Providing Secure Coordinated Access to Grid Services

David Chadwick, Linying Su,
Romain Laborde

University of Kent

d.w.chadwick@kent.ac.uk

 © 2006 University of Kent

Contents

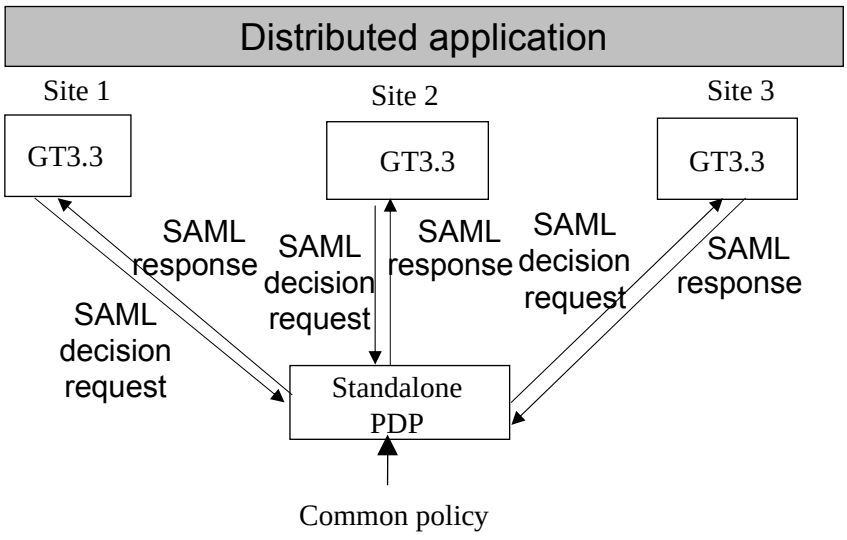
 © 2006 University of Kent

Motivation/Problem Statement

- Sometimes one access control decision depends upon prior decisions
  - E.g. You can only draw £250 from ATM machines in a day
  - E.g. You are only entitled to use 5GB memory per grid job
- Decision may depend upon previous decisions at the same or different resources in the distributed system
- Relatively easy to solve if only one PDP is involved
  - Have a stateful PDP
  - Already implemented in Globus Toolkit, from v3.3 onwards using GGF Specification "Use of SAML for OGSI Authorization"

© 2006 University of Kent

Current Implementation



© 2006 University of Kent

Disadvantages of Current Solution

- Requires a stateful PDP, but most PDPs today are stateless
- In many VOs and grids there are multiple PDPs
  - So we need to share state information between them all
- Conclusion. We would like to provide a stateless solution for both single and multiple PDP systems

