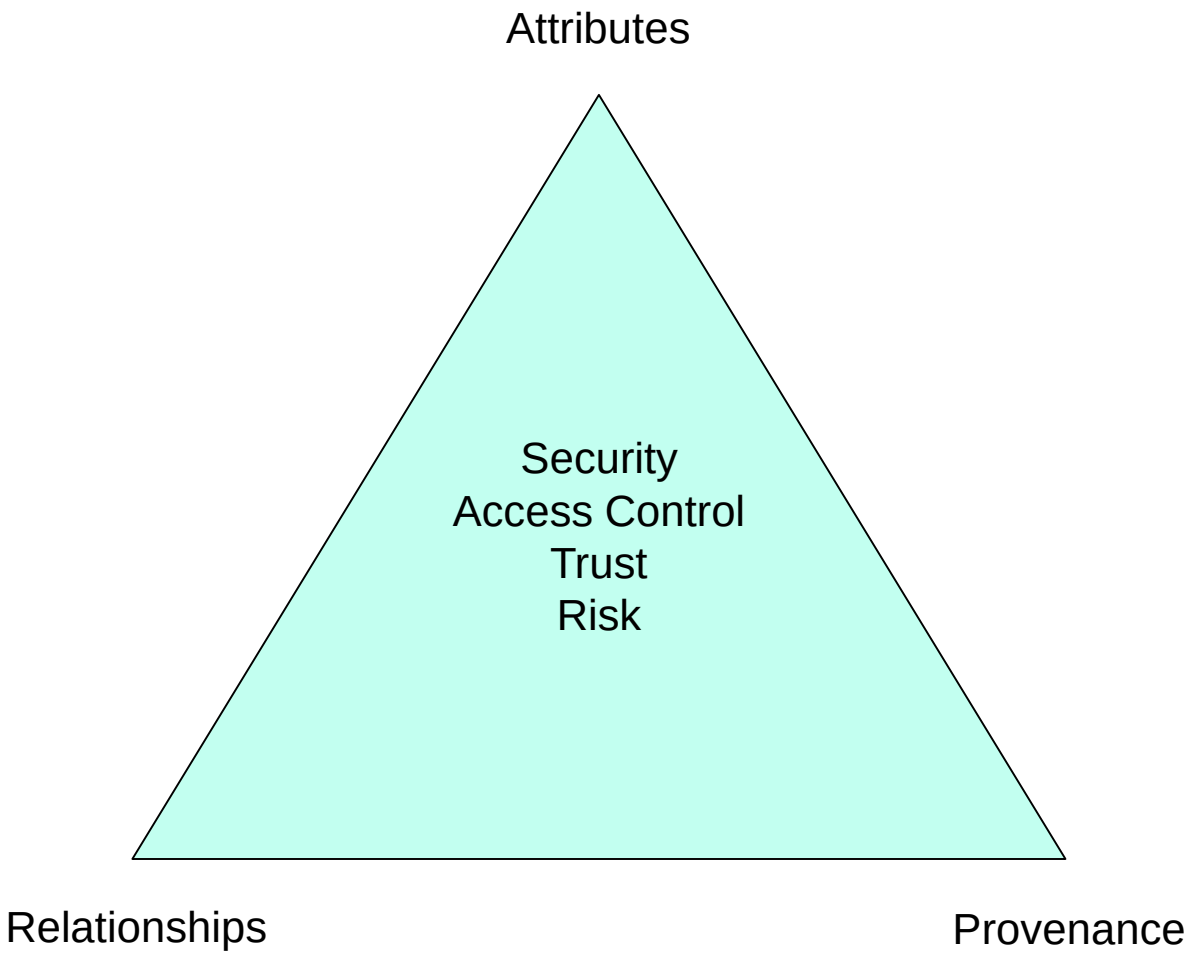
Provenance-Based Access Control (PBAC)

Prof. Ravi Sandhu
Executive Director and Endowed Chair

April 15, 2016

ravi.sandhu@utsa.edu
www.profsandhu.com

Ultimate Unified Model



What is Provenance?

Art definition of provenance

- Essential in judging authenticity and evaluating worth.

Data provenance in computing systems

- Is different from log data.
- Contains linkage of information pieces.
- Is utilized in different computing areas.

Provenance Data

- Information of operations/transactions performed against data objects and versions
  - Actions that were performed against data
  - Acting Users/Subjects who performed actions on data
  - Data Objects used for actions
  - Data Objects generated from actions
  - Additional Contextual Information of the above entities

- Directed Acyclic Graph (DAG)
- Causality dependencies between entities (acting users / subjects, action processes and data objects)
- Dependency graph can be traversed for the discovery of Origin, usage, versioning info, etc.

