Colorado State University
Department of Computer Science

Access Control Policies Specification and Analysis for Multi-Institutional Collaborative Projects

Abhimanyu Chawla, Mahmoud Abdelgawad, Indrakshi Ray

IEEE CIC 2025, Pittsburgh, PA, USA

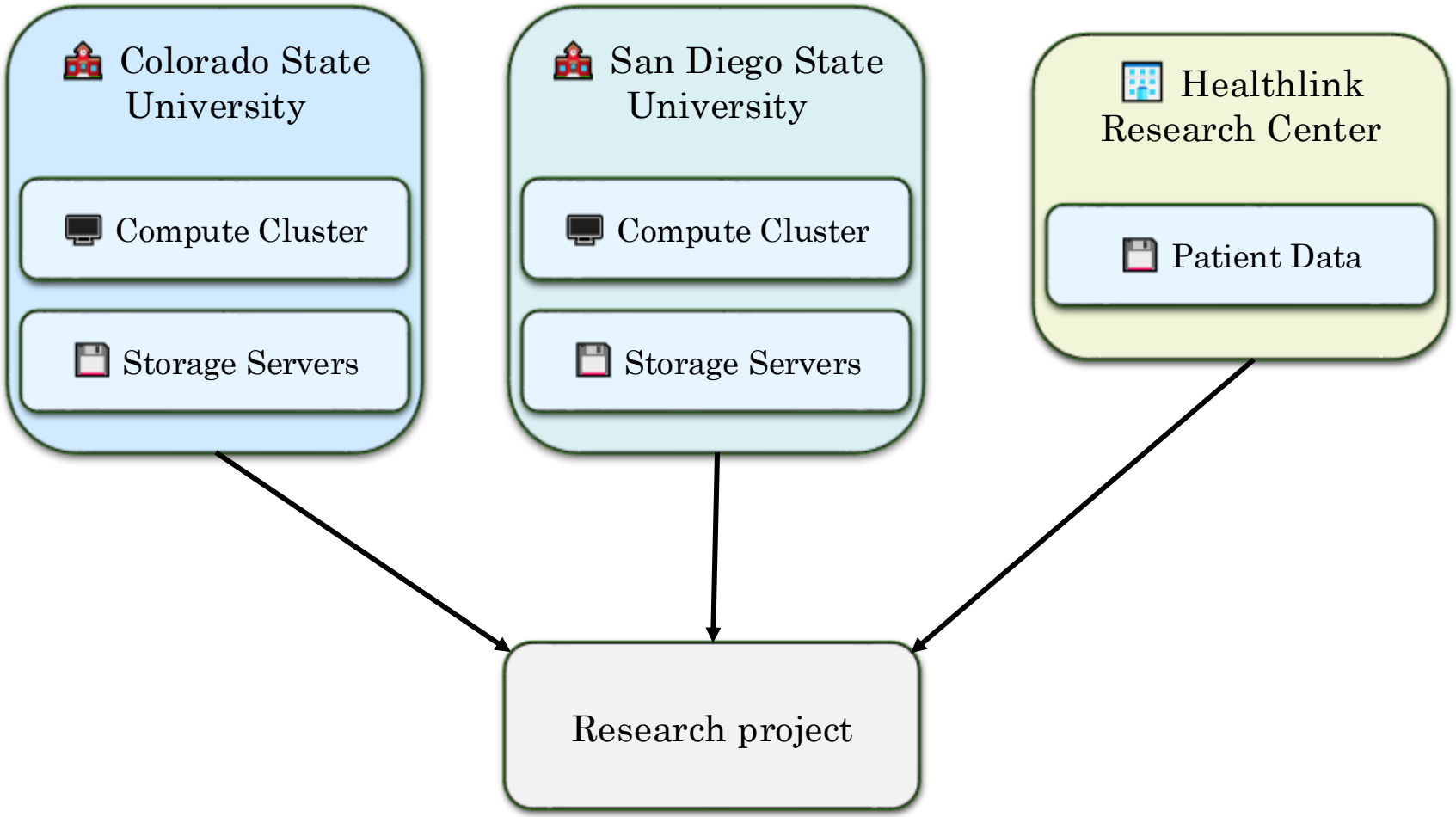
Outline

- Research Computing Infrastructure (RCI)
- Need for RCI Policies
- Compliance with RCI policies
- Security of RCI policies
- Conclusion & Future Work

