Adopting Provenance-based Access Control in OpenStack Cloud IaaS

October , 2014
NSS Presentation

Institute for Cyber Security
University of Texas at San Antonio

Cloud Computing

- Cloud computing has been the "next big thing."
- Has 3 primary service models:
  - Software-as-a-Service (SaaS)
  - Platform-as-a-Service (PaaS)
  - Infrastructure-as-a-Service (IaaS)
- We focus on PBAC for IaaS
  - Specifically, multi-tenant single-cloud systems.
  - OpenStack Nova / Glance.

Access Control Aspects

- DSOD concerns for virtual resources management and protection
  - Ex: Only virtual images up-loaders are allowed to delete.

- Multi-tenant concerns
  - A virtual image may be created in one tenant, copied to another tenant and modified, and used to launch a virtual machine instance in another.

Background: what is provenance?

Art definition of provenance

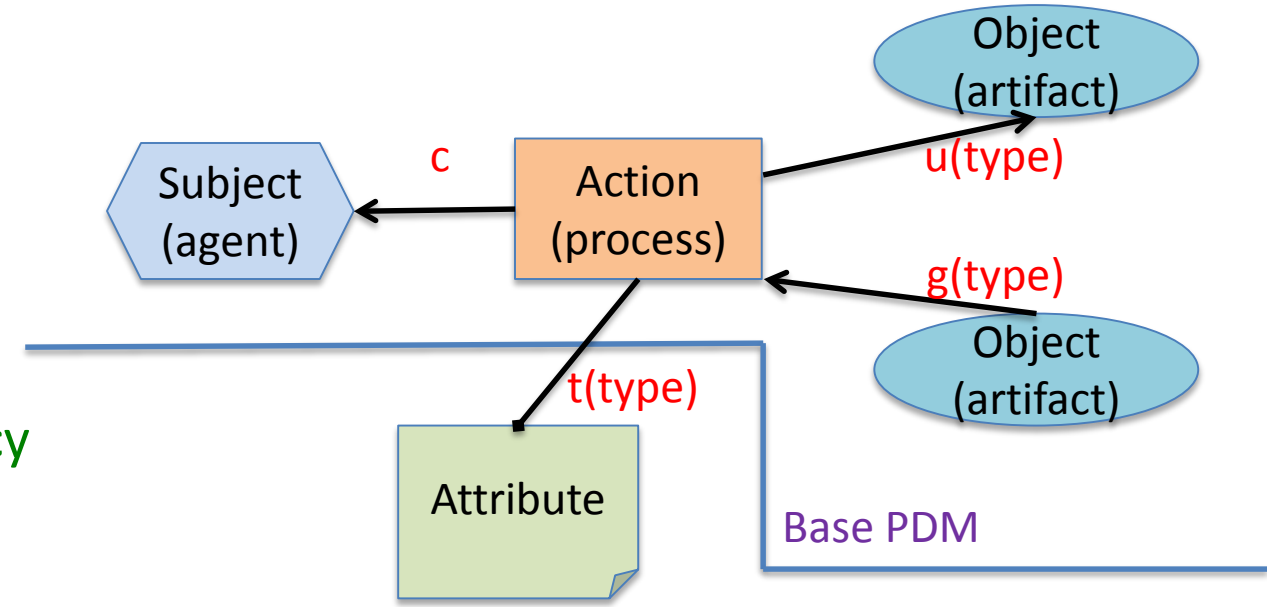
- Essential in judging authenticity and evaluating worth.

Data provenance in computing systems

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

Provenance Data Model
[inspired by OPM]

- 4 Node Types
  - Object (Artifact)
  - Action (Process)
  - Subject (Agent)
  - Attribute
- 3 Causality dependency edge Types
  (not a dataflow) and
  Attribute Edge



Contextual Extension

c	wasControlledBy
u	used
g	wasGeneratedBy
t	hasAttribute

→ Dep. edge

→ Attrb. edge

Inverse edges are enabled for usage in queries, but **cycle-avoidant**.

