

Auditing Compliance with a Hippocratic Database

Rakesh Agrawal, Roberto Bayardo, Christos Faloutsos,
Jerry Kiernan, Ralf Rantau, and Ramakrishnan Srikant

Presentation By: Manish Singh

9/30/09

EECS 584, Fall 2009

1

Outline

- The 10 Guiding principles of a Hippocratic Database
 - Compliance
- System Properties & Architecture
- How to specify the sensitive information?
- Modules
 - Query Log
 - Temporal Database
- Determining suspicious queries
- Performance

9/30/09

EECS 584, Fall 2009

2

Recap of Hippocratic Database

- Importance of Data Privacy
 - Privacy Accidents
 - Ethically questionable behavior
- “Strawman” design

Hippocratic Oath: *“And about whatever I may see or hear in treatment, or even without treatment, in the life of human beings – things that should not ever be blurted out outside – I will remain silent, holding such things to be unutterable.”*

9/30/09

EECS 584, Fall 2009

3

The 10 Guiding Principles

- Purpose Specification
- Consent
- Limited Collection
- Limited Use
- Limited Disclosure
- Limited Retention
- Accuracy
- Safety
- Openness
- Compliance

9/30/09

EECS 584, Fall 2009

4

Outcome of the Vision paper

- Lots of research has been done on various principles such as:
 - Limited Disclosure
 - Purpose Specification
 - Consent
 - Limited Use
 - Safety
- Today’s goal
 - Discussion on the **Compliance** principle.

9/30/09

EECS 584, Fall 2009

5

Auditing Compliance

- Compliance
 - Database adheres to its declared data disclosure property.
- Auditing compliance
 - Auditing Framework
 - Audit Expression
 - Specifies the sensitive data subjected to disclosure review.
 - Similar to SQL queries
 - Identify the suspicious queries

9/30/09

EECS 584, Fall 2009

6

System Properties

- Non-disruptive
- Fast and precise
- Fine-grained
- Convenient
- Assumptions
 - Single query suspiciousness
 - Single select clause

9/30/09 EECS 584, Fall 2009 7

System Architecture (1)

9/30/09 EECS 584, Fall 2009 8

System Architecture (2)

9/30/09 EECS 584, Fall 2009 9

Audit Expression

- Duplicate preserving SELECT query

*otherthan purpose-recipient pairs
during start-time to end-time
audit audit-list
from table-list
where condition-list*
- **U** is the cross product of all tables in the table-list.
- Sensitive information are the cells marked in **U**.

9/30/09 EECS 584, Fall 2009 10

Running Example

Customer (cid, name, address, phone, zip, contact)
Treatment (pcid, rcid, did, disease, duration, date)
Doctor (did, name)

lid	uname	timestamp	sql
L1	alice	2009-08-04 09:30:00	SELECT c.zip FROM Customer c, Treatment t WHERE c.cid = t.pcid and t.disease = 'diabetes'
L2	bob	2009-08-04 10:18:00	SELECT * FROM Customer WHERE zip = '95120'
L3	bob	2009-08-04 10:20:00	SELECT t.did FROM Customer c, Treatment t WHERE c.cid = t.pcid and t.name = 'Alice'

9/30/09 EECS 584, Fall 2009 11

Indispensable Tuple

cid	name	add	zip	age
1	alice	X1	95120	16
2	jack	X2	95135	35

pcid	did	disease
1	12	diabetes
2	13	cancer

Customer Treatment

cid	name	add	zip	age	pcid	did	disease
1	alice	X1	95120	16	1	12	diabetes
1	alice	X1	95120	16	2	13	cancer
2	jack	X2	95135	35	1	12	diabetes
2	jack	X2	95135	35	2	13	cancer

A: where c.cid = t.pcid and c.zip = '95120'
L1: where c.cid = t.pcid and t.disease = 'diabetes'

9/30/09 EECS 584, Fall 2009 12

Terminologies

Tuple t , Query Q , Audit A

- **Indispensable tuple:** Omitting t makes a difference on Q
- **Candidate query:** Q accesses all the audit list columns of A .
- **Suspicious query:** Q and A share an indispensable tuple

9/30/09

EECS 584, Fall 2009

13

Indispensability

- The paper discusses about determining indispensability for following types of query:
 - SPJ queries
 - Queries with aggregation without having, and
 - Queries with aggregation and having.

