may require recusal.

21.6 Dissent

Allow:

- Minority technical reports
- unresolved issue register
- alternative interpretations
- publication of disagreement where security permits

21.7 Appeals

Appeals should be:

- Time-bounded
- evidence-based
- reviewed by qualified people not responsible for the original decision
- documented
- capable of correcting error

21.8 Funding

Funding sources should be disclosed.

No payer should obtain:

- Task access beyond role
- result suppression
- favorable scoring
- unilateral change authority
- exclusive ownership of a public-interest protocol

21.9 Oversight

Oversight can include:

- Board committee
- independent council
- public auditor
- government institute
- multi-institution consortium
- accreditation body

21.10 Governance Review

Review governance after:

- Major leak
- disputed result
- funding change
- personnel concentration
- protocol expansion
- international adoption

---

22. Compromise Detection and Response

22.1 Signs of Possible Compromise

Indicators include:

- Unexplained performance jump
- exact solution reproduction
- suspicious phrasing
- benchmark-specific behavior
- unusual access logs
- unauthorized downloads
- model probing of hidden files
- leaked task fragments
- employee report
- public appearance of content
- hash mismatch
- missing logs
- changed scoring code

22.2 Detection Methods

Possible methods:

- Transcript review
- similarity analysis
- canary tasks
- access analytics
- file-integrity monitoring
- red-team testing
- audit
- developer disclosure
- environment monitoring
- controlled replication

22.3 Initial Response

- Preserve evidence
- restrict access
- pause administration if necessary
- notify security and protocol owners
- assess scope
- avoid premature public accusation

22.4 Investigation

Determine:

- What was exposed
- when
- to whom
- how
- which results are affected
- whether scoring changed
- whether the model may have encountered content
- whether the task bank remains usable

22.5 Result Status

Possible statuses:

- Valid
- valid with caveat
- under review
- suspended
- withdrawn

- superseded
- invalid

22.6 Remediation

Options:

- Replace items
- rotate bank
- revise generator
- re-evaluate
- change access
- patch infrastructure
- retrain personnel
- disclose incident
- retire protocol

22.7 Public Notice

A public notice should state enough to prevent continued misuse of invalid results.

22.8 Learning Review

Every material compromise should generate:

- Root-cause analysis
- corrective action
- governance review
- update to `FAILURE_DATABASE.md`
- update to `VERSION_HISTORY.md`

---

23. Lifecycle of a Held-Out Evaluation

Stage 1: Need Identification

Determine why public evaluation is insufficient.

Output:

- Holdout justification

Stage 2: Threat Modeling

Identify:

- Assets
- adversaries
- exposure paths
- security objectives
- residual risk

Output:

- Threat model

Stage 3: Construct Design

Define:

- Capability
- subdomains
- task universe
- decision use
- exclusions

Output:

- Construct map

Stage 4: Governance Charter

Define:

- Owner
- roles
- conflicts
- access
- appeals
- publication
- funding

Output:

- Charter

Stage 5: Task Development

Create:

- Items

- generators
- environments
- scoring
- provenance

Output:

- Candidate bank

Stage 6: Validation

Assess:

- Correctness
- difficulty
- coverage
- reliability
- fairness
- security
- contamination

Output:

- Validation report

Stage 7: Security Build

Implement:

- Classification
- access
- storage
- execution
- logging
- incident response

Output:

- Security plan

Stage 8: Pilot

Run on:

- Reference systems
- human baselines
- controlled configurations

Output:

- Pilot report

Stage 9: Independent Review

Review:

- Science
- security
- fairness
- governance
- interpretation

Output:

- Review and response

Stage 10: Approval

Authorize a version for defined use.

Stage 11: Administration

Conduct controlled evaluation.

Stage 12: Reporting

Issue public and restricted outputs.

Stage 13: Monitoring

Track:

- Exposure
- leakage
- saturation
- validity
- disputes
- cost
- result use

Stage 14: Renewal

Add, replace, or recalibrate tasks.

Stage 15: Retirement

Withdraw active authority and archive appropriately.

---

24. Specialized Domains

24.1 Cybersecurity

Held-out cyber evaluations may protect:

- Vulnerabilities
- target environments
- credentials
- attack chains
- scoring triggers
- defensive configurations

Requirements may include:

- Sandboxing
- network isolation
- safe targets
- logging
- exploit handling
- responsible disclosure
- domain experts

AISI's Inspect Cyber work provides standardized infrastructure for agentic cyber evaluations. [^inspect-cyber]

24.2 Biology and CBRN

Held-out biological evaluation may involve:

- Sensitive knowledge
- expert judgment
- safe proxies
- access controls
- staged disclosure
- institutional biosafety review

Task design should avoid creating harmful operational guidance unnecessarily.