Dependency List

- Dependency List (DL): A set of identified dependencies that consists of pairs of
  - Dependency Name: abstracted dependency names (DNAME) and
  - regular expression-based dependency path pattern (DPATH)

- Examples
  - < wasModifiedVof, $g_{modify}.u_{input}$ >
  - < wasUploadedBy, wasCopiedVof?.wasModifiedVof $*.g_{upload}.c$ >

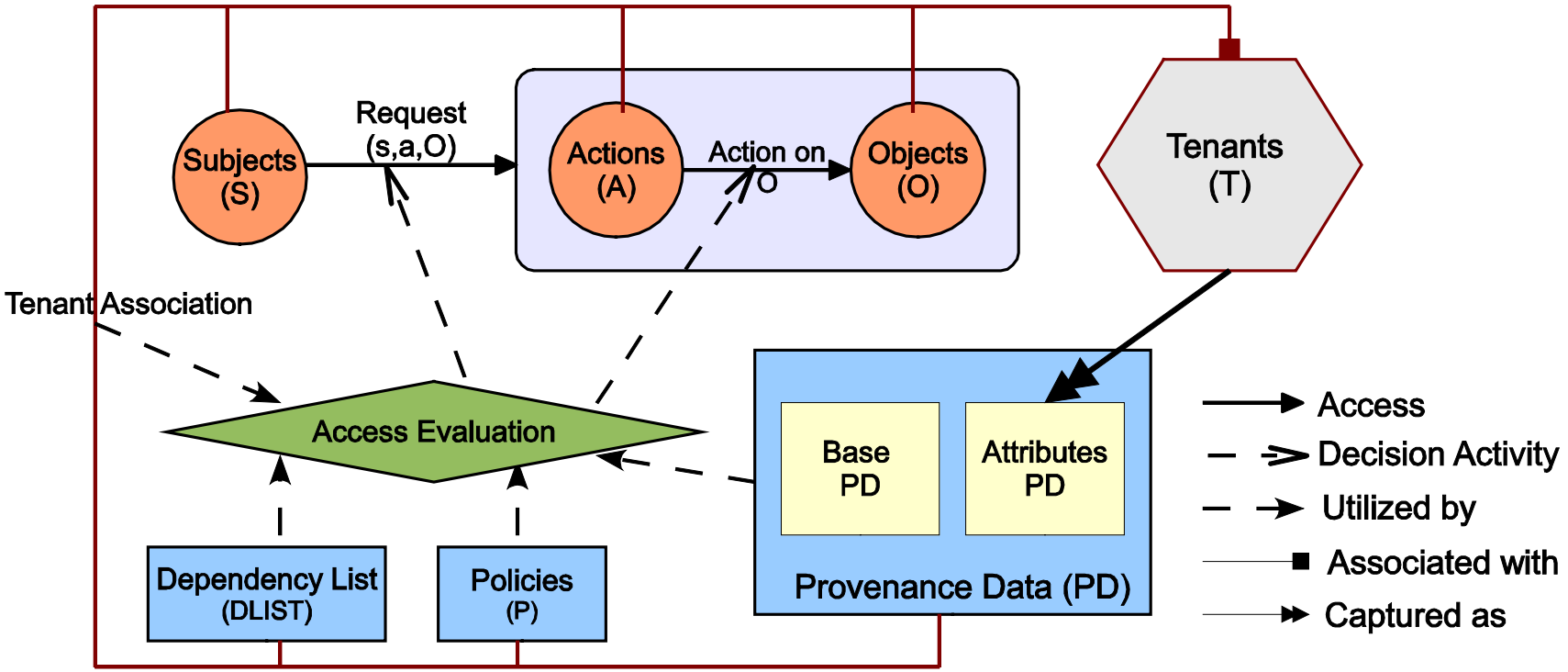
PBAC Models

- **$PBAC_B$**: utilizes base data model
  - Does not capture contextual information

