Data Protection in Outsourcing Scenarios

Sabrina De Capitani di Vimercati
Dipartimento di Tecnologie dell'Informazione
Università degli Studi di Milano
sabrina.decapitani@unimi.it

3rd International Workshop on Autonomous and Spontaneous Security
(SETOP 2010)

Motivation (1)

Recent advances in the communications and information technology have led new emerging scenarios

- Outsourcing (data and services)
  - data storage and service access through honest-but-curious servers
- Pervasive and ubiquitous computing
  - computing and communication services anytime and anywhere
- Ambient intelligence
  - seamless support for the different activities and interactions of users acting within a controlled environment
- Cloud computing
  - Internet-based access to data and applications shared among different clients

Motivation (2)

- The availability of online services anytime and anywhere and the ability to process and store sensitive data securely are becoming crucial
- Our data will be no longer remain on personal hard disks: they will be stored in remote systems
  - can move around in different locations
  - can be distributed and fragmented among different protection domains (i.e., different data centers)
  - should be accessible only to the authorized parties
  - should be managed according to possible restrictions on their storage and usage
  - ...

Issues to be addressed

- Data protection
- Query execution
- Private access
- Data integrity and correctness
- Access control enforcement
- Support for selective write privileges
- Data publication and utility
- Private collaborative computation

Issues to be addressed

- Data protection: fragmentation and encryption
- Query execution
- Private access
- Data integrity and correctness
- Access control enforcement
- Support for selective write privileges
- Data publication and utility: fragmentation and loose associations
- Private collaborative computation

Fragmentation and encryption

- Encryption proposed in outsourcing scenarios makes query evaluation more expensive or not always possible
- Often what is sensitive is the association between values of different attributes, rather than the values themselves
  - e.g., association between employee's names and salaries

  $\Longrightarrow$ protect associations by breaking them, rather than encrypting
- Recent solutions for enforcing privacy requirements couple:
  - encryption
  - data fragmentation

Confidentiality constraints

- Privacy requirements are represented as a set of confidentiality constraints that capture sensitivity of attributes and associations
  - sets of attributes such that the (joint) visibility of values of the attributes in the sets should be protected
- Sensitive attributes: the values assumed by some attributes are considered sensitive and cannot be stored in the clear
  $\Longrightarrow$ singleton constraints
- Sensitive associations: the association between values of given attributes is sensitive and should not be released
  $\Longrightarrow$ non-singleton constraints

Outline

- Non-communicating pair of servers [Aggarwal et al., CIDR'05]
- Multiple fragments [ESORICS'07, ACM TISSEC'10]
- Departing from encryption: Keep a few [ESORICS'09]
- Fragments and loose associations [PVLDB'10]

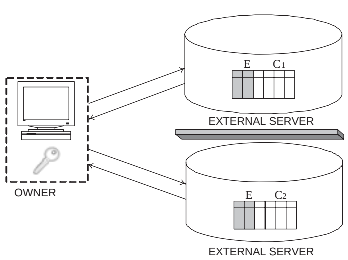
P. Samarati, S. De Capitani di Vimercati, "Data Protection in Outsourcing Scenarios: Issues and Directions," in *Proc. of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2010)*, Beijing, China, April, 2010.

Non-Communicating Pair of Servers

G. Aggarwal, M. Bawa, P. Ganesan, H. Garcia-Molina, K. Kenthapadi, R. Motwani, U. Srivastava, D. Thomas, Y. Xu, "Two Can Keep a Secret: A Distributed Architecture for Secure Database Services," in *Proc. of the Conference on Innovative Data Systems Research* Asilomar, CA, USA, January 4-7, 2005.