NIST work on AI-assisted protein design has used safe biological proxies as part of a TEVV approach, illustrating how evaluation design can reduce experimental risk while investigating relevant capability.[^nist-protein]

24.3 Autonomous Agents

Held-out agent evaluation may protect:

- Environment state
- objectives
- monitoring
- credentials
- adversarial events
- task sequence

Security should account for active model attempts to inspect or escape the environment.

24.4 Safeguard Evaluation

Protecting attack methods can preserve test value and reduce misuse.

AISI's principles for evaluating misuse safeguards emphasize threat modeling, representative attacks, and careful interpretation.[^aisi-safeguards]

24.5 Persuasion and Human Interaction

Tasks may involve human subjects, private conversations, or experimental conditions.

Additional requirements:

- Consent
- privacy
- ethics review
- de-identification
- secure retention
- human-risk monitoring

24.6 Critical Infrastructure

Evaluation may require simulated or digital-twin environments rather than live systems.

24.7 Scientific Capability

Held-out tasks should distinguish:

- Knowledge recall
- hypothesis quality
- experimental design
- tool use
- result interpretation
- real-world validation

25. Technical Options for Protected Evaluation

25.1 Secure API Evaluation

Evaluator sends tasks to a controlled model endpoint.

Benefits:

- Developer retains model control
- limited integration

Risks:

- Developer may observe tasks
- endpoint may differ from evaluated model
- logs may leak content

25.2 Evaluator-Controlled Deployment

Model is deployed in evaluator infrastructure.

Benefits:

- Strong evaluator control
- richer observability

Risks:

- Weight transfer
- IP
- security
- hardware requirements

25.3 Joint Secure Facility

Developer and evaluator operate within a controlled facility.

Benefits:

- Shared trust
- direct troubleshooting

Risks:

- Cost
- scheduling
- personnel access

25.4 Trusted Execution Environment

Protected computation can reduce the need for either party to expose all assets.

Attestation can help verify the environment.[^ccc-attestation]

Limitations must be assessed case by case.

25.5 Cryptographic Commitments

Commit to:

- Task bank
- sampling seed
- scoring code
- threshold
- report draft

before administration or disclosure.

25.6 Secure Multi-Party Computation

Potentially allows computation across parties without revealing all inputs.

Current practicality depends on workload and implementation.

25.7 Homomorphic Encryption

Potentially enables computation on encrypted data.

It may be too costly or limited for many frontier model evaluations.

25.8 Federated Evaluation

Tasks remain with custodians while systems or outputs move through controlled interfaces.

25.9 Remote Desktop or Clean Room

Authorized personnel work in a monitored environment without unrestricted export.

25.10 Air-Gapped Evaluation

Useful for highly sensitive content, but operationally expensive and not immune to insider risk.

25.11 Technical Neutrality

Standards Body should specify desired security properties rather than prematurely mandating one technology.

---

26. International Interoperability

26.1 Cross-Border Challenge

Evaluation materials may cross:

- Legal regimes
- privacy systems
- export controls
- security classifications
- languages
- professional standards

26.2 Shared Metadata

Institutions should align on:

- Protocol identifier
- version
- construct
- task category
- access tier
- evaluator qualification
- result status
- compromise status
- expiration
- confidence

26.3 Mutual Recognition

One institution may recognize another's result when there is confidence in:

- Competence
- security
- governance
- methodology
- administration
- reporting
- appeals

26.4 Distributed Custody

Different national institutes may hold different task subsets.

Advantages:

- Reduced single-point compromise
- regional expertise
- resilience

Risks:

- Form equivalence
- geopolitical distrust
- inconsistent access
- coordination

26.5 Shared Retired Banks

Retired tasks can support international research without exposing active holdouts.

26.6 No Forced Centralization

Interoperability should not require one global task bank controlled by one institution.

26.7 Institute Cooperation

International AI safety and security institutes have increasingly collaborated on evaluation practices and research. Such cooperation can support shared methods while preserving national authority.

---

27. Maturity Model

Level 0: Informal Private Test

Characteristics:

- Private questions
- no formal governance
- weak provenance
- no access record
- no compromise plan

Use:

- Internal exploration only

Level 1: Documented Holdout

Characteristics:

- Defined purpose
- named owner
- task records
- basic access controls
- versioning
- scoring

Use:

- Research validation

Level 2: Controlled Held-Out Protocol

Characteristics:

- Threat model
- chain of custody
- validated tasks
- reproducible administration
- uncertainty
- result expiration
- compromise response

Use:

- Serious organizational decisions

Level 3: Independently Reviewed Secure Evaluation

Characteristics:

- Separation of duties
- independent review
- formal security
- appeals
- controlled replication
- rotating task bank
- public methodology

Use:

- High-stakes pre-deployment decisions

Level 4: Interoperable Evaluation Ecosystem

Characteristics:

- Multiple qualified evaluators
- shared metadata
- mutual recognition
- accreditation
- international coordination
- distributed custody
- incident exchange
- formal retirement

Use:

- Mature standards and conformity-assessment ecosystem

---

28. Implementation Pathway

Phase 1: Select the Decision

Start with a specific question.

