Data Protection in the Cloud

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

ARO Workshop on Cloud Security

Fairfax, Virginia, USA - March 12, 2013

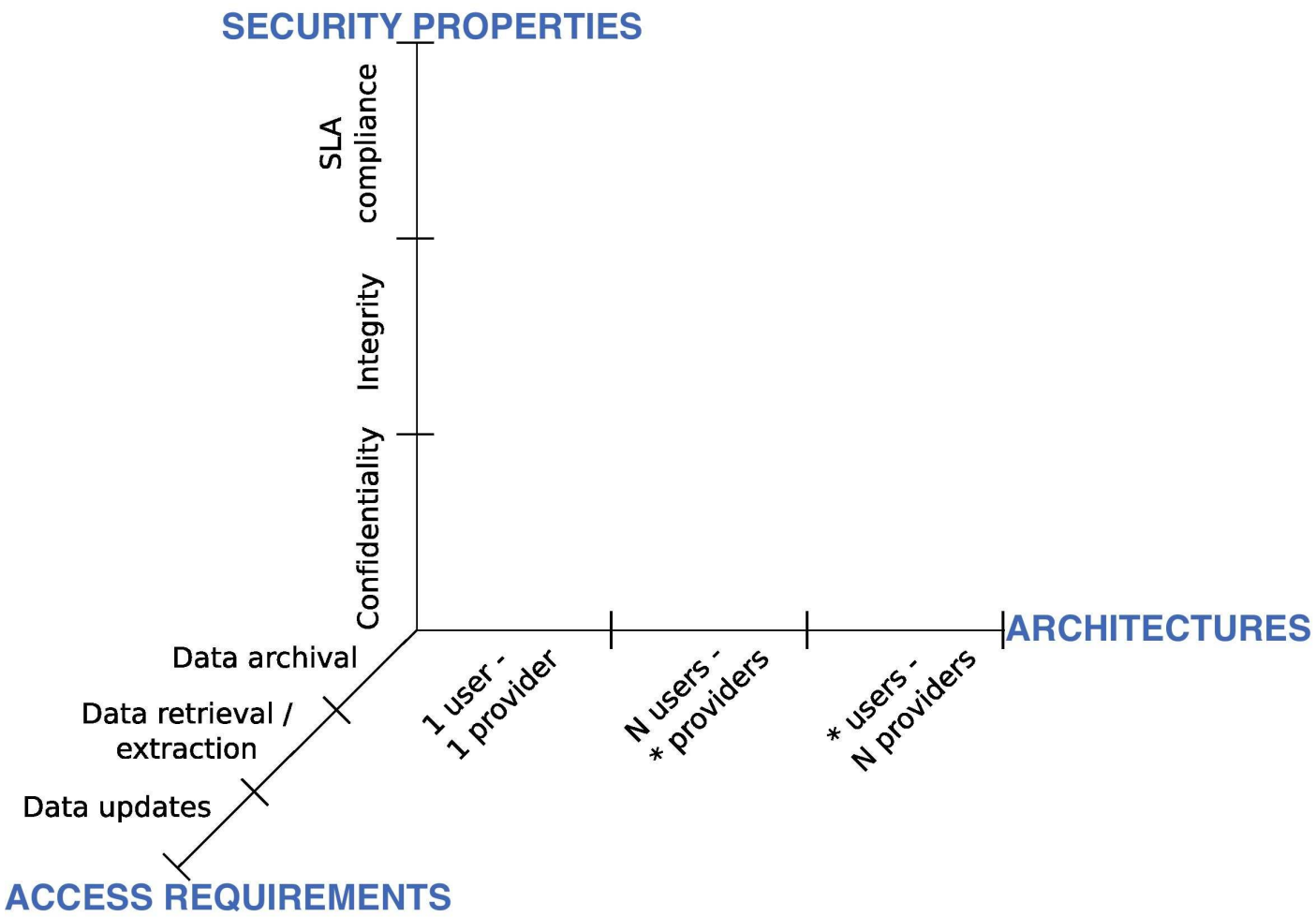
Based on joint work with: S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi

Cloud computing technology

- 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

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 also 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 can rely 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

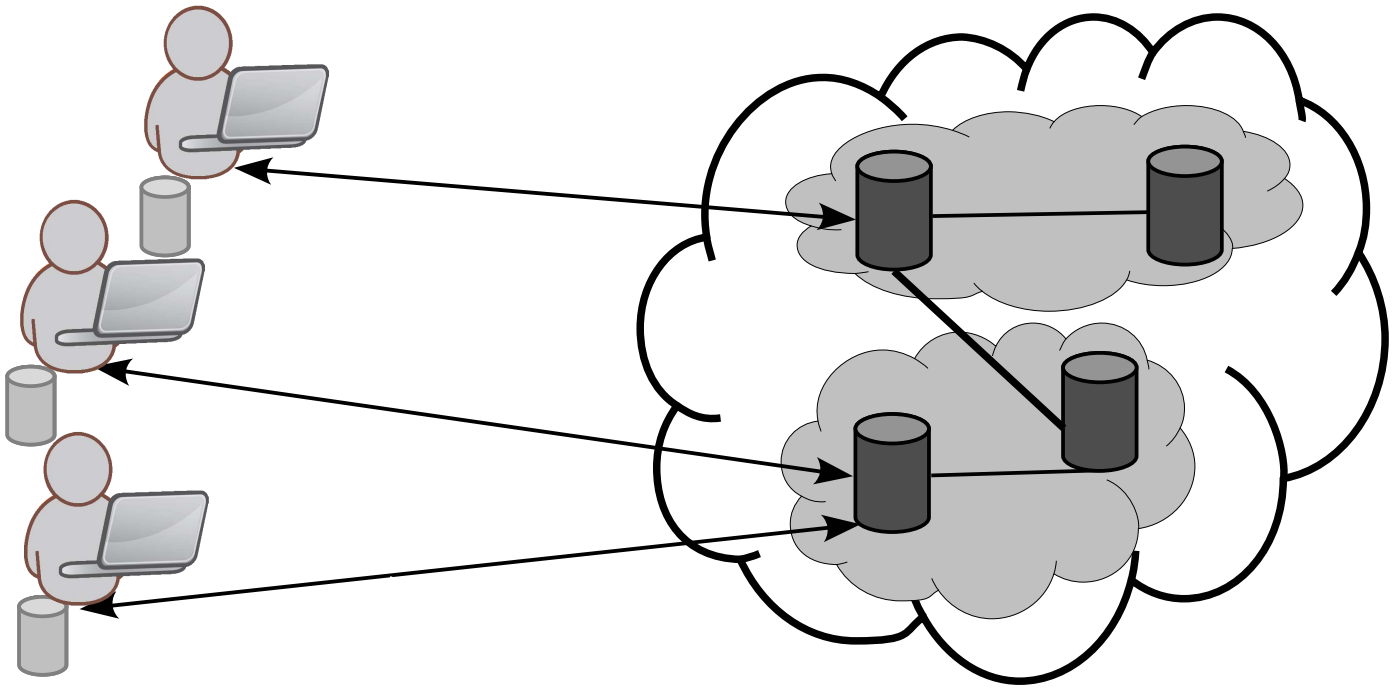
- Multiple providers: one or more users adopt multiple servers for data storage and access