Non-communicating pair of servers

- Confidentiality constraints are enforced by splitting information over two independent servers that cannot communicate (need to be completely unaware of each other)
  - Sensitive associations are protected by distributing the involved attributes between the two servers
  - Encryption is applied only when explicitly demanded by the confidentiality constraints or when storing the attribute in any of the servers would expose at least a sensitive association



- $E \cup C_1 \cup C_2 = R$
- $C_1 \cup C_2 \subseteq R$

Enforcing confidentiality constraints

- Confidentiality constraints $\mathscr{C}$ defined over a relation R are enforced by decomposing R as $\langle R_1, R_2, E \rangle$ where:
  - R_1 and R_2 include a unique tuple ID needed to ensure lossless decomposition
  - $R_1 \cup R_2 = R$
  - E is the set of encrypted attributes and $E \subseteq R_1$, $E \subseteq R_2$
  - for each $c \in \mathscr{C}$, $c \not\subseteq (R_1 - E)$ and $c \not\subseteq (R_2 - E)$

Confidentiality constraints – Example (1)

R = (Name,DoB,Gender,Zip,Position,Salary,Email,Telephone)

- {Telephone}, {Email}
  - attributes Telephone and Email are sensitive (cannot be stored in the clear)
- {Name,Salary}, {Name,Position}, {Name,DoB}
  - attributes Salary, Position, and DoB are private of an individual and cannot be stored in the clear in association with the name
- {DoB,Gender,Zip,Salary}, {DoB,Gender,Zip,Position}
  - attributes DoB, Gender, Zip can work as quasi-identifier
- {Position,Salary}, {Salary,DoB}
  - association rules between Position and Salary and between Salary and DoB need to be protected from an adversary

Enforcing confidentiality constraints – Example (2)

R = (Name,DoB,Gender,Zip,Position,Salary,Email,Telephone)

{Telephone}
{Email}
{Name,Salary}
{Name,Position}
{Name,DoB}
{DoB,Gender,Zip,Salary}
{DoB,Gender,Zip,Position}
{Position,Salary}
{Salary,DoB}

$\Longrightarrow R$ = (Name,DoB,Gender,Zip,Position,Salary,Email,Telephone)

- R_1: (ID,Name,Gender,Zip,Salary^e,Email^e,Telephone^e)
- R_2: (ID,Position,DoB,Salary^e,Email^e,Telephone^e)

Note that Salary is encrypted even if non sensitive per se since storing it in the clear in any of the two fragments would violate at least a constraint

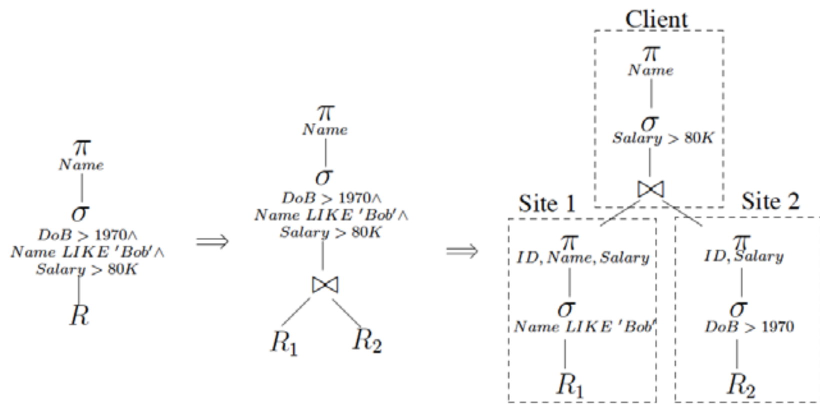
Query execution

At the logical level: replace R with $R_1 \bowtie R_2$
Query plans:

- Fetch R_1 and R_2 from the servers and execute the query locally
  - extremely expensive
- Involve servers S_1 and S_2 in the query evaluation
  - can do the usual optimizations, e.g., push down selections and projections
  - selections on encrypted attributes cannot be pushed down
  - different options for executing queries:
    - send sub-queries to both S_1 and S_2 in parallel, and join the results at the client
    - send only one of the two sub-queries, say to S_1; the tuple IDs of the result from S_1 are then used to perform a semi-join with the result of the sub-query of S_2 to filter R_2

Query execution – Example

- R_1: (ID,Name,Gender,Zip,Salary^e,Email^e,Telephone^e)
- R_2: (ID,Position,DoB,Salary^e,Email^e,Telephone^e)



Identifying the optimal decomposition

Brute force approach for optimizing wrt workload W:

- For each possible safe decomposition of R:
  - optimize each query in W for the decomposition
  - estimate the total cost for executing the queries in W using the optimized query plans
- Select the decomposition that has the lowest overall query cost

Too expensive! $\Longrightarrow$ Exploit affinity matrix

Multiple Fragments

V. Ciriani, S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, “Combining Fragmentation and Encryption to Protect Privacy in Data Storage,” in *ACM Transactions on Information and System Security (TISSEC)*, vol. 13, no. 3, July, 2010.