Example:

"Does this system demonstrate a defined level of long-horizon cyber capability under controlled tool access?"

Phase 2: Justify the Holdout

Explain why public tests are insufficient.

Phase 3: Build the Construct Map

Define subcapabilities and exclusions.

Phase 4: Create Governance

Assign roles, conflicts, access, and appeals.

Phase 5: Develop the Threat Model

Identify leakage and active-agent risks.

Phase 6: Build the Task Bank

Use multiple task sources.

Phase 7: Validate

Test tasks with experts, reference models, and human baselines.

Phase 8: Build Secure Infrastructure

Implement access, logging, packaging, and incident response.

Phase 9: Pilot

Run on non-consequential reference systems first.

Phase 10: Independent Review

Review science and security separately.

Phase 11: Launch

Publish methodology and access rules.

Phase 12: Monitor Exposure

Track every use.

Phase 13: Rotate

Replace tasks based on exposure and evidence.

Phase 14: Audit the Holdout

Evaluate whether the holdout remains worth its cost and authority.

---

29. Proposed Standards Body Pilot

29.1 Candidate Pilot

Held-Out Evaluation Protocol for Long-Horizon Technical Agent Tasks

This would complement the pilot proposed in Foundation 1.

29.2 Purpose

Measure whether AI agents can complete unfamiliar, verifiable technical tasks without prior access to the active task set.

29.3 Public Components

- Construct
- task taxonomy
- administration rules
- scoring principles
- governance
- result format
- limitations

29.4 Held-Out Components

- Active tasks
- reference solutions
- sampling seed
- selected environment details
- adversarial variants

29.5 Task Sources

- Expert-authored software tasks
- newly created repositories
- procedural variants
- recent dependency changes
- incident-derived repair tasks

29.6 Security

- Controlled repository access
- isolated execution
- no public network unless required
- signed task packages
- complete logging
- task exposure accounting

29.7 Evaluation Tracks

Standard Track

Common evaluator-defined scaffold.

Developer-Elicited Track

Developer proposes configuration without task access.

Independent Optimization Track

Evaluator attempts stronger elicitation.

29.8 Outputs

- Protocol
- threat model
- task-development manual
- security plan
- pilot results
- exposure ledger
- compromise exercise
- public methodology
- controlled replication package

29.9 Success Criteria

- Low evidence of contamination
- reliable scoring
- useful difficulty spread
- reproducible administration
- fair integration
- manageable cost
- credible external review
- clear result limits

---

30. Metrics for Evaluating the Holdout System

30.1 Integrity Metrics

- Unauthorized access events
- leaked items
- hash mismatches
- unexplained task changes
- compromised results
- access-review findings

30.2 Measurement Metrics

- Discrimination
- reliability
- coverage
- uncertainty
- human agreement
- external validity
- public-private score gap

30.3 Exposure Metrics

- Number of administrations
- number of viewers
- model families tested
- transcript copies
- retired-item rate
- residual item value

30.4 Operational Metrics

- Cost
- administration time
- integration failure
- evaluator burden
- security burden
- task-development throughput

30.5 Fairness Metrics

- Access approvals
- approval time
- participation by smaller actors
- geographic participation
- appeals
- form-equivalence findings

30.6 Governance Metrics

- Conflict disclosures
- recusals
- independent reviews
- unresolved dissent
- funder concentration
- change-record completeness

30.7 Decision Utility

- Decisions informed
- errors corrected
- false confidence avoided
- correlation with later evidence
- result misuse
- user understanding

---

31. Failure Modes and Safeguards

31.1 Secret but Invalid

Failure: The test is confidential but poorly designed.

Safeguard: Independent validity review and public construct explanation.

31.2 Single-Point Custodian

Failure: One person controls the full bank and scoring.

Safeguard: Separation of duties and access logging.

31.3 Permanent Holdout

Failure: The same private set is reused until it becomes stale or informally known.

Safeguard: Exposure limits, rotation, expiration.

31.4 Security as Authority

Failure: Maintainers dismiss criticism by citing confidentiality.

Safeguard: Controlled independent audit and public methodological disclosure.

31.5 Developer Capture

Failure: Access dependence gives the developer control over findings.

Safeguard: Contractual independence, diversified access, no result veto.

31.6 Evaluator Capture

Failure: A private evaluator benefits from making its test indispensable.

Safeguard: Multiple evaluators, protocol registry, interoperability, review.

31.7 Unequal Form Difficulty

Failure: Participants receive materially different tests.

