DUMA Workshop
San Jose, USA

Architecture, Workflows, and Prototype for Stateful Data Usage Control in Cloud

Aliaksandr Lazouski, Gaetano Mancini, Fabio Martinelli, Paolo Mori

Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche

May 17th, 2014
Agenda

- Motivation: Cloud and Data Security
- Usage Control Model
- Prototype
  - Architecture
  - Data Usage Policy
  - Management of Mutable Attributes
  - Implementation
- Conclusions
EU Projects

• New challenges for data usage control
  • COCO aims at allowing the cloud users to securely and privately share their data in the cloud and the mobile
• CNR past involvement
  • NESSOS
  • contrail open computing infrastructures for elastic services
CoCoCloud Pilots

- Public Administrations
  - PA1
  - PA2

- e-Health
  - User
  - Hospital X
  - Hospital Y
Motivation

- Distributed environment
  - Multiplies copies of data objects (DOs) in distributed environment
    - DOs can be replicated
    - DOs can move outside originator's domain
    - Accesses to DOs are long-lasting
    - Regulate usage of many DOs
- Policy Example
  - A user is allowed to make X replicas of DO and all these DOs must reside in a certain geographic area until deleted
  - Consider the most appropriate policy enforcement mechanism
Usage Control Model

- Based on UCON model of R. Sandhu et. al.
- Mutability of attributes
  - Stateful usage control
- Continuous Policy Enforcement
  - Revocation of access rights
- Distributed authorization infrastructure for usage control
  - Concurrent management of attributes
  - Data usage control policy based on U-XACML language
If policy is violated, access is revoked
U-XACML Policy

- We extend XACML language and enforcement architecture to implement UCON features
  - Continuous evaluation (adding specific attribute to the <Condition> Element)
  - Attribute Updates
    - via Obligations in XACML 3.0
Distributed Architecture

- SM (Session Manager)
- LM (Lock Manager)
- AM (Attribute Manager)
Allocation of Components

- Many Policy Decision Points (authorization systems) may exist in distributed settings
  - Who makes access decision?
- Many Policy Information Points
  - Where to store attributes of data?
Data Usage Policy

- PDP Allocation Policy
  - Which PDP/AS to use for access decision
    - based on trust, performance, compliance, etc.
    - evaluated and enforced by PEP
- PIP Allocation Policy
  - Who to query for mutable attributes
    - only 1 location for a given mutable attribute
    - concurrency control on attribute queries
    - evaluated and enforced by CH
- U-XACML policy
  - Access and Usage policy
  - evaluated by PDP, enforced by PEP
Concurrent Management of Attributes

- Many PDP/AS may request to retrieve/update attributes concurrently
- Race Conditions
  - Inconsistent retrieval
  - Lost update
    - Policy: It is allowed to make 5 replicas of DO in the system
    - User 1 and 2 requests to make a DO replica
    - Decision process 1 loses an attribute update at t4. The update at t3 is lost (overwritten) at t4 by decision process 2
    - Both requests are granted – policy violation

<table>
<thead>
<tr>
<th>t</th>
<th>User1 (PEP1)</th>
<th>User2 (PEP2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Retrieve n=4</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Retrieve n=4</td>
</tr>
<tr>
<td>3</td>
<td>Update n=5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>Update n=5</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Concurrent Management of Attributes (2)

- Distributed Two-phase Locking Protocol
  - All attributes are locked before updating/evaluating the UXACML policy. Wait until an attribute is locked by another decision process.
  - Locks are released afterwards
- Deadlock occurs when 2 or more decision processes are in a simultaneous *wait state*
  - Attributes are locked in a fixed predefined order: \( #a < #b < #c \)
  - Such protocol is deadlock free
Implementation Notes

- Authorization System (CH, PDP, Session Manager, AT)
  - Stateful Web Service in Axis2 (for Cloud), Android App (for mobile)
  - Protocols: SOAP-SAML-XACML
  - Extended Balana PDP engine for UXACML evaluation
- PEP
  - Cloud: OpenNebula
    - DO is Virtual Machine Image
    - Data usage policies are placed into the OVF package of VMI
  - Mobile device: Android
    - DOs are Activity, Service, Content Provider, internal resources of Apps
- Concurrency Control
  - PIP is Web Service in Axis2
  - AM and LM - PostgreSQL
Conclusions

- Stateful Data Usage Control based on UXACML model and XACML standard
  - Prototype enforces the policy by monitoring (and updating if needed) attributes of all peers that possess the DO copies in the system
- Components allocation policies and concurrency control on mutable attributes
- Reference architecture and initial implementation for Cloud and mobile
Future Work

• Relax assumptions on the trustworthiness of data consumers enforcing data usage policies

• Test prototype in real test-bed with a large number of distributed components

• Study what policies can be enforced and prove properties that our design holds
Thank you!

• Q?