Security and Privacy in the Cloud

Pierangela Samarati
Dipartimento di Informatica
Università degli Studi di Milano
pierangela.samarati@unimi.it

6th International Conference on Cloud Computing and Services Science
(CLOSER 2016)
Rome, Italy – April 25, 2016

Cloud computing

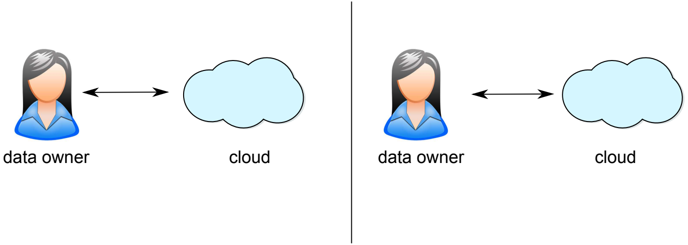
- The Cloud allows users and organizations to rely on external providers for storing, processing, and accessing their data
  - + high configurability and economy of scale
  - + data and services are always available
  - + scalable infrastructure for applications
- Users lose control over their own data
  - – new security and privacy problems
- Need solutions to protect data and to securely process them in the cloud

The data protection challenge

- Huge amount of data collected, generated, and shared
- Growing use of SaaS business applications
- Growing amount of pervasive and mobile applications relying on data availability anytime anywhere

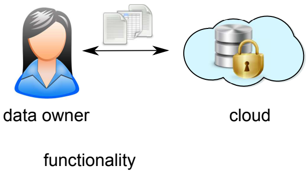
Cloud Computing: Today

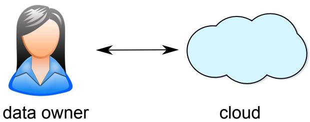
Cloud Service Providers (CSPs) apply security measures in the services they offer but these measures protect only the perimeter and storage against outsiders



Cloud Computing: Today

Cloud Service Providers (CSPs) apply security measures in the services they offer but these measures protect only the perimeter and storage against outsiders

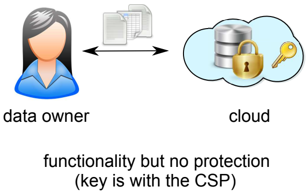


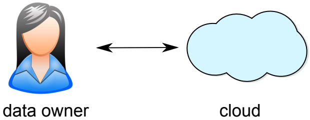


- functionality

Cloud Computing: Today

Cloud Service Providers (CSPs) apply security measures in the services they offer but these measures protect only the perimeter and storage against outsiders

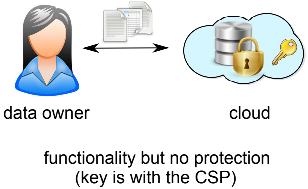


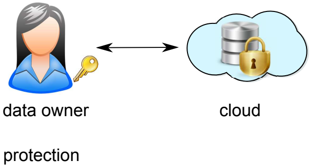


- functionality implies full trust in the CSP that has full access to the data (e.g., Google Cloud Storage, iCloud)

Cloud Computing: Today

Cloud Service Providers (CSPs) apply security measures in the services they offer but these measures protect only the perimeter and storage against outsiders

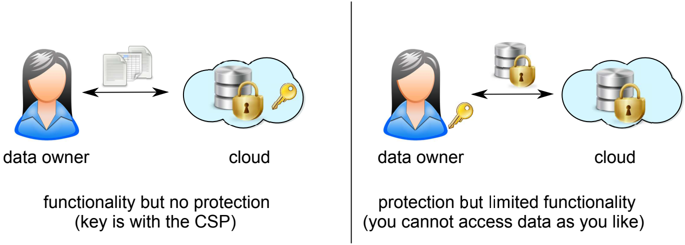




- functionality implies full trust in the CSP that has full access to the data (e.g., Google Cloud Storage, iCloud)
- protection

Cloud Computing: Today

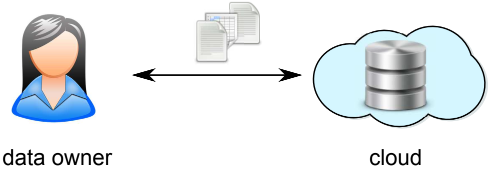
Cloud Service Providers (CSPs) apply security measures in the services they offer but these measures protect only the perimeter and storage against outsiders



- functionality implies full trust in the CSP that has full access to the data (e.g., Google Cloud Storage, iCloud)
- protection but limited functionality since the CSP cannot access data (e.g., Boxcryptor, SpiderOak)

Cloud computing: ESCUDO-CLOUD's vision

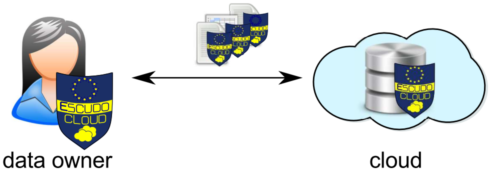
Solutions that provide protection guarantees giving the data owners both: full control over their data and cloud functionality over them



H2020 project "Enforceable Security in the Cloud to Uphold Data Ownership" (ESCUDO-CLOUD).

Cloud computing: ESCUDO-CLOUD's vision

Solutions that provide protection guarantees giving the data owners both: full control over their data and cloud functionality over them



- client-side trust boundary: only the behavior of the client should be considered trusted

  $\Longrightarrow$ techniques and implementations supporting direct processing of encrypted data in the cloud

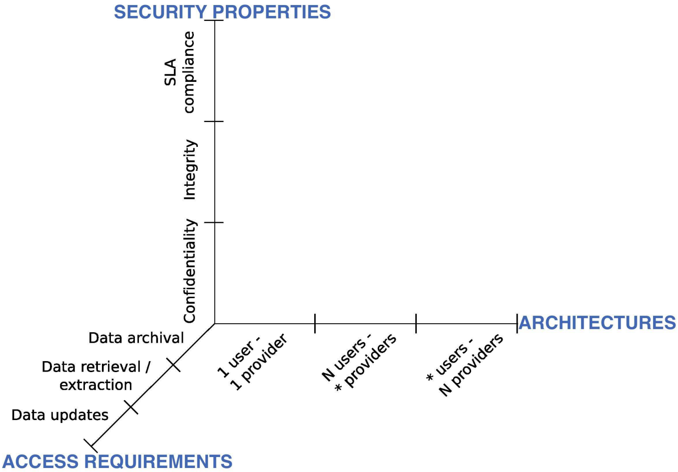
---

H2020 project "Enforceable Security in the Cloud to Uphold Data Ownership" (ESCUDO-CLOUD).

Characterization of
Data Protection Challenges

Scientific and technical challenges

Three dimensions characterize the problems and challenges



Security properties

- Confidentiality: protection of the data externally stored, the identity of the users accessing the data, the actions that users perform on the data
- Integrity: authenticity and integrity of the stored data as well as of the result of queries over them
- Availability (SLA): satisfaction by external providers of the data storage and access requirements users may wish to enforce (i.e., SLAs established between users and providers)

Access requirements

- Data archival: access to data is a primitive upload/download
  $\Longrightarrow$ protection of data in storage

- Data retrieval/extraction: access to data requires fine-grained data retrieval and execution of queries
  $\Longrightarrow$ protection of computations and query results

- Data update: access to data entails both access retrieval and enforcement of updates
  $\Longrightarrow$ protection of the actions as well as of their effect on the data

Architectures

- One user-one provider: a user relies on the cloud for enjoying external storage for her own use and access

  $\Longrightarrow$ protection of data at rest; fine-grained retrieval; query privacy

- Multiple users: a user relies on external storage for making her data available to others, and sharing and disseminating them in a selective way

  $\Longrightarrow$ authorizations and access control; multiple writers

- Multiple providers: one or more users adopt multiple servers for data storage and access

  $\Longrightarrow$ controlled data sharing and computation

Combinations of the dimensions

- Every combination of the different instances of the dimensions identifies new problems and challenges
- The security properties to be guaranteed can depend on the access requirements and on the trust assumption on the providers involved in storage and/or processing of data
- Providers can be:
  - curious
  - lazy
  - malicious

Some Challenges in Data Protection

Some issues and opportunities

- Protection of and fine-grained access to outsourced data
  - confidentiality (and integrity) of data at rest
  - fine-grained retrieval and query execution
- Selective information sharing
  - access control on resources in the cloud
- Integrity
  - integrity of stored data and query results

Protection of and Fine-Grained Access to Outsourced Data

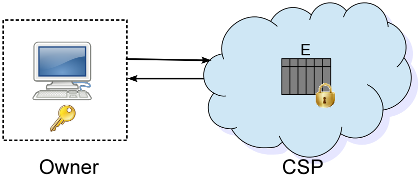
P. Samarati, S. De Capitani di Vimercati, "Cloud Security: Issues and Concerns," in *Encyclopedia on Cloud Computing,* S. Murugesan, I. Bojanova (eds.), Wiley, 2016.