9/30/09

EECS 584, Fall 2009

14

Temporal Database

- To determine if a query Q has actually accessed any sensitive information.
- Maintain a backlog table for each table T .
 - Recreate the database instance at which Q was executed.
 - Time stamped.
 - Interval stamped.

9/30/09

EECS 584, Fall 2009

15

Time Stamped

name	id	add	time stamp	op
Alice	1	X1	2009-08-04 08:30:00	ins
Jack	2	X2	2009-08-04 09:00:00	ins
Xavier	3	X3	2009-08-04 11:00:00	ins
Xavier	3	X4	2009-08-08 10:30:00	upd
Jack	2	X2	2009-08-08 11:30:00	del
Matt	4	X5	2009-08-09 08:30:00	ins

Patient^b

name	id	add
Alice	1	X1
Xavier	3	X4
Matt	4	X5

Patient_{t=2009-08-09}
10:30:00

9/30/09

EECS 584, Fall 2009

16

Time Stamped

- A tuple in T^b has two additional columns:
 - TS: time of storage
 - OP: operation {'insert', 'delete', 'update'}
- Triggers are used to capture updates
- State of T , at time τ , is given by:

$$T^\tau = \pi_{P_{C_1, \dots, C_m}}(\{t \mid t \in T^b \wedge t.TS \leq \tau \wedge t.OP \neq \text{'delete'} \wedge \nexists r \in T^b \text{ s.t. } r.P = t.P \wedge r.TS \leq \tau \wedge r.TS > t.TS\})$$

9/30/09

EECS 584, Fall 2009

17

Interval Stamped

name	id	add	TS(start time)	TE(end time)
Alice	1	X1	2009-08-04 08:30:00	--
Jack	2	X2	2009-08-04 09:00:00	2009-08-08 11:30:00
Xavier	3	X3	2009-08-04 11:00:00	2009-08-08 10:30:00
Xavier	3	X4	2009-08-08 10:30:00	--
Matt	4	X5	2009-08-09 08:30:00	--

Patient^b

name	id	add
Alice	1	X1
Xavier	3	X4
Matt	4	X5

Patient_{t=2009-08-09}
10:30:00

9/30/09

EECS 584, Fall 2009

18

Interval Stamped

- Period of time for which each tuple was alive:
 - TS: time of storage
 - TE: end time
- Insert trigger adds tuple t to T^b , and sets TE to null
- Update trigger searches for tuple b such that $b.P=t.P$ and $b.TE=null$, and sets $b.TE$ to the current time and inserts new tuple t
- Delete trigger searches for tuple b such that $b.P=t.P$ and $b.TE=null$, and sets $b.TE$ to the current time

9/30/09

EECS 584, Fall 2009

19

Hospital Database

Customer (cid, name, address, phone, zip, contact)
 Treatment (pcid, rcid, did, disease, duration, date)
 Doctor (did, name)

9/30/09

EECS 584, Fall 2009

20

Example

- Audit if the disease information of anyone living zipcode = 95120 was disclosed.

```
audit  disease
from   Customer c, Treatment t
where  c.cid = t.pcid and c.zip = '95120'
```

- The disease cell of table Customer * Treatment, where the condition $c.cid = t.pcid$ and $c.zip = '95120'$ is marked as sensitive information.

9/30/09

EECS 584, Fall 2009

21

Example Contd.

```
audit  disease
from   Customer c, Treatment t
where  c.cid = t.pcid and c.zip = '95120'
```

- Given query Q:

```
select address
from   Customer c, Treatment t
where  c.cid = t.pcid and t.disease = 'diabetes'
```

- Q is a candidate query:
 - Q accesses the disease information that A is auditing.
- Suspicious
 - Q would be suspicious with respect to A, if there was a customer who lived in the ZIP code 95120 and was treated for diabetes

9/30/09

EECS 584, Fall 2009

22

Determining Suspicious Queries

- Involves two steps:
 - Syntactic Analysis
 - Find candidate queries using query log
 - Generate Audit Query
 - Determine if a candidate query is actually suspicious.
 - Execute over a temporal database to check the actual query result.