Provenance-aware Systems

- Capturing provenance data
- Storing provenance data
- Querying provenance data

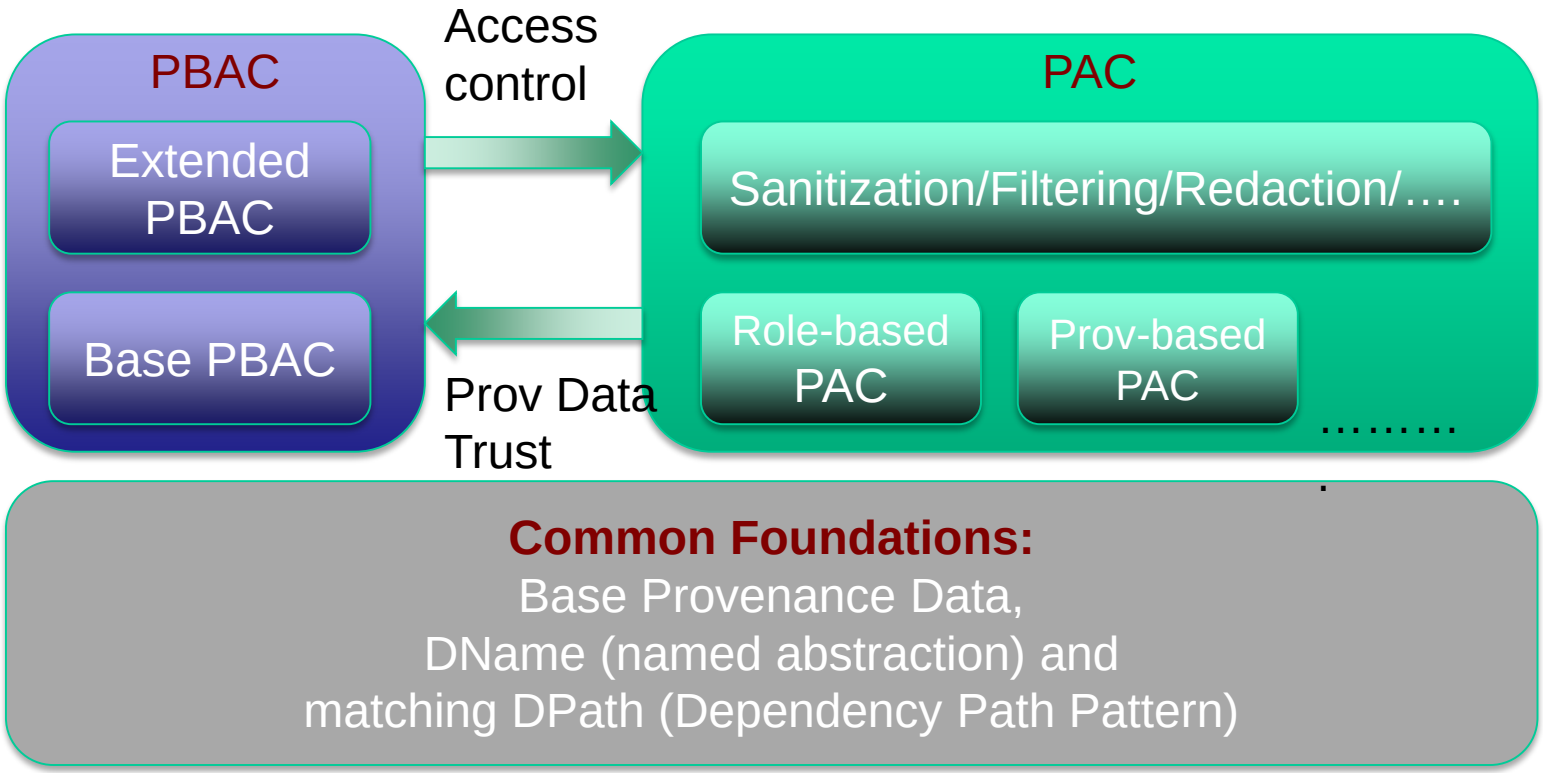
Provenance Data Model

- Using provenance data
- Securing provenance data

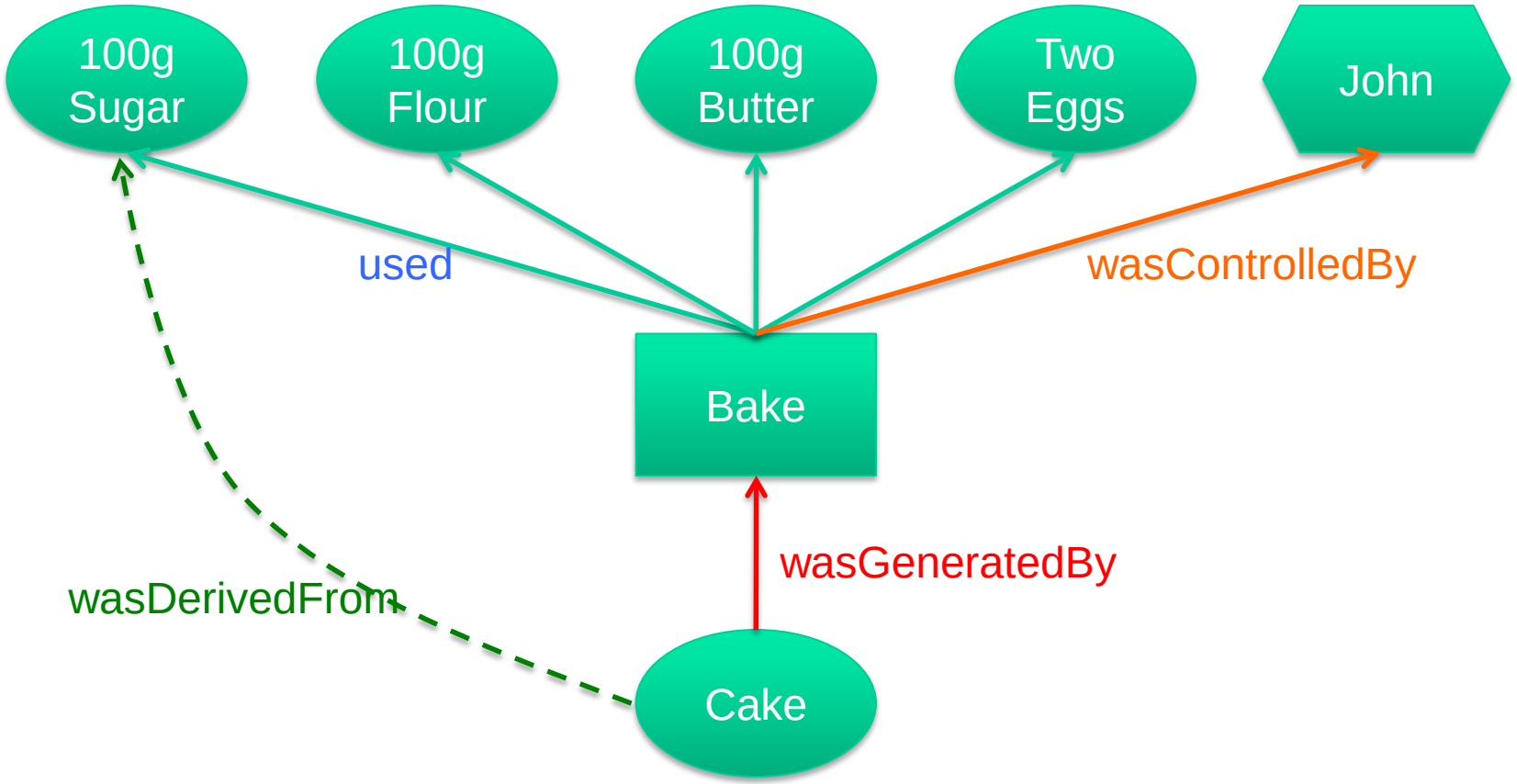
Access Control

- Provenance Access Control (PAC)
  - Controlling access to provenance data which could be more sensitive than the underlying data