  $\Longrightarrow$ controlled data sharing and computation

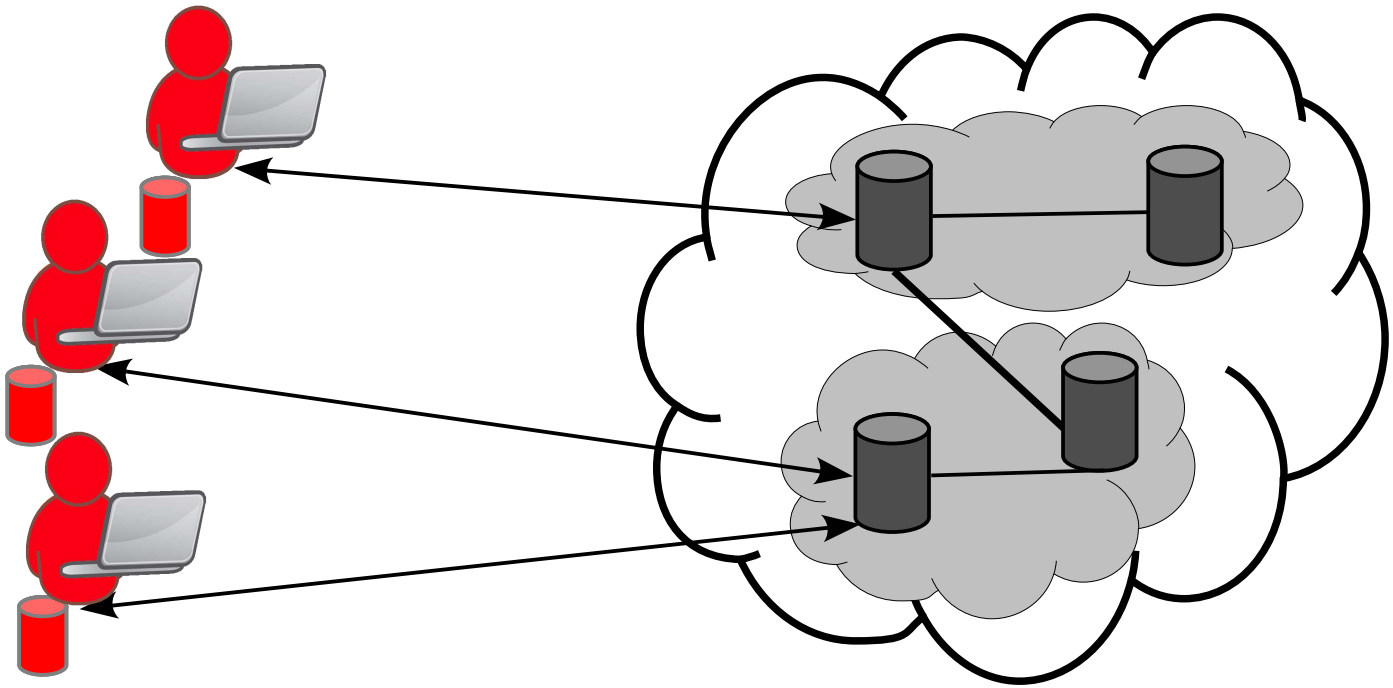
Combinations of the dimensions

- Every combination of the difference 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:
  - lazy
  - curious
  - malicious