S. De Capitani di Vimercati et al., "Encryption and Fragmentation for Data Confidentiality in the Cloud," in *Foundations of Security Analysis and Design VII,* A. Aldini, J. Lopez, F. Martinelli (eds.), Springer, 2014.

S. De Capitani di Vimercati, S. Foresti, P. Samarati, "Selective and Fine-Grained Access to Data in the Cloud," in *Secure Cloud Computing,* S. Jajodia, K. Kant, P. Samarati, V. Swarup, C. Wang (eds.), Springer, 2014.

The role of encryption in protecting data

- Current solutions put their focus on encryption services that can easily protect data at rest
- The CSP can be honest-but-curious and should not have access to the resource content
- Data confidentiality is provided by wrapping a layer of encryption around sensitive data (e.g., Boxcryptor, SpiderOak)



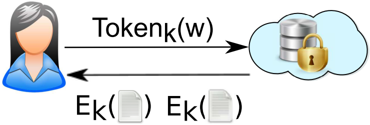
Fine-grained access to data in the cloud

- For confidentiality reasons, CSPs storing data cannot decrypt them for data processing/access

- Need mechanisms to support access to the outsourced data
  - effective and efficient
  - should not open the door to inferences

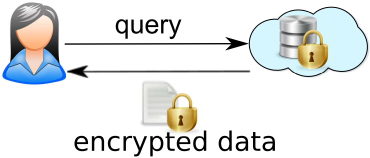
Fine-grained access: Approaches – 1

Keyword-based searches directly on the encrypted data: supported by specific cryptographic techniques (e.g., [CWLRL-11])



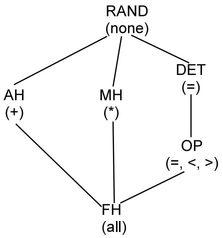
Fine-grained access: Approaches – 2

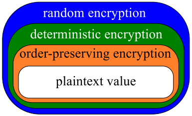
Homomorphic encryption: supports the execution of operations directly on the encrypted data (e.g., Gentry's system)

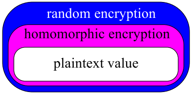


Fine-grained access: Approaches – 3

- Encryption schemas: each column can be encrypted with a different encryption schema, depending on the conditions to be evaluated on it (e.g., Google encrypted BigQuery)
- Onion encryption (CryptDB): different onion layers each of which supports the execution of a specific SQL operation (e.g., HanaDB SEEED framework) [PRZB-11]

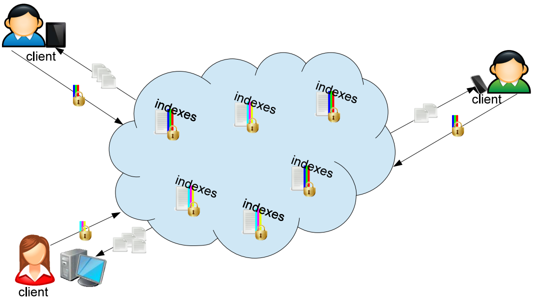






Fine-grained access: Approaches – 4

Indexes: metadata attached to the data and used for fine-grained information retrieval and query execution (e.g., [CDDJPS-05, HIML-02, WL-06])



can also be complementary to encryption (even with encryption users want to have the ability to perform searches based on metadata)

Encryption and indexes

Indexes associated with attributes are used by the provider to select data to be returned in response to a query

Patients

SSN	**Name**	**Disease**
123-45-6789	Alice	Asthma
234-56-7891	Bob	Asthma
345-67-8912	Carol	Asthma
456-78-9123	David	Bronchitis
567-89-1234	Eva	Gastritis

Patients^k

Counter	**Etuple**	I_S	I_N	I_D
1	x4Z3tfX2ShOSM	π	κ	α
2	mNHg1oC010p8w	ϖ	κ	α
3	WsIaCvfyF1Dxw	ξ	λ	α
4	JpO8eLTVgwV1E	ρ	κ	β
5	qctG6XnFNDTQc	ι	μ	α

Query on plaintext translated to a query on indexes and some postprocessing at the client

Encryption and indexes

Indexes associated with attributes are used by the provider to select data to be returned in response to a query

Patients

SSN	Name	Disease
123-45-6789	Alice	Asthma
234-56-7891	Bob	Asthma
345-67-8912	Carol	Asthma
456-78-9123	David	Bronchitis
567-89-1234	Eva	Gastritis

Patients^k

Counter	Etuple	I_S	I_N	I_D
1	x4Z3tfX2ShOSM	π	κ	α
2	mNHg1oC010p8w	ϖ	κ	α
3	WsIaCvfyF1Dxw	ξ	λ	α
4	JpO8eLTVgwV1E	ρ	κ	β
5	qctG6XnFNDTQc	ι	μ	α

Query on plaintext translated to a query on indexes and some postprocessing at the client

Original query

```
SELECT *
FROM Patients
WHERE Disease = 'Asthma'
```

Encryption and indexes

Indexes associated with attributes are used by the provider to select data to be returned in response to a query

Patients

SSN	**Name**	**Disease**
123-45-6789	Alice	Asthma
234-56-7891	Bob	Asthma
345-67-8912	Carol	Asthma
456-78-9123	David	Bronchitis
567-89-1234	Eva	Gastritis

Patients^k

Counter	**Etuple**	I_S	I_N	I_D
1	x4Z3tfX2ShOSM	π	κ	α
2	mNHg1oC010p8w	ϖ	κ	α
3	WsIaCvfyF1Dxw	ξ	λ	α
4	JpO8eLTVgwV1E	ρ	κ	β
5	qctG6XnFNDTQc	ι	μ	α

Query on plaintext translated to a query on indexes and some postprocessing at the client

Original query

SELECT *
FROM Patients
WHERE Disease ='Asthma'

At server

r = SELECT Etuple
FROM Patients^k
WHERE $I_D = \alpha$

Encryption and indexes

Indexes associated with attributes are used by the provider to select data to be returned in response to a query

Patients

SSN	Name	Disease
123-45-6789	Alice	Asthma
234-56-7891	Bob	Asthma
345-67-8912	Carol	Asthma
456-78-9123	David	Bronchitis
567-89-1234	Eva	Gastritis

Patients^k

Counter	Etuple	I_S	I_N	I_D
1	x4Z3tfX2ShOSM	π	κ	α
2	mNHg1oC010p8w	ϖ	κ	α
3	WslaCvfyF1Dxw	ξ	λ	α
4	JpO8eLTVgwV1E	ρ	κ	β
5	qctG6XnFNDTQc	ι	μ	α

Query on plaintext translated to a query on indexes and some postprocessing at the client

Original query

SELECT Name,Disease
FROM Patients
WHERE Disease ='Asthma'

At server

r = SELECT Etuple
FROM Patients^k
WHERE $I_D = \alpha$

At client

SELECT *
FROM Decrypt(r, *key*)
WHERE Disease = 'Asthma'

Encryption and indexes

Indexes associated with attributes are used by the provider to select data to be returned in response to a query

Patients

SSN	**Name**	**Disease**
123-45-6789	Alice	Asthma
234-56-7891	Bob	Asthma
345-67-8912	Carol	Asthma
456-78-9123	David	Bronchitis
567-89-1234	Eva	Gastritis

Patients^k

Counter	**Etuple**	I_S	I_N	I_D
1	x4Z3tfX2ShOSM	π	κ	α
2	mNHg1oC010p8w	ϖ	κ	α
3	WslaCvfyF1Dxw	ξ	λ	α
4	JpO8eLTVgwV1E	ρ	κ	β
5	qctG6XnFNDTQc	ι	μ	α

Query on plaintext translated to a query on indexes and some postprocessing at the client

Original query

SELECT Name,Disease
FROM Patients
WHERE Disease ='Asthma'

At server

r = SELECT Etuple
FROM Patients^k
WHERE $I_D = \alpha$

At client

SELECT *
FROM Decrypt(r, *key*)
WHERE Disease = 'Asthma'

Indexes – 1

Different choices for indexes [CDDJPS-05, HIML-02, WL-06]

- Actual attribute value, $t[I_i] = t[A_i]$ (very limited applicability)
- Direct index: each plaintext value is mapped onto one index value and viceversa ($t[I_i] = E_k(t[A_i])$)
  - \+ simple and precise for equality queries
  - – preserves plaintext value distinguishability (inference attacks)

[3] Asthma —— α [3]

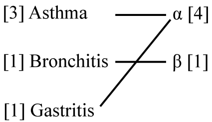
[1] Bronchitis —— β [1]

[1] Gastritis —— γ [1]

Indexes – 2

Different choices for indexes [CDDJPS-05, HIML-02, WL-06]