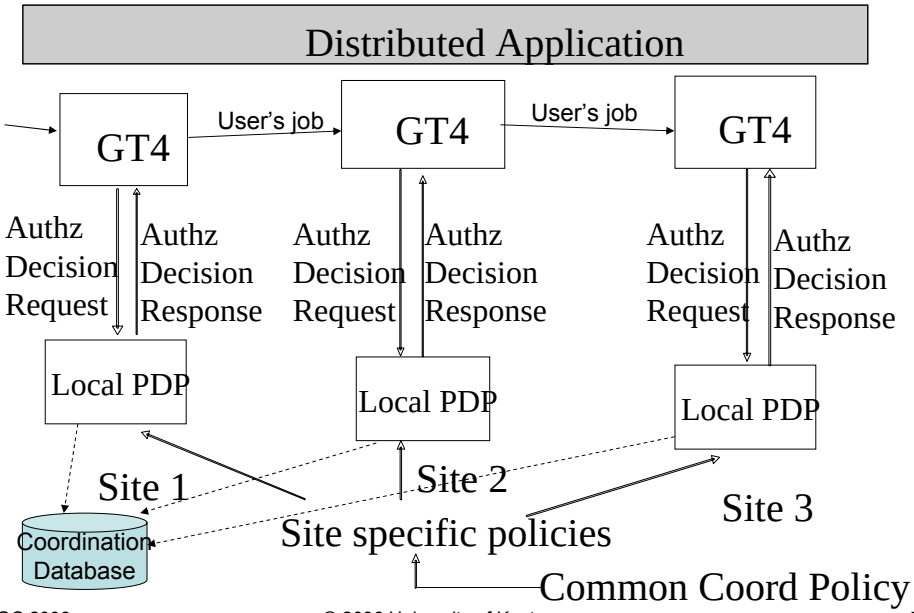
© 2006 University of Kent

Conceptual Solution in Brief

- Store state information in coordination attributes of a coordination object
- Introduce a coordination policy for the distributed application (as part of the access control policy)
  - Access control decisions depend upon values of these coordination attributes [as well as subject, resource, action and environmental attributes]
  - Obligations are used to update these coordination attributes
- Specify how to refine the coordination policy for each resource when multiple PDPs are involved
- Implement coordination object and attributes in a database grid service (DB provides stable storage, fast lookup, distribution, replication etc.)

© 2006 University of Kent

Conceptual Solution



 © 2006 University of Kent

When is Coordination Required?

- Policy constraints are normally placed on Subject attributes, Action attributes, Resource attributes and/or Environmental attributes. E.g.
  - only Fred or Mary but not both ….. (subject con)
  - cannot create an exam paper and answer it….(action con)
  - use more than 10GB of memory in total in the grid…(resource con)
  - If you enter before 9am you cannot stay after 3pm…. (environment con)
- If any access control decision will produce state changes in the history of these attributes (known as *Retained Access control Decision Information* in ISO 10181-4) which will affect future access control decision, then coordination between the access control decisions is required.

 © 2006 University of Kent

The Coordination Object

- A persistent and stateful object holding multiple attributes about the subject, resource, action and environmental state
  - Similar to environmental object and environmental attributes
  - The difference is that the PDP only needs to read environmental attributes but needs to read and update coordination attributes
- Coordination object may be distributed or centralised, replicated or single copy, but this is irrelevant from an access control and policy perspective (its an implementation issue)
- PDP does not need to know semantics of coordination attributes, just how to compare then
  - PDP does not know semantics of any of the attributes (subject, resource, action or environmental)

© 2006 University of Kent

Notation for Coordination Attributes

- Notation
  - att(O) means attribute of object O
  - E.g. role(S) means role of subject, date(E) means current date
- We add constraining dimensions for subject, action, resource and environment to coordination object (C)
- att[SubDim, ResDim, ActDim, EnvDim](C)
  - E.g. usage (C)
  - means coordinate usage for all subjects accessing all resources for all actions over all environments
  - E.g. usage[{date(E)}](C)
  - means coordinate usage on each date for all actions by all subjects accessing all resources
  - E.g. usage[{birthDate(S), lastName(S), firstName(S)}, {date(E)}](C)
  - means coordinate usage on each date by subjects identified by their first name, last name and date of birth for all actions on all resources

© 2006 University of Kent

Access Control Policies with Coordination

- Simply include the coordination attributes into logical expressions of access control policies
  - E.g. users, identified by their userIDs, cannot use more than 3GB of store each
  - type(R)=storage ^ type(A)=use ^ (amount(A)+storeUsage[{userID(S)}](C)≤3)
- Pass coordination attributes to PDP along with the subject, action, resource and environmental attributes. PDP does not understand semantics of any attributes or the syntax of their names. Its just a string name.

