###-### title - PI - area

CERIAS

the center for education and research in information assurance and security

Realizing Privacy-Preserving Features in Hippocratic Databases

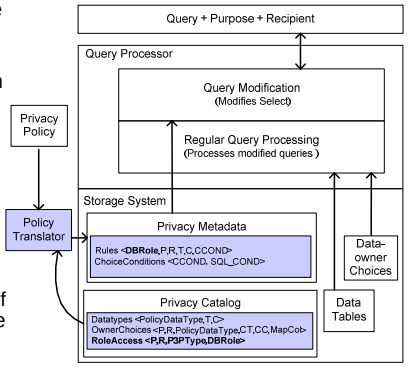
Yasin Laura-Silva and Walid G. Aref

The Privacy problem and Hippocratic Databases (HDB)

- Companies need to comply with privacy laws
- How to manage/share information without violating privacy policies and data owner preferences?
- HDB has privacy as a core principle. It allows automated, fine-grained data disclosure at the database level
- There are still several problems that need to be addressed before HDBs can support efficiently the requirements of real-world systems
  1. Inadequate support of policy retention time
  2. Lack of support of policy versions
  3. Lack of an effective and flexible way to ensure that users only use purposes and recipients that they are supposed to use
  4. Lack of a way to restrict access to DML operations other than SELECT

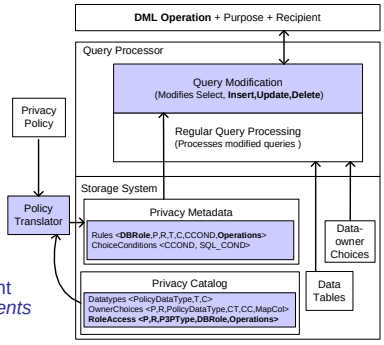
1. Mapping purpose, recipient, and data type of a policy with DB roles

- In previous work all the rules translated from P3P to the DB are assigned to the role Public independently of the Purpose-Recipient pair of the rule
- In the real world, a DB user/role should use only certain combinations of Purpose-Recipient pairs
- We propose to use the relationship between purpose-recipient-data type and database roles during privacy policy translation
- This mapping can be viewed as a way to specify the database roles that can access specific sections of the data using a particular combination of purpose and recipient
- After policy translation, each role will have its own set of rules only for those (P,R) pairs that it is supposed to use



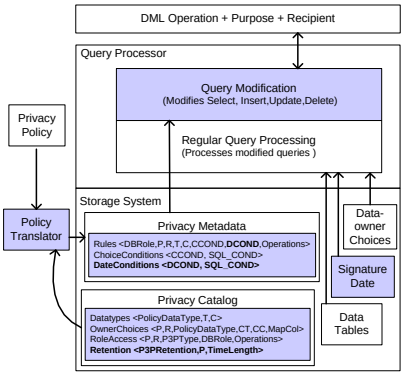
2. Support of multiple DML operations

- Previous work only focuses on the SELECT operation
- Our contribution to support multiple DML operations includes:
  - The study of the semantics of privacy rules and preferences for other DML operations
  - The algorithms to implement these operations
- With the two first extensions, we are able to enforce restrictions like:
  1. User Mary should use only recipient *Doctors* when accessing table *Patients* for the purpose *Treatment*
  2. For purpose *Treatment* and recipient *Doctors*, allow *sysadmin* to access all the columns of table *Patient*, and *doctors1* a subset of them
  3. For purpose Treatment and recipient Doctors, allow *sysadmin* to perform SELECT and UPDATE over table *Patient* but only SELECT to *doctors1*



3. Support of retention time

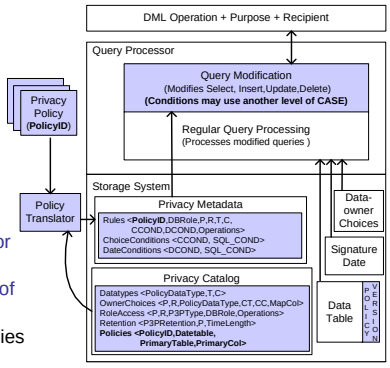
- Data should be retained only as long as necessary for the fulfillment of the purposes for which it was collected
- The original HDB architecture suggests the deletion of all data items that have outlived their purpose
- Our approach to support retention time is similar to the one used to support opt-in/opt-out preferences
- It does not require deleting the information after the allowed retention time
- It uses SQL conditions, which constitutes a flexible mechanism to express complex restrictions
- It uses the element *Retention* of P3P privacy rules. This element can have several predefined values: *no-retention*, *stated-purpose*, *legal-requirement*, etc



4. Support of policy versions

- 80% of organizations use different privacy policies for employees and clients, 42% have multiple policies for clients, and 75% require support of policy versions
- Different cases of multiple versions/policies requirements:
  1. Multiple policies
  2. Single policy, multiple data owners
  3. Multiple policies over time
  4. Multiple versions. Two cases:
     a) The policy for patients is updated only for new patients
     b) Two policy versions for different groups of patients are simultaneously used

  This last case requires the use of two policies associated with the same database entity *Patient*, this case is not supported by the frameworks for limiting disclosure proposed in previous work.



5. Support of generalization hierarchies

- Previous support of opt-in/opt-out choices is very limited; data owners can only give either full access to the data or deny it completely; there is not the option to give access to a generalized version of the data
- We propose the study of the integration of HDB and anonymization/generalization techniques
- We present a design to introduce generalization hierarchies into the limiting disclosure framework for HDBs

```
Select P.Name, DP.dName from Patient P, DiseasePatient DP where
                P.pid=DP.pid
            Purpose = Research; Recipient = Lab
```

⇩

```
Select P.Name, D.dName from
  (Select pno from Patient) AS P
  (Select pno,
      CASE (select diseaseName_option from options_disease
              where DiseasePatient.pno=options_disease.pno) as level
      WHEN 0 THEN NULL
      WHEN 1 THEN dname
      ELSE  generalize("DiseasePatient","dName",dname,level)
      END AS dname
  From DiseasePatient) AS DP
where P.pno=DP.pno and DP.dno=D.dno
```

Query modification with generalization hierarchies

Without generalization

Name	Disease
Mike	Flu
John	Pneumonia
Maria	Bronchitis
Peter	Flu

⇨

With generalization

Name	Disease
Mike	Respiratory System Problem
Maria	Respiratory Infection
Peter	Flu

PURDUE UNIVERSITY

CERIAS

Discovery Park e-Enterprise Center