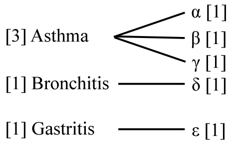
- Bucket index: each plaintext value is mapped onto one index value, with collisions (partition-based or hash-based)
  - + support for equality queries
  - + collisions remove plaintext distinguishability
  - – result may contain spurious tuples (post-processing query)
  - – still vulnerable to inference attacks



Indexes – 3

Different choices for indexes [CDDJPS-05, HIML-02, WL-06]

- Flattened index: each plaintext value is mapped onto one or more index values; all index values have the same number of occurrences (flattening), but each index value represents one plaintext value
  - \+ decreases exposure to inference attacks
  - – remains vulnerable in dynamic scenarios



Fragmentation and encryption

- Encryption makes query evaluation and application execution 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

- Alternative solutions limit encryption by coupling:
  - encryption
  - data fragmentation

Confidentiality constraints

- Sets of attributes such that the (joint) visibility of values of the attributes in the sets should be protected

- Sensitive attributes: the values of some attributes are considered sensitive and should not be visible
  $\Longrightarrow$ singleton constraints

- Sensitive associations: the associations among values of given attributes are sensitive and should not be visible
  $\Longrightarrow$ non-singleton constraints

Confidentiality constraints – Example

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

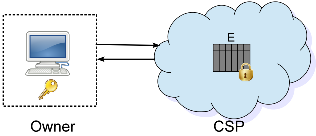
- {Telephone}, {Email}
  - attributes Telephone and Email are sensitive (cannot be stored in the clear)
- {Name,Salary}, {Name,Position}, {Name,DoB}
  - attributes Salary, Position, and DoB are private of an individual and cannot be stored in the clear in association with the Name
- {DoB,Gender,Zip,Salary}, {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

Fragmentation

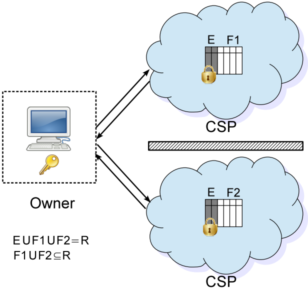
- Fragmentation partitions attributes of original relation to provide (maximal) availability of attributes in plaintext form for access
  - no sensitive attribute visible in external fragments
  - no sensitive association visible in external fragments
  - ensure unlinkability of fragments (no attribute in common)
- Different approaches:
  - Two can keep a secret splits information over two independent servers that cannot communicate [ABGGKMSTX-05]
  - Multiple unlinkable fragments allows for more than two non-linkable fragments [CDFJPS-10]
  - Keep a few involves the data owner as a trusted party to maintain a limited amount of data [CDFJPS-09, CDFJPS-11]

Fragmentation and encryption: Approaches

Encryption

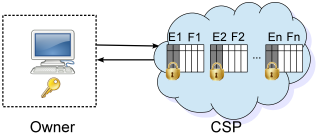


Encryption and fragmentation (two can keep a secret)



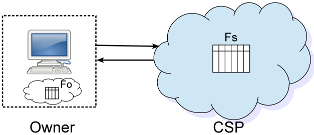
$E \cup F1 \cup F2 = R$
$F1 \cup F2 \subseteq R$

Encryption and fragmentation (multiple unlinkable fragments)



$E1 \cup F1 = \ldots = En \cup Fn = R$
$F1 \cup \ldots \cup Fn \subseteq R$

Fragmentation (keep a few)



$Fo \cup Fs = R$

Fragmentation and encryption – Examples

P

SSN	Name	YoB	Job	Disease	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Two can keep a secret

F_1

tid	Name	YoB	SSN^k	$Disease^k$

F_2

tid	Job	Doctor	SSN^k	$Disease^k$

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Multiple unlinkable fragments

F_1

$salt_1$	enc_1	Name	YoB

F_2

$salt_2$	enc_2	Job

F_3

$salt_3$	enc_3	Disease	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Keep a few

F_o

tid	SSN	Name	Disease

F_s

tid	YoB	Job	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Fragmentation and encryption – Examples

P

<u>SSN</u>	Name	YoB	Job	Disease	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Two can keep a secret

F_1

<u>tid</u>	Name	YoB	SSN^k	Disease^k

F_2

<u>tid</u>	Job	Doctor	SSN^k	Disease^k

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Multiple unlinkable fragments

F_1

<u>salt_1</u>	enc_1	Name	YoB

F_2

<u>salt_2</u>	enc_2	Job

F_3

<u>salt_3</u>	enc_3	Disease	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Keep a few

F_o

<u>tid</u>	SSN	Name	Disease

F_s

<u>tid</u>	YoB	Job	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Fragmentation and encryption – Examples

P

SSN	Name	YoB	Job	Disease	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Two can keep a secret

F_1

tid	Name	YoB	SSN^k	$Disease^k$

F_2

tid	Job	Doctor	SSN^k	$Disease^k$

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Multiple unlinkable fragments

F_1

$salt_1$	enc_1	Name	YoB

F_2

$salt_2$	enc_2	Job

F_3

$salt_3$	enc_3	Disease	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Keep a few

F_o

tid	SSN	Name	Disease

F_s

tid	YoB	Job	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Fragmentation and encryption – Examples

P

<u>SSN</u>	Name	YoB	Job	Disease	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Two can keep a secret

F_1

<u>tid</u>	Name	YoB	SSN^k	$Disease^k$

F_2

<u>tid</u>	Job	Doctor	SSN^k	$Disease^k$

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Multiple unlinkable fragments

F_1

<u>$salt_1$</u>	enc_1	Name	YoB

F_2

<u>$salt_2$</u>	enc_2	Job

F_3

<u>$salt_3$</u>	enc_3	Disease	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Keep a few

F_o

<u>tid</u>	SSN	Name	Disease

F_s

<u>tid</u>	YoB	Job	Doctor

c_0 = {SSN}
c_1 = {Name,Disease}
c_2 = {Name,Job}
c_3 = {Job,Disease}

Fragmentation and inference

- Fragmentation assumes attributes to be independent

- In presence of data dependencies:
  - sensitive attributes/associations may be indirectly exposed
  - fragments may be indirectly linkable

S. De Capitani di Vimercati, S. Foresti, S. Jajodia, G. Livraga, S. Paraboschi, P. Samarati, "Fragmentation in Presence of Data Dependencies," in *IEEE Transactions on Dependable and Secure Computing (TDSC)*, vol. 11, n. 6, November/December 2014, pp. 510-523.

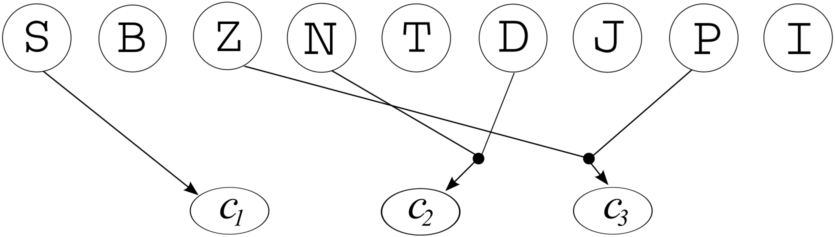
Fragmentation and inference – Example

R(SSN, Birth, ZIP, Name, Treatment, Disease, Job, Premium, Insurance)

S B Z N T D J P I

Fragmentation and inference – Example

R(SSN, Birth, ZIP, Name, Treatment, Disease, Job, Premium, Insurance)

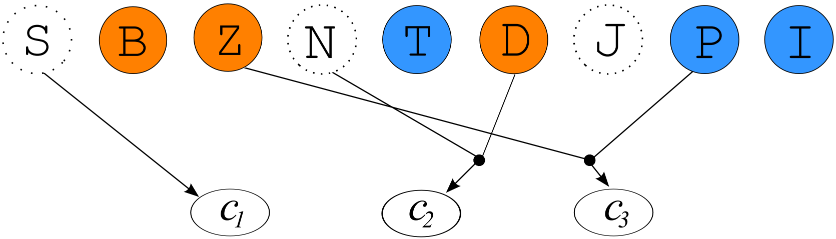


Constraints

c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

Fragmentation and inference – Example

R(SSN, Birth, ZIP, Name, Treatment, Disease, Job, Premium, Insurance)

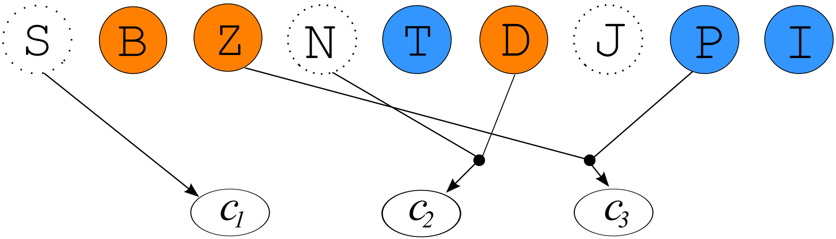


Constraints

c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

Fragmentation and inference – Example

R(SSN, Birth, ZIP, Name, Treatment, Disease, Job, Premium, Insurance)