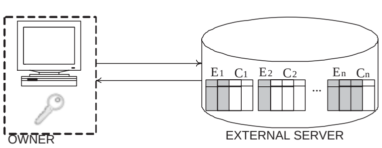
Multiple fragments (1)

Coupling fragmentation and encryption interesting and promising, but, limitation to two servers:

– too strong and difficult to enforce in real environments

– limits the number of associations that can be solved by fragmenting data, often forcing the use of encryption

$\Longrightarrow$ allow for more than two non-linkable fragments



- $E_1 \cup C_1 = \ldots = E_n \cup C_n = R$
- $C_1 \cup \ldots \cup C_n \subseteq R$

Multiple fragments (2)

- A fragmentation of R is a set of fragments $\mathcal{F} = \{F_1, \ldots, F_m\}$, where $F_i \subseteq R$, for $i = 1, \ldots, m$
- A fragmentation $\mathcal{F}$ of R correctly enforces a set $\mathcal{C}$ of confidentiality constraints iff the following conditions are satisfied:
  - $\forall F \in \mathcal{F}, \forall c \in \mathcal{C} : c \not\subseteq F$ (each individual fragment satisfies the constraints)
  - $\forall F_i, F_j \in \mathcal{F}, i \neq j : F_i \cap F_j = \emptyset$ (fragments do not have attributes in common)

Multiple fragments (3)

- Each fragment F is mapped to a physical fragment containing:
  - all the attributes in F in the clear
  - all the other attributes of R encrypted (a salt is applied on each encryption)
- Fragment $F_i = \{A_{i_1}, \ldots, A_{i_n}\}$ of R mapped to physical fragment F_i^e(salt,enc,$A_{i_1}, \ldots, A_{i_n}$):
  - each $t \in r$ over R is mapped to a tuple $t^e \in f_i^e$ with f_i^e a relation over F_i^e and:
    - t^e[*enc*] = $E_k(t[R - F_i] \otimes t^e$[*salt*])
    - $t^e[A_{i_j}] = t[A_{i_j}]$, for $j = 1, \ldots, n$

Multiple fragments – Example (1)

MEDICALDATA

SSN	Name	DoB	Zip	Illness	Physician
123-45-6789	Nancy	65/12/07	94142	hypertension	M. White
987-65-4321	Ned	73/01/05	94141	gastritis	D. Warren
963-85-2741	Nell	86/03/31	94139	flu	M. White
147-85-2369	Nick	90/07/19	94139	asthma	D. Warren

c_0= {SSN}
c_1= {Name, DoB}
c_2= {Name, Zip}
c_3= {Name, Illness}
c_4= {Name, Physician}
c_5= {DoB, Zip, Illness}
c_6= {DoB, Zip, Physician}

Multiple fragments – Example (1)

MEDICALDATA

SSN	Name	DoB	Zip	Illness	Physician
123-45-6789	Nancy	65/12/07	94142	hypertension	M. White
987-65-4321	Ned	73/01/05	94141	gastritis	D. Warren
963-85-2741	Nell	86/03/31	94139	flu	M. White
147-85-2369	Nick	90/07/19	94139	asthma	D. Warren

c_0= {SSN}
c_1= {Name, DoB}
c_2= {Name, Zip}
c_3= {Name, Illness}
c_4= {Name, Physician}
c_5= {DoB, Zip, Illness}
c_6= {DoB, Zip, Physician}

F_1

salt	enc	Name
s_1	α	Nancy
s_2	β	Ned
s_3	γ	Nell
s_4	δ	Nick

F_2

salt	enc	DoB	Zip
s_5	ε	65/12/07	94142
s_6	ζ	73/01/05	94141
s_7	η	86/03/31	94139
s_8	θ	90/07/19	94139

F_3

salt	enc	Illness	Physician
s_9	ι	hypertension	M. White
s_{10}	κ	gastritis	D. Warren
s_{11}	λ	flu	M. White
s_{12}	μ	asthma	D. Warren

Executing queries on fragments

- Every physical fragment of R contains all the attributes of R $\Longrightarrow$ no more than one fragment needs to be accessed to respond to a query
- If the query involves an encrypted attribute, an additional query may need to be executed by the client