  - Needs access control models/mechanisms (e.g, RBAC)
  - (Meaningful) control granularity? Right level of abstraction?
- Provenance-based Access Control (PBAC)
  - Using provenance data to control access to the underlying data
  - Provenance-based policy specification

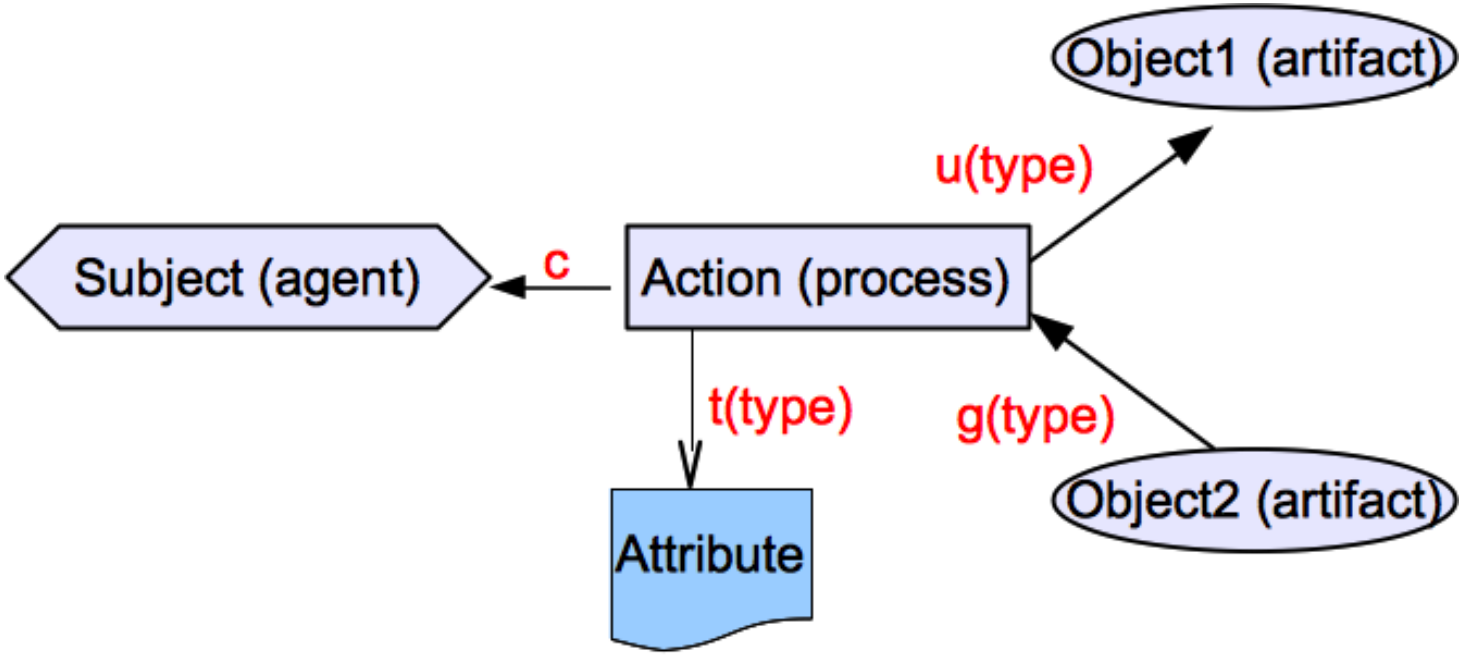
Access control in Provenance-aware Systems