Constraints

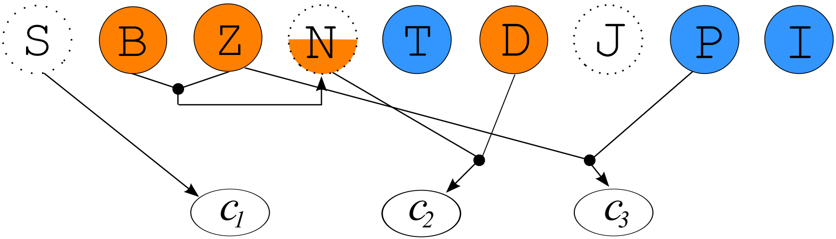
c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

Dependencies

d_1 = {Birth, ZIP} $\rightsquigarrow$ Name
d_2 = {Treatment} $\rightsquigarrow$ Disease
d_3 = {Disease} $\rightsquigarrow$ Job
d_4 = {Insurance, Premium} $\rightsquigarrow$ Job

Fragmentation and inference – Example

R(SSN, Birth, ZIP, <u>Name</u>, Treatment, Disease, Job, Premium, Insurance)



Constraints

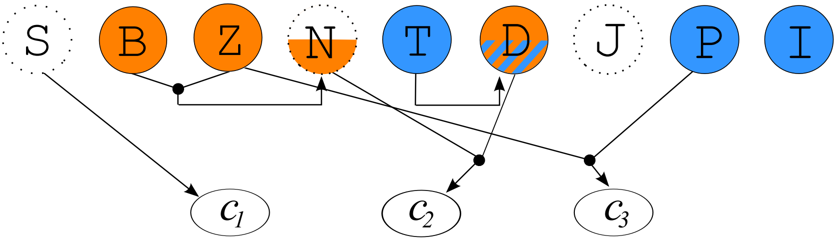
c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

Dependencies

d_1 = {Birth, ZIP}$\rightsquigarrow$ Name
d_2 = {Treatment}$\rightsquigarrow$ Disease
d_3 = {Disease}$\rightsquigarrow$ Job
d_4 = {Insurance, Premium}$\rightsquigarrow$ Job

Fragmentation and inference – Example

R(SSN, Birth, ZIP, Name, Treatment, Disease, Job, Premium, Insurance)



Constraints

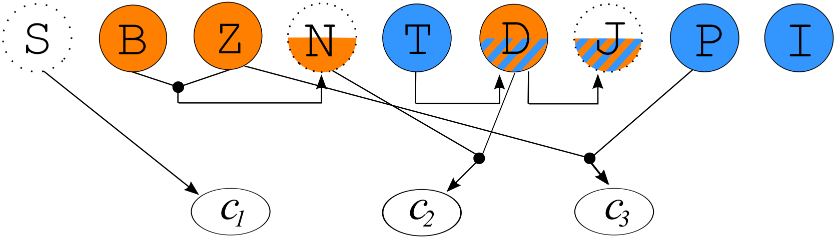
c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

Dependencies

d_1 = {Birth, ZIP} $\rightsquigarrow$ Name
d_2 = {Treatment} $\rightsquigarrow$ Disease
d_3 = {Disease} $\rightsquigarrow$ Job
d_4 = {Insurance, Premium} $\rightsquigarrow$ Job

Fragmentation and inference – Example

R(SSN, Birth, ZIP, Name, Treatment, Disease, Job, Premium, Insurance)



Constraints

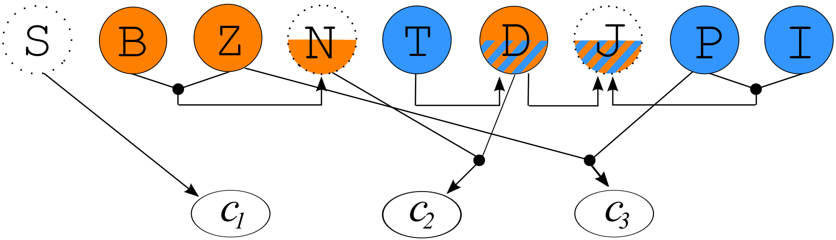
c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

Dependencies

d_1 = {Birth, ZIP} $\rightsquigarrow$ Name
d_2 = {Treatment} $\rightsquigarrow$ Disease
d_3 = {Disease} $\rightsquigarrow$ Job
d_4 = {Insurance, Premium} $\rightsquigarrow$ Job

Fragmentation and inference – Example

R(SSN, Birth, ZIP, Name, Treatment, Disease, Job, Premium, Insurance)



Constraints

c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

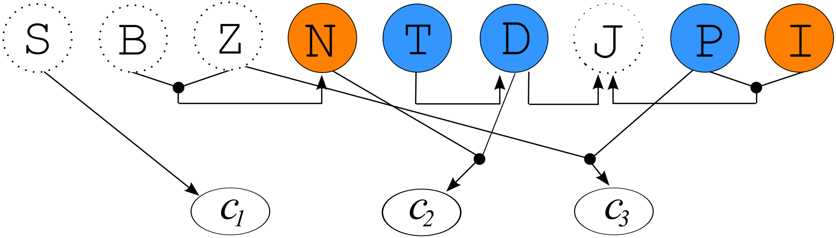
Dependencies

d_1 = {Birth, ZIP} $\rightsquigarrow$ Name
d_2 = {Treatment} $\rightsquigarrow$ Disease
d_3 = {Disease} $\rightsquigarrow$ Job
d_4 = {Insurance, Premium} $\rightsquigarrow$ Job

Fragmenting with data dependencies

Take into account data dependencies in fragmentation

- Fragments should not contain sensitive attributes/associations neither directly nor indirectly



Constraints

c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

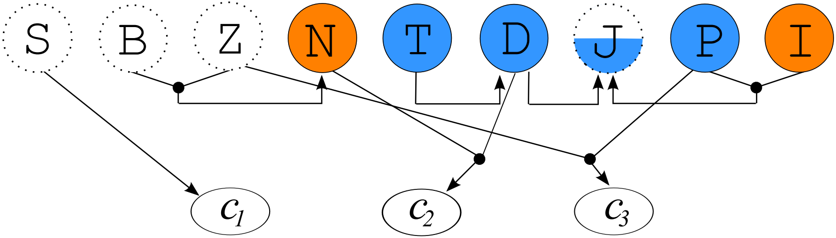
Dependencies

d_1 = {Birth, ZIP} $\rightsquigarrow$ Name
d_2 = {Treatment} $\rightsquigarrow$ Disease
d_3 = {Disease} $\rightsquigarrow$ Job
d_4 = {Insurance, Premium} $\rightsquigarrow$ Job

Fragmenting with data dependencies

Take into account data dependencies in fragmentation

- Fragments should not contain sensitive attributes/associations neither directly nor indirectly



Constraints

c_1 = {SSN}
c_2 = {Name, Disease}
c_3 = {ZIP, Premium}

Dependencies

d_1 = {Birth, ZIP} $\rightsquigarrow$ Name
d_2 = {Treatment} $\rightsquigarrow$ Disease
d_3 = {Disease} $\rightsquigarrow$ Job
d_4 = {Insurance, Premium} $\rightsquigarrow$ Job

Variations/open issues . . .

- Fragmentation quality metrics (e.g., maximize number/size of attributes in plaintext, optimize wrt workload/visibility requirements) [CDFJPS-11]
- Joint application of indexes and fragments (need to control information leakage) [DFJPS-13a]
- Data fragmentation in hybrid clouds
- Support for different kinds of query

Selective Information Sharing

S. De Capitani di Vimercati, S. Foresti, G. Livraga, P. Samarati, "Selective and Private Access to Outsourced Data Centers," in *Handbook on Data Centers,* S.U. Khan, A.Y. Zomaya (eds.), Springer, 2015.

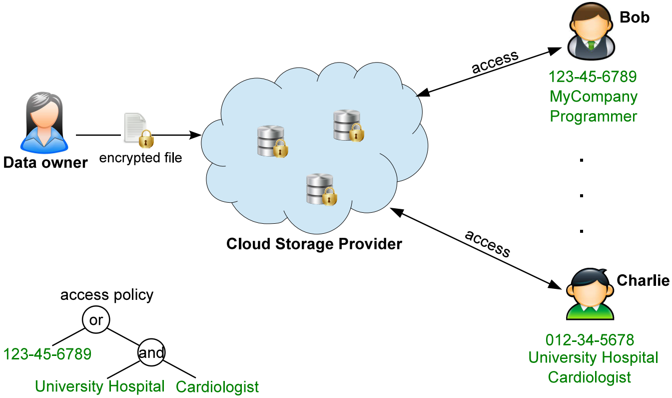
S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Encryption Policies for Regulating Access to Outsourced Data," in *ACM Transactions on Database Systems (TODS),* vol. 35, n. 2, April 2010, pp. 12:1-12:46.