Safeguard: Calibration, bridging, reference systems, uncertainty.

31.8 Task-Bank Leakage

Failure: Active content enters public or training systems.

Safeguard: Compromise plan, rotation, investigation, result withdrawal.

31.9 Generator Leakage

Failure: Protected task generator becomes a training target.

Safeguard: Generator versioning, defense in depth, alternative generators.

31.10 Hidden Threshold Manipulation

Failure: Decision thresholds move after results are observed.

Safeguard: Pre-commitment, governance, change records.

31.11 Overclaiming

Failure: Passing a private test is described as proof of safety.

Safeguard: Claims boundary and complementary evidence.

31.12 Under-Elicitation

Failure: The evaluator misses capability because of weak configuration.

Safeguard: Multiple elicitation tracks and developer input without task disclosure.

31.13 Model Awareness

Failure: The model detects evaluation and changes behavior.

Safeguard: varied contexts, environment analysis, monitoring, operational evidence.

31.14 Insider Threat

Failure: Authorized person leaks or modifies content.

Safeguard: least privilege, logging, dual control, monitoring, culture.

31.15 Excessive Cost

Failure: Security and task renewal become unsustainable.

Safeguard: proportionality, shared infrastructure, prioritization.

31.16 Exclusion of Open Communities

Failure: Only large corporations can access evaluation.

Safeguard: supervised access, subsidies, shared facilities, representative governance.

31.17 Harmful Evaluation Content

Failure: The evaluation itself creates misuse or safety risk.

Safeguard: domain review, safe proxies, sandboxing, restricted disclosure.

31.18 Retired-Task Mishandling

Failure: Old tasks are released in ways that expose active methods or sensitive content.

Safeguard: retirement review and staged disclosure.

---

32. Serious Objections

Objection 1: Science Requires Open Tests

Open access supports replication and criticism.

This objection is strong.

Response:

- Publish methodology
- release retired tasks
- enable controlled replication
- use public anchors
- permit independent audit
- document governance

Residual concern:

Some scientific scrutiny will remain weaker while active tasks are private.

Objection 2: Secret Evaluations Are Unaccountable

They can become arbitrary instruments of power.

Response:

- Fair notice
- public construct
- independent review
- pre-registered scoring
- appeals
- change records
- conflict disclosure

Residual concern:

Information asymmetry cannot be eliminated.

Objection 3: Holdouts Will Leak Eventually

Often true.

Response:

Design for rotation, compromise detection, and recovery.

A holdout should be renewable infrastructure, not a permanent vault.

Objection 4: Developers Need Tasks to Debug Integration

Response:

Use public shadow tasks, retired examples, synthetic tasks, and supervised troubleshooting.

Residual concern:

Some active-task exposure may occasionally be necessary. It should be recorded and those items reconsidered.

Objection 5: Private Tests Advantage Insiders

Response:

Use formal access policy, exposure records, evaluator rotation, and independent governance.

Residual concern:

People close to the evaluation ecosystem may still learn its style.

Objection 6: A New Private Set Can Still Resemble Public Training Data

Correct.

Response:

Use provenance, post-cutoff sourcing, expert creation, procedural generation, and contamination analysis.

Residual concern:

Proving complete non-exposure is difficult.

Objection 7: Holdouts Encourage Test-Centric Governance

Response:

Require complementary operational and contextual evidence.

Objection 8: Held-Out Tests Are Too Expensive

Response:

Use them selectively where public exposure materially weakens high-value decisions.

Objection 9: Cryptographic and Secure-Compute Solutions Are Premature

Often true for complex workloads.

Response:

Specify security properties, pilot technologies, and avoid dependence on one mechanism.

Objection 10: Developers Can Train Against the General Domain Anyway

Yes.

A holdout does not prevent legitimate domain improvement.

Its purpose is to test generalization beyond exact known content, not to stop learning.

Objection 11: Confidential Results Can Be Misrepresented

Response:

Public reporting minimums, independent verification, and result status.

Objection 12: Open-Weight Models Cannot Receive the Same Developer Support

Response:

Report the difference and create standardized community participation paths.

---

33. Evidence Gaps

33.1 Magnitude of Contamination Effects

How much do different forms of contamination inflate different benchmark results?

33.2 Detection

Which detection methods work without access to training data?

33.3 Retro-Holdout Validity

When do newly created analogues measure the same construct as older public tests?

33.4 Task Exposure

How many administrations can a task tolerate before its value declines?

33.5 Model Awareness

Can systems reliably detect controlled evaluation contexts and alter behavior?

33.6 Controlled Replication

Which institutional models produce credible replication without broad disclosure?

33.7 Secure Compute

When do trusted execution environments and related methods provide practical assurance for frontier evaluation?

