A Language for Provenance Access Control

**Tyrone Cadenhead**, Vaibhav Khadilkar, Murat Kantarcioglu and Bhavani Thuraisingham

The University of Texas at Dallas

**CODASPY 2011**
February 21 - 23, 2011
Overview

• Provenance what it is and why we need it
• The challenges in protecting provenance
• Why we need a new access control language for provenance
• Our policy language and implementations
• Experiments
• Conclusions
Provenance: Example
(in healthcare domain)
Provenance example Cont’d

Flow of a patient’s record

– Three visits by the patient:
  ❖ A primary physician performed a checkup on the patient
  ❖ Patient returned for a follow-up visit at the physician’s request
  ❖ Surgery. The entire history is available for decisions

Queries:

– Find all physicians involved in a decision process leading to surgery
– Find the blood test results that were involved in the decision
– What is the outcome of the surgery?
Provenance Example

Features:

- **Directed edges** indicating that an event happened before another event
- **Causal dependencies** between the node entities
  - Edges start at a node called *the effect* and points to another node called *the cause* of the event
- **Acyclic**, indicating that history is non-cyclic and immutable
Sensitivity of Provenance

- **Entire provenance may contain both sensitive resources and sensitive relationships among resources**
  - A patient had surgery. Further provenance metadata/annotations reveal who performed the surgery, the date, and why.
  - But information about why a surgery is necessary may not be released to third parties without prior consent.

- **Existence of sensitive information requires some access control**
  - policy and mechanism which specify how to protect the provenance information
Challenge: Protecting Provenance

- Defining a provenance resource
  - Granularity
    - The directed graph
    - A node indicating a single resource,
    - A edge indicating the relationship among resources,
    - A path tracing the history back to its original sources,
    - A Subgraph, etc.

- Protection mechanisms
  - Protect both the relationships and objects
  - Based on access control
    - Specify who has access, when access is to be granted, where it should be granted, which format the accessed resource takes on.
Traditional Access Control

Current Drawbacks:

- Typically defined for systems with single data items.
- The number of resources in a provenance graph is exponential in the number of single resources.
- To identify all these resources, we need to iterate all of them.
- Lead to administration burdens, when done manually.

No support the provenance directed graph structure

- The relationships between the single data items is what sets a provenance access policy apart from the existing access control policies.)
Requirements for Provenance Access control

- Define a policy without enumerating all the resources
- Extend traditional access control
  - use traditional access control mechanisms to protect the single data items,
  - use the provenance access control mechanism to protect provenance resources
- Flexible
- Fine grained
- Simplification
- Manageability
A Proposed Policy Specification

A Graph query is a query which traverse a graph database by using regular expressions over the paths in the graph.

(\text{med:Doc1}_2, \text{[wasGeneratedBy]/[wasControlledBy]}, ?x)
Policy Specification Features

✓ **Integrity and Access Control**
  
  – *Integrity* - use the record only if it is along a specified path in the provenance graph (e.g. it was generated by the process operated by a physician).
  
  – *Access Control* - protect a path in the provenance graph if a particular record is along that path (e.g. a sensitive resource along a path describes the patient’s disease).

✓ **Fine-grained access control** - The record value is allowed to be any (indivisible) part of a provenance graph.

✓ **Arbitrary Paths** - The *restriction element* defines policies over paths of arbitrary lengths in a provenance graph that apply to a subject or record.

✓ **Flexible** - Policy specification is in XML,
  
  – XML language is open and extensible.

✓ **Compatible** - Specification is both customizable and readily supports integration of other policies (many existing languages already use XML).
Prototype

- Translate Policy Specification
  - Executed over a preferred Data format
    - Format represent and store provenance
  - Parsed into a query over the provenance graph
    - Query support for regular expressions

- A Semantic Approach:
  - RDF data model
    - Support Graph Structure
  - SPARQL
    - Query the RDF graph
  - Support Graph Querying
    - SPARQL + Regular Expressions
The Provenance graph

• RDF Triples

\[
\begin{align*}
&<\text{opm:Process}> <\text{opm:WasControlledBy}> <\text{opm:Agent}> \\
&<\text{opm:Process}> <\text{opm:Used}> <\text{opm:Artifact}> \\
&<\text{opm:Artifact}> <\text{opm:WasDerivedFrom}> <\text{opm:Artifact}> \\
&<\text{opm:Artifact}> <\text{opm:WasGeneratedBy}> <\text{opm:Process}> \\
&<\text{opm:Process}> <\text{opm:WasTriggeredBy}> <\text{opm:Process}> \\
\end{align*}
\]

\( V = \{\text{WasControlledBy, Used, WasDerivedFrom, WasGeneratedBy, WasTriggeredBy}\} \)

• Path \((<s_1>, (P), <o_n>))\)

  – Define \(P\) over \(V\) using regular expressions
  \((x, [p]^*, y)\) and \((x, [p]^+, y)\)
SPARQL

SELECT $\tilde{B}$ WHERE P

- SELECT ?x

  \{ med:doc4 [opm:WasDerivedFrom]+ ?x \}

- SELECT ?x

  \{ med:doc2 [WasGeneratedBy]/[WasControlledBy]) ?x \}
Example 1. (Why Query)

```
med:Doc1_2 gleen:OnPath("([opm:WasDerivedFrom] | [opm:WasGeneratedBy] | [opm:WasTriggeredBy] | [opm:WasControlledBy] | [Used])* ?x).
```
SELECT ?x
WHERE
{
    Med:Doc1_2
    gleen:onPath ('([opm:wasGeneratedBy]/
        [opm:wasControlledBy])' ?x)
}

Figure 2: A resource protected by a policy
Experiment

Implement prototype using the following open sources technologies:

- Jena
- XML 1.0
- Java 1.6
- Glleen regular expression library
- The OPM toolbox
We construct four graph block structures:
- varying number of provenance entities and RDF triples (as annotations)
- same diameter per block (which is the longest path in the graph)
- different branching factors (multiple parents)

- Concatenate the basic blocks to from new blocks
- Using both, the open provenance model (OPM) vocabulary and a synthetic domain vocabulary.

<table>
<thead>
<tr>
<th>Block #</th>
<th>Diameter (longest path)</th>
<th>no. of Artifacts</th>
<th>no. of Processes</th>
<th>no. of Agents</th>
<th>Annotated Graph (triples)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>13</td>
<td>24</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>41</td>
<td>23</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>25</td>
<td>26</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5347</td>
<td>7214</td>
<td>6743</td>
<td>8533</td>
</tr>
</tbody>
</table>
Graph structures

(a) Block 1 Structure

(b) Block 4 Structure
Experimental Setup

• Setup
  – Execute the template queries over large in memory Jena models.
  – Daisy chain graphs belonging to the same block to build the Jena models
  – Create a composite block
    • by daisy chaining six sets of block 1, followed by six sets of block 2, followed by six sets of block 3, followed by six sets of block 4. We then daisy chain these composite blocks together
Experimental Results

Results

- Query execution time varies with the graph size (number of RDF triples in graph), branching factor, diameter and the query templates.

- The points on the graph are the diameters.

- The closure queries (Why and How) are the most expensive, while the others are efficient since the answer to these queries usually involve few hops from the starting entity.
Experimental Results

(a) Block 1 chain

(b) Block 4 chain

(c) Composite chain
Conclusions

- Propose Regular Expressions on the provenance graph as an extension to traditional access control
- Present a XML-based policy and associated grammar
- Implement a prototype to test effectiveness of our policy language