Open Provenance Model Example



Provenance Data Model



c wasControlledby
u used
g wasGeneratedBy
t
hasAttributeOf

→ Attrb. edge
→ Dep. edge

Direct vs. Indirect Dependencies

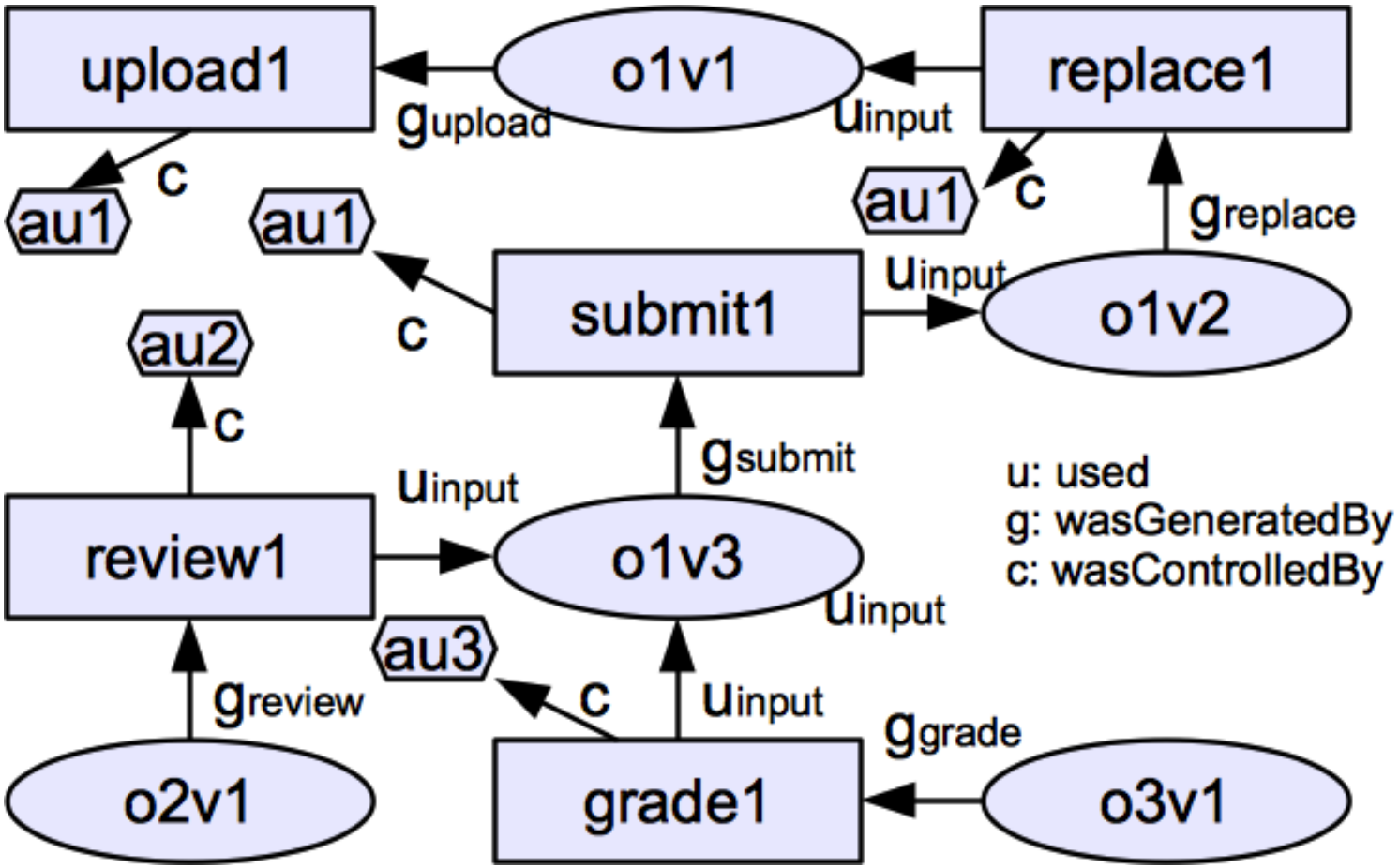
- Direct dependencies
  - Used (u), wasGeneratedBy (g), wasControlledBy (c)
  - Captured from transactions as base provenance data\
- Indirect dependencies
  - System-computable dependencies
    - using pre-defined dependency names and matching dependency path patterns
  - User-declared dependencies
    - using pre-defined dependency names