Scientific Research

- Scientific research is *collaborative* in nature
- *Multiple institutions* with their *own rules and regulations*
  - Universities, Industries, Government Organizations
  - Institutions may span multiple countries
- *Multiple stakeholders* with their *own interests*
- Various types of users involved in the collaboration
- *Dynamic nature* of the collaboration
  - Users, organizations join/leave the collaboration
  - Resources (infrastructure, code, data) generated and consumed
- *Problem: how to correctly specify the access control policies of such a scientific collaboration?*

Scientific Collaboration Example



Research Computing Infrastructure

- Access control policies are needed to protect the *research computing infrastructure* in the scientific collaboration
- Research computing infrastructure comprises
  - Hardware
  - Software
  - Data
  - Code/Algorithms/Scientific Artifacts
- Goal
  - *Formulate access control policies that can be enforced uniformly across multiple institutions*

Access Control Requirements

- Derived from various documents
  - Domain specific regulations pertaining to project (HIPAA)
  - Institutional policies
  - State and national policies
  - International regulations
- Typically specified in English

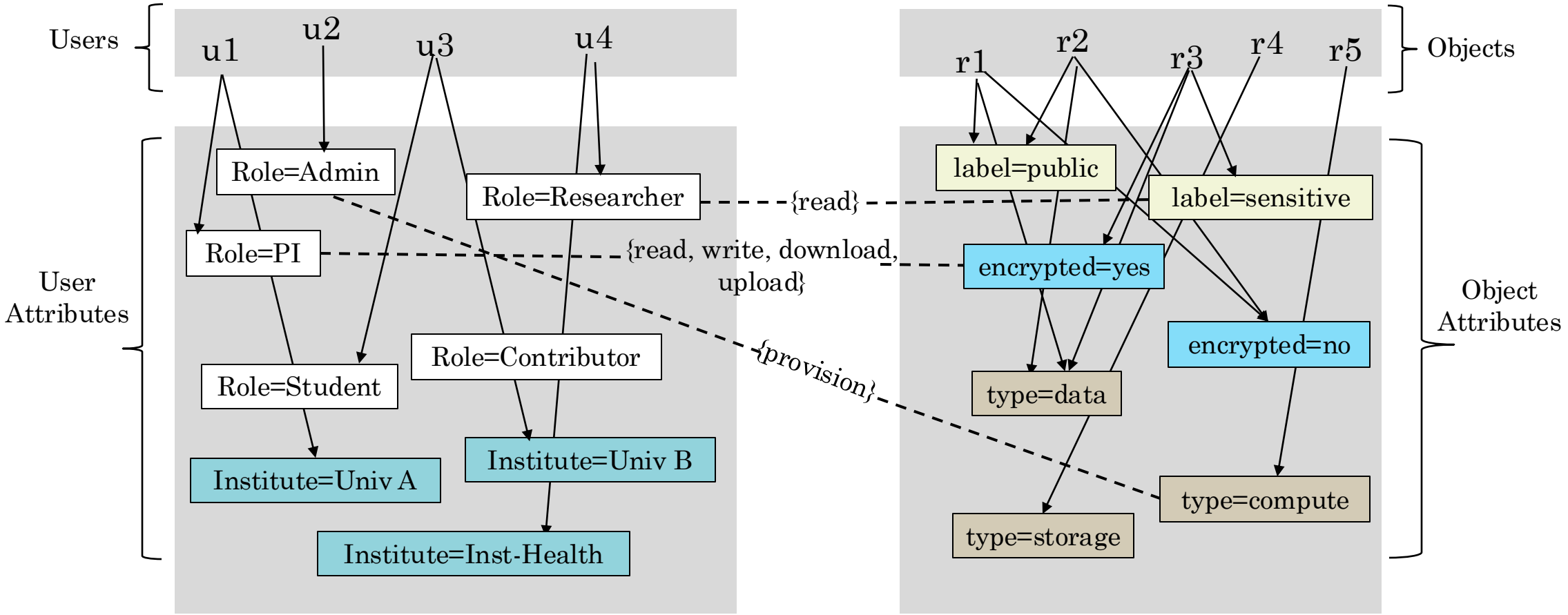
Example Access Control Policies

Policy ID	Policy Description
P1	PI can read, write, download, and upload sensitive data
P2	Researchers and students from Univ-A, Univ-B may read but not write sensitive data
P3	Encryption of sensitive data must be done by PI using approved algorithms
P4	Decryption of sensitive data requires PI
P5	Access to secure HPC nodes is granted to PI or researchers.
P6	Only admin can provision HPC resources and perform system operations
P7	Only researchers with IRB training can conduct user studies

Access Control Model

- Access control requirements must be translated into the form of an access control model that can be *analyzed* and *enforced*
- Which access control model to use?
  - Cannot be based on just identity or role of user
  - Attribute-based access control to support fine-grained access
  - Should be able to support *dynamic policies*