Security and privacy problems

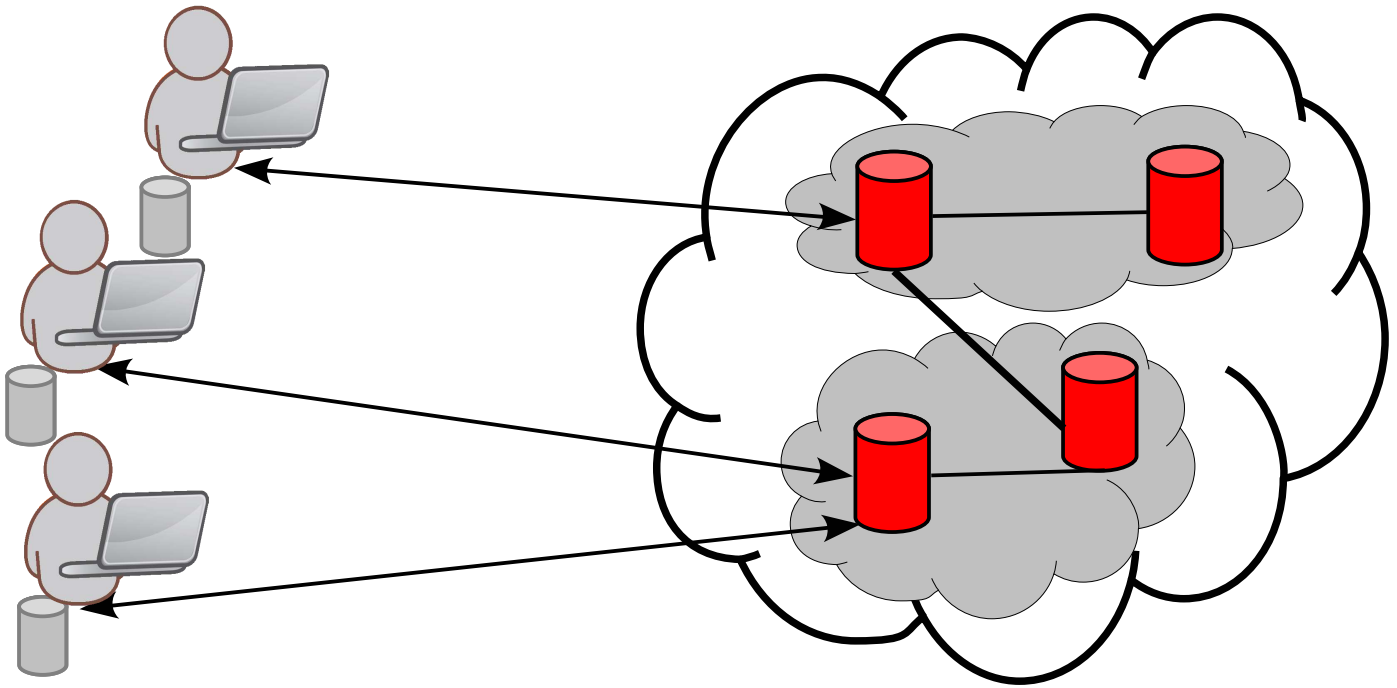


Security and privacy problems



Privacy of users

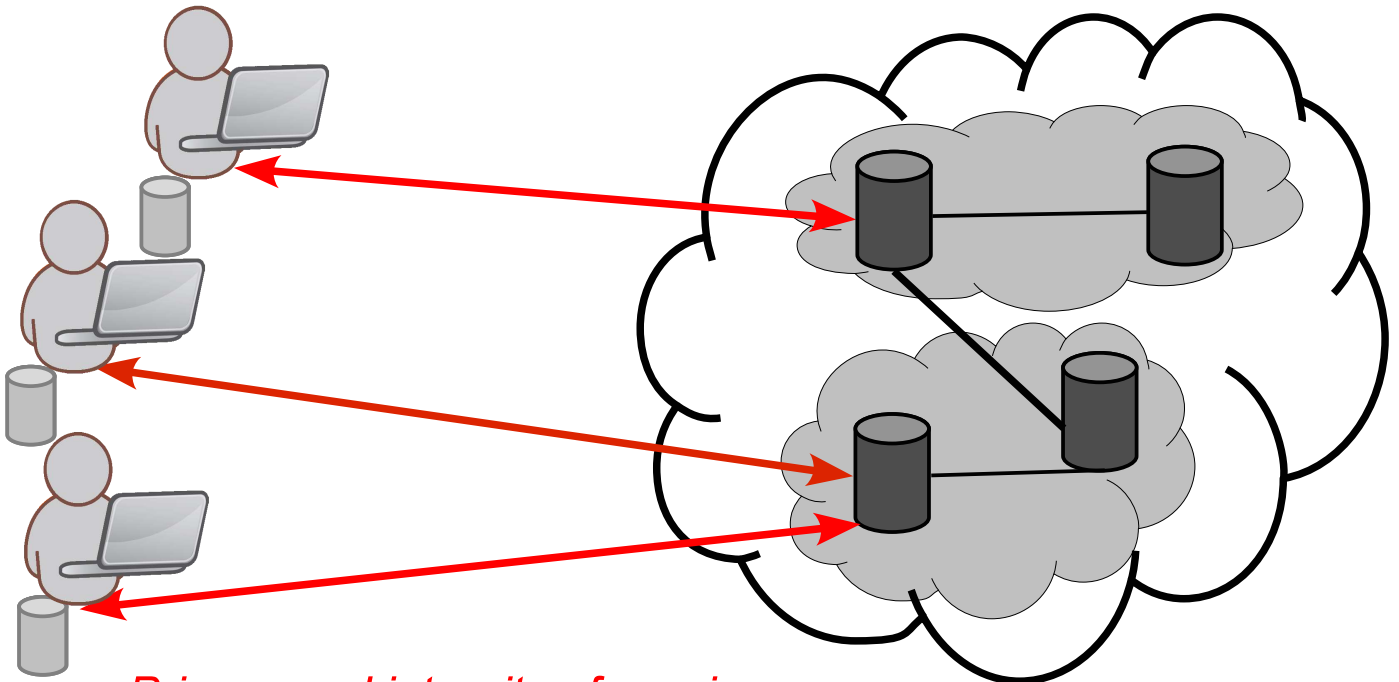
Security and privacy problems



Privacy of users

Privacy and integrity of data storage

Security and privacy problems

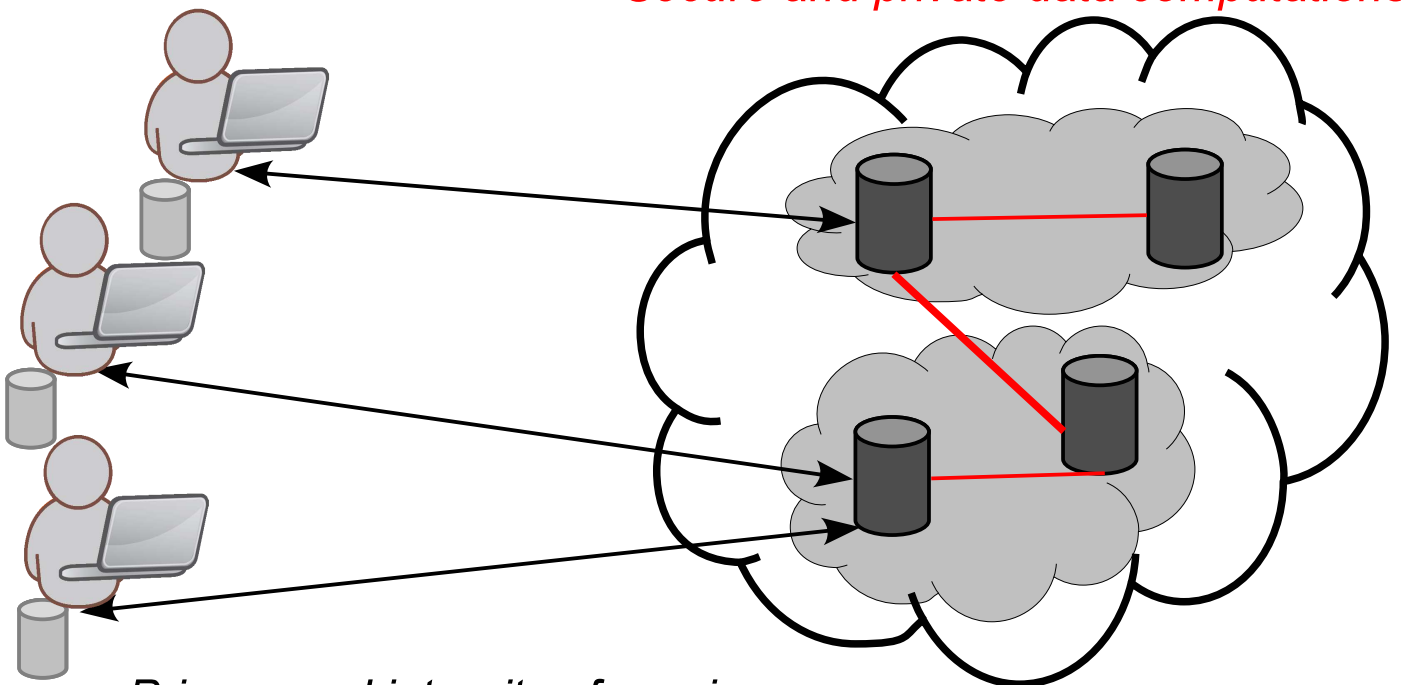


Privacy and integrity of queries and computations

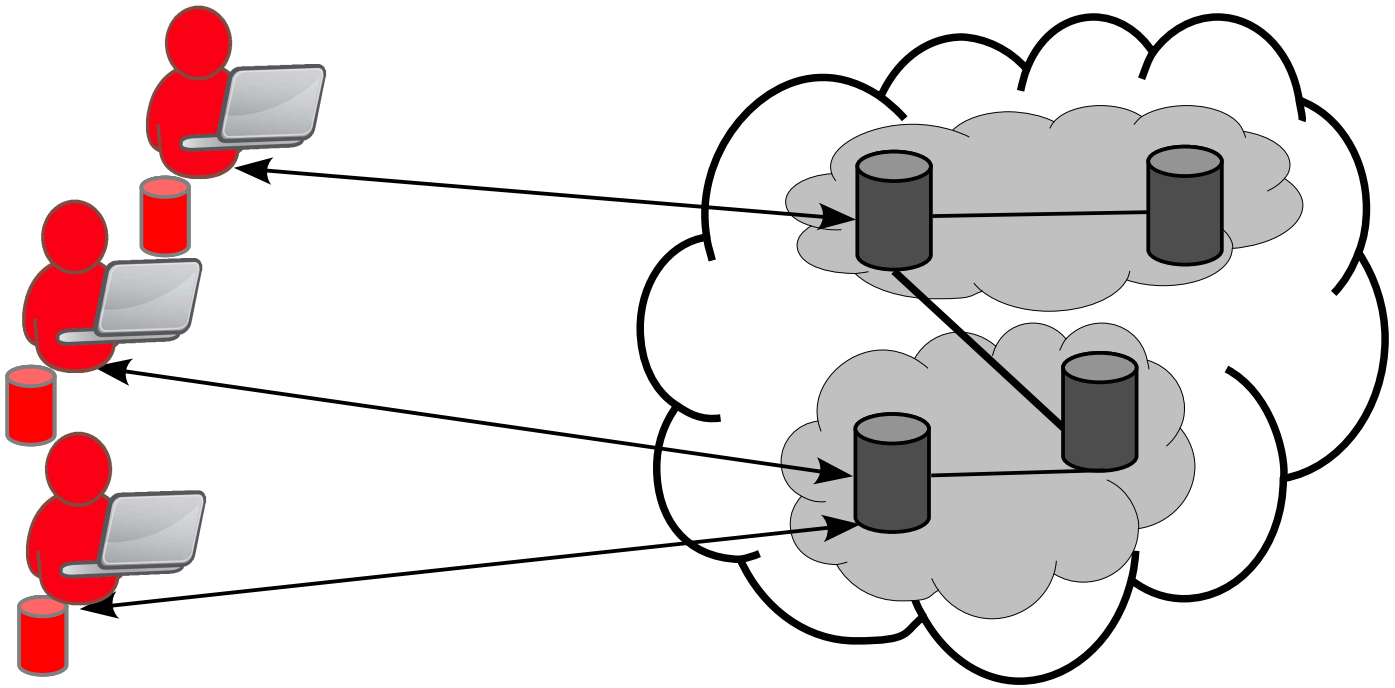
Privacy of users

Privacy and integrity of data storage

Security and privacy problems



Privacy of users



Privacy of users

Privacy of users' identities

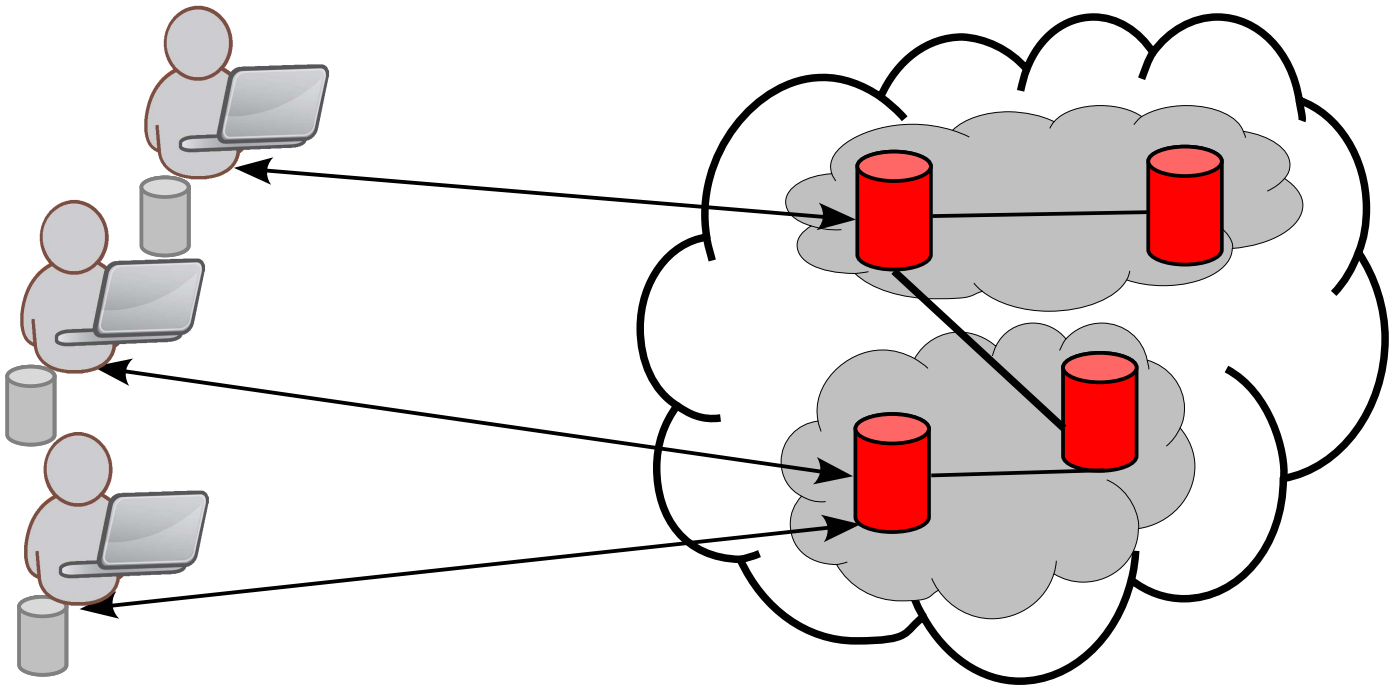
Users may wish to remain anonymous or to not disclose much information about themselves when operating in the cloud

- Anonymous communication techniques (e.g., Mix networks, onion routing, Tor, Crowds)

- Attribute-based access control (departing from user identities) [Bonatti, Samarati, JCS 2002]
  - instead of declaring their identities, users prove they satisfy properties needed for the access
  - changes the way access control process works