Object Dependency List (DL_O)

- A set of pairs of
  - abstracted dependency names (DNAME) and
  - regular expression-based object dependency path patterns (DPATH)

- Examples
  - < wasSubmittedVof, $g_{submit}.u_{input}$ >
  - < wasAuthoredBy, wasSubmittedVof?.wasReplacedVof $*.g_{upload}.c$ >

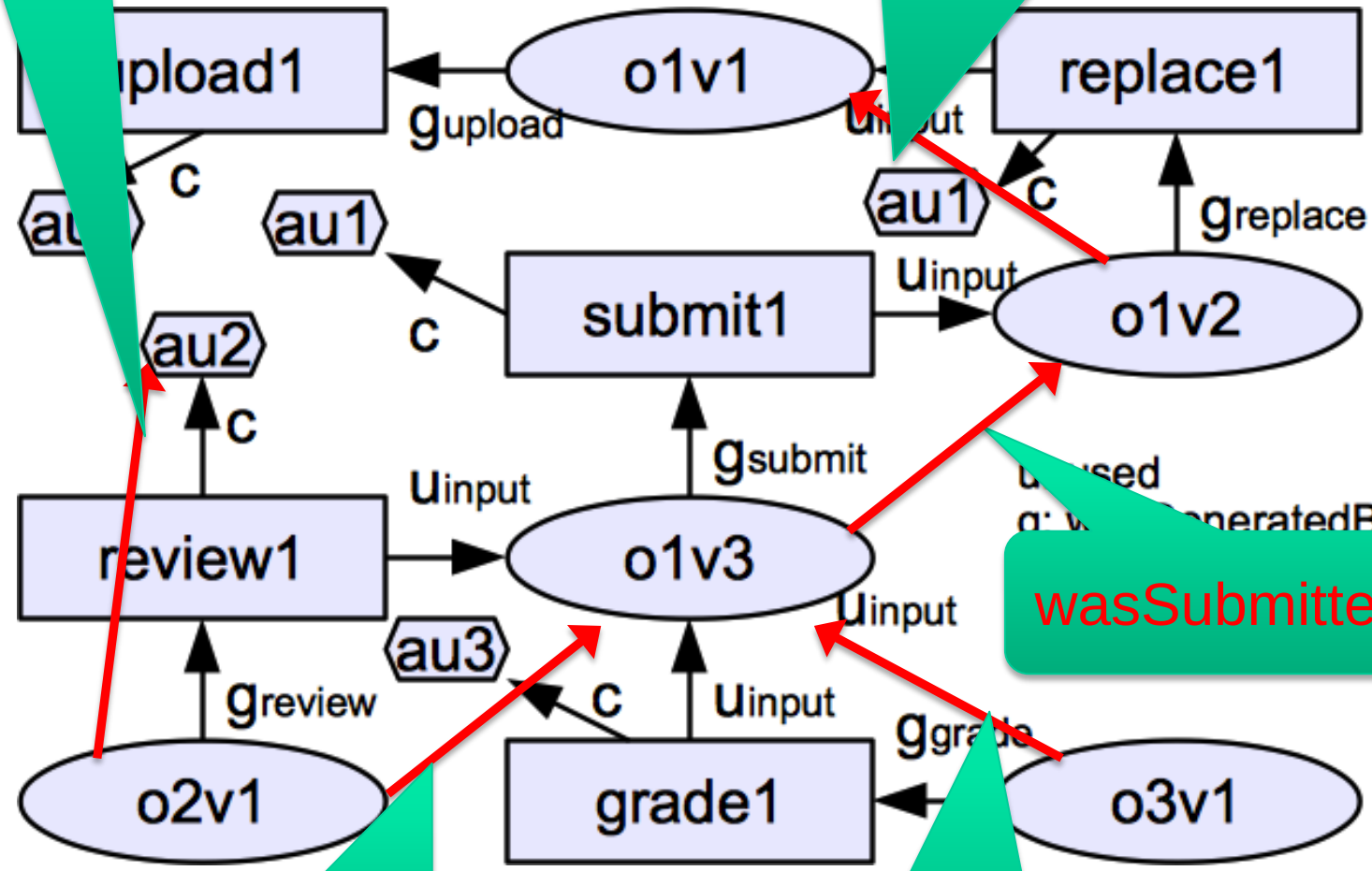
A Sample Base Provenance Data



Sample Object Dependency List (DL_O)