- *NIST Next Generation Access Control (NGAC) is used*

NGAC Access Control Model



Workflows

- Scientific projects involve the execution of various workflows
  - *createDataset*, *uploadData*
- System administrators execute workflows to support project
  - *addUser*, *performBackup*, *secureDelete*
- Workflow has tasks coordinated by control-flow dependencies
- *W = uploadData(patientData)*
- *W = If(T1){T2;T3} T4*
  - *T1 = check(patientData.label == sensitive)*
  - *T2 = generateKey(k1)*
  - *T3 = encrypt(PatientData,k1)*
  - *T4 = write(PatientData, ServerD)*

Workflow Authorization

- *W = uploadData(patientData)*
- *W = If (t1) {t2;t3;t4} t5*
  - *t1 = check(patientData.label == sensitive)*
  - *t2 = determineEncType(patientData)*
  - *t3 = generateKey(k1)*
  - *t4 = encrypt(PatientData,k1)*
  - *t5 = write(PatientData, ServerD)*
- Not all workflows can be executed by all users
- Workflow authorization/system authorization matrix states which user and resource attributes needed to perform task

Authorizations

- Authorizations are specified in the form of matrices
- Matrices are derived from NGAC rules

W	**User Attributes**		**Resource Attributes**	
	Role = PI	Inst = Univ A	Own = HI	Label = Sens.
W1	1	1	1	1
W2	1	0	1	0

Workflow Authorization Matrix

T	User Attributes					Resource Attributes						
	Inst = UnA	Inst = HI	Role = Stud	Role = Res	Role = PI	Own = HI	Own = UnA	Lbl = Sens	Enc = Yes	Type = Data	Type = Str	Id = SvrD
t1	1	1	0	0	1	1	0	1`	0	1	0	0
t2	1	1	0	0	1	1	0	1	0	1	0	0
t3	1	1	0	0	1	0	0	0	0	0	0	0
t4	1	1	0	0	1	1	0	1	0	1	0	0
t5	1	0	0	0	1	0	1	1	0	0	1	1

We have a similar matrix for system operations

Authorization Matrix Properties

- **Property 1:** Every task must map to a valid combination of existing attributes
- **Property 2:** Every attribute in the matrix must be used by at least one task, or should be flagged as unused
- **Property 3: E**very task must have at least one required attribute
- We can enforce these properties for all the matrices

Compliance Inconsistencies

W	**User Attributes**		**Resource Attributes**	
	Role = PI	Inst = Univ A	Own = HI	Label = Sens.
W1	1	1	1	1
W2	1	0	1	0

- What happens if the policies change without updating this matrix?
- We get *compliance inconsistencies*