33.8 Fair Access

How can smaller evaluators and open communities participate without weakening security?

33.9 Threshold Fairness

What procedural protections are required when held-out results influence deployment or legal status?

33.10 Real-World Predictive Value

Do held-out results predict deployment behavior better than public benchmarks?

33.11 Evaluator Incentives

How do funding and access relationships affect conclusions?

33.12 International Recognition

What evidence should support mutual recognition of confidential results?

---

34. Research Agenda

Priority 1: Contamination Measurement

Compare:

- Exact exposure
- semantic exposure
- template exposure
- repeated benchmark optimization
- post-training exposure

Priority 2: Holdout Construction

Study:

- Expert-authored
- post-cutoff
- procedural
- event-sourced
- adversarial
- hybrid methods

Priority 3: Retro-Holdouts

Develop methods for testing whether a new private set is genuinely comparable to an older public benchmark.

Priority 4: Exposure Accounting

Model task value as a function of:

- Administrations
- viewers
- model families
- time
- transcript retention
- public derivatives

Priority 5: Secure Evaluation Infrastructure

Pilot:

- TEEs
- attestation
- secure APIs
- joint facilities
- federated evaluation
- cryptographic commitments

Priority 6: Controlled Reproducibility

Compare:

- Independent reruns
- retired-task release
- method replication
- multi-party verification

Priority 7: Model Awareness and Cheating

Develop methods to detect:

- Environment probing
- judge manipulation
- hidden-data search
- sandbox escape
- strategic underperformance
- test-specific behavior

Priority 8: Fairness

Study access models for:

- Small labs
- academia
- public-interest groups
- open-source communities
- international evaluators

Priority 9: Governance

Pilot:

- Distributed custody
- appeals
- public-interest oversight
- evaluator rotation
- funding safeguards

Priority 10: Decision Utility

Measure whether held-out evidence improves actual deployment and policy decisions.

---

35. Near-Term Experiments

Experiment 1: Public Versus Held-Out Performance

Create matched public and private task forms.

Compare:

- Performance
- ranking
- confidence
- error type
- model-family effects

Experiment 2: Exposure Decay

Administer a task bank repeatedly and measure whether performance changes after controlled exposure.

Experiment 3: Retro-Holdout Validation

Construct a private analogue of a known benchmark and test construct equivalence.

Experiment 4: Generator Security

Compare public and protected procedural generators.

Experiment 5: Controlled Replication

Have two independent evaluators administer the same protected protocol.

Experiment 6: Elicitation Tracks

Compare standard, developer-elicited, and evaluator-optimized performance without exposing active tasks.

Experiment 7: Attested Execution

Pilot a trusted execution environment for a limited evaluation workload.

Experiment 8: Compromise Drill

Simulate a task-bank leak and test response time, result-status changes, and replacement.

Experiment 9: Open-Source Participation

Create a supervised evaluation path for an open-weight model community.

Experiment 10: Delayed Disclosure

Release retired tasks and assess whether the disclosure improves scientific scrutiny without harming active evaluation.

---

36. Implications for Future Standards

A future held-out evaluation standard could require:

36.1 Holdout Justification

Explain why restriction is necessary.

36.2 Protected-Surface Definition

Identify exactly what is held out.

36.3 Threat Model

Define assets, threats, controls, and residual risk.

36.4 Governance

Specify ownership, roles, conflicts, review, and appeals.

36.5 Provenance

Maintain task and solution history.

36.6 Access Control

Define tiers, authorization, review, and revocation.

36.7 Administration

Standardize configuration, logging, deviations, and scoring.

36.8 Validation

Demonstrate correctness, reliability, coverage, and fairness.

36.9 Reproducibility

Provide controlled replication or equivalent assurance.

36.10 Disclosure

Publish methodology, results, limitations, and restriction rationale.

36.11 Compromise Response

Define detection, investigation, status, remediation, and notice.

36.12 Rotation and Retirement

Define exposure limits, renewal, expiration, and archival treatment.

Such a standard should be developed through the future `STANDARDS_DEVELOPMENT_PROCESS.md`.

---

37. Relationship to the Other Foundations

Foundation 1: Dynamic Evaluation Protocols

A holdout must evolve as tasks leak, saturate, and lose relevance.

Foundation 3: High-Stakes Capability Evaluation

The higher the potential consequence, the stronger the case for protected, independently administered evidence.

Foundation 4: Independent Expert Review

Restricted content requires qualified external review to maintain accountability.

Foundation 5: Third-Party Auditor Ecosystem

Held-out evaluation depends on trustworthy organizations capable of secure administration.

Foundation 6: Progressive Standards and Requirements

Held-out protocols may evolve from research practice into procurement, certification, or regulatory evidence.

Foundation 7: Incentives and Prestige

Participation should reward genuine scrutiny, not private-test mystique.