Selective information sharing

- Different users might need to enjoy different views on the outsourced data
- Enforcement of the access control policy requires the data owner to mediate access requests
  $\Longrightarrow$ impractical (if not inapplicable)
- Authorization enforcement may not be delegated to the provider
  $\Longrightarrow$ data owner should remain in control

Selective information sharing: Approaches – 1

- Attribute-based encryption (ABE): allow derivation of a key only by users who hold certain attributes (based on asymmetric cryptography)



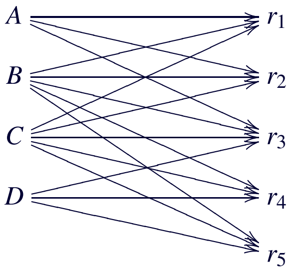
Selective information sharing: Approaches – 2

- Selective encryption: the authorization policy defined by the data owner is translated into an equivalent encryption policy
  - users will be able to access only the resources for which they have the key

Selective encryption – 1

- Selective encryption: different keys are used to encrypt different data and users can know (or can derive) the keys of the data they can access [DFJPS-10, DFJPS-07]
  - data themselves need to directly enforce access control
  - authorization to access a resource translated into knowledge of the key with which the resource is encrypted

	r_1	r_2	r_3	r_4	r_5
A	1	1	1	0	0
B	1	1	1	1	1
C	1	1	1	1	1
D	0	0	1	1	1



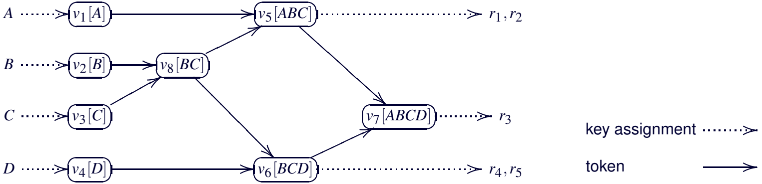
Selective encryption – 2

Requirements:

- one version of data (no replication); one key per user

Basic idea:

- key derivation method: via public tokens a user can derive all keys of the resources she is allowed to access



- user A can access $\{r_1, r_2, r_3\}$
- users B and C can access $\{r_1, r_2, r_3, r_4, r_5\}$
- user D can access $\{r_3, r_4, r_5\}$

Selective encryption – 3

Exploit ACLs to minimize number of keys and tokens

- Keys:
  - one key per user
  - an additional key for each non-singleton ACL
- Resources are encrypted with the key of their ACLs
- Tokens allow users to derive the keys of the ACLs to which they belong (to limit the number of tokens additional keys might be inserted for 'factoring' derivation paths)

Construction of the key and token graph

Start from an authorization policy $\mathscr{A}$

1. Create a vertex/key for each user and for each non-singleton *acl* (initialization)

2. For each vertex v corresponding to a non-singleton *acl*, find a cover without redundancies (covering)
   - for each user u in v.*acl*, find an ancestor v' of v with $u \in v'$.*acl*

3. Factorize common ancestors (factorization)

Key and token graph – Example

	r_1	r_2	r_3	r_4	r_5
A	1	1	1	0	0
B	1	1	1	1	1
C	1	1	1	1	1
D	0	0	1	1	1

Initialization

$v_1[A]$ $v_5[ABC]$

$v_2[B]$

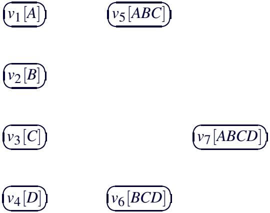
$v_3[C]$ $v_7[ABCD]$

$v_4[D]$ $v_6[BCD]$

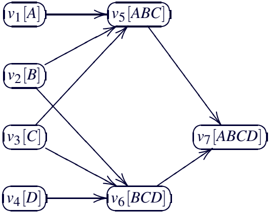
Key and token graph – Example

	r_1	r_2	r_3	r_4	r_5
A	1	1	1	0	0
B	1	1	1	1	1
C	1	1	1	1	1
D	0	0	1	1	1

Initialization

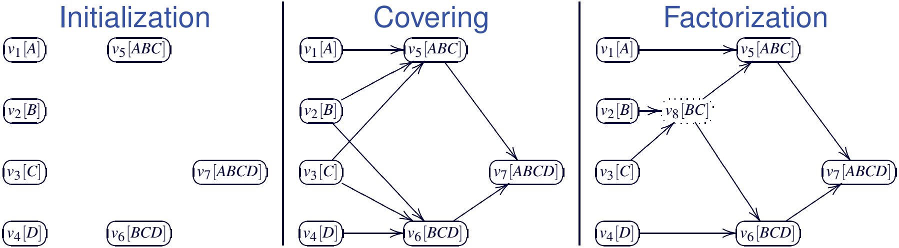


Covering



Key and token graph – Example

	r_1	r_2	r_3	r_4	r_5
A	1	1	1	0	0
B	1	1	1	1	1
C	1	1	1	1	1
D	0	0	1	1	1



Policy updates

- When authorizations dynamically change, the data owner needs to:
  - download the resource from the provider
  - create a new key for the resource
  - decrypt the resource with the old key
  - re-encrypt the resource with the new key
  - upload the resource to the provider and communicate the public catalog updates

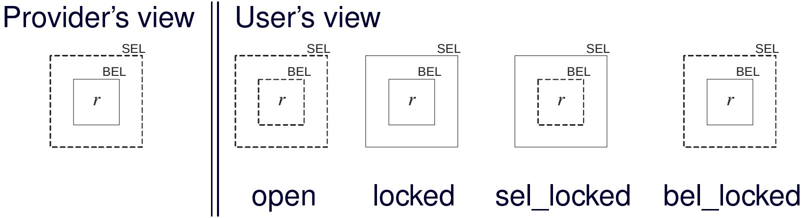
  $\Longrightarrow$ inefficient

- Possible solution: over-encryption [DFJPS-10, DFJPS-07]

Over-encryption

- Resources are encrypted twice
  - by the owner, with a key shared with the users and unknown to the provider (Base Encryption Layer - BEL level)
  - by the provider, with a key shared with authorized users (Surface Encryption Layer - SEL level)
- To access a resource a user must know both the corresponding BEL and SEL keys
- Grant and revoke operations may require
  - the addition of new tokens at the BEL level
  - the update of the SEL level according to the operations performed

Over-encryption



- Each layer is depicted as a fence
  - discontinuous, if the key is known
  - continuous, if the key is not known (protection cannot be passed)

Variations/open issues ...

- Support of write authorizations [DFJLPS-13]
- Support of multi-owners scenario [DFJPPS-10]
- Combination of selective encryption and indexes [DFJPS-11]

Integrity of Data Storage and Computation

S. De Capitani di Vimercati, S. Foresti, S. Jajodia, G. Livraga, S. Paraboschi, P. Samarati, “Integrity for Distributed Queries,”in *Proc. of the 2nd IEEE Conference on Communications and Network Security (CNS 2014)*, CA, USA, October 2014.

S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, “Integrity for Join Queries in the Cloud,” in *IEEE Transactions on Cloud Computing (TCC),* vol. 1, n. 2, July-December 2013, pp. 187-200.

Integrity of storage and query computation – 1

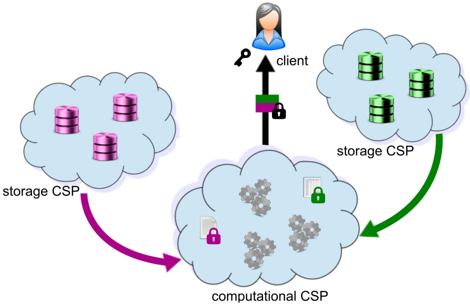
- Data owner and users need mechanisms that provide integrity for query results:
  - correctness: computed on genuine data
  - completeness: computed on the whole data collection
  - freshness: computed on the most recent version of the data
- Two approaches:
  - deterministic: uses authenticated data structures (e.g., signature chains, Merkle hash trees, skip lists) or encryption-based solutions (e.g., verifiable homomorphic encryption schema [LDPW-14])
  - probabilistic: exploits insertion of fake tuples in query results, replication of tuples in query results, pre-computed tokens (e.g., [DFJPS-13b,DFJPS-14,DFJLPS-14b,XWYM-07])

Integrity of storage and query computation – 2

- Other approaches consider the verification of the integrity of query results of complex queries (joins):
  - Merkle hash tree or its variations [LHKR-06, YPPK-09]
    - support only joins on which the Merkle hash tree has been constructed
  - fake tuples [XWYM-07]
    - spurious tuples
    - network overhead

Computation with multiple providers

- Different CSPs are available on the market, offering a variety of services (e.g., storage, computation) at different prices
- Users can select the CSP that better matches their security, economic, and functional requirements
- Multiple CSPs can help enhancing security but
  $\Longrightarrow$ need solutions to verify the correct behavior of these CSPs



