Obligation Standardization

David Chadwick, University of Kent

Mario Lischka, NEC Laboratories Europe
Problems with Existing Model

- Obligations have not been handled fully, they are simply attribute assignments which are consumed by an Obligations Service and must be evaluated simultaneously with the user’s action.
- In reality we have 3 different temporal types of obligations, Before, With and After each with their own semantics.
- Some obligations need handling by remote systems.
- Thus multiple places for obligations to be enforced.
- Many obligations are application independent so don’t need to handled by the application dependent PEP.
- Syntax and semantic of obligations are not specified.
- Conflicts among obligations are neither specified nor handled.
Contribution/Proposal

- Obligation Conceptual Model
  - Obligation subject
  - Temporal constraints
  - Fall backs
- Obligation specification & conflict resolution
Problems with existing XACML obligation model and language

- No standardised parameters for conceptual entities
  - Subject to perform obligation
  - Action to be performed
  - Target of obligation
  - Constraints?
  - Failure Semantics
- No temporal positioning of the obligation
  - Before, With or After the user’s action
- No failure semantics
  - If obligation fails then Exception/Fall backs/Final Decision
- No ability to direct the obligation to an enforcement subject
- No ability to have delayed obligations
  - Do X in one week’s time
  - PEP still needs acknowledgment that the obligation has been recorded
0. User's request

1a. Will enforce Authz Decisions

1b. Will validate presented credentials and pull more

2a. Will coordinate "before" obligation enforcement

2b. Will validate presented credentials and pull more

3. AA

4. CVS

5. Will evaluate each policy according to the languages they support

6. App Indep PEP

7. Policy

8. Policy

9. Policy

10. Will Enforce Conflict Resolution Policy

11. Will Enforce Conflict Resolution Policy

12. AppDep PEP

13. Target Resource

14. Obligations Service

Obligations Service

Obligations Service
Direct communications

Indirect communications via sticky policies

Site A

Site B

Site C
Obligation Specification

Obligation on granted (or denied access) are recognized in XACML, but specification is completely open

→ Problems as Obligations Service (PEP) and Obligation Writer (PDP/PAP) should agree how to handle Obligations

- Schema which allows the description of obligations
- Specification of generic obligations used in the policies
- Negotiation of supported obligations between PDP and PEP

Examples for Types of Obligations
- resource control (read/write locks, logging)
- Obfuscation/Transformation
- Dynamic process workflow
Initialization of Obligation Handling

- PEP/Obligations Service and PDP can agree on common obligations supported by each side
- pre-check of supported obligations

- PDP does not have to analyze the obligation semantic/ only works on syntactical level
  - PDP can verify the support of obligation during parsing of the policies
  - implementation parsing verification could be done generic i.e. independent of the obligations used in any instance
  - additional obligation could be specified without modification of the PDP
Conflicts

- **Relation between Obligations**
  - Unrelated
  - Conflict
  - Contradiction
  - Inclusion
  - Subordinated/super-ordinated

- **Conflict between two obligations defined on**
  - as general
  - partial obligation values
  - all obligation values

- **Conflict detection**
  - on the set of combined obligations
  - Based on Specification of conflicts

- **Generic Conflict resolution for**
  - Inclusion
  - Subordinated/super-ordinated
Conclusion

- temporal types of obligations (Before, With and After) are required
- Obligations handling by remote systems, enforcement at multiple places
- Syntax and semantic of obligations and relation among them have to be specified