Foundation 8: Global Interoperability

Institutions need compatible metadata, access, security, and recognition systems.

---

38. Canonical Standards Body Positions

Standards Body adopts the following working positions.

1. Public benchmarks remain essential to open science.
2. Some frontier evaluations require held-out components because public exposure can weaken validity.

3. Held-out evaluation is a spectrum, not a binary category.
4. Every restriction should have a defined purpose, scope, owner, duration, and review path.
5. Confidentiality does not establish scientific validity.
6. Transparent governance can coexist with protected content.
7. The construct and broad methodology should normally be disclosed.
8. Exact tasks, solutions, attack libraries, or environment details may remain protected when disclosure would materially reduce validity or create harm.
9. Held-out results should not be the sole basis for broad safety claims.
10. High-consequence held-out evaluations require independent review.
11. Task provenance and chain of custody are core evaluation evidence.
12. Access should follow least privilege and separation of duties.
13. Developers should receive fair notice without active-task disclosure.
14. Developers should be able to challenge material errors but should not control publication.
15. Open-source and smaller actors require meaningful participation pathways.
16. A private evaluator is not automatically independent.
17. Result uncertainty should be reported even when the task set is confidential.
18. Held-out tasks should rotate, expire, or retire.
19. Compromise should change result status transparently.
20. Security controls should be proportionate and auditable.
21. Cryptographic or confidential-computing methods can support assurance but cannot replace governance.
22. Retired-task release should be considered when safe and useful.
23. International interoperability should favor shared requirements over one global task bank.
24. Passing a held-out evaluation is not proof of safety.
25. The holdout system itself should be regularly evaluated.

---

39. Decision Rules

A held-out component is justified when:

- Public exposure would materially increase contamination or gaming
- the decision consequence is substantial
- task replacement is costly or slow
- attack disclosure would undermine safeguard testing
- the domain contains sensitive or harmful content
- independent generalization evidence is otherwise unavailable
- the protocol can support secure and fair administration

A held-out component is not justified merely because:

- Secrecy makes the evaluation appear authoritative
- maintainers want exclusive control
- public criticism is inconvenient
- the methodology is weak
- a sponsor prefers confidentiality
- the evaluator wants commercial lock-in

A task should be rotated when:

- Exposure exceeds policy
- leakage is suspected
- performance saturates
- validity declines
- environment changes
- solutions become common
- the task is used for development

A result should be suspended when:

- Active tasks may have leaked
- model identity is uncertain
- configuration materially deviated
- scoring integrity failed
- logs are incomplete
- independent review finds a serious flaw

A protocol should be retired when:

- The holdout can no longer be protected
- task renewal is not feasible
- construct validity fails
- governance legitimacy fails
- cost exceeds decision value
- a stronger successor exists

40. Held-Out Evaluation Plan Template

A. Identity

- Protocol name
- identifier
- version
- owner
- status
- date

B. Purpose

- Decision question
- intended users
- intended use
- prohibited use
- holdout justification

C. Construct

- Definition
- subdomains
- exclusions
- deployment context

D. Protected Surface

- Tasks
- solutions
- scoring
- generator
- environment
- thresholds
- results
- identities

E. Threat Model

- Assets
- adversaries
- attack surfaces
- controls

- residual risk

F. Governance

- Roles
- conflicts
- funding
- review
- appeals
- disclosure authority

G. Task Bank

- Sources
- provenance
- validation
- difficulty
- coverage
- exposure policy
- retirement

H. Access

- Tiers
- qualification
- approval
- monitoring
- revocation

I. Infrastructure

- Storage
- transfer
- execution
- logging
- backup
- deletion
- attestation

J. Administration

- Model verification
- configuration
- tools

- retries
- dry run
- deviations

K. Scoring

- Metrics
- judges
- adjudication
- uncertainty
- thresholds

L. Reporting

- Public report
- restricted annex
- developer review
- result status
- expiration

M. Compromise Response

- Detection
- investigation
- suspension
- remediation
- notice

N. Renewal

- Rotation triggers
- review cycle
- successor
- retirement

---

41. Access Request Template

Applicant:
Organization:
Role:
Requested protocol:
Requested access tier:

Purpose:
Duration:

Competence

Describe relevant technical and domain expertise.

Security Controls

Describe identity, device, storage, execution, logging, and incident controls.

Conflicts

Disclose relevant financial, professional, competitive, and intellectual interests.

Data Handling

Explain how protected materials and outputs will be handled.

Publication

Describe intended outputs and review process.

Subcontractors

Identify all third parties.

Prior Incidents

Disclose relevant security or integrity incidents.