Compliance Inconsitency Example

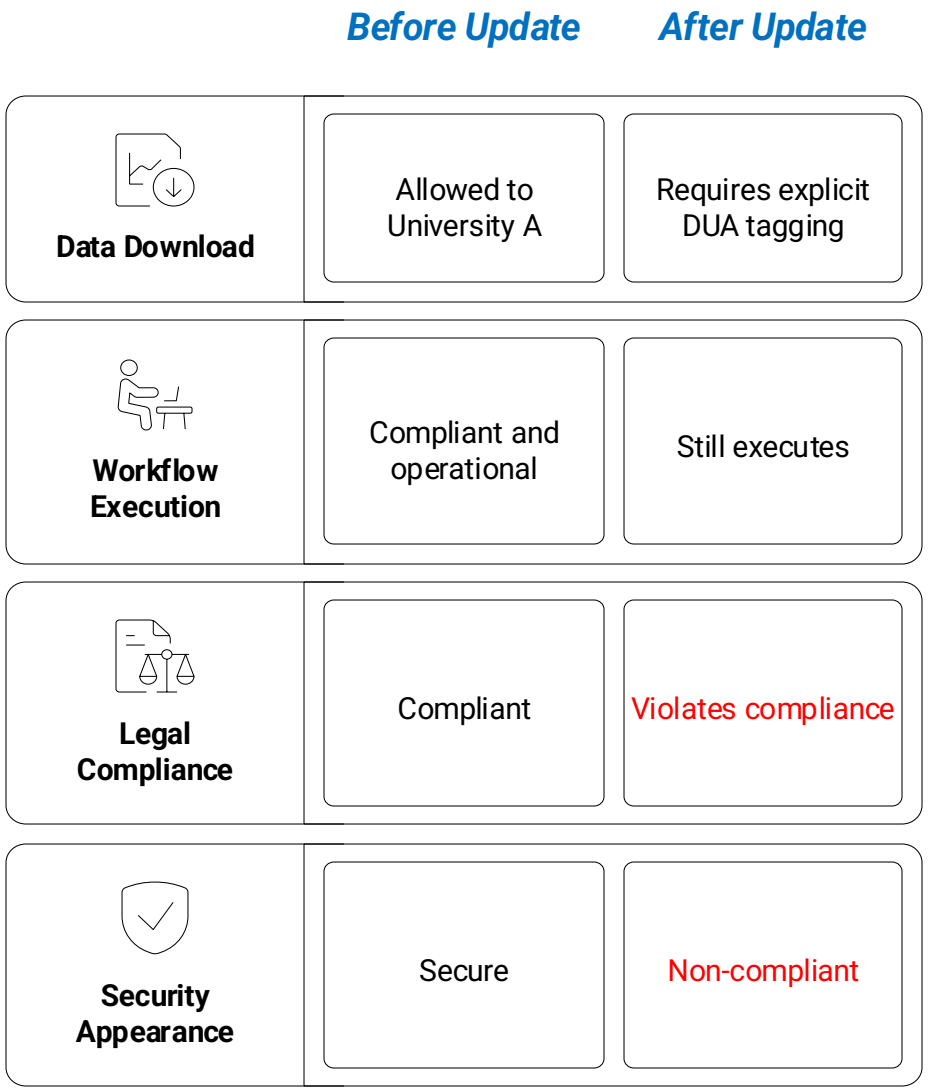
Compliance Inconsistencies

Example:

- Healthcare Institute (HI) updates its HIPAA compliance interpretation.
- New regulation now forbids export of any patient-level data to non-HI environments unless an explicit "Data Use Agreement" field is set

Impact

- Compliance inconsistency arises between workflow and updated regulatory policy
- The system *appears secure* but is **actually non-compliant** under the new rules



Refinement Inconsistencies

- What happens if a user has the attributes needed to execute the workflow, but does not have the attributes for performing the tasks in the workflow?

- What happens if a user has the attributes needed to execute the task, but does not have the attributes for performing the system level operations in the workflow?

Refinement Inconsistencies

T	User Attributes					Resource Attributes						
	Inst = UnA	Inst = HI	Role = Stud	Role = Res	Role = PI	Own = HI	Own = UnA	Lbl = Sens	Enc = Yes	Type = Data	Type = Str	Id = SvrD
t1	1	1	0	0	1	1	0	1`	0	1	0	0
t2	1	1	0	0	1	1	0	1	0	1	0	0
t3	1	1	0	0	1	0	0	0	0	0	0	0
t4	1	1	0	0	1	1	0	1	0	1	0	0
t5	1	0	0	0	1	0	1	1	0	0	1	1