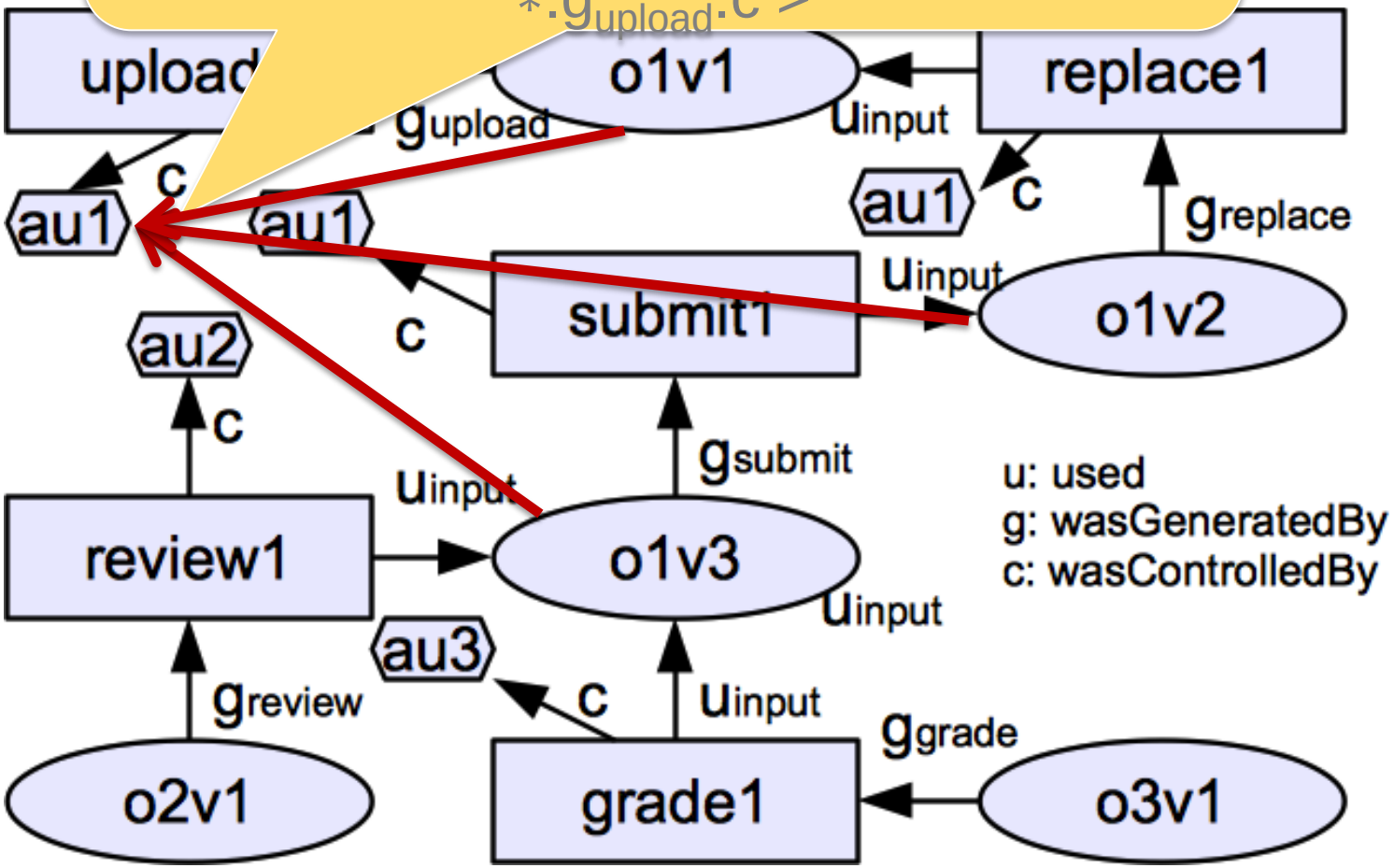
- < wasReplacedVof, $g_{replace}.u_{input}$ >
- < wasSubmittedVof, $g_{submit}.u_{input}$ >
- < wasReviewedOof, $g_{review}.u_{input}$ >
- < wasReviewedOby, $g_{review}.c$ >
- < wasGradedOof, $g_{grade}.u_{input}$ >
- < wasAuthoredBy, wasSubmittedVof?.wasReplacedVof $*$.gupload.c >
- < wasReviewedBy, wasReviewedOof−1. wasReviewedOby >