Original query on R	**Translation over fragment F_3^e**
Q := SELECT SSN, Name FROM MedicalData WHERE (Illness='gastritis' OR Illness='asthma') AND Physician='D. Warren' AND Zip='94141'	Q^3 := SELECT salt, enc FROM F_3^e WHERE (Illness='gastritis' OR Illness='asthma') AND Physician='D. Warren' Q' := SELECT SSN, Name FROM *Decrypt*(Q^3, *Key*) WHERE Zip='94141'

Optimization criteria

- Goal: find a fragmentation that makes query execution efficient
- The fragmentation process can then take into consideration different optimization criteria:
  - number of fragments [ESORICS'07]
  - affinity among attributes [ACM TISSEC'10]
  - query workload [ICDCS'09]
- All criteria obey maximal visibility
  - only attributes that appear in singleton constraints (sensitive attributes) are encrypted
  - all attributes that are not sensitive appear in the clear in one fragment

Departing from Encryption: Keep a Few

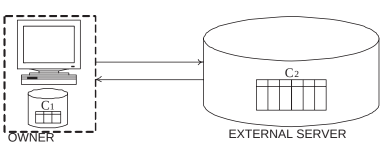
V. Ciriani, S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, “Keep a Few: Outsourcing Data while Maintaining Confidentiality,” in *Proc. of the 14th European Symposium On Research In Computer Security (ESORICS 2009)*, Saint Malo, France, September 21-25, 2009.

Keep a few

Basic idea:

– encryption makes query execution more expensive and not always possible
– encryption brings overhead of key management

⟹ Depart from encryption by involving the owner as a trusted party to maintain a limited amount of data



- $C_1 \cup C_2 = R$

Fragmentation

Given:

- $R(A_1,\ldots,A_n)$: relation schema
- $\mathscr{C} = \{c_1,\ldots,c_m\}$: confidentiality constraints over R

Determine a fragmentation $\mathscr{F} = \langle F_o, F_s \rangle$ for R, where F_o is stored at the owner and F_s is stored at a storage server, and

- $F_o \cup F_s = R$ (completeness)
- $\forall c \in \mathscr{C}, c \not\subseteq F_s$ (confidentiality)
- $F_o \cap F_s = \emptyset$ (non-redundancy)   /* can be relaxed */

At the physical level F_o and F_s have a common attribute (additional tid or non-sensitive key attribute) to guarantee lossless join

Fragmentation – Example

PATIENT

SSN	Name	DoB	Race	Job	Illness	Treatment	HDate
123-45-6789	Nancy	65/12/07	white	waiter	hypertension	ace	09/01/02
987-65-4321	Ned	73/01/05	black	nurse	gastritis	antibiotics	09/01/06
963-85-2741	Nell	86/03/31	red	banker	flu	aspirin	09/01/08
147-85-2369	Nick	90/07/19	asian	waiter	asthma	anti-inflammatory	09/01/10

c_0={SSN}
c_1={Name,Illness}
c_2={Name,Treatment}
c_3={DoB,Race,Illness}
c_4={DoB,Race,Treatment}
c_5={Job,Illness}

F_o

tid	SSN	Illness	Treatment
1	123-45-6789	hypertension	ace
2	987-65-4321	gastritis	antibiotics
3	963-85-2741	flu	aspirin
4	147-85-2369	asthma	anti-inflammatory

F_s

tid	Name	DoB	Race	Job	HDate
1	Nancy	65/12/07	white	waiter	09/01/02
2	Ned	73/01/05	black	nurse	09/01/06
3	Nell	86/03/31	red	banker	09/01/08
4	Nick	90/07/19	asian	waiter	09/01/10

Query evaluation

- Queries formulated on R need to be translated into equivalent queries on F_o and/or F_s

- Queries of the form: SELECT A FROM R WHERE C
  where C is a conjunction of basic conditions
  - C_o: conditions that involve only attributes stored at the client
  - C_s: conditions that involve only attributes stored at the sever
  - C_{so}: conditions that involve attributes stored at the client and attributes stored at the server

Query evaluation – Example

- F_o={SSN,Illness,Treatment}, F_s={Name,DoB,Race,Job,HDate}

- q = SELECT SSN, DoB
  FROM Patient
  WHERE (Treatment=“antibiotic”)
  AND (Job=“nurse”)
  AND (Name=Illness)