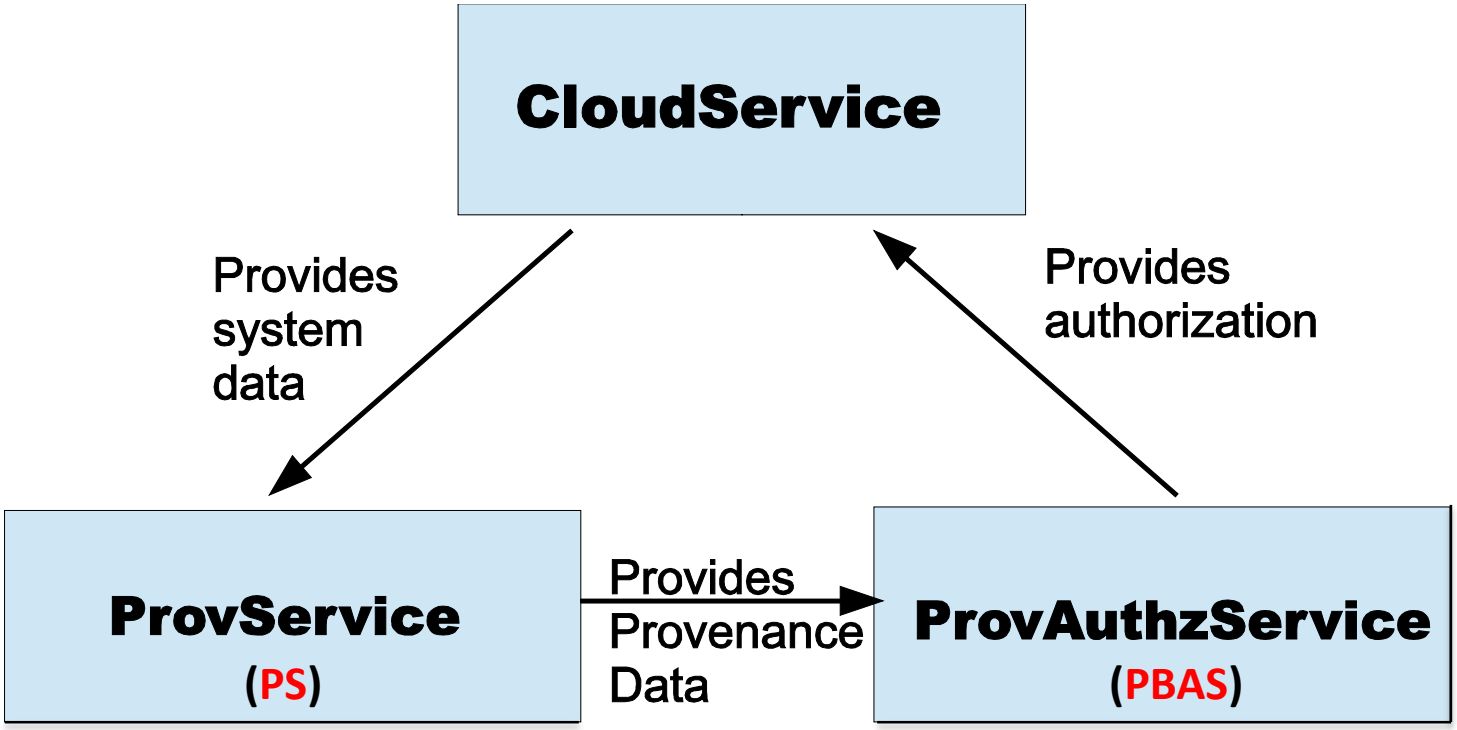
- **$PBAC_C$**: extending the base model
  - Incorporate *contextual information* associated with the main entities (**Subjects**, etc.)
  - Extend base data model with attributes

Tenant-aware PBAC

Tenants as contextual information.