9/30/09

EECS 584, Fall 2009

23

Syntactic Analysis

- It involves various steps to prune out non-suspicious queries:

- Check attributes
- Check for contradictions between predicates

```
audit  address
from   Customer c, Treatment t
where  c.cid = t.pcid and t.disease = 'cancer'

select address
from   Customer c, Treatment t
where  c.cid = t.pcid and t.disease = 'diabetes'
```

- Set of candidate queries $Q = \{Q_1, \dots, Q_n\}$

9/30/09

EECS 584, Fall 2009

24

Audit Query Generation for a SPJ Query

- Candidate Query(Q):**

```

select T.disease
from Treatment T, Doctor D
where T.did = D.did and D.name = 'Phil'

```
- Audit Expression(A)**

```

audit T.disease
from Customer C, Treatment T
where C.cid = T.cid and C.name = 'Alice'

```
- Combine A and Q's query expression to determine existence of a common indispensable tuple.

9/30/09 EECS 584, Fall 2009 25

Audit Query Generation

The diagram shows a tree structure where a 'T.S' node at the top contains a 'select' statement: 'select := T.j = D.d and D.m = Phil'. This node is connected to a 'T.S' node on the left containing an 'audit expression := C.c = T.p and C.n = Alice'. The 'T.S' node on the left is connected to three leaf nodes: 'p,r,i,s,u,t Treatment', 'd,m Doctor', and 'c,n,a,h,z,o Customer'. Edges are labeled with 'T', 'D', and 'C'.

9/30/09 EECS 584, Fall 2009 26

Audit Query Generation

The diagram shows a tree structure where a 'Q1' node at the top contains an 'audit expression := C.c = X.p and C.n = Alice'. This node is connected to a 'T.p' node on the left containing a 'select := T.j = D.d and D.m = Phil'. The 'T.p' node is connected to three leaf nodes: 'p,r,i,s,u,t Treatment', 'd,m Doctor', and 'c,n,a,h,z,o Customer'. Edges are labeled with 'X', 'T', and 'D'.

9/30/09 EECS 584, Fall 2009 27

Performance

- What is the overhead on normal query processing?
- What is the cost of conducting audits?
- Terminology**
 - TS -> Time Stamped Organization
 - IS -> Interval Stamped Organization
 - Simple -> Indexed on Primary Key
 - Composite -> Indexed on Time Stamp & Primary Key

9/30/09 EECS 584, Fall 2009 28

Performance

- Query Overhead**
 - 10 times overhead for TS organization.
 - Without index IS is not feasible.
 - Many applications have more read queries as compared to update operations.

The graph plots 'Time (minutes)' on the y-axis (0 to 250) against '# of versions per tuple' on the x-axis (5 to 50). The legend includes: IS-Composite, IS-Simple, TS-Composite, TS-Simple, TS-No Index, and No Triggers. IS-Composite shows the highest time, increasing from ~50 to ~220 minutes. TS-Composite is the next highest, increasing from ~50 to ~180 minutes. TS-Simple and TS-No Index show similar, lower times, increasing from ~50 to ~120 minutes. IS-Simple and No Triggers show the lowest times, remaining near 50 minutes.

Cost of maintaining backlog tables

9/30/09 EECS 584, Fall 2009 29

Performance of Audit

- Both TS and IS do better with composite index.
- TS outperforms IS with increase in number of versions of tuples.
- TS allows lazy update of index.

The graph plots 'Time (msec.)' on the y-axis (1 to 100) against '# of versions per tuple' on the x-axis (1 to 50). The legend includes: IS-Simple, TS-Simple, IS-Composite, and TS-Composite. IS-Composite shows the highest time, increasing from ~10 to ~100 msec. TS-Composite is the next highest, increasing from ~10 to ~80 msec. IS-Simple and TS-Simple show the lowest times, remaining near 10 msec.

Execution time of an audit query

9/30/09 EECS 584, Fall 2009 30



Thank You!

Questions?