Decision

- Approved
- approved with conditions
- supervised access only
- deferred
- rejected

Conditions

Expiration

---

42. Compromise Incident Template

Incident identifier:
Date discovered:
Reporter:
Protocol:
Affected version:

Description

Assets Affected

Suspected Exposure

Timeline

Immediate Actions

Evidence Preserved

Results Potentially Affected

Investigation Lead

Status

- Suspected
- confirmed
- contained
- remediated
- closed

Result Action

- No change
- caveat
- suspend
- withdraw
- re-evaluate

Root Cause

Corrective Action

Public Notice

Governance Review

Follow-Up Date

---

43. Holdout Scorecard

Dimension	Core Question
Purpose	Is the measurement reason for holding content out explicit?
Construct	Is the evaluated capability clearly defined?
Protected surface	Is it clear what is restricted and why?
Provenance	Can each task and solution be traced?
Coverage	Does the bank represent the intended task universe?
Validity	Does evidence support the intended interpretation?
Reliability	Are results sufficiently consistent?
Integrity	Are contamination and manipulation risks addressed?
Security	Are controls proportionate to the threat model?
Access	Is authorization role-based and reviewable?
Chain of custody	Is every material access and change recorded?
Administration	Can qualified evaluators run the protocol consistently?
Scoring	Are rules, judges, and uncertainty defensible?
Transparency	Is enough public for accountability?
Reproducibility	Can independent parties verify the process?
Fairness	Are participants treated consistently?
Independence	Is evaluator judgment protected from interested parties?
Appeals	Can material error be challenged?
Compromise response	Can the system detect, suspend, recover, and disclose?
Renewal	Are tasks rotated before authority decays?
Retirement	Can obsolete or compromised evidence be withdrawn?
Accessibility	Can qualified smaller and open actors participate?
Interoperability	Can other institutions understand and recognize the result?

Dimension	Core Question
Decision utility	Does the holdout improve a real decision?

---

44. Final Perspective

A benchmark loses independence when the people building the system know exactly how success will be measured.

That does not make public evaluation useless.

It means public evaluation answers a different question.

A public benchmark can show how well a system performs on a shared, inspectable target.

A held-out evaluation can provide additional evidence about how the system behaves when the exact target has not become part of development.

Both matter.

The future evaluation ecosystem should preserve the openness that allows science to progress while protecting enough unseen evidence to test genuine generalization, resilience, and threshold capability.

That balance is difficult.

Too little protection produces contaminated or gameable measurement.

Too much secrecy produces unaccountable authority.

The answer is not a permanent secret benchmark.

The answer is a governed holdout system.

Such a system should know:

- What it protects
- why it protects it
- who can access it
- how access is recorded
- how tasks are validated
- how results are reproduced
- how participants appeal
- how compromise is detected
- how tasks are renewed
- when evidence expires
- when content can be disclosed
- when the protocol should end

The second foundation of Standards Body is therefore not secrecy.

It is protected evidence under accountable control.

References and Research Basis

[^nist-tevv]: National Institute of Standards and Technology, **AI Test, Evaluation, Validation and Verification**. https://www.nist.gov/ai-test-evaluation-validation-and-verification-tevv

[^nist-rmf]: National Institute of Standards and Technology, **AI Risk Management Framework**. https://www.nist.gov/itl/ai-risk-management-framework

[^nist-statistical]: National Institute of Standards and Technology, **Expanding the AI Evaluation Toolbox with Statistical Models**, 2026. https://www.nist.gov/news-events/news/2026/02/new-report-expanding-ai-evaluation-toolbox-statistical-models

[^nist-cheating]: National Institute of Standards and Technology, **Practices for Detecting and Preventing Evaluation Cheating**, 2025. https://www.nist.gov/caisi/cheating-ai-agent-evaluations/4-practices-detecting-and-preventing-evaluation-cheating

[^nist-protein]: National Institute of Standards and Technology, **Experimental Evaluation of AI-Driven Protein Design Risks Using Safe Biological Proxies**, 2025. https://www.nist.gov/publications/experimental-evaluation-ai-driven-protein-design-risks-using-safe-biological-proxies

[^aisi-lessons]: UK AI Security Institute, **Early Lessons from Evaluating Frontier AI Systems**, 2024. https://www.aisi.gov.uk/blog/early-lessons-from-evaluating-frontier-ai-systems

[^aisi-qa]: UK AI Security Institute, **Early Insights from Developing Question-Answer Evaluations for Frontier AI**, 2024. https://www.aisi.gov.uk/blog/early-insights-from-developing-question-answer-evaluations-for-frontier-ai

[^aisi-sandbox]: UK AI Security Institute, **The Inspect Sandboxing Toolkit: Scalable and Secure AI Agent Evaluations**, 2025. https://www.aisi.gov.uk/blog/the-inspect-sandboxing-toolkit-scalable-and-secure-ai-agent-evaluations