Architecture Overview



Deployment Architecture

Variations:

- Integrated Deployment
- Stand-alone Deployment
- Hybrid Deployment

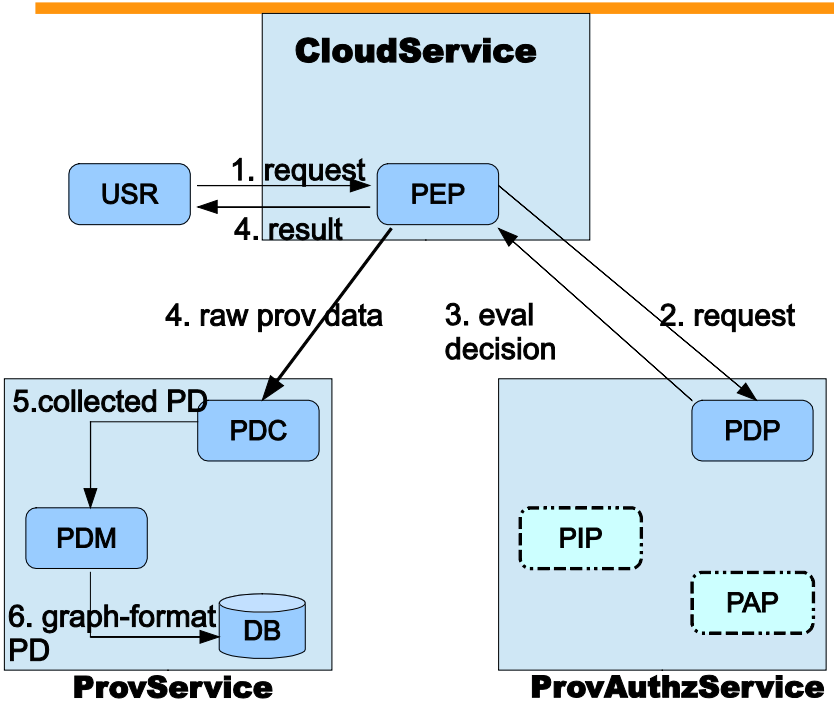
Design pros & cons:

Ease of integration -

Communication latency -

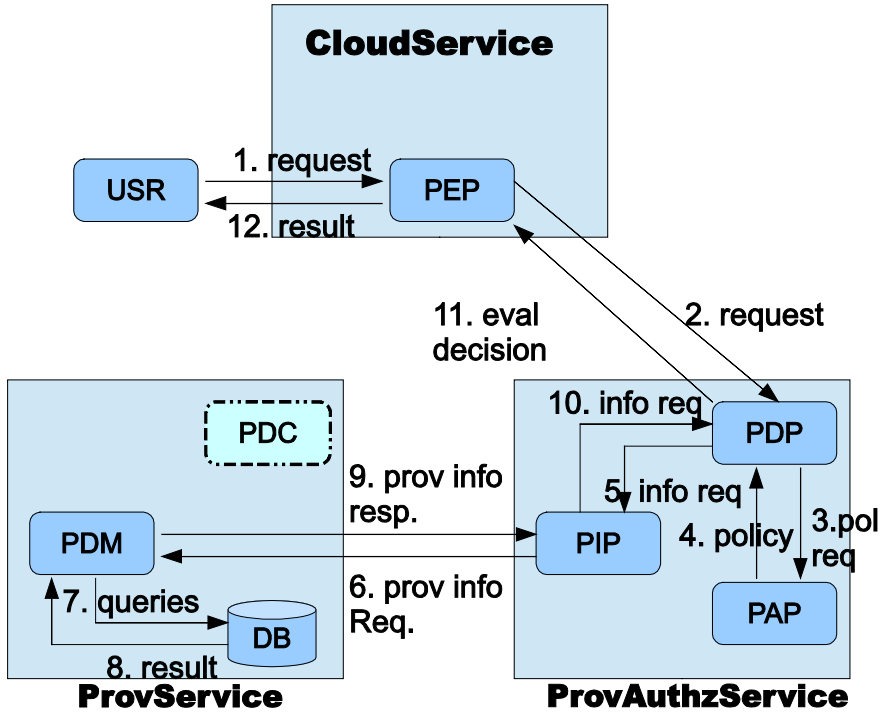
Provenance data sharing -

Logical Architecture

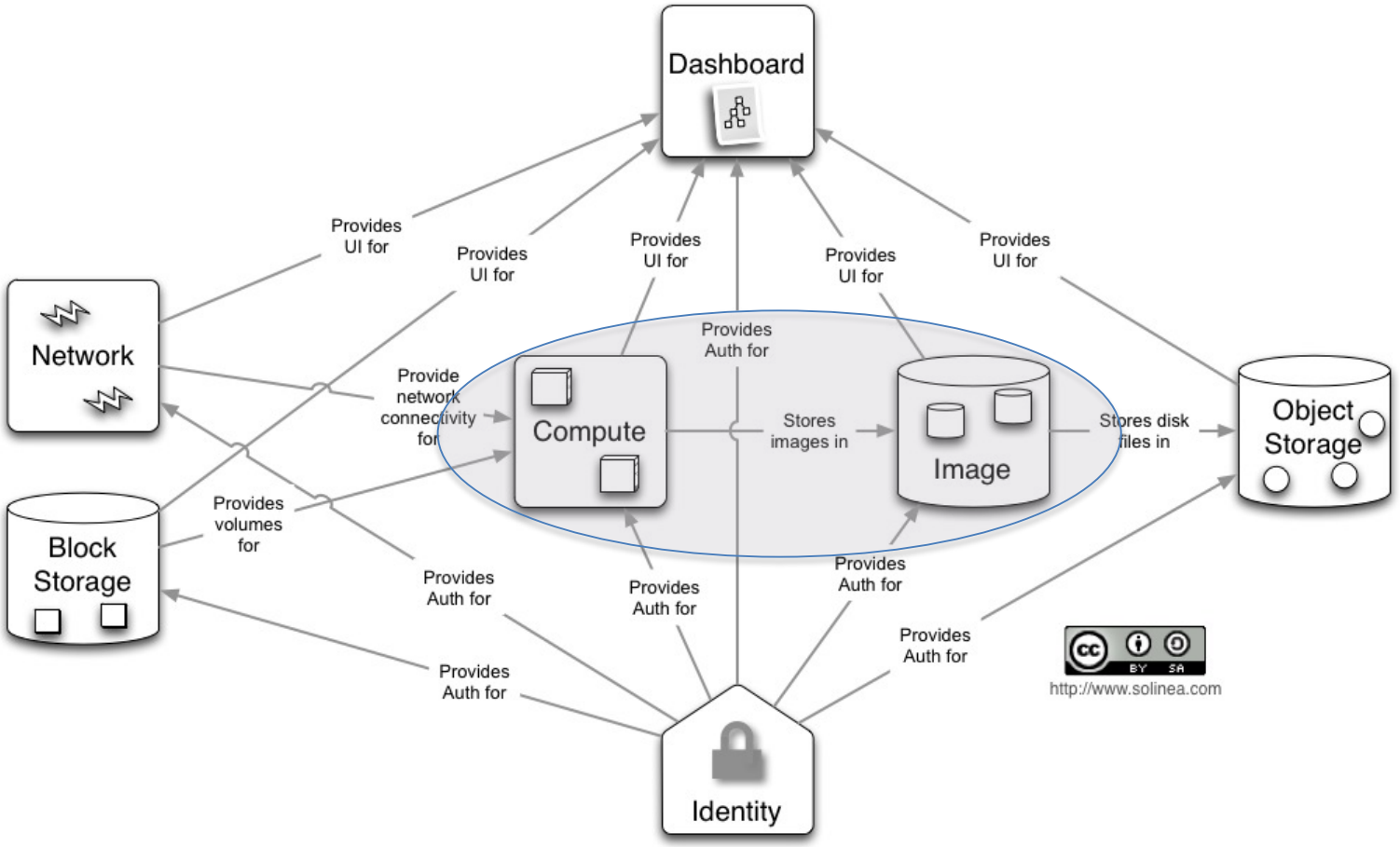


PROV-SERVICE
Dataflow

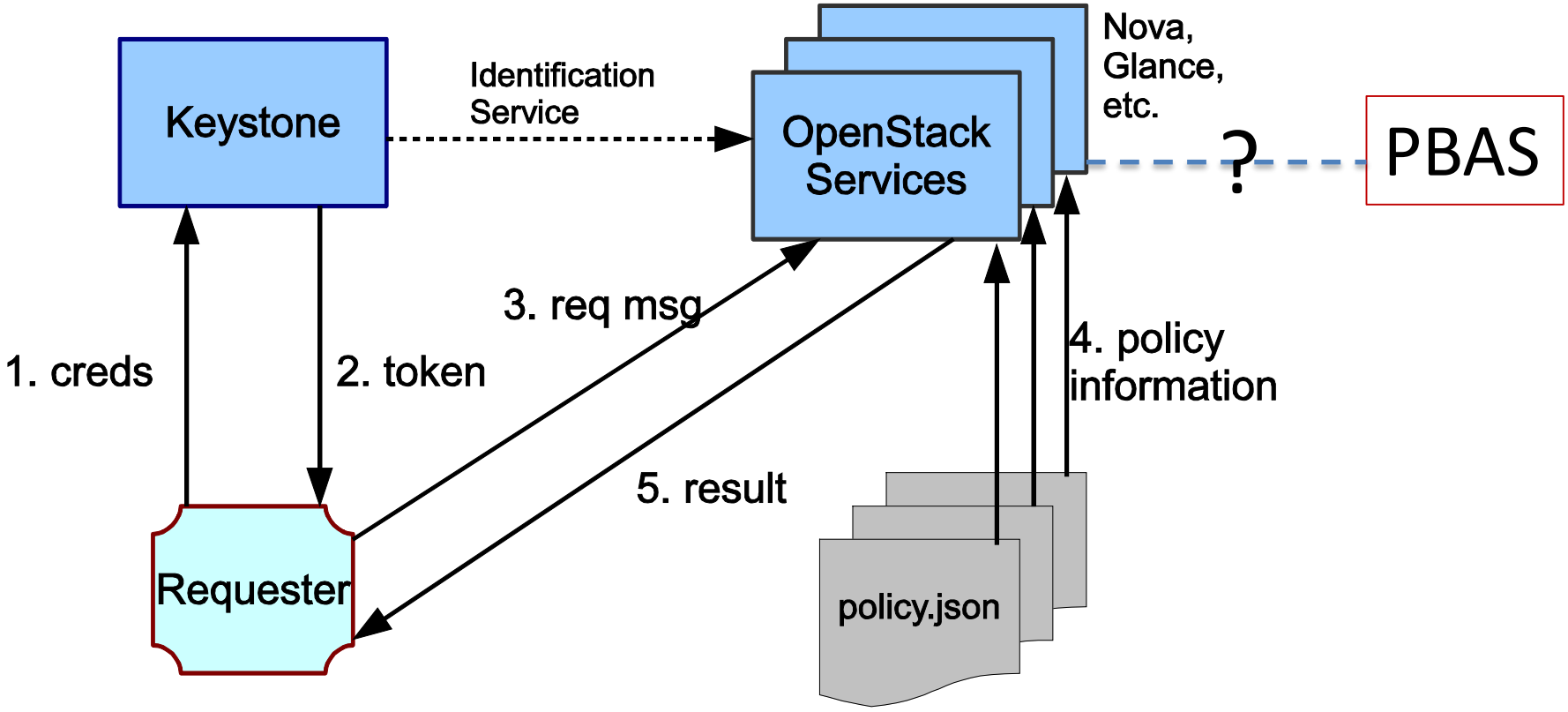
PROVAUTHZ-SERVICE
Dataflow



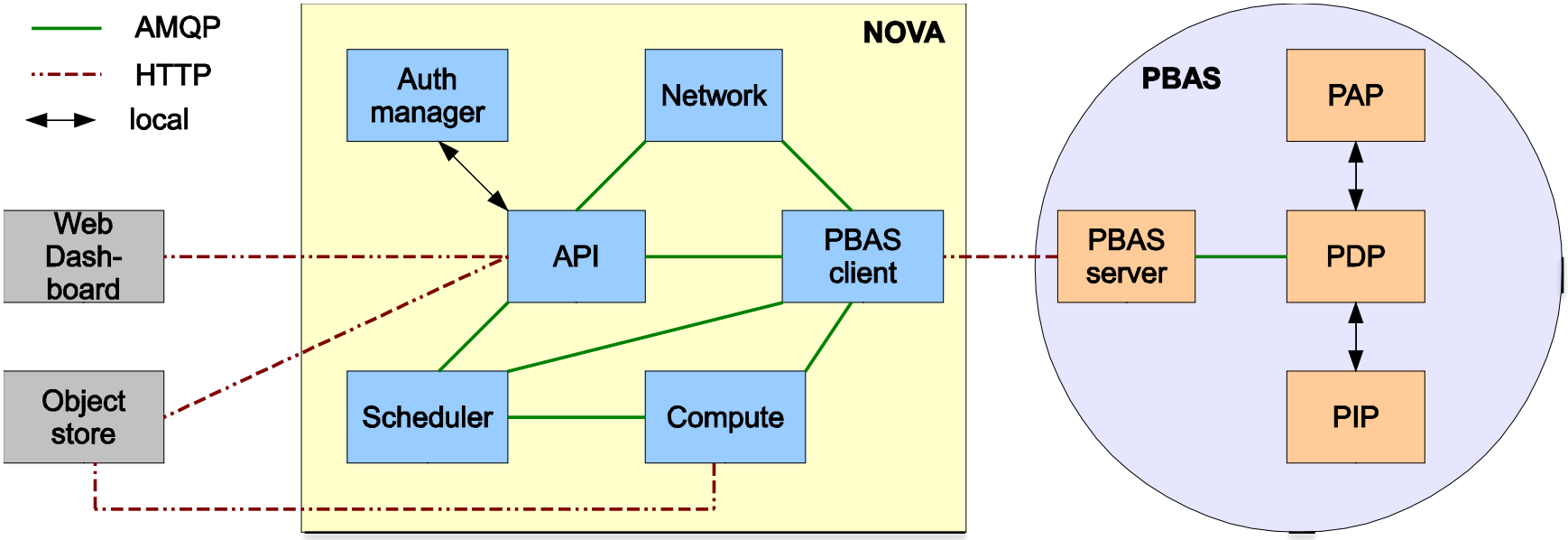
OpenStack Conceptual Architecture



OpenStack Authorization



Nova PBAS Implementation



Experiments

- Measure the time an authorization process takes from the time of request until decision is returned.
  - nova list
  - glance image-list