- Techniques for allowing users to effectively define privacy preferences on the release of their information [Chen et al., INFOCOM 2005; Yao et al., ACM TISSEC 2008; Kärger et al., SDM 2008; Ardagna et al., WPES 2010, PASSAT, 2010, IJIPSI 2012]

Privacy and integrity of data storage



Privacy of users

Privacy and integrity of data storage

Contributions and advancements

The research community has been very active and produced several contributions and advancements. E.g.,:

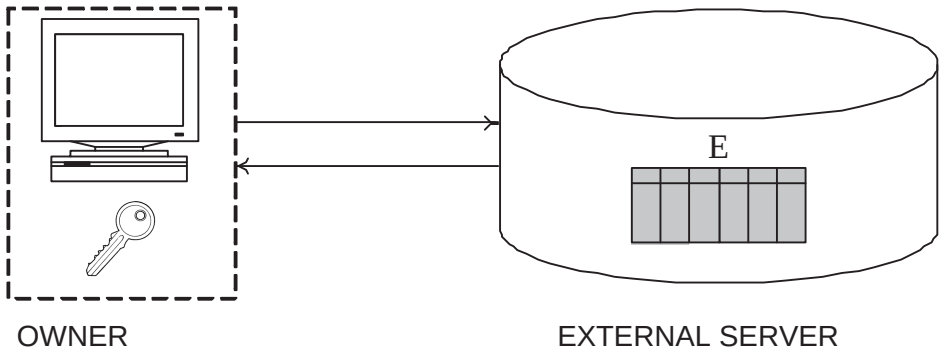
- Solutions for protecting data [Aggarwal et al., CIDR 2005; Hacigümüş et al., SIGMOD 2002; Ciriani et al., ESORICS 2009; Ciriani et al., ACM TISSEC 2010]
- Indexes supporting different types of queries [Ceselli et al., ACM TISSEC 2005; Hacigümüş et al., SIGMOD 2002; Wang et al., VLDB 2006]
- Selective access to outsourced data [De Capitani di Vimercati et al., ACM TODS 2010]
- Data integrity [Sion, VLDB 2005; Xie et al., VLDB 2007; Wang et al., CIKM 2008]
- Inference exposure evaluation [Ceselli et al., ACM TISSEC 2005]

Data protection

- Solutions for protecting data can be based on:
  - encryption
  - encryption and fragmentation
  - fragmentation

Encryption

- Data confidentiality is provided by wrapping a layer of encryption around sensitive data [Hacigümüş et al., SIGMOD 2002]
  - for performance reasons, encryption is typically applied at the tuple level



Encryption and indexes

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

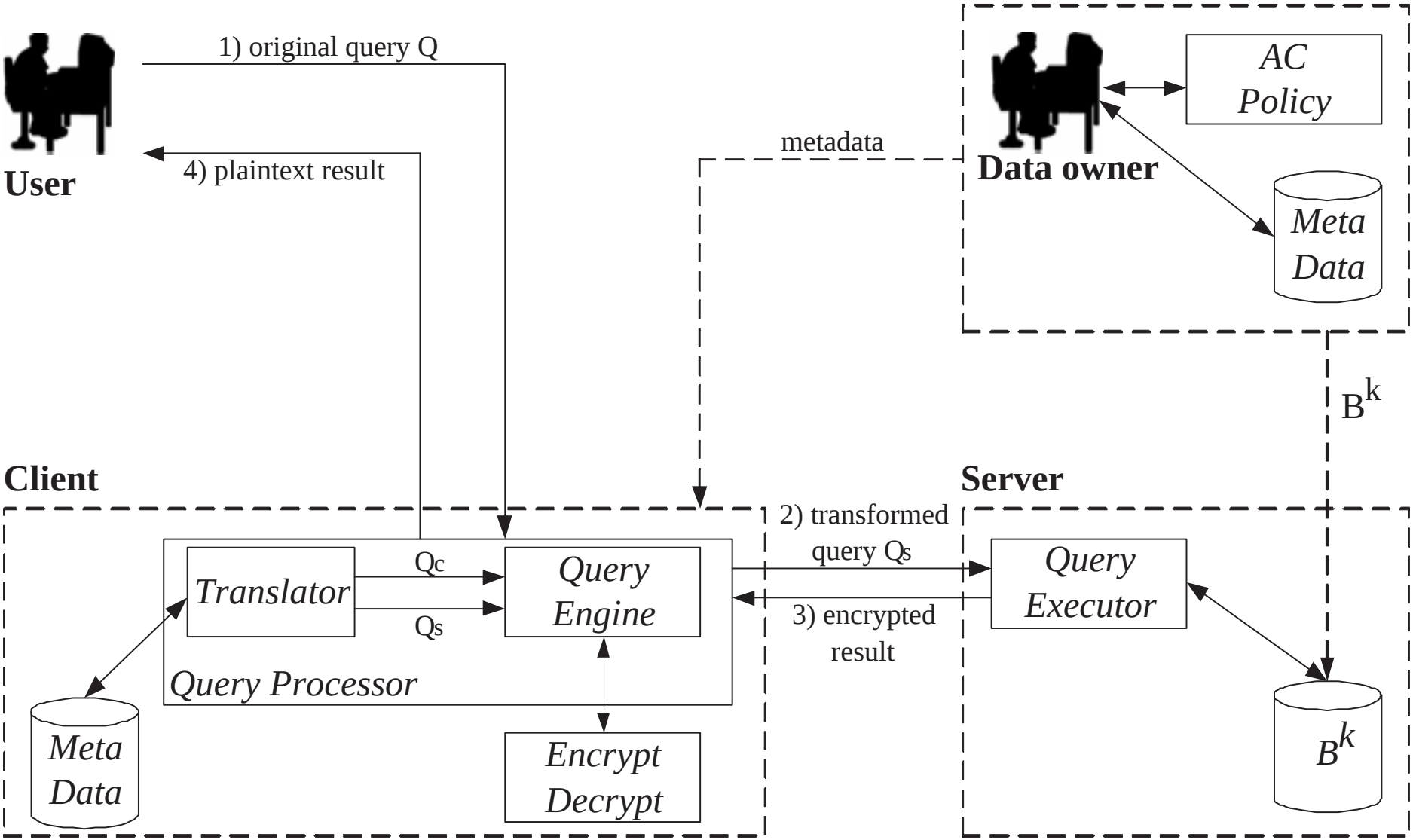
MedicalData

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

MedicalData^k

Counter	Etuple	I_S	I_N	I_D	I_Z	I_I	I_P
1	x4Z3tfX2ShOSM	π	α	μ	θ	δ	ω
2	mNHg1oC010p8w	ϖ	β	κ	θ	ι	Λ
3	WsIaCvfyF1Dxw	ξ	γ	η	ε	κ	ω
4	JpO8eLTVgwV1E	ρ	δ	κ	ε	ι	Λ

Query evaluation process



Indexes – 1

Different choices for indexes [Ceselli et al., ACM TISSEC 2005; Hacigümüş et al., SIGMOD 2002; Wang et al., VLDB 2006]

- 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)
- 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 (postprocessing query)
  - – still vulnerable to inference attacks

Indexes – 2

- 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

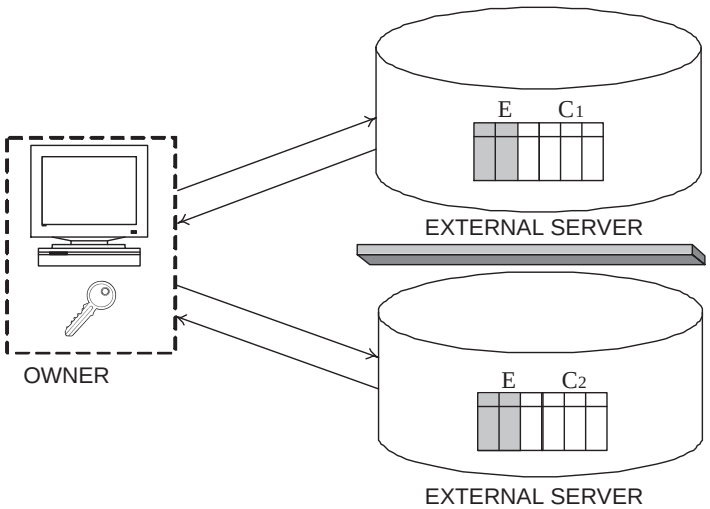
- Often what is sensitive is the association between values of different attributes, rather than the values themselves
  - e.g., association between employee's names and salaries

  $\Longrightarrow$protect associations by breaking them, rather than encrypting

- Recent solutions for enforcing privacy requirements couple:
  - encryption
  - data fragmentation

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) [Aggarwal et al., CIDR 2005]
  - Sensitive associations are protected by distributing the involved attributes among the two servers
  - Encryption is applied only when explicitly demanded by the confidentiality constraints or when storing an attribute in any of the server would expose at least a sensitive association