- The conditions in the WHERE clause are split as follows
  - C_o = {Treatment = “antibiotic”}
  - C_s = {Job = “nurse”}
  - C_{so} = {Name = Illness}

Query evaluation strategies

Server-Client strategy

- server: evaluate C_s and return result to client
- client: receive result from server and join it with F_o
- client: evaluate C_o and C_{so} on the joined relation

Client-Server strategy

- client: evaluate C_o and send tid of tuples in result to server
- server: join input with F_s, evaluate C_s, and return result to client
- client: join result from server with F_o and evaluate C_{so}

Server-client strategy – Example

q = SELECT SSN, DoB
FROM Patient
WHERE (Treatment = "antibiotic")
AND (Job = "nurse")
AND (Name = Illness)

C_o={Treatment = "antibiotic"}
C_s={Job = "nurse"}
C_{so}={Name = Illness}

q_s = SELECT tid,Name,DoB
FROM F_s
WHERE Job = "nurse"

q_{so} = SELECT SSN, DoB
FROM F_o JOIN r_s
ON F_o.tid=r_s.tid
WHERE (Treatment = "antibiotic") AND (Name = Illness)

Client-server strategy – Example

q = SELECT SSN, DoB
FROM Patient
WHERE (Treatment = "antibiotic")
AND (Job = "nurse")
AND (Name = Illness)

C_o={Treatment = "antibiotic"}
C_s={Job = "nurse"}
C_{so}={Name = Illness}

q_o = SELECT tid
FROM F_o
WHERE Treatment = "antibiotic"

q_s = SELECT tid,Name,DoB
FROM F_s JOIN r_o ON F_s.tid=r_o.tid
WHERE Job = "nurse"

q_{so} = SELECT SSN, DoB
FROM F_o JOIN r_s ON F_o.tid=r_s.tid
WHERE Name = Illness

Server-client vs client-server strategies

- If the storage server knows or can infer the query
  - Client-Server leaks information: the server infers that some tuples are associated with values that satisfy C_o
- If the storage server does not know and cannot infer the query
  - Server-Client and Client-Server strategies can be adopted without privacy violations
  - possible strategy based on performances: evaluate most selective conditions first

Minimal fragmentation

- The goal is to minimize the owner's workload due to the management of F_o
- Weight function w takes a pair $\langle F_o, F_s \rangle$ as input and returns the owner's workload (i.e., storage and/or computational load)
- A fragmentation $\mathscr{F} = \langle F_o, F_s \rangle$ is minimal iff:
  1. $\mathscr{F}$ is correct (i.e., it satisfies the completeness, confidentiality, and non-redundancy properties)
  2. $\nexists \mathscr{F}'$ such that $w(\mathscr{F}') < w(\mathscr{F})$ and $\mathscr{F}'$ is correct

Fragmentation metrics

Different metrics could be applied splitting the attributes between F_o and F_s, such as minimizing:

- storage
  - number of attributes in F_o (*Min-Attr*)
  - size of attributes in F_o (*Min-Size*)
- computation/traffic
  - number of queries in which the owner needs to be involved (*Min-Query*)
  - number of conditions within queries in which the owner needs to be involved (*Min-Cond*)

The metrics to be applied may depend on the information available

Modeling of the minimization problems

- All problems of minimizing storage or computation/traffic aim at identifying a hitting set
  - F_o must contain at least an attribute for each constraint
- Different metrics correspond to different criteria according to which the hitting set should be minimized
- The problem is to compute the hitting set of attributes with minimum weight

$\Longrightarrow$ NP-hard problem

Fragments and Loose Associations

S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Fragments and Loose Associations: Respecting Privacy in Data Publishing," in *Proc. of the VLDB Endowment*, vol. 3, no. 1, 2010.