- 4 experimental configurations:
  - E1: normal Nova and Glance authorization.
  - E2: integrated PBAS/PS services with Nova and Glance.
  - E3: integrated PBAS/PS service, stand-alone from Nova and Glance.
  - E4: separate PBAS and PS services, stand-alone from Nova and Glance.
- Deployment Configurations:
  - 4GB RAM, 2.5 GHz quad-core CPU.
  - OpenStack Devstack (Grizzly) on 12.04 Ubuntu.
- Mainly test deep-shaped provenance graphs.
  - Generate mock data for virtual images and machines scenario.

Results and Evaluation

Traversal Distance	Glance (e1)	Glance (e2)	Glance (e3)	Glance (e4)
No PBAC	0.55	-	-	-
20 Edges	-	0.575	0.607	.642
1000 edges	-	.612	.788	.852

Traversal Distance	Nova (e1)	Nova (e2)	Nova (e3)	Nova (e4)
No PBAC	0.75	-	-	-
20 Edges	-	0.84	0.902	1.062
1000 edges	-	2.292	.362	4.102

Future Work and Directions

- Expanding provenance data model to include **user-declared** provenance data.

- **Collaborated** PBAC usage
  - Multi-cloud.
  - Distributed systems.

- Full-cycle implementation and evaluation
  - including **provenance capturing** service.

Thank you!!!

Questions and Comments?