[^aisi-sandbox-learning]: UK AI Security Institute, **What Can Sandboxed AI Agents Learn About Their Evaluation Environments?**, 2026. https://www.aisi.gov.uk/blog/what-can-sandboxed-ai-agents-learn-about-their-evaluation-environments

[^aisi-breakout]: UK AI Security Institute, **Can AI Agents Escape Their Sandboxes?**, 2026. https://www.aisi.gov.uk/blog/can-ai-agents-escape-their-sandboxes-a-benchmark-for-safely-measuring-container-breakout-capabilities

[^aisi-safeguards]: UK AI Security Institute, **Principles for Safeguard Evaluation**, 2025. https://www.aisi.gov.uk/blog/principles-for-safeguard-evaluation

[^inspect]: UK AI Security Institute, **Inspect AI**, evaluation framework. https://inspect.aisi.org.uk/

[^inspect-cyber]: UK AI Security Institute, **A New Standard for Agentic Cyber Evaluations**, 2025. https://www.aisi.gov.uk/blog/inspect-cyber

[^openai-pf]: OpenAI, **Preparedness Framework v2**, 2025. https://cdn.openai.com/pdf/18a02b5d-6b67-4cec-ab64-68cdfbddebcd/preparedness-framework-v2.pdf

[^openai-external]: OpenAI, **Strengthening Our Safety Ecosystem with External Testing**, 2025. https://openai.com/index/strengthening-safety-with-external-testing/

[^deepmind-fsf]: Google DeepMind, **Frontier Safety Framework**, updated 2025. https://deepmind.google/blog/updating-the-frontier-safety-framework/

[^deepmind-warning]: Google DeepMind, **An Early Warning System for Novel AI Risks**, 2023. https://deepmind.google/blog/an-early-warning-system-for-novel-ai-risks/

[^anthropic-rsp]: Anthropic, **Responsible Scaling Policy**, Version 3.2, 2026. https://www.anthropic.com/responsible-scaling-policy

[^extreme-risk]: Toby Shevlane et al., **Model Evaluation for Extreme Risks**, 2023. https://arxiv.org/abs/2305.15324

[^dangerous-capabilities]: Mary Phuong et al., **Evaluating Frontier Models for Dangerous Capabilities**, 2024. https://arxiv.org/abs/2403.13793

[^external-access]: Jacob Charnock et al., **Expanding External Access to Frontier AI Models for Dangerous Capability Evaluations**, 2026. https://arxiv.org/abs/2601.11916

[^contamination-survey]: Cheng Xu et al., **Benchmark Data Contamination of Large Language Models: A Survey**, 2024. https://arxiv.org/abs/2406.04244

[^contamination-trust]: Yihong Dong et al., **Data Contamination and Trustworthy Evaluation for Large Language Models**, 2024. https://arxiv.org/abs/2402.15938

[^retro-holdout]: Jack Haimes et al., **Benchmark Inflation: Revealing LLM Performance Gaps Using Retro-Holdouts**, 2024. https://arxiv.org/abs/2410.09247

[^livebench]: Colin White et al., **LiveBench: A Challenging, Contamination-Limited LLM Benchmark**, 2024. https://arxiv.org/abs/2406.19314

[^c2leva]: Yanyang Li et al., **C2LEVA: Toward Comprehensive and Contamination-Free Language Model Evaluation**, 2024. https://arxiv.org/abs/2412.04947

[^clean-eval]: Wenhong Zhu et al., **CLEAN-EVAL: Clean Evaluation on Contaminated Large Language Models**, 2023. https://arxiv.org/abs/2311.09154

[^ccc-technical]: Confidential Computing Consortium, **A Technical Analysis of Confidential Computing**. https://confidentialcomputing.io/wp-content/uploads/sites/10/2023/03/CCC-A-Technical-Analysis-of-Confidential-Computing-v1.3_unlocked.pdf

[^ccc-attestation]: Confidential Computing Consortium, **Why Is Attestation Required for Confidential Computing?**, 2023. https://confidentialcomputing.io/2023/04/06/why-is-attestation-required-for-confidential-computing/

Revision Record

Version 1.0

Date: July 16, 2026

Change type: Complete replacement

Summary: Replaces the earlier outline edition with a fully developed canonical working white paper. Adds first-principles rationale, definitions, design taxonomy, contamination analysis, holdout threat modeling, task provenance, statistical design, security architecture, access tiers, chain of custody, administration, reporting, controlled reproducibility, transparency, fairness, developer-evaluator relationships, governance, compromise response, lifecycle, domain-specific requirements, protected-compute options, international interoperability, maturity model, implementation pathway, a Standards Body pilot, metrics, failure analysis, objections, evidence gaps, research agenda, standards implications, operational templates, scorecard, and primary-source research basis.

Status: Ready for internal review and future expert critique.