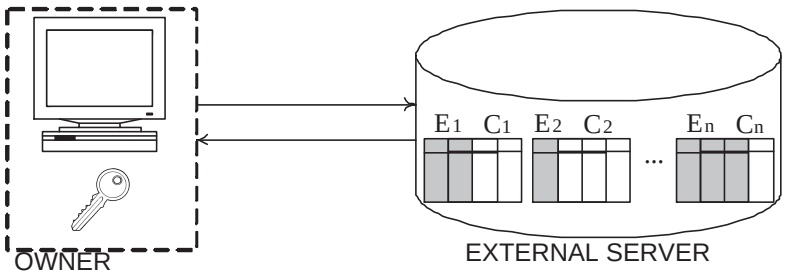
- $E \cup C_1 \cup C_2 = R$
- $C_1 \cup C_2 \subseteq R$

Multiple fragments

Coupling fragmentation and encryption is interesting and promising, but assumption of two non-communicating 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 [Ciriani et al., ACM TISSEC 2010]



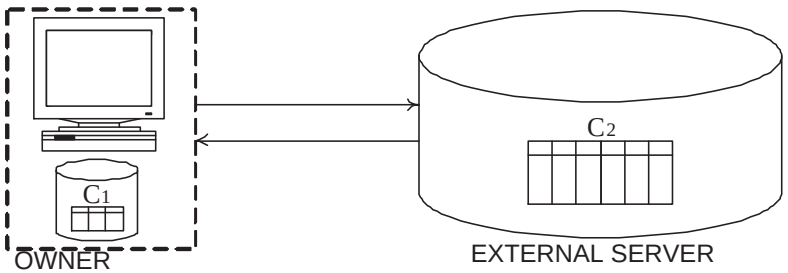
- $E_1 \cup C_1 = \ldots = E_n \cup C_n = R$
- $C_1 \cup \ldots \cup C_n \subseteq R$

Keep a few

Basic idea:

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

$\Longrightarrow$ Depart from encryption by involving the owner as a trusted party to maintain a limited amount of data [Ciriani et al., ESORICS 2009]



- $C_1 \cup C_2 = R$

Selective Encryption

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), April 2010.

Selective encryption – 1

- 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
- Existing approaches for data outsourcing can support the use of different keys for encrypting different data

  $\Longrightarrow$ selective encryption as a means to enforce selective access

  [De Capitani di Vimercati et al., ACM TODS 2010]

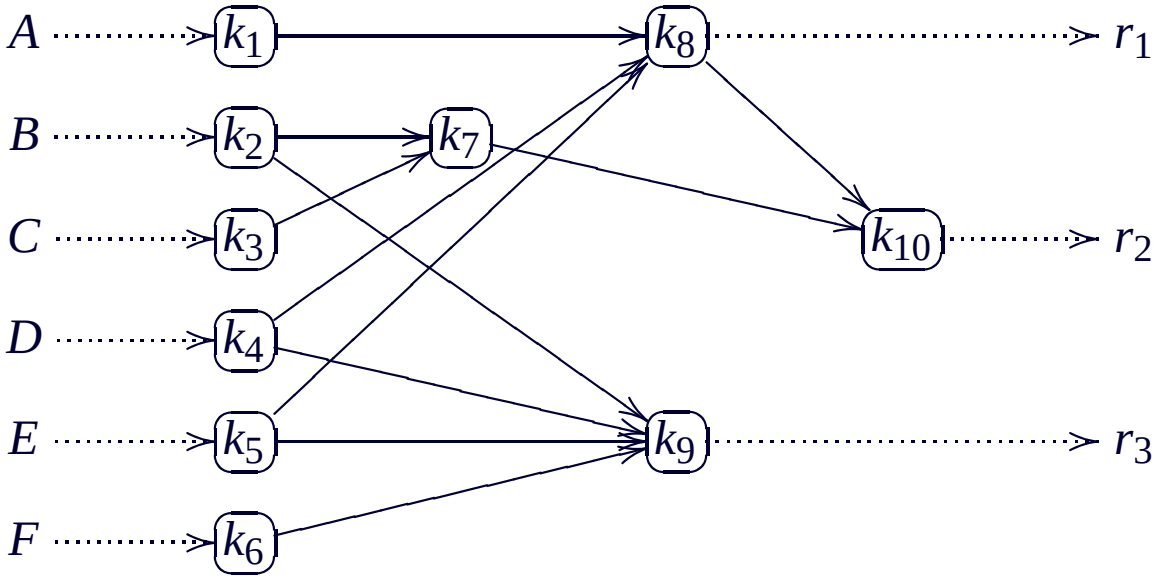
Selective encryption – 2

Basic idea:

- different ACLs implies different encryption keys
- key derivation method to limit number of keys
  - via public tokens a user can derive all keys of the resources she is allowed to access
- over-encryption to support policy updates

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 23-28, 2007.

Selective encryption – Example



- user A can access $\{r_1, r_2\}$
- user B can access $\{r_2, r_3\}$
- user C can access $\{r_2\}$
- user D can access $\{r_1, r_2, r_3\}$
- user E can access $\{r_1, r_2, r_3\}$
- user F can access $\{r_3\}$

key assignment ⋯⋯→

token ⟶

Exposure of confidential information

- Indexes, fragmentation, and selective encryption are all solutions providing the required security and privacy guarantees but...
- ...What happens when such solutions are combined?

Exposure of confidential information

- Indexes, fragmentation, and selective encryption are all solutions providing the required security and privacy guarantees but...
- ...What happens when such solutions are combined?

$\Longrightarrow$ They may open the door to inferences by users

Exposure of confidential information

- Indexes, fragmentation, and selective encryption are all solutions providing the required security and privacy guarantees but...
- ...What happens when such solutions are combined?

$\Longrightarrow$ They may open the door to inferences by users

  - Indexes and selective encryption
  - Indexes and fragmentation

Indexes and Selective Encryption

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 2011*, Chicago, Illinois, USA, October 17, 2011.

Indexes and selective encryption: User knowledge

Each user knows the:

- index functions ι used to define indexes in the encrypted relation
- plaintext tuples that she is authorized to access
- encrypted relation in its entirety

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1	001	NY	2010	600
t_2	002	Rome	2010	700
t_3	003	Rome	2011	600
t_4	004	NY	2011	700
t_5	005	Oslo	2011	700

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι(700)
3	γ	ι(Rome)	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Indexes and selective encryption: User knowledge

Each user knows the:

- index functions ι used to define indexes in the encrypted relation
- plaintext tuples that she is authorized to access
- encrypted relation in its entirety

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1				
t_2	002	Rome	2010	700
t_3	003	Rome	2011	600
t_4				
t_5				

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι(700)
3	γ	ι(Rome)	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1				
t_2	002	Rome	2010	700
t_3	003	Rome	2011	600
t_4				
t_5				

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι(700)
3	γ	ι(Rome)	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1				
t_2	002	Rome	**2010**	700
t_3	003	Rome	2011	600
t_4				
t_5				

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι**(2010)**	ι(700)
3	γ	ι(Rome)	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1			**2010**	
t_2	002	Rome	**2010**	700
t_3	003	Rome	2011	600
t_4				
t_5				

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι**(2010)**	ι(600)
2	β	ι(Rome)	ι**(2010)**	ι(700)
3	γ	ι(Rome)	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
$\boldsymbol{t_3}$	$\boldsymbol{B}$
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1			2010	
t_2	002	Rome	2010	700
t_3	003	Rome	**2011**	600
t_4				
t_5				

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι(700)
3	γ	ι(Rome)	ι**(2011)**	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1			2010	
t_2	002	Rome	2010	700
t_3	003	Rome	**2011**	600
t_4			**2011**	
t_5			**2011**	

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι(700)
3	γ	ι(Rome)	ι**(2011)**	ι(600)
4	δ	ι(NY)	ι**(2011)**	ι(700)
5	ε	ι(Oslo)	ι**(2011)**	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1			2010	
t_2	002	Rome	2010	**700**
t_3	003	Rome	2011	600
t_4			2011	
t_5			2011	

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι**(700)**
3	γ	ι(Rome)	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1			2010	
t_2	002	Rome	2010	**700**
t_3	003	Rome	2011	600
t_4			2011	**700**
t_5			2011	**700**

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι**(700)**
3	γ	ι(Rome)	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι**(700)**
5	ε	ι(Oslo)	ι(2011)	ι**(700)**

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
$\boldsymbol{t_3}$	$\boldsymbol{B}$
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1			2010	
t_2	002	Rome	2010	700
t_3	003	Rome	2011	**600**
t_4			2011	700
t_5			2011	700

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι(700)
3	γ	ι(Rome)	ι(2011)	ι**(600)**
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1			2010	**600**
t_2	002	Rome	2010	700
t_3	003	Rome	2011	**600**
t_4			2011	700
t_5			2011	700