Data publication

- Fragmentation can also be used to protect sensitive associations in data publishing
  $\Longrightarrow$ publish/release to external parties only views (fragments) that do not expose sensitive associations

- To increase utility of published information fragments could be coupled with some associations in sanitized form
  $\Longrightarrow$ loose associations: associations among groups of values (in contrast to specific values)

Loose association

Given two fragments F_l and F_r containing sub-tuples involved in a sensitive association:

- partition the tuples of F_l and F_r in different groups of size k_l and k_r
- associations among tuples induce associations among groups
- need to ensure that induced group associations guarantee a proper privacy degree

Loose association – Example

SSN	Name	DoB	Race	Illness
123-45-6789	Nancy	65/12/07	white	hypertension
987-65-4321	Ned	73/01/05	black	gastritis
963-85-2741	Nell	86/03/31	red	flu
147-85-2369	Nick	90/07/19	asian	asthma
782-90-5280	Nicole	55/05/22	white	gastritis
816-52-7272	Noel	32/11/22	red	obesity
872-62-5178	Nora	68/08/14	asian	measles
712-81-7618	Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Ned	black
Nell	red
Nick	asian
Nicole	white
Noel	red
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
32/11/22	obesity
68/08/14	measles
73/01/05	hypertension

Data Protection in Outsourcing Scenarios 39/45

Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Data Protection in Outsourcing Scenarios 39/45

Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

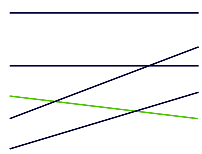
Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic



F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

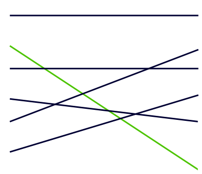
Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic



F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

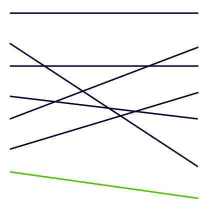
Loose association – Example

	Name	DoB	Race	Illness
	Nancy	65/12/07	white	hypertension
	Ned	73/01/05	black	gastritis
	Nell	86/03/31	red	flu
	Nick	90/07/19	asian	asthma
	Nicole	55/05/22	white	gastritis
	Noel	32/11/22	red	obesity
⟹	Nora	68/08/14	asian	measles
	Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic



F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

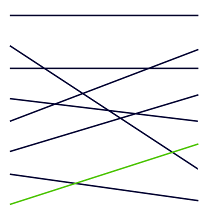
Loose association – Example

	Name	DoB	Race	Illness
	Nancy	65/12/07	white	hypertension
	Ned	73/01/05	black	gastritis
	Nell	86/03/31	red	flu
	Nick	90/07/19	asian	asthma
	Nicole	55/05/22	white	gastritis
	Noel	32/11/22	red	obesity
	Nora	68/08/14	asian	measles
⟹	Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic



F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

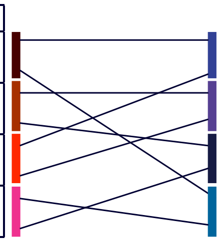
Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic



F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Loose association – Example

Name	DoB	Race	Illness
Nancy	65/12/07	white	hypertension
Ned	73/01/05	black	gastritis
Nell	86/03/31	red	flu
Nick	90/07/19	asian	asthma
Nicole	55/05/22	white	gastritis
Noel	32/11/22	red	obesity
Nora	68/08/14	asian	measles
Norman	73/01/05	hispanic	hypertension

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race	G
Nancy	white	nr2
Noel	red	nr2
Nell	red	nr3
Nicole	white	nr3
Ned	black	nr1
Nick	asian	nr1
Nora	asian	nr4
Norman	hispanic	nr4

A

G_l	G_r
nr1	id1
nr1	id2
nr2	id1
nr2	id3
nr3	id2
nr3	id4
nr4	id3
nr4	id4

F_r

G	DoB	Illness
id1	65/12/07	hypertension
id1	73/01/05	gastritis
id2	86/03/31	flu
id2	90/07/19	asthma
id4	55/05/22	gastritis
id4	73/01/05	hypertension
id3	32/11/22	obesity
id3	68/08/14	measles

k-loose association

- An association is k-loose if every group association indistinguishably corresponds to at least k distinct associations among tuples
- The degree of looseness characterizes the privacy (and utility) of the associations
  - the probability of an association to exist in the original relation may change from 1/card(relation) to $1/k$
- If grouping satisfies given heterogeneity properties, the group association is guaranteed to be k-loose with $k = k_l \cdot k_r$
  - group heterogeneity
  - association heterogeneity
  - deep heterogeneity

Group heterogeneity

No group can contain tuples that have the same values for the attributes involved in constraints covered by F_l and F_r