© 2006 University of Kent

Updating Coordination Attributes

- We add obligations to the policy that tell the PEP what actions it must undertake when a coordinated action is granted
- We have a single parameter, Chronicle, that tells the PEP when to perform the obligation (either before, simultaneously with or after the user's action is enforced) at the choice of the policy writer

© 2006 University of Kent

Implications of Chronicle parameter

- *Before* means that a *denied access* is possible if the original access fails
- *After* means that a *lost access* is possible if the coordination update fails, but preferable for some applications such as ATM withdrawals
- *With* is the most accurate, but it depends upon the PEP's capabilities for supporting transactions and two phase commit, and is impractical for grid jobs that last for hours
- Note. XACML does not have the Chronicle concept, and *With* is implied for all obligations

© 2006 University of Kent

Coordination Database

- Implemented in a MySQL database
- Each coordination attribute becomes one table of size (|SubDim|+|ResDim|+|ActDim|+|EnvDim|+1)xN
  Where N is the no of rows ≡ no of different accesses
- E.g. The coordination table (called balance) for balance(C[{userID(S)}]) after 2 user accesses

userID(S)	value
CN=fred,O=kent,C=uk	0.5
CN=mary,O=huhhot,C=cn	1

© 2006 University of Kent

Looking up the Coordination Data

- Assume a subject (id = X) wants to access a file (mode = M, fileName = F) on a resource (id = Y), with restrictions on numberOfAccesses per subject per access mode per file per resource per day
- The current value of the numberOfAccesses may be located from table T (=numberOfAccessess table) by the following SQL command: SELECT value FROM T WHERE id(S)=X AND id(R)=Y AND mode(A)=M AND fileName(A)=F AND date(E)=<today's date>

 © 2006 University of Kent

Coordination Database Service

- Supports 7 methods
- checkWS - checks if the service is available or not
- getCoordAttrVal - returns the value of a coordination attribute
  - If no value currently exists the service creates a new one initialized with a value specified in the coordination object database schema
- setCoordAttrVal - sets the value of a coordination attribute
- isCoordAttr - queries if a coordination attribute of this name exists in the coordination database
- getAttributeDefinition returns an XML element which contains the definition of the coordination attribute including the attributes that are embedded in its name
- lockCoordAttrs – read or write locks multiple attributes in the database
- unlockCoordAttrs - removes all the locks in the database held by the current thread

 © 2006 University of Kent

Knowing which Attribute Values are needed by the PDP

- Three possible choices (at least)
- Configure PEP with the list of attributes needed by the PDP for this application and policy (implemented by GT4)
- PEP calls a getAttributes method on PDP at initialisation, which returns the set of attributes needed by the current policy (our choice)
- PDP returns the set of attributes needed by the current authorisation decision request (adopted by XACML)