Probabilistic approach for join queries

- A client, with the cooperation of the storage servers, can assess the integrity of joins performed by a computational cloud
- Protection techniques [DFJPS-13b,DFJPS-14]:
  - encryption makes data unintelligible
  - markers, fake tuples not recognizable as such by the computational cloud (and not colliding with real tuples)
  - twins, replication of existing tuples
- A marker missing or a twin appearing solo $\Longrightarrow$ integrity violation
- Probabilistic guarantee depending on the amount of control (markers and twins) inserted

On-the-fly encryption

- Server S encrypts $B(I, Att)$, obtaining $B_k(I_k, B.Tuple_k)$
  - For each t in B, there is τ in B_k: $\tau[I_k]=E_k(t[I])$ and $\tau[B.Tuple_k]=E_k(t)$
  - E is a symmetric encryption function with key k
  - k is defined by the client and changes at every query
- Encryption provides data confidentiality

R_l

	I	Attr
l_1	a	Ann
l_2	b	Beth
l_3	c	Cloe

R_r

	I	Attr
r_1	a	flu
r_2	a	asthma
r_3	b	ulcer
r_4	e	hernia
r_5	e	flu
r_6	e	cancer

J

	L.I	L.Attr	R.I	R.Attr	
l_1	a	Ann	a	flu	r_1
l_1	a	Ann	a	asthma	r_2
l_2	b	Beth	b	ulcer	r_3

On-the-fly encryption

- Server S encrypts $B(I, Att)$, obtaining $B_k(I_k, B.Tuple_k)$
  - For each t in B, there is τ in B_k: $\tau[I_k]=E_k(t[I])$ and $\tau[B.Tuple_k]=E_k(t)$
  - E is a symmetric encryption function with key k
  - k is defined by the client and changes at every query
- Encryption provides data confidentiality

R_{lk}

I_k	$L.Tuple_k$
α	λ_1
β	λ_2
γ	λ_3

R_{rk}

I_k	$R.Tuple_k$
α	ρ_1
α	ρ_2
β	ρ_3
ε	ρ_4
ε	ρ_5
ε	ρ_6

J_k

$L.I_k$	$L.Attr_k$	$R.I_k$	$R.Attr_k$
α	λ_1	α	ρ_1
α	λ_1	α	ρ_2
β	λ_2	β	ρ_3

Markers

- Artificial tuples injected into R_l by S_l and R_r by S_r
  - not recognizable by the computational server
  - do not generate spurious tuples
  - inserted in a concerted manner to guarantee that they belong to the join result
- The absence of markers signals incompleteness of the join result

R_l

	I	Attr
l_1	a	Ann
l_2	b	Beth
l_3	c	Cloe

R_r

	I	Attr
r_1	a	flu
r_2	a	asthma
r_3	b	ulcer
r_4	e	hernia
r_5	e	flu
r_6	e	cancer

J

	L.I	L.Attr	R.I	R.Attr	
l_1	a	Ann	a	flu	r_1
l_1	a	Ann	a	asthma	r_2
l_2	b	Beth	b	ulcer	r_3

Markers

- Artificial tuples injected into R_l by S_l and R_r by S_r
  - not recognizable by the computational server
  - do not generate spurious tuples
  - inserted in a concerted manner to guarantee that they belong to the join result
- The absence of markers signals incompleteness of the join result

$R_l{}^*$

	I	Attr
l_1	a	Ann
l_2	b	Beth
l_3	c	Cloe
m_1	x	*marker*$_1$

$R_r{}^*$

	I	Attr
r_1	a	flu
r_2	a	asthma
r_3	b	ulcer
r_4	e	hernia
r_5	e	flu
r_6	e	cancer
m_2	x	*marker*$_2$

J^*

	L.I	L.Attr	R.I	R.Attr	
l_1	a	Ann	a	flu	r_1
l_1	a	Ann	a	asthma	r_2
l_2	b	Beth	b	ulcer	r_3
m_1	x	*marker*$_1$	x	*marker*$_2$	m_2

Twins

- Duplicates of tuples that satisfy condition C_{twin} that
  - is defined on the join attribute I
  - tunes the percentage p_t of twins
  - is defined by the client and communicated to S_l and S_r
- Twin pairs are not recognizable by the computational server
- A twin appearing solo signals incompleteness of the join result

R_l

	I	Attr
l_1	a	Ann
l_2	b	Beth
l_3	c	Cloe

R_r

	I	Attr
r_1	a	flu
r_2	a	asthma
r_3	b	ulcer
r_4	e	hernia
r_5	e	flu
r_6	e	cancer

J

	L.I	L.Attr	R.I	R.Attr	
l_1	a	Ann	a	flu	r_1
l_1	a	Ann	a	asthma	r_2
l_2	b	Beth	b	ulcer	r_3

Twins

- Duplicates of tuples that satisfy condition C_{twin} that
  - is defined on the join attribute I
  - tunes the percentage p_t of twins
  - is defined by the client and communicated to S_l and S_r
- Twin pairs are not recognizable by the computational server
- A twin appearing solo signals incompleteness of the join result

R_l^*

	I	Attr
l_1	a	Ann
l_2	b	Beth
l_3	c	Cloe
$\bar{l}_2$	$\bar{\text{b}}$	Beth

R_r^*

	I	Attr
r_1	a	flu
r_2	a	asthma
r_3	b	ulcer
r_4	e	hernia
r_5	e	flu
r_6	e	cancer
$\bar{r}_3$	$\bar{\text{b}}$	ulcer

J^*

	L.I	L.Attr	R.I	R.Attr	
l_1	a	Ann	a	flu	r_1
l_1	a	Ann	a	asthma	r_2
l_2	b	Beth	b	ulcer	r_3
$\bar{l}_2$	$\bar{\text{b}}$	Beth	$\bar{\text{b}}$	ulcer	$\bar{r}_3$