John.role = PI
John.institute = HI

John can execute W
John cannot execute t5 in W

Refinement inconsistency

Refinement Inconsistency Example

Refinement Inconsistencies

Example:

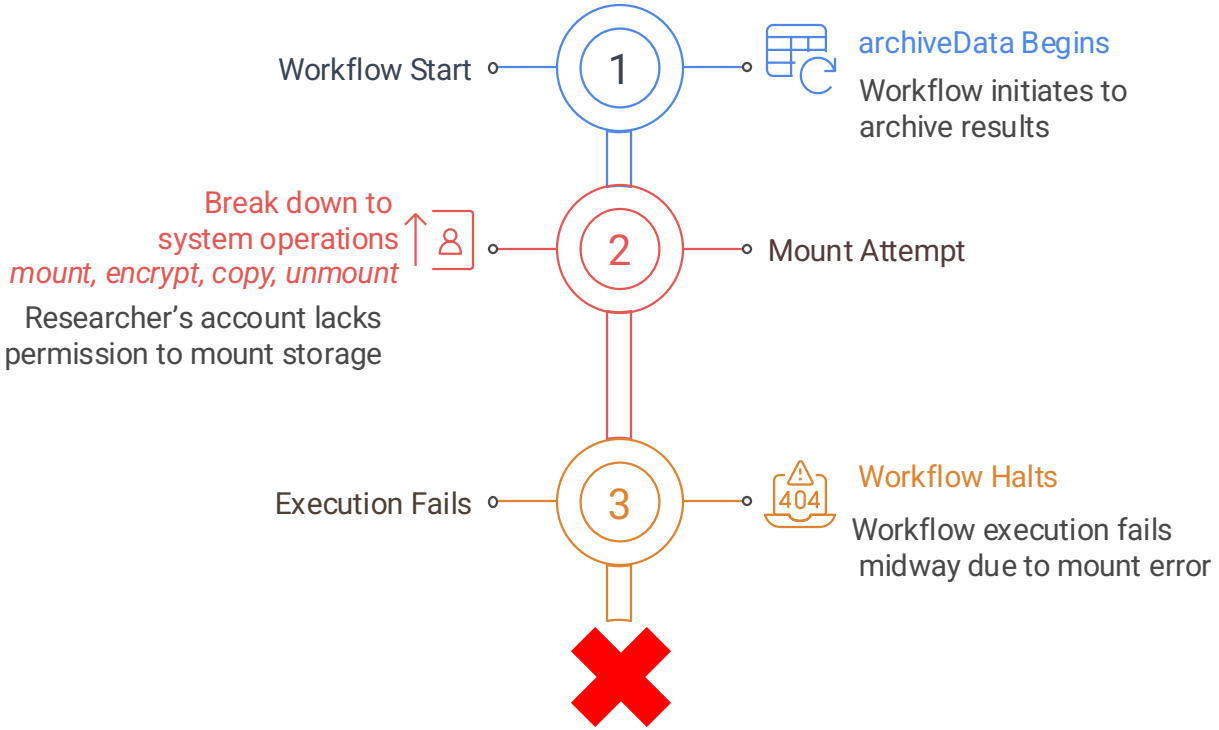
- Task **archiveData** at University A is defined to store anonymized results to long-term encrypted storage.
- The workflow grants the ***Researcher*** role at University A permission to execute **archiveData**.

Problem:

- The Researcher's account lacks permission to perform **mount** operation on the storage partition (that operation requires elevated privileges).