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι**(600)**
2	β	ι(Rome)	ι(2010)	ι(700)
3	γ	ι(Rome)	ι(2011)	ι**(600)**
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1			2010	600
t_2	002	**Rome**	2010	700
t_3	003	**Rome**	2011	600
t_4			2011	700
t_5			2011	700

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι**(Rome)**	ι(2010)	ι(700)
3	γ	ι**(Rome)**	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Direct index – 1

- Plaintext values are always represented by the same index value and viceversa

  $\Longrightarrow$ cells having the same plaintext values are exposed

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1		~~Rome~~	2010	600
t_2	002	**Rome**	2010	700
t_3	003	**Rome**	2011	600
t_4		~~Rome~~	2011	700
t_5		~~Rome~~	2011	700

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι**(NY)**	ι(2010)	ι(600)
2	β	ι**(Rome)**	ι(2010)	ι(700)
3	γ	ι**(Rome)**	ι(2011)	ι(600)
4	δ	ι**(NY)**	ι(2011)	ι(700)
5	ε	ι**(Oslo)**	ι(2011)	ι(700)

Exposure risk: Direct index – 2

- Each user knows index function ι

  $\Longrightarrow$ all index-plaintext value correspondences are exposed to brute-force attacks

  $\Longrightarrow$ the whole outsourced relation is exposed to brute-force attacks

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1		NY	2010	600
t_2	002	Rome	2010	700
t_3	003	Rome	2011	600
t_4		NY	2011	700
t_5		Oslo	2011	700

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι(NY)	ι(2010)	ι(600)
2	β	ι(Rome)	ι(2010)	ι(700)
3	γ	ι(Rome)	ι(2011)	ι(600)
4	δ	ι(NY)	ι(2011)	ι(700)
5	ε	ι(Oslo)	ι(2011)	ι(700)

Exposure risk: Flattened and bucket/hash-based index

- Flattened index: an index value always represents the same plaintext value and users know the index function

  $\Longrightarrow$ cells having the same plaintext values are exposed

  $\Longrightarrow$ all index-plaintext value correspondences are exposed to brute-force attacks

  $\Longrightarrow$ the whole outsourced relation is exposed to brute-force attacks

- Bucket/hash-based index: the same index value may represent different plaintext values

  $\Longrightarrow$ users can only infer with certainty that certain values do not correspond to given cells

Indexes guided by access control restrictions

Intuitive:

- Indexes based on the ACLs (complicate query execution)

Alternative:

- Use different indexes for different users who can access the tuple
  - one index for every user
  - same value, overlapping ACLs $\Longrightarrow$ different index values
  - salts are used for providing such diversity

Indexes guided by access control restrictions – Example

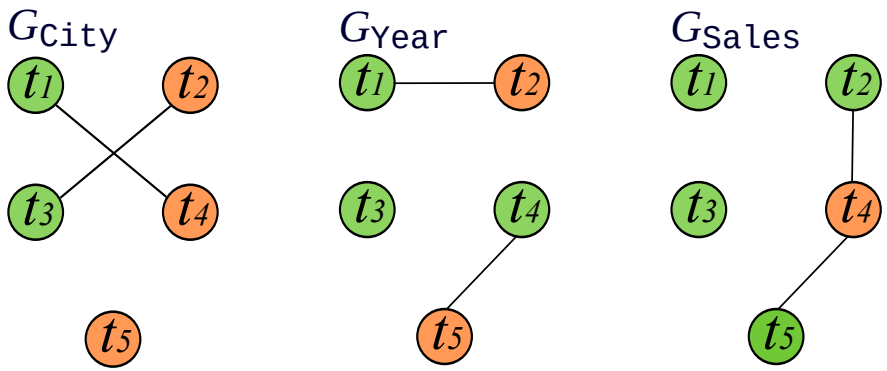
Index function ι_u for user u over attribute A is defined applying randomly generated salts to tuples

- same value, overlapping ACLs $\Longrightarrow$ different salts

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

SHOPS

	Id	City	Year	Sales
t_1	001	NY	2010	600
t_2	002	Rome	2010	700
t_3	003	Rome	2011	600
t_4	004	NY	2011	700
t_5	005	Oslo	2011	700



SHOPS^e

tid	etuple	I_c	I_y	I_s
1	α	ι_A(NY,s_A)	ι_A(2010,s_A)	ι_A(600,s_A)
2	β	ι_A(Rome,s'_A)ι_B(Rome,s_B)	ι_A(2010,s'_A)ι_B(2010,s_B)	ι_A(700,s_A)ι_B(700,s_B)
3	γ	ι_B(Rome,s'_B)	ι_B(2011,s'_B)	ι_B(600,s_B)
4	δ	ι_A(NY,s'_A)ι_C(NY,s_C)	ι_A(2011,s_A)ι_C(2011,s_C)	ι_A(700,s'_A)ι_C(700,s_C)
5	ε	ι_C(Oslo,s_C)	ι_C(2011,s'_C)	ι_C(700,s'_C)

Open issues...

- Protection against the server observing multiple queries
- Protection against collusion between users and server

Open issues...

- Protection against the server observing multiple queries
- Protection against collusion between users and server

SHOPS

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

	Id	City	Year	Sales
t_1	001	NY	2010	600
t_2	002	Rome	2010	700
t_3	003	Rome	2011	600
t_4	004	NY	2011	700
t_5	005	Oslo	2011	700

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	$\iota_A(\text{NY},s_A)$	$\iota_A(2010,s_A)$	$\iota_A(600,s_A)$
2	β	$\iota_A(\text{Rome},s'_A)\iota_B(\text{Rome},s_B)$	$\iota_A(2010,s'_A)\iota_B(2010,s_B)$	$\iota_A(700,s_A)\iota_B(700,s_B)$
3	γ	$\iota_B(\text{Rome},s'_B)$	$\iota_B(2011,s'_B)$	$\iota_B(600,s_B)$
4	δ	$\iota_A(\text{NY},s'_A)\iota_C(\text{NY},s_C)$	$\iota_A(2011,s_A)\iota_C(2011,s_C)$	$\iota_A(700,s'_A)\iota_C(700,s_C)$
5	ε	$\iota_C(\text{Oslo},s_C)$	$\iota_C(2011,s'_C)$	$\iota_C(700,s'_C)$

Open issues...

- Protection against the server observing multiple queries
- Protection against collusion between users and server

SHOPS

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

	Id	City	Year	Sales
t_1				
t_2	002	Rome	2010	700
t_3	003	Rome	2011	600
t_4				
t_5				

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	$\iota_A(\text{NY},s_A)$	$\iota_A(2010,s_A)$	$\iota_A(600,s_A)$
2	β	$\iota_A(\text{Rome},s'_A)\iota_B(\text{Rome},s_B)$	$\iota_A(2010,s'_A)\iota_B(2010,s_B)$	$\iota_A(700,s_A)\iota_B(700,s_B)$
3	γ	$\iota_B(\text{Rome},s'_B)$	$\iota_B(2011,s'_B)$	$\iota_B(600,s_B)$
4	δ	$\iota_A(\text{NY},s'_A)\iota_C(\text{NY},s_C)$	$\iota_A(2011,s_A)\iota_C(2011,s_C)$	$\iota_A(700,s'_A)\iota_C(700,s_C)$
5	ε	$\iota_C(\text{Oslo},s_C)$	$\iota_C(2011,s'_C)$	$\iota_C(700,s'_C)$

Query by B, who has 2 salts for `Year`

```
SELECT City, Sales
FROM   SHOPS
WHERE  Year=2010
```

$\Longrightarrow$

Open issues...

- Protection against the server observing multiple queries
- Protection against collusion between users and server

SHOPS

	acl
t_1	A
t_2	A,B
t_3	B
t_4	A,C
t_5	C

	Id	City	Year	Sales
t_1				
t_2	002	Rome	2010	700
t_3	003	Rome	2011	600
t_4				
t_5				

SHOPS^e

tid	etuple	$\mathtt{I}_c$	$\mathtt{I}_y$	$\mathtt{I}_s$
1	α	ι_A(NY,s_A)	ι_A(2010,s_A)	ι_A(600,s_A)
2	β	ι_A**(Rome,**s'_A**)**ι_B**(Rome,**s_B**)**	ι_A**(2010,**s'_A**)**ι_B**(2010,**s_B**)**	ι_A**(700,**s_A**)**ι_B**(700,**s_B**)**
3	γ	ι_B(Rome,s'_B)	ι_B(2011,s'_B)	ι_B(600,s_B)
4	δ	ι_A(NY,s'_A)ι_C(NY,s_C)	ι_A(2011,s_A)ι_C(2011,s_C)	ι_A(700,s'_A)ι_C(700,s_C)
5	ε	ι_C(Oslo,s_C)	ι_C(2011,s'_C)	ι_C(700,s'_C)

Query by B, who has 2 salts for `Year` translates to

```
SELECT City, Sales
FROM   SHOPS
WHERE  Year=2010
```

$\Longrightarrow$

```
SELECT etuple
FROM   SHOPS^e
WHERE  I_y IN {ι_B(2010,s_B), ι_B(2010,s'_B)}
```

Indexes and Fragmentation

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 2013*, Brisbane, Australia, April 8, 2013.