ample Base

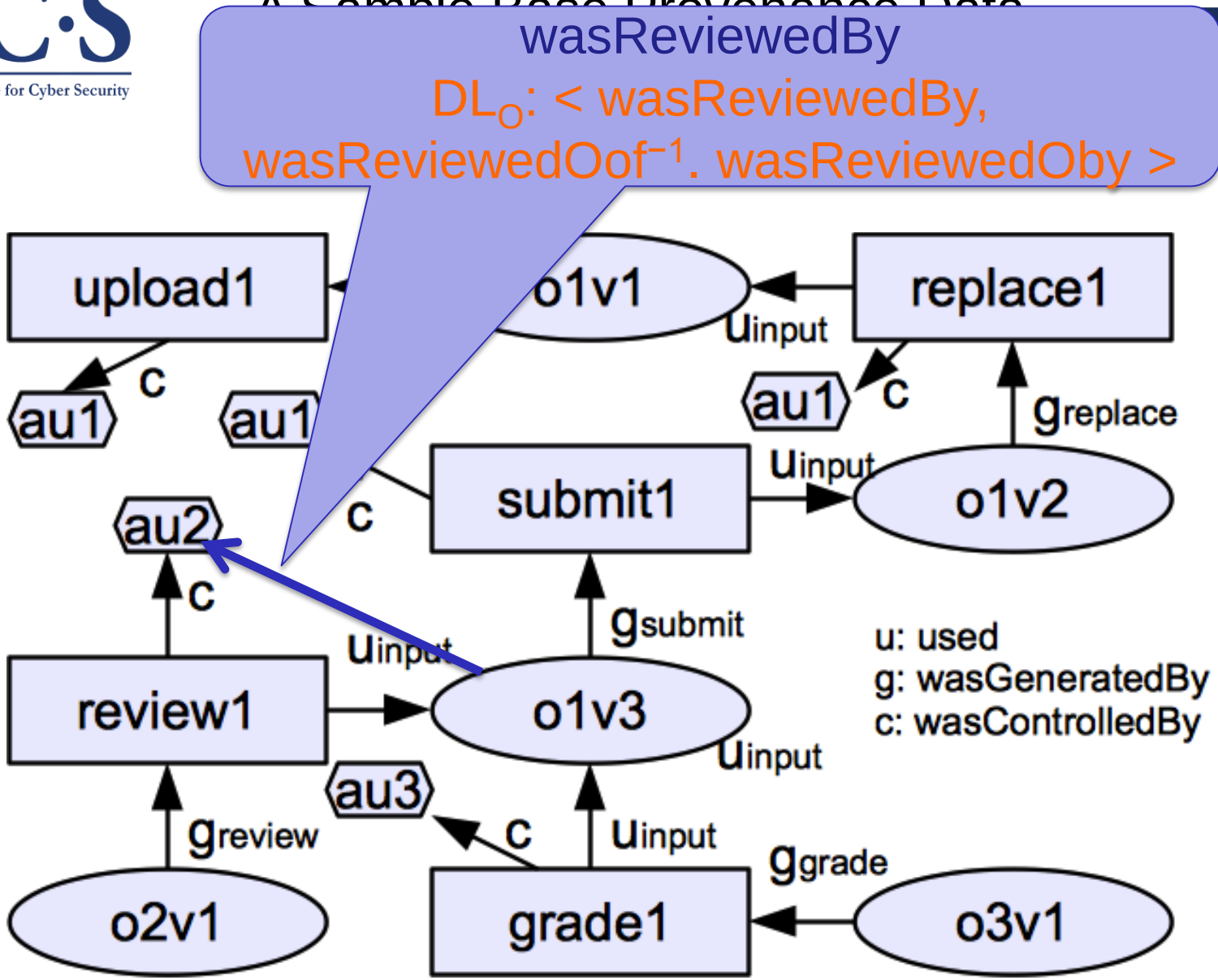


wasAuthoredBy

DL_O: <wasAuthoredBy, wasSubmittedVof?. wasReplacedVof $*.g_{upload}.c$ >



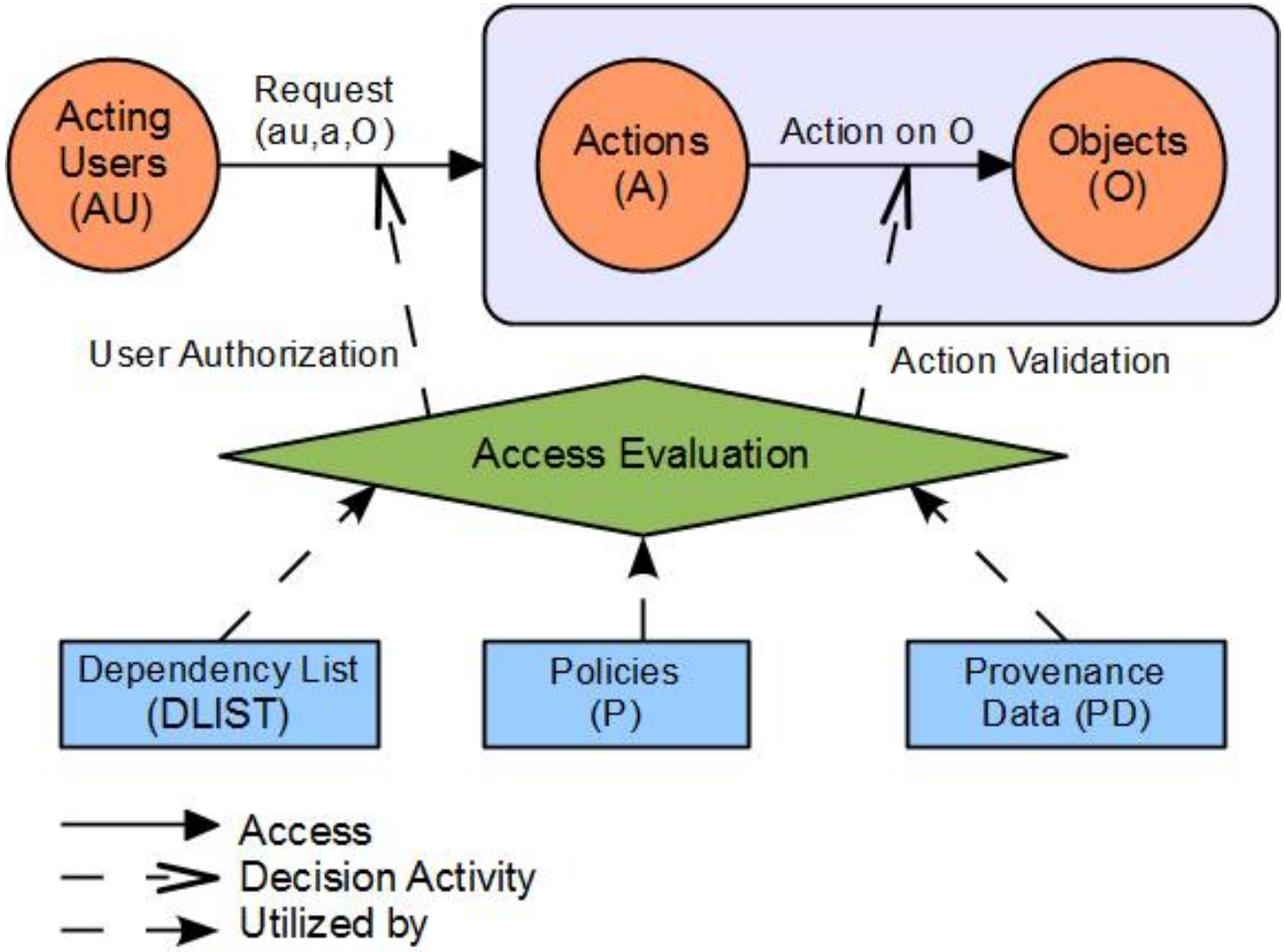
A Sample Base Provenance Data



A Homework Grading System

1. Anyone can upload a homework.
2. A user can replace a homework if she uploaded it (origin-based control) and the homework is not submitted yet.
3. A user can submit a homework if she uploaded it and the homework is not submitted already. (workflow control)