Impact

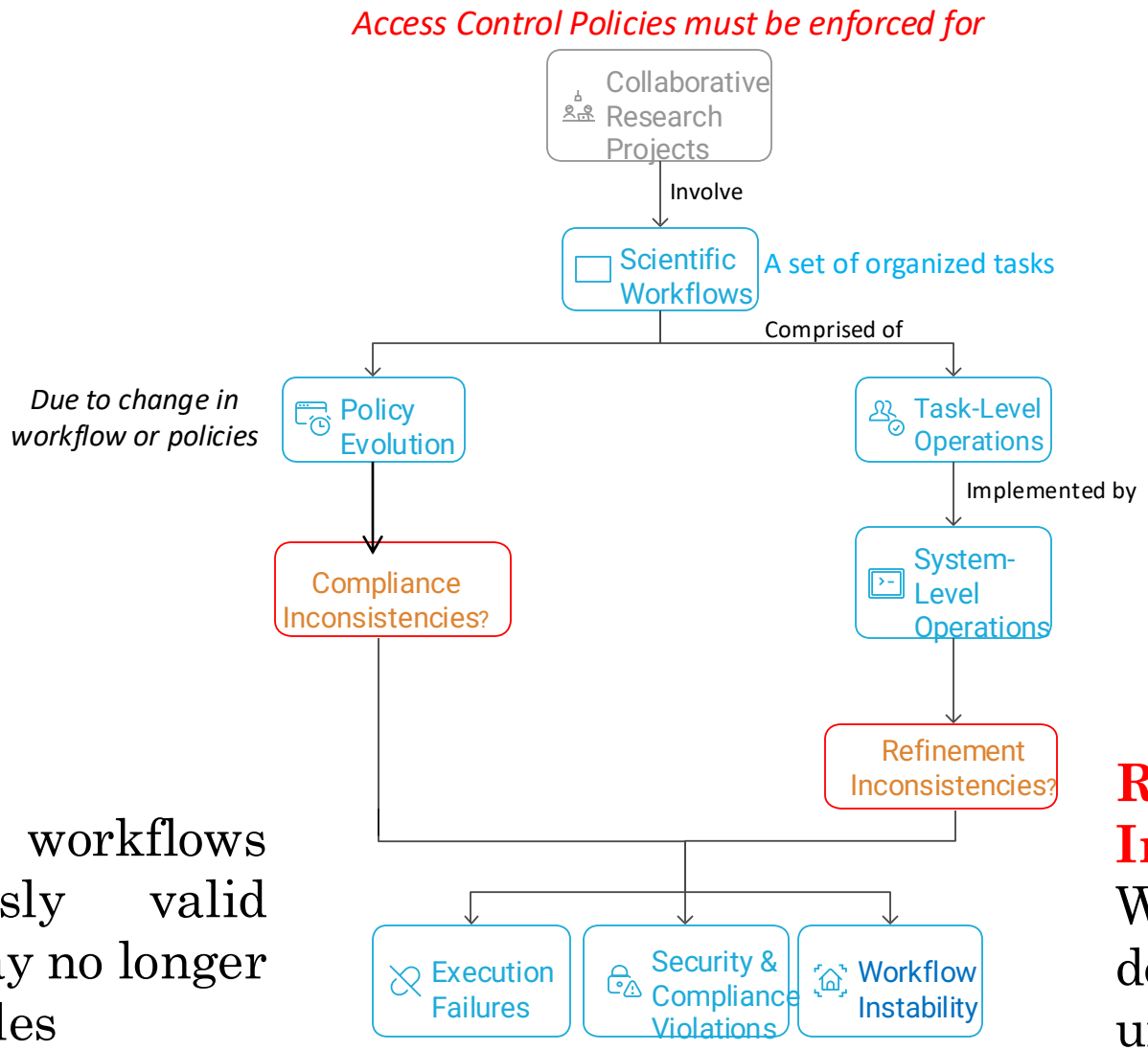
- Refinement inconsistency between task-level authorization (allowed) and system-level enforcement (partially denied).
- *Execution fails* midway.
- We are also preventing some unauthorized malicious task from executing system level operations



Refinement Consistency Enforcement

- Describes a workflow using its constituent tasks
  - $W = \{t_1, t_2, t_3, t_4, \ldots t_n\}\}$
- Describes a task in terms of low-level operations
  - $t_k = \{op_1, op_2, \ldots. op_m\}$
- Translational logic decomposes a workflow or task into its constituent components
- If a user (possessing some attributes) is authorized to execute a workflow, he should be allowed to execute the tasks in the workflow
- If a user (possessing some attributes) is authorized to execute a task, he should be allowed to execute the operations comprising the tasks