Query execution – Example

CLIENT

COMPUTATIONAL CLOUD

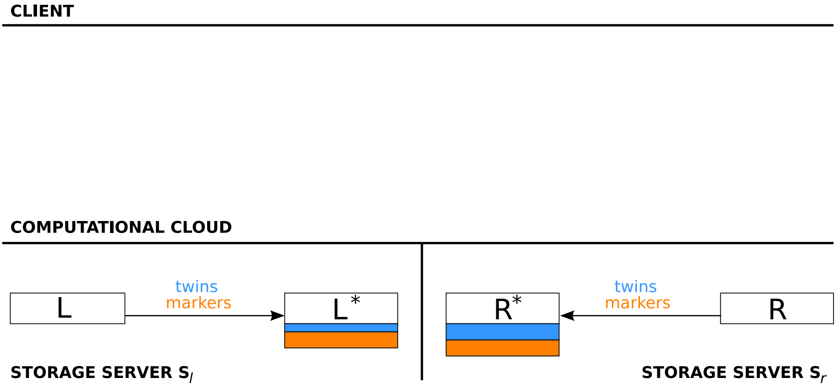
L

R

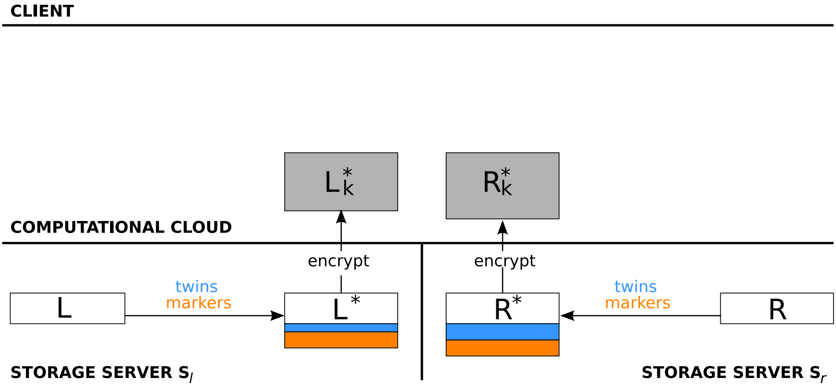
STORAGE SERVER S_l

STORAGE SERVER S_r

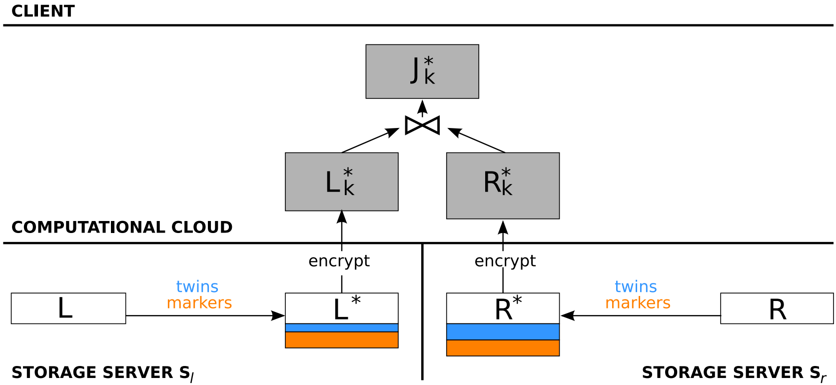
Query execution – Example



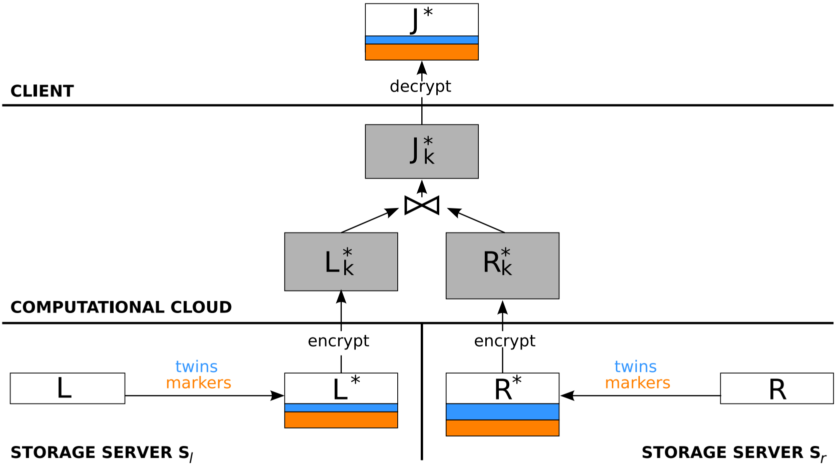
Query execution – Example



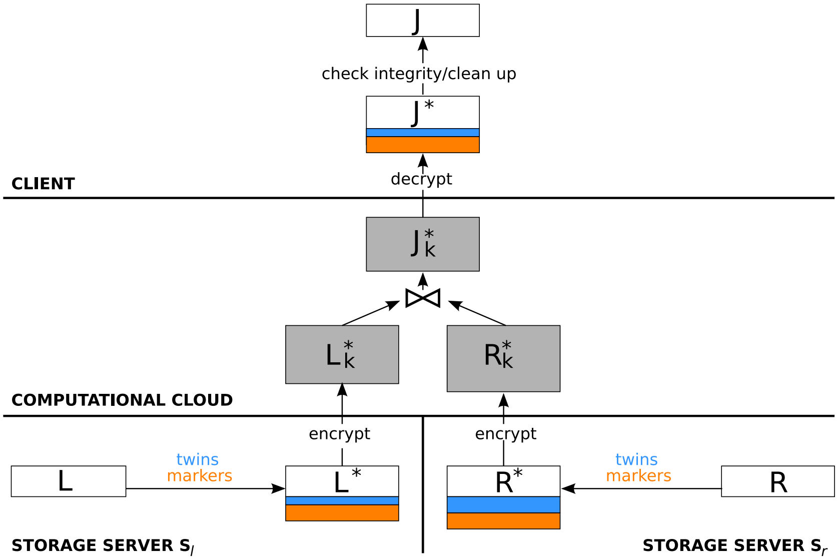
Query execution – Example



Query execution – Example



Query execution – Example



Markers and twins: Integrity guarantees

- The guarantee offered by markers and twins can be measured as the probability of the computational cloud to go undetected when omitting tuples

- Markers and twins offer complementary protection:
  - Twins are twice as effective as markers, but loose their effectiveness when the computational cloud omits a large fraction of tuples (extreme case: all tuples omitted)
  - Markers allow detecting extreme behavior (all tuples omitted) and provide effective when the computational cloud omits a large fraction of tuples

Variations/open issues ...

- Execution of a join as a semi-join to support n:m joins and protect join profile [DFJPS-14]
- Application of the techniques to only a portion of the data (verification object) [DFJPS-14]
- Application of the techniques in a distributed computation scenario (e.g., MapReduce) [DFJLPS-14b]
- Consideration of different trust levels
- Removal of trust assumptions in the storage servers

Conclusions

- Novel scenarios provide great convenience and benefit in the management and access to the information but require solutions to protect data

- Need to provide users and data owners with control over their data

- Data protection solutions are beneficial to both:
  - users and data owners (empowered with control)
  - CSPs and data controllers (increased confidence of users, decreased liability)

References – 1

- [ABGGKMSTX-05] 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 CIDR*, Asilomar, CA, USA, January 2005.
- [AT-83] S. Akl, P. Taylor, "Cryptographic Solution to a Problem of Access Control in a Hierarchy," *ACM TOCS*, vol. 1, no. 3, August 1983.
- [B-70] B.H. Bloom, "Trade-offs in Hash Coding with Allowable Error," in *Communication of the ACM*, vol. 13, no. 7, July 1970.
- [BS-04] S.M. Bellovin, B. Cheswick, "Privacy-enhanced Searches Using Encrypted Bloom Filters," in *Cryptology ePrint*, 2004.
- [CDFJPS-09] 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 ESORICS*, Saint-Malo, France, September 2009.
- [CDFJPS-10] 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 TISSEC*, vol. 13, no. 3, July 2010.
- [CDFJPS-11] V. Ciriani, S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Selective Data Outsourcing for Enforcing Privacy," in *JCS*, vol. 19, n. 3, 2011.

References – 2

- [CKGS-98] B. Chor, E. Kushilevitz, O. Goldreich, M. Sudan, "Private Information Retrieval," in *JACM*, vol. 45, no. 6, 1998.
- [CWLRL-11] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-Preserving Multikeyword Ranked Search over Encrypted Cloud Data," in *Proc. of INFOCOM*, Shanghai, China, April 2011.
- [CDDJPS-05] A. Ceselli, E. Damiani, S. De Capitani di Vimercati, S. Jajodia, S. Paraboschi, P. Samarati, "Modeling and Assessing Inference Exposure in Encrypted Databases," in *ACM TISSEC*, vol. 8, no. 1, February 2005.
- [CMW-06] J. Crampton, K. Martin, P. Wild, "On Key Assignment for Hierarchical Access Control," in *Proc. of CSFW*, Venice, Italy, July 2006.
- [DEFJLS-14] S. De Capitani di Vimercati, R.F. Erbacher, S. Foresti, S. Jajodia, G. Livraga, P. Samarati, "Encryption and Fragmentation for Data Confidentiality in the Cloud," in *Foundations of Security Analysis and Design VII,* A. Aldini, J. Lopez, F. Martinelli (eds.), Springer, 2014.
- [DFJLPS-13] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, G. Livraga, S. Paraboschi, P. Samarati, "Enforcing Dynamic Write Privileges in Data Outsourcing," in *Computers & Security*, vol. 39, November 2013.

References – 3

- [DFJLPS-14a] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, G. Livraga, S. Paraboschi, P. Samarati, "Fragmentation in Presence of Data Dependencies," in *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 2014.
- [DFJLPS-14b] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, G. Livraga, S. Paraboschi, P. Samarati, "Integrity for Distributed Queries," in *Proc. of CNS*, San Francisco, CA, USA, October 2014.
- [DFJPPS-10] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, G. Pelosi, P. Samarati, "Encryption-based Policy Enforcement for Cloud Storage," in *Proc. of SPCC 2010*, Genova, Italy, June 2010.
- [DFJPS-07] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Over-encryption: Management of Access Control Evolution on Outsourced Data," in *Proc. of VLDB 2007*, Vienna, Austria, September 2007.
- [DFJPS-10] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Encryption Policies for Regulating Access to Outsourced Data," in *ACM Transactions on Database Systems (TODS)*, vol. 35, n. 2, April 2010.
- [DFJPS-11] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Private Data Indexes for Selective Access to Outsourced Data," in *Proc. of WPES*, Chicago, Illinois, USA, October 2011.

References – 4

- [DFJPS-13a] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "On Information Leakage by Indexes over Data Fragments," in *Proc. of PrivDB*, Brisbane, Australia, April 2013.
- [DFJPS-13b] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Integrity for Join Queries in the Cloud," in *IEEE TCC*, vol. 1, n. 2, July-December 2013.
- [DFJPS-14] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, "Optimizing Integrity Checks for Join Queries in the Cloud," in *Proc. of DBSec*, Vienna, Austria, July 2014.
- [DFLS-15] S. De Capitani di Vimercati, S. Foresti, G. Livraga, P. Samarati, "Selective and Private Access to Outsourced Data Centers," in *Handbook on Data Centers,* S.U. Khan, A.Y. Zomaya (eds.), Springer, 2015.
- [DFM-04] A. De Santis, A.L. Ferrara, B. Masucci, "Cryptographic Key Assignment Schemes for any Access Control Policy," in *Inf. Process. Lett.*, vol. 92, no. 4, 2004.
- [DFPPS-11] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, P. Samarati, "Efficient and Private Access to Outsourced Data," in *Proc. of ICDCS,* Minneapolis, Minnesota, USA, June 2011.