Information exposure

+ Provides effectiveness and efficiency in query execution
  - enables the partial server-side evaluation of selection conditions over encrypted attributes
- Indexes combined with fragmentation can cause information leakage of confidential (encrypted or fragmented) information
  - exposure to leakage varies depending on the kind of indexes

Fragments and indexes – Example

F_1^e

salt	enc	Name	State
s_{11}	t_{11}^e	Adams	VA
s_{12}	t_{12}^e	Brown	MN
s_{13}	t_{13}^e	Cooper	CA
s_{14}	t_{14}^e	Davis	VA
s_{15}	t_{15}^e	Eden	NY
s_{16}	t_{16}^e	Falk	CA
s_{17}	t_{17}^e	Green	NY
s_{18}	t_{18}^e	Hack	NY

F_2^e

salt	enc	Disease
s_{21}	t_{21}^e	Flu
s_{22}	t_{22}^e	Flu
s_{23}	t_{23}^e	Flu
s_{24}	t_{24}^e	Diabetes
s_{25}	t_{25}^e	Diabetes
s_{26}	t_{26}^e	Gastritis
s_{27}	t_{27}^e	Arthritis
s_{28}	t_{28}^e	Arthritis

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t_{11}^e	Adams	VA	α
s_{12}	t_{12}^e	Brown	MN	α
s_{13}	t_{13}^e	Cooper	CA	α
s_{14}	t_{14}^e	Davis	VA	β
s_{15}	t_{15}^e	Eden	NY	β
s_{16}	t_{16}^e	Falk	CA	γ
s_{17}	t_{17}^e	Green	NY	δ
s_{18}	t_{18}^e	Hack	NY	δ

F_2^e

salt	enc	Disease
s_{21}	t_{21}^e	Flu
s_{22}	t_{22}^e	Flu
s_{23}	t_{23}^e	Flu
s_{24}	t_{24}^e	Diabetes
s_{25}	t_{25}^e	Diabetes
s_{26}	t_{26}^e	Gastritis
s_{27}	t_{27}^e	Arthritis
s_{28}	t_{28}^e	Arthritis

Direct index

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t_{11}^e	Adams	VA	α
s_{12}	t_{12}^e	Brown	MN	α
s_{13}	t_{13}^e	Cooper	CA	α
s_{14}	t_{14}^e	Davis	VA	β
s_{15}	t_{15}^e	Eden	NY	β
s_{16}	t_{16}^e	Falk	CA	γ
s_{17}	t_{17}^e	Green	NY	δ
s_{18}	t_{18}^e	Hack	NY	δ

vertical knowledge

salt	enc	Disease
s_{21}	t_{21}^e	Flu
s_{22}	t_{22}^e	Flu
s_{23}	t_{23}^e	Flu
s_{24}	t_{24}^e	Diabetes
s_{25}	t_{25}^e	Diabetes
s_{26}	t_{26}^e	Gastritis
s_{27}	t_{27}^e	Arthritis
s_{28}	t_{28}^e	Arthritis

Direct index

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t^e_{11}	Adams	VA	α
s_{12}	t^e_{12}	Brown	MN	α
s_{13}	t^e_{13}	Cooper	CA	α
s_{14}	t^e_{14}	Davis	VA	β
s_{15}	t^e_{15}	Eden	NY	β
s_{16}	t^e_{16}	Falk	CA	γ
s_{17}	t^e_{17}	Green	NY	δ
s_{18}	t^e_{18}	Hack	NY	δ

vertical knowledge

salt	enc	Disease
s_{21}	t^e_{21}	Flu
s_{22}	t^e_{22}	Flu
s_{23}	t^e_{23}	Flu
s_{24}	t^e_{24}	Diabetes
s_{25}	t^e_{25}	Diabetes
s_{26}	t^e_{26}	Gastritis
s_{27}	t^e_{27}	Arthritis
s_{28}	t^e_{28}	Arthritis

Direct index

- ι(Flu) = α
- ι(Gastritis) = γ

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t^e_{11}	Adams	VA	α
s_{12}	t^e_{12}	Brown	MN	α
s_{13}	t^e_{13}	Cooper	CA	α
s_{14}	t^e_{14}	Davis	VA	β
s_{15}	t^e_{15}	Eden	NY	β
s_{16}	t^e_{16}	Falk	CA	γ
s_{17}	t^e_{17}	Green	NY	δ
s_{18}	t^e_{18}	Hack	NY	δ

vertical knowledge

salt	enc	Disease
s_{21}	t^e_{21}	Flu
s_{22}	t^e_{22}	Flu
s_{23}	t^e_{23}	Flu
s_{24}	t^e_{24}	Diabetes
s_{25}	t^e_{25}	Diabetes
s_{26}	t^e_{26}	Gastritis
s_{27}	t^e_{27}	Arthritis
s_{28}	t^e_{28}	Arthritis

Direct index

- ι(Flu) = $\alpha \Longrightarrow$ Adams, Brown, Cooper have Flu
- ι(Gastritis) = $\gamma \Longrightarrow$ Falk has Gastritis
- the other patients have Diabetes or Arthritis with $p = 50\%$

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t_{11}^e	Adams	VA	ζ
s_{12}	t_{12}^e	Brown	MN	ζ
s_{13}	t_{13}^e	Cooper	CA	ζ
s_{14}	t_{14}^e	Davis	VA	η
s_{15}	t_{15}^e	Eden	NY	η
s_{16}	t_{16}^e	Falk	CA	ζ
s_{17}	t_{17}^e	Green	NY	θ
s_{18}	t_{18}^e	Hack	NY	θ

vertical knowledge

salt	enc	Disease
s_{21}	t_{21}^e	Flu
s_{22}	t_{22}^e	Flu
s_{23}	t_{23}^e	Flu
s_{24}	t_{24}^e	Diabetes
s_{25}	t_{25}^e	Diabetes
s_{26}	t_{26}^e	Gastritis
s_{27}	t_{27}^e	Arthritis
s_{28}	t_{28}^e	Arthritis

Bucket index

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t^e_{11}	Adams	VA	ζ
s_{12}	t^e_{12}	Brown	MN	ζ
s_{13}	t^e_{13}	Cooper	CA	ζ
s_{14}	t^e_{14}	Davis	VA	η
s_{15}	t^e_{15}	Eden	NY	η
s_{16}	t^e_{16}	Falk	CA	ζ
s_{17}	t^e_{17}	Green	NY	θ
s_{18}	t^e_{18}	Hack	NY	θ

vertical knowledge

salt	enc	Disease
s_{21}	t^e_{21}	Flu
s_{22}	t^e_{22}	Flu
s_{23}	t^e_{23}	Flu
s_{24}	t^e_{24}	Diabetes
s_{25}	t^e_{25}	Diabetes
s_{26}	t^e_{26}	Gastritis
s_{27}	t^e_{27}	Arthritis
s_{28}	t^e_{28}	Arthritis

Bucket index

- ι(Flu) = ι(Gastritis) = ζ

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t_{11}^e	Adams	VA	ζ
s_{12}	t_{12}^e	Brown	MN	ζ
s_{13}	t_{13}^e	Cooper	CA	ζ
s_{14}	t_{14}^e	Davis	VA	η
s_{15}	t_{15}^e	Eden	NY	η
s_{16}	t_{16}^e	Falk	CA	ζ
s_{17}	t_{17}^e	Green	NY	θ
s_{18}	t_{18}^e	Hack	NY	θ

vertical knowledge

salt	enc	Disease
s_{21}	t_{21}^e	Flu
s_{22}	t_{22}^e	Flu
s_{23}	t_{23}^e	Flu
s_{24}	t_{24}^e	Diabetes
s_{25}	t_{25}^e	Diabetes
s_{26}	t_{26}^e	Gastritis
s_{27}	t_{27}^e	Arthritis
s_{28}	t_{28}^e	Arthritis

Bucket index

- ι(Flu) = ι(Gastritis) = $\zeta \Longrightarrow$ Adams, Brown, Cooper, and Falk have Flu with $p = 75\%$, Gastritis with $p = 25\%$

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t^e_{11}	Adams	VA	κ
s_{12}	t^e_{12}	Brown	MN	λ
s_{13}	t^e_{13}	Cooper	CA	μ
s_{14}	t^e_{14}	Davis	VA	ν
s_{15}	t^e_{15}	Eden	NY	ξ
s_{16}	t^e_{16}	Falk	CA	π
s_{17}	t^e_{17}	Green	NY	ρ
s_{18}	t^e_{18}	Hack	NY	σ

vertical knowledge

salt	enc	Disease
s_{21}	t^e_{21}	Flu
s_{22}	t^e_{22}	Flu
s_{23}	t^e_{23}	Flu
s_{24}	t^e_{24}	Diabetes
s_{25}	t^e_{25}	Diabetes
s_{26}	t^e_{26}	Gastritis
s_{27}	t^e_{27}	Arthritis
s_{28}	t^e_{28}	Arthritis