Compliance and Refinement Inconsistency Checks



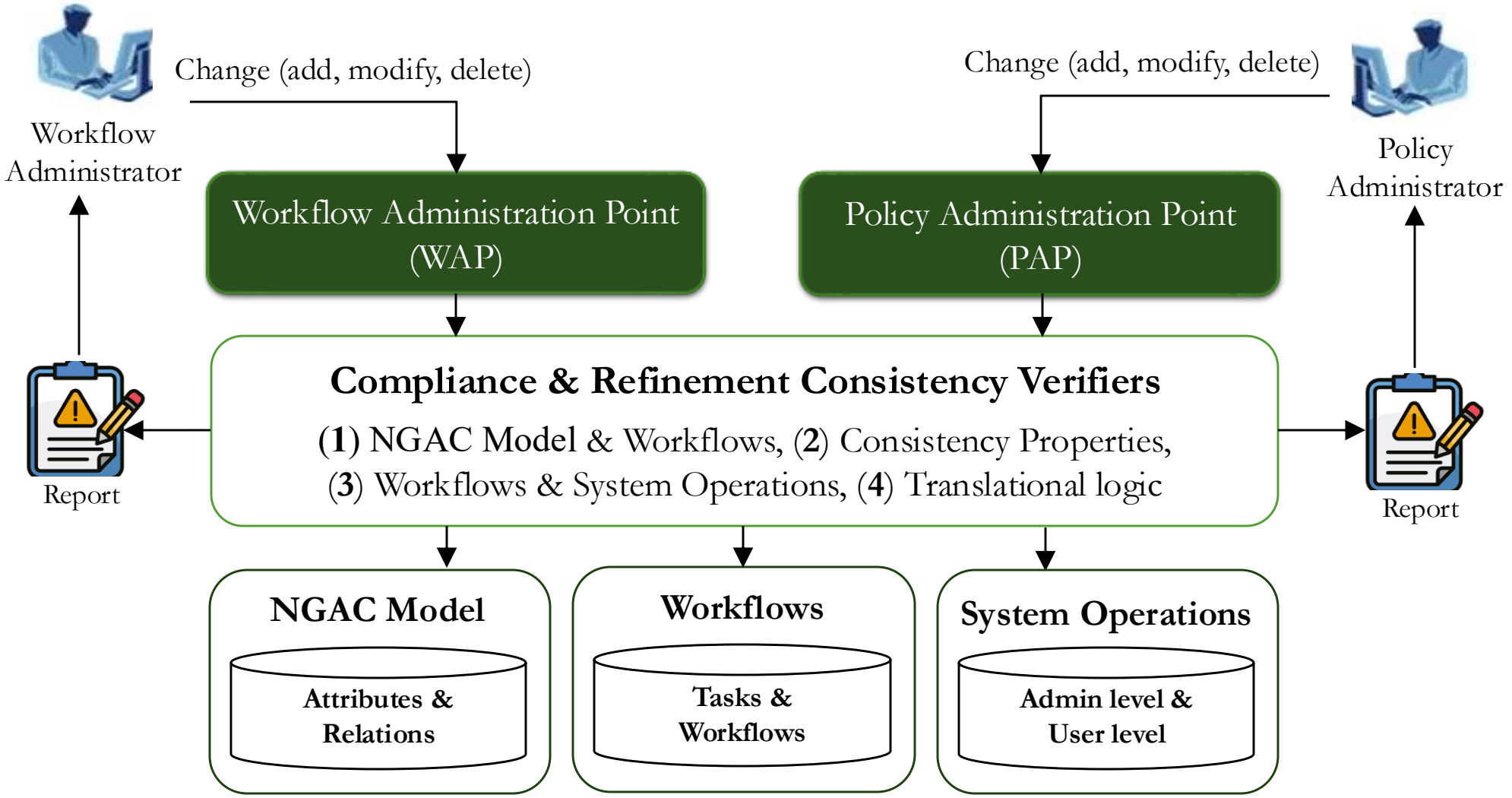
Compliance Inconsistencies

When policies or workflows evolve, previously valid authorizations may no longer align with new rules