4. A user can review a homework if she is not the author of the homework (DSOD), the user did not review the homework earlier, and the homework is submitted already but not graded yet.
5. A user can grade a homework if the homework is reviewed but not graded yet.

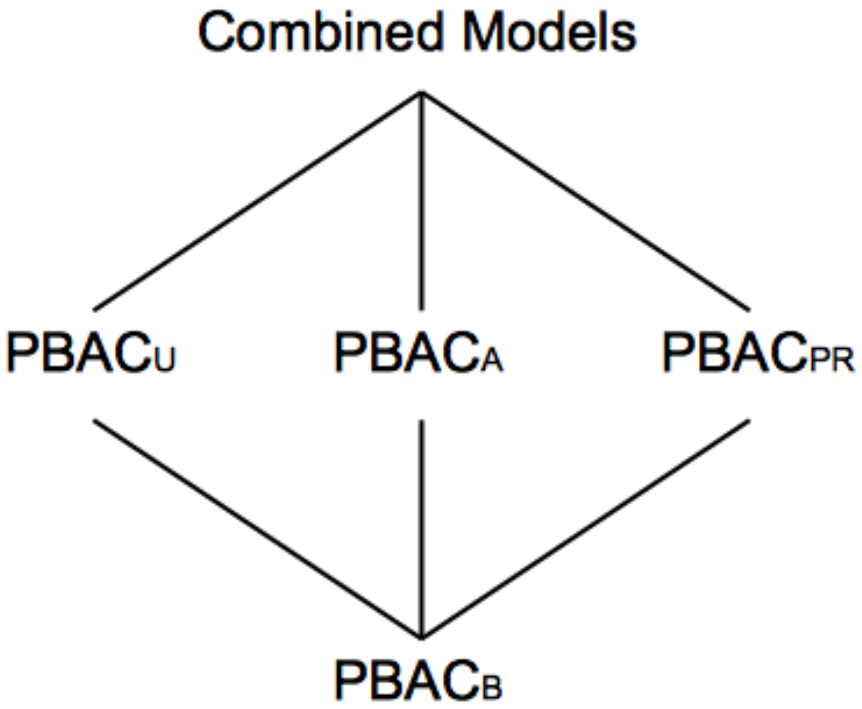
$PBAC_B$ Model Components



$PBAC_B$: A Base Model

- System-captured Base Provenance Data only
  - Using only 3 direct dependencies (u, g, c)
  - No user-declared provenance data
- Object dependency only
- Policy is readily available
  - No policy retrieval required

A Family of PBAC Models



PBAC Model Components