Flattened index

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t_{11}^e	Adams	VA	κ
s_{12}	t_{12}^e	Brown	MN	λ
s_{13}	t_{13}^e	Cooper	CA	μ
s_{14}	t_{14}^e	Davis	VA	ν
s_{15}	t_{15}^e	Eden	NY	ξ
s_{16}	t_{16}^e	Falk	CA	π
s_{17}	t_{17}^e	Green	NY	ρ
s_{18}	t_{18}^e	Hack	NY	σ

vertical knowledge

salt	enc	Disease
s_{21}	t_{21}^e	Flu
s_{22}	t_{22}^e	Flu
s_{23}	t_{23}^e	Flu
s_{24}	t_{24}^e	Diabetes
s_{25}	t_{25}^e	Diabetes
s_{26}	t_{26}^e	Gastritis
s_{27}	t_{27}^e	Arthritis
s_{28}	t_{28}^e	Arthritis

Flattened index

+ blocks inference exposure

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t^e_{11}	Adams	VA	κ
s_{12}	t^e_{12}	Brown	MN	λ
s_{13}	t^e_{13}	Cooper	CA	μ
s_{14}	t^e_{14}	Davis	VA	ν
s_{15}	t^e_{15}	Eden	NY	ξ
s_{16}	t^e_{16}	Falk	CA	π
s_{17}	t^e_{17}	Green	NY	ρ
s_{18}	t^e_{18}	Hack	NY	σ

vertical knowledge

salt	enc	Disease
s_{21}	t^e_{21}	Flu
s_{22}	t^e_{22}	Flu
s_{23}	t^e_{23}	Flu
s_{24}	t^e_{24}	Diabetes
s_{25}	t^e_{25}	Diabetes
s_{26}	t^e_{26}	Gastritis
s_{27}	t^e_{27}	Arthritis
s_{28}	t^e_{28}	Arthritis

Flattened index

+ blocks inference exposure
– exposed to inferences exploiting dynamic observations

EXAMPLE

Disease = ‘Flu’ translates to i_d IN $\{\kappa,\lambda,\mu\} \Longrightarrow \iota(\text{Flu}) = \{\kappa,\lambda,\mu\}$

Fragments and indexes – Example

F_1^e

salt	enc	Name	State	i_d
s_{11}	t^e_{11}	Adams	VA	κ
s_{12}	t^e_{12}	Brown	MN	λ
s_{13}	t^e_{13}	Cooper	CA	μ
s_{14}	t^e_{14}	Davis	VA	ν
s_{15}	t^e_{15}	Eden	NY	ξ
s_{16}	t^e_{16}	Falk	CA	π
s_{17}	t^e_{17}	Green	NY	ρ
s_{18}	t^e_{18}	Hack	NY	σ

vertical knowledge

salt	enc	Disease
s_{21}	t^e_{21}	Flu
s_{22}	t^e_{22}	Flu
s_{23}	t^e_{23}	Flu
s_{24}	t^e_{24}	Diabetes
s_{25}	t^e_{25}	Diabetes
s_{26}	t^e_{26}	Gastritis
s_{27}	t^e_{27}	Arthritis
s_{28}	t^e_{28}	Arthritis

Flattened index

+ blocks inference exposure
– exposed to inferences exploiting dynamic observations

EXAMPLE

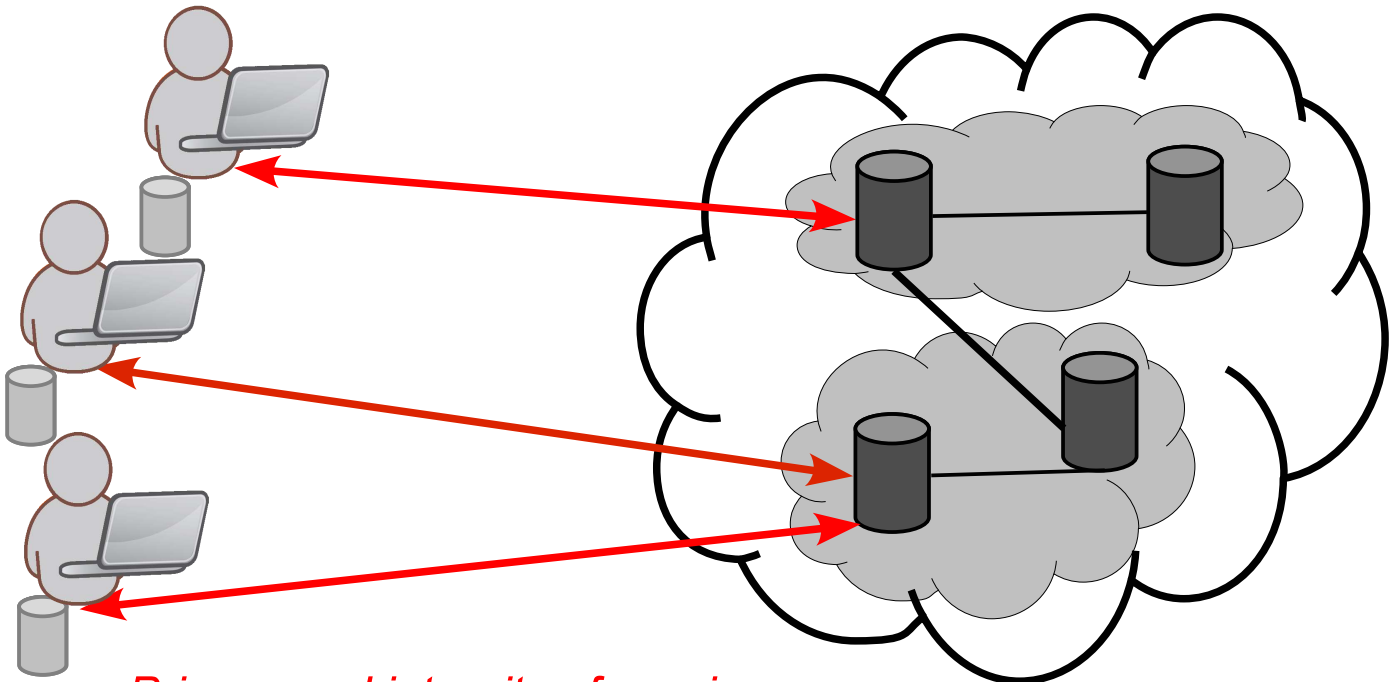
Disease = 'Flu' translates to i_d IN $\{\kappa,\lambda,\mu\} \Longrightarrow \iota(\text{Flu}) = \{\kappa,\lambda,\mu\}$

$\iota(\text{Flu}) = \{\kappa,\lambda,\mu\} \Longrightarrow$ Adams, Brown, and Cooper have Flu

Open issues...

- Protection against observation of accesses to fragments
- Protection against the release of multiple indexes
  - multiple indexes in the same fragment
  - indexes on the same attribute in multiple fragments
  - two attributes appear one in plaintext and the other indexed in one fragment and reversed in another fragment
- Protection against observer's external knowledge
- Definition of metrics for assessing exposures due to indexes

Privacy and integrity of queries and computations



Privacy and integrity of queries and computations

Privacy of users

Privacy and integrity of data storage

Access and pattern confidentiality

Guaranteeing privacy of outsourced data entails protecting the confidentiality of the data (content confidentiality) as well as the accesses to them

- Access confidentiality: confidentiality of the fact that an access aims at a specific data

- Pattern confidentiality: confidentiality of the fact that two accesses aim at the same data

Approaches for protecting data accesses

- Private Information Retrieval (PIR) proposals (e.g., [Chor et al., JACM 1998; Sion et al., NDSS 2007])
- Oblivious traversal of tree-structured data/indexes [Lin et al., WOSIS 2004]
- Pyramid-shaped database layout of Oblivious RAM [Williams et al., CCS 2008; Williams et al., CCS 2012]
- Shuffle index based on the definition of a B+-tree structure with dynamic allocation of data ([De Capitani di Vimercati et al., ICDCS 2011])

Shuffle Index

S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, P. Samarati, “Efficient and Private Access to Outsourced Data,” in *Proc. of ICDCS 2011,* Minneapolis, Minnesota, USA, June 20-24, 2011.

Shuffle index: Rationale of the approach

- Destroy the correspondence between the frequencies with which blocks are accessed and the frequencies of accesses to different values

- Combine three strategies:
  - cover searches
    - provide confusion in individual accesses (the target of an access is hid within a group of other requests)
  - cached searches
    - allow protection of accesses to the same values (local cache of nodes in the path to the target for counteracting intersection attacks)
  - shuffling
    - dynamically changes node allocation to blocks at every access, so destroying the fixed node-block correspondence

Access execution – Example