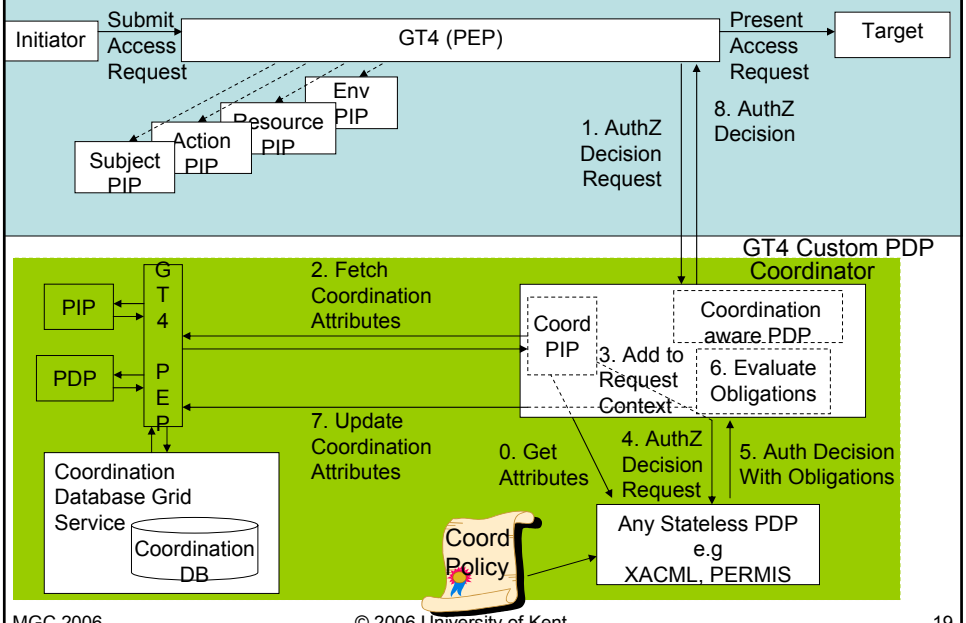
 © 2006 University of Kent

Securing Access to Coordination Database Service

- Make it a grid service, and use Grid Security Infrastructure to protect requests and responses
- Control access to DB service by having its own standard RBAC PDP
  - Only role Coordinator has read and write access to the coordination database service
  - Allocate role of Coordinator to PEPs of Application Grid Services

 © 2006 University of Kent

Implementation in GT4



 © 2006 University of Kent

Obligations Service

- Has 3 methods
- getChronicle – returns value of Chronicle parameter
- evaluateObligation – evaluates attribute assignments

  e.g. balance[{id(S)}](C) + amount(A) ← balance[{id(S)}](C)
- performObligation – updates Coordination Database

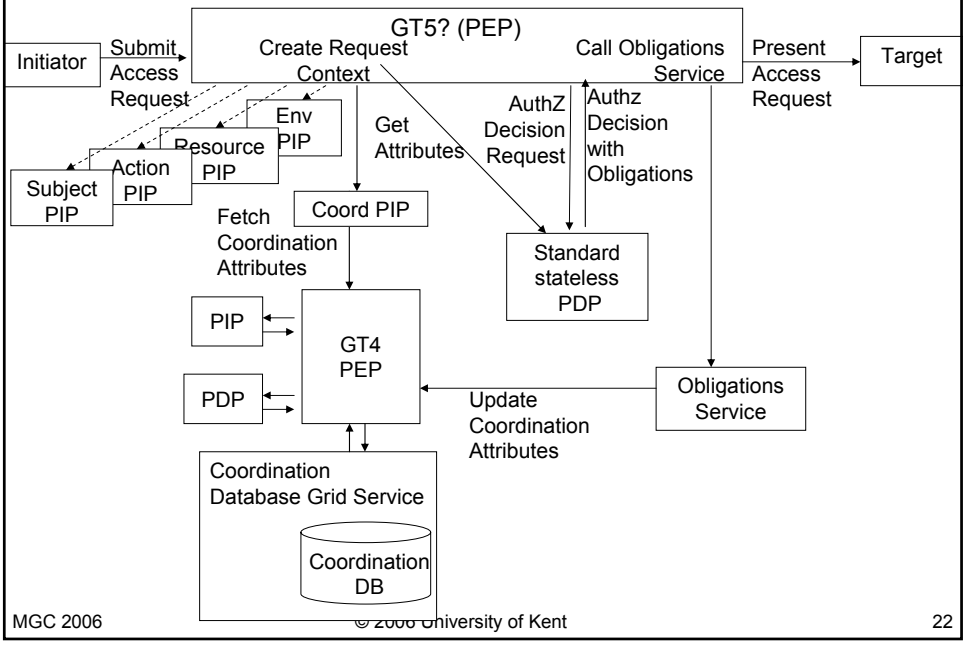
 © 2006 University of Kent

Limitation of Current Implementation

- Only supports Chronicle=Before
- Can be resolved by more tightly integrating coordination into GT4
  - Call Coordination PIP directly from GT4 PEP
  - Add support for obligations to GT4 PEP

© 2006 University of Kent

Future Integrated System



© 2006 University of Kent

Any Questions?

- ??????????????????????????
  ??????????????????????????
  ??????????????????????????
  ??????????????????????????
  ?????????????????????????
  ????? 

© 2006 University of Kent