- it ensures diversity of tuples within groups

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness	
65/12/07	hypertension	NO
73/01/05	hypertension	
86/03/31	flu	
90/07/19	asthma	
55/05/22	gastritis	NO
73/01/05	gastritis	
32/11/22	obesity	
68/08/14	measles	

Group heterogeneity

No group can contain tuples that have the same values for the attributes involved in constraints covered by F_l and F_r

- it ensures diversity of tuples within groups

$c_0 =$ {SSN}
$c_1 =$ {Name,Illness}
$c_2 =$ {Name,DoB}
$c_3 =$ {Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Association heterogeneity

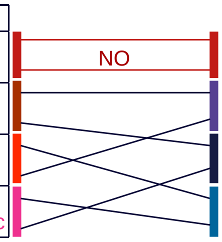
No group can be associated twice with another group (the group association cannot contain any duplicate)

- it ensures that for each real tuple in the original relation there are at least $k_l \cdot k_r$ pairs in the group association that may correspond to it

$c_0 =$ {SSN}
$c_1 =$ {Name,Illness}
$c_2 =$ {Name,DoB}
$c_3 =$ {Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic



F_r

DoB	Illness
65/12/07	hypertension
32/11/22	obesity
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
73/01/05	gastritis
68/08/14	measles

Association heterogeneity

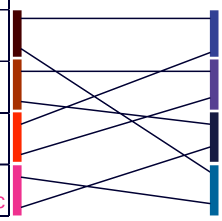
No group can be associated twice with another group (the group association cannot contain any duplicate)

- it ensures that for each real tuple in the original relation there are at least $k_l \cdot k_r$ pairs in the group association that may correspond to it

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic



F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Deep heterogeneity

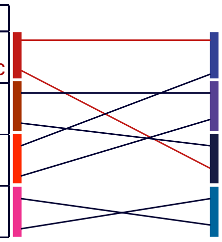
No group can be associated with two groups that contain tuples that have the same values for the attributes involved in a constraint covered by F_l and F_r

- it ensures that all $k_l \cdot k_r$ pairs in the group association to which each tuple could correspond contain diverse values for attributes involved in constraints

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}

F_l

Name	Race
Nancy	white
Norman	hispanic
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Noel	red



F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

NO (65/12/07 hypertension, 73/01/05 gastritis)

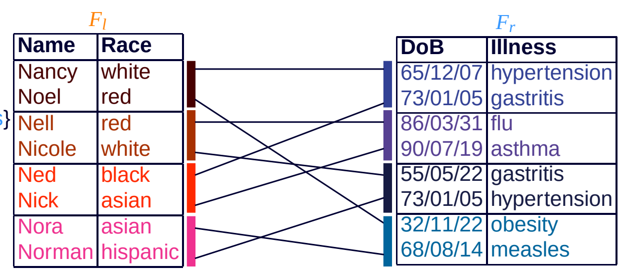
NO (55/05/22 gastritis, 73/01/05 hypertension)

Deep heterogeneity

No group can be associated with two groups that contain tuples that have the same values for the attributes involved in a constraint covered by F_l and F_r

- it ensures that all $k_l \cdot k_r$ pairs in the group association to which each tuple could correspond contain diverse values for attributes involved in constraints

c_0 ={SSN}
c_1 ={Name,Illness}
c_2 ={Name,DoB}
c_3 ={Race,DoB,Illness}



F_l

Name	Race
Nancy	white
Noel	red
Nell	red
Nicole	white
Ned	black
Nick	asian
Nora	asian
Norman	hispanic

F_r

DoB	Illness
65/12/07	hypertension
73/01/05	gastritis
86/03/31	flu
90/07/19	asthma
55/05/22	gastritis
73/01/05	hypertension
32/11/22	obesity
68/08/14	measles

Research directions

- Balance between encryption and fragmentation
- Schema vs. instance constraints
- Data dependencies not captured by confidentiality constraints
- Enforcement of different kinds of queries
- Visibility requirements
- Balance privacy and utility
- External knowledge

Conclusions

- The development of the Information technologies presents:
  - new needs and risks for privacy
  - new opportunities for protecting privacy
- Lots of opportunities for new open issues to be addressed

... towards allowing society to fully benefit from information technology while enjoying security and privacy