References – 5

- [DFPPS-13a] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, P. Samarati, “Supporting Concurrency and Multiple Indexes in Private Access to Outsourced Data,” in *JCS*, vol. 21, n. 3, 2013.
- [DFPPS-13b] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, P. Samarati, “Distributed Shuffling for Preserving Access Confidentiality,” in *Proc. of ESORICS 2013*, Egham, U.K., September 2013.
- [DFPPS-14] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, P. Samarati, “Protecting Access Confidentiality with Data Distribution and Swapping,” in *Proc. of BDCloud*, Sydney, Australia, December 2014.
- [DFPPS-15a] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, P. Samarati, “Shuffle Index: Efficient and Private Access to Outsourced Data,” in *ACM TOS*, 2015.
- [DFPPS-15b] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, P. Samarati, “Three-Server Swapping for Access Confidentiality,” in *IEEE TCC*, 2015 (to appear).
- [DFS-14] S. De Capitani di Vimercati, S. Foresti, P. Samarati, “Selective and Fine-Grained Access to Data in the Cloud,” in *Secure Cloud Computing,* S. Jajodia, K. Kant, P. Samarati, V. Swarup, C. Wang (eds.), Springer, 2014.

References – 6

- [FPS-15] S. Foresti, V. Piuri, G. Soares, “On the Use of Fuzzy Logic in Dependable Cloud Management,” in *IEEE CNS,* Florence, Italy, September 2014 (poster session).
- [G-03] E. Goh, “Secure Indexes,” in *IACR Cryptology ePrint Archive*, 2003.
- [G-80] E. Gudes, “The Design of a Cryptography based Secure File System,” in *IEEE TSE*, vol. 6, no. 5, September 1980.
- [HIML-02] H. Hacigümüş, B. Iyer, S. Mehrotra, C. Li, “Executing SQL over Encrypted Data in the Database-Service-Provider Model,” in *Proc. of the ACM SIGMOD*, Madison, Wisconsin, USA, June 2002.
- [HL-90] L. Harn, H. Lin, “A Cryptographic Key Generation Scheme for Multilevel Data Security,” *Computers and Security*, vol. 9, no. 6, October 1990.
- [HY-03] M. Hwang, W. Yang, “Controlling Access in Large Partially Ordered Hierarchies using Cryptographic Keys,” *The Journal of Systems and Software*, vol. 67, no. 2, August 2003.
- [JP-13] R. Jhawar, V. Piuri, “Adaptive Resource Management for Balancing Availability and Performance in Cloud Computing”, in *Proc. of SECRYPT 2013,* Reykjavik, Iceland, July 2013.

References – 7

- [PRZB-11] R.A. Popa, C.M.S. Redfield, N. Zeldovich, H. Balakrishnan, “CryptDB: Protecting Confidentiality with Encrypted Query Processing,” in *Proc. of SOSP*, Cascais, Portugal, October 2011.
- [RVBM-09] M. Raykova, B. Vo, S.M. Bellovin, T. Malkin, “Secure Anonymous Database Search,” in *Proc. of ACM CCSW*, Chicago, IL, USA, November 2009.
- [JP-12] R. Jhawar, V. Piuri, “Fault Tolerance Management in IaaS Clouds”, in *Proc. of ESTEL,* Rome, Italy, October 2012.
- [JPS-12a] R. Jhawar, V. Piuri, P. Samarati, “Supporting Security Requirements for Resource Management in Cloud Computing,” in *Proc. of CSE*, Paphos, Cyprus, December 2012.
- [JPS-12b] R. Jhawar, V. Piuri, M. Santambrogio, “A Comprehensive Conceptual System-Level Approach to Fault Tolerance in Cloud Computing”, in *Proc. of SysCon 2012*, Vancouver, BC, Canada, March 2012.
- [JPS-13] R. Jhawar, V. Piuri, M. Santambrogio, “Fault Tolerance Management in Cloud Computing: A System-Level Perspective”, in *IEEE Systems Journal*, vol. 7, n. 2, June 2013.

References – 8

- [LC-04] P. Lin, K.S. Candan, “Hiding Traversal of Tree Structured Data from Untrusted Data Stores,” in *Proc. of WOSIS*, Porto, Portugal, April 2004.
- [LCLJML-13] J. Li, X. Chen, J. Li, C, Jia, J. Ma, W. Lou, “Fine-grained access control system based on outsourced attribute-based encryption,” in *Proc. of ESORICS 2013*, Egham, U.K., September 2013.
- [LDGW-13] J. Lai, R.H. Deng, C. Guan, J. Weng, “Attribute-Based Encryption With Verifiable Outsourced Decryption,” in *IEEE TIFS*, 8(8):1343-1354, August 2013.
- [LDL-12] J. Lai, R.H. Deng, Y. Li, “Expressive CP-ABE with Partially Hidden Access Structures,” in *Proc. of ASIACCS*, Seoul, Korea, May 2012.
- [LDLW-14] J. Lai, R.H. Deng, Y. Li, J. Weng, “Fully Secure Key-Policy Attribute-based Encryption with Constant-Size Ciphertexts and Fast Decryption,” in *Proc. of ASIACCS*, Kyoto, Japan, June 2014.
- [LDPW-14] J. Lai, R.H. Deng, H. Pang, J. Weng, “Verifiable Computation on Outsourced Encrypted Data,” in *Proc. of ESORICS*, Wroclaw, Poland, September 2014.

References – 9

- [LHKR-06] F. Li, M. Hadjieleftheriou, G. Kollios, L. Reyzin, “Dynamic Authenticated Index Structures for Outsourced Databases,” in *Proc. of SIGMOD*, Chicago, IL, USA, June 2006.
- [LWL-89] H. Liaw, S. Wang, C. Lei, “On the Design of a Single-Key-Lock Mechanism Based on Newton’s Interpolating Polynomial,” in *IEEE TSE*, vol. 15, no. 9, September 1989.
- [M-85] S. MacKinnon et al., “An Optimal Algorithm for Assigning Cryptographic Keys to Control Access in a Hierarchy,” in *IEEE TC,* vol. 34, no. 9, September 1985.
- [SD-16] P. Samarati, S. De Capitani di Vimercati, “Cloud Security: Issues and Concerns,” in *Encyclopedia on Cloud Computing,* S. Murugesan, I. Bojanova (eds.), Wiley, 2016.
- [S-87] R. Sandhu, “On Some Cryptographic Solutions for Access Control in a Tree Hierarchy,” in *Proc. of the 1987 Fall Joint Computer Conference on Exploring Technology: Today and Tomorrow*, Dallas, TX, USA, October 1987.
- [S-88] R. Sandhu, “Cryptographic Implementation of a Tree Hierarchy for Access Control,” in *Information Processing Letters*, vol. 27, no. 2, 1988.

References – 10

- [SC-02] V. Shen, T. Chen, “A Novel Key Management Scheme Based on Discrete Logarithms and Polynomial Interpolations,” in *Computers and Security*, vol. 21, no. 2, March 2002.
- [SC-07] R. Sion, B. Carbunar, “On the Computational Practicality of Private Information Retrieval,” in *Proc. of NDSS*, San Diego, CA, USA, February/March 2007.
- [SvSFRYD-13] E. Stefanov, M. van Dijk, E. Shi, C. Fletcher, L. Ren, X. Yu, S. Devadas, “Path ORAM: An Extremely Simple Oblivious RAM protocol,” in *Proc. of CCS*, Berlin, Germany, November 2013.
- [XWYM-07] M. Xie, H. Wang, J. Yin, X. Meng, “Integrity Auditing of Outsourced Data,” in *Proc. of VLDB*, Vienna, Austria, September 2007.
- [WL-06] H. Wang, Laks V. S. Lakshmanan, “Efficient Secure Query Evaluation over Encrypted XML Databases,” in *Proc. of VLDB*, Seoul, Korea, September 2006.
- [WSC-08] P. Williams, R. Sion, B. Carbunar, “Building Castles Out of Mud: Practical Access Pattern Privacy and Correctness on Untrusted Storage,” in *Proc of CCS*, Alexandria, USA, October 2008.
- [WS-12] P. Williams, R. Sion, “Single Round Access Privacy on Outsourced Storage,” in *Proc. of CCS*, Raleigh, NC, USA, October 2012.

References – 11

- [YPPK-09] Y. Yang, D. Papadias, S. Papadopoulos, P. Kalnis, "Authenticated Join Processing in Outsourced Databases," in *Proc. of SIGMOD*, Providence, RI, USA, June 2009.
- [YWRL-10] S. Yu, C. Wang, K. Ren, W. Lou, "Attribute Based Data Sharing with Attribute Revocation," in *Proc. of ASIACCS*, Beijing, China, April 2010.