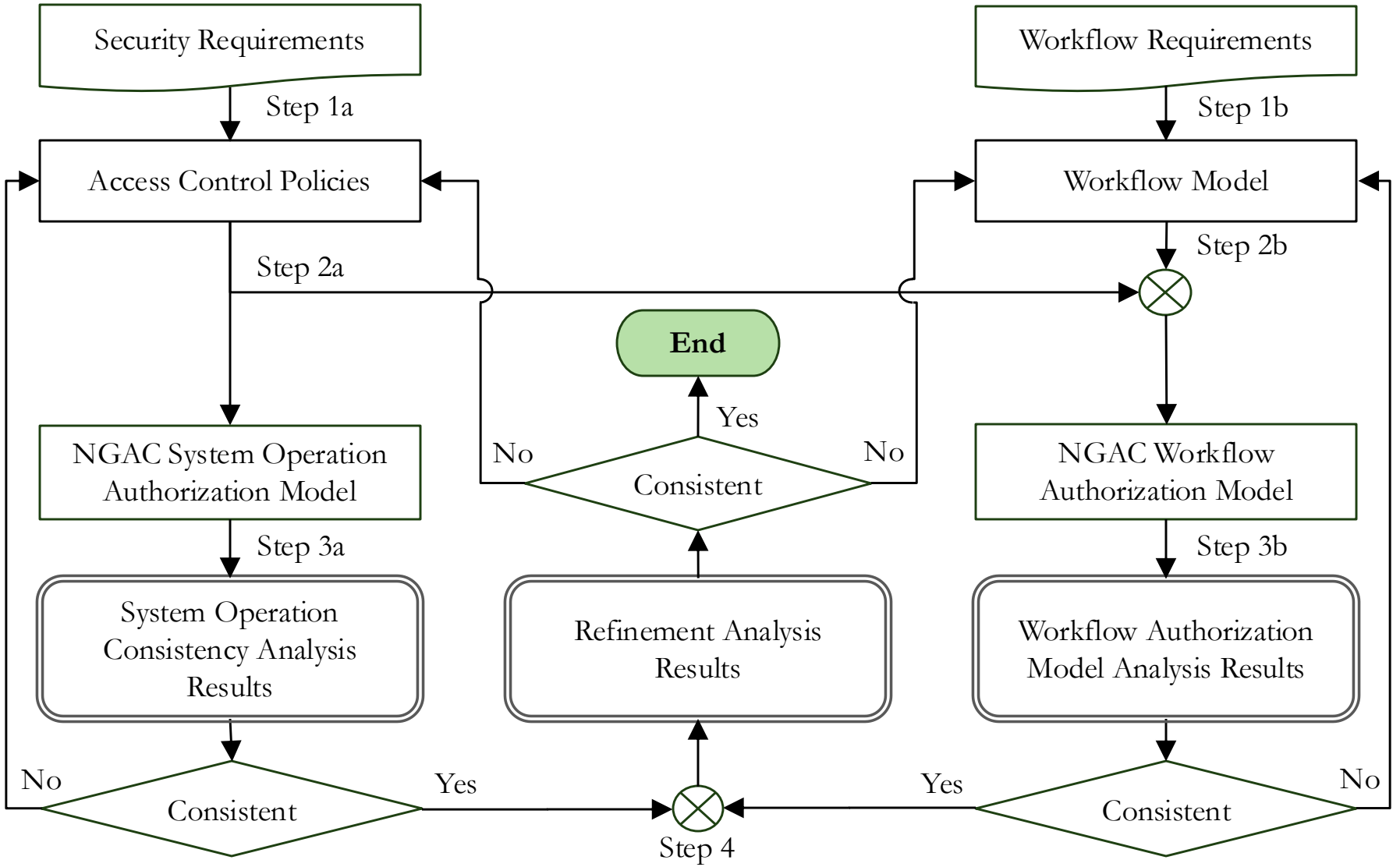
Refinement Inconsistencies

Workflow/task authorization does not match that of underlying system operation

Overview of Approach



Consistency, Refinement, Compliance Checks



Conclusion

- **NGAC is suitable for RCI environments**
- **Automated checks for compliance and refinement inconsistencies**
- **Future work will handle more constraints including obligations**
- **Effectiveness in a collaborative setting must be evaluated**
- **Scalability and performance must be assessed**

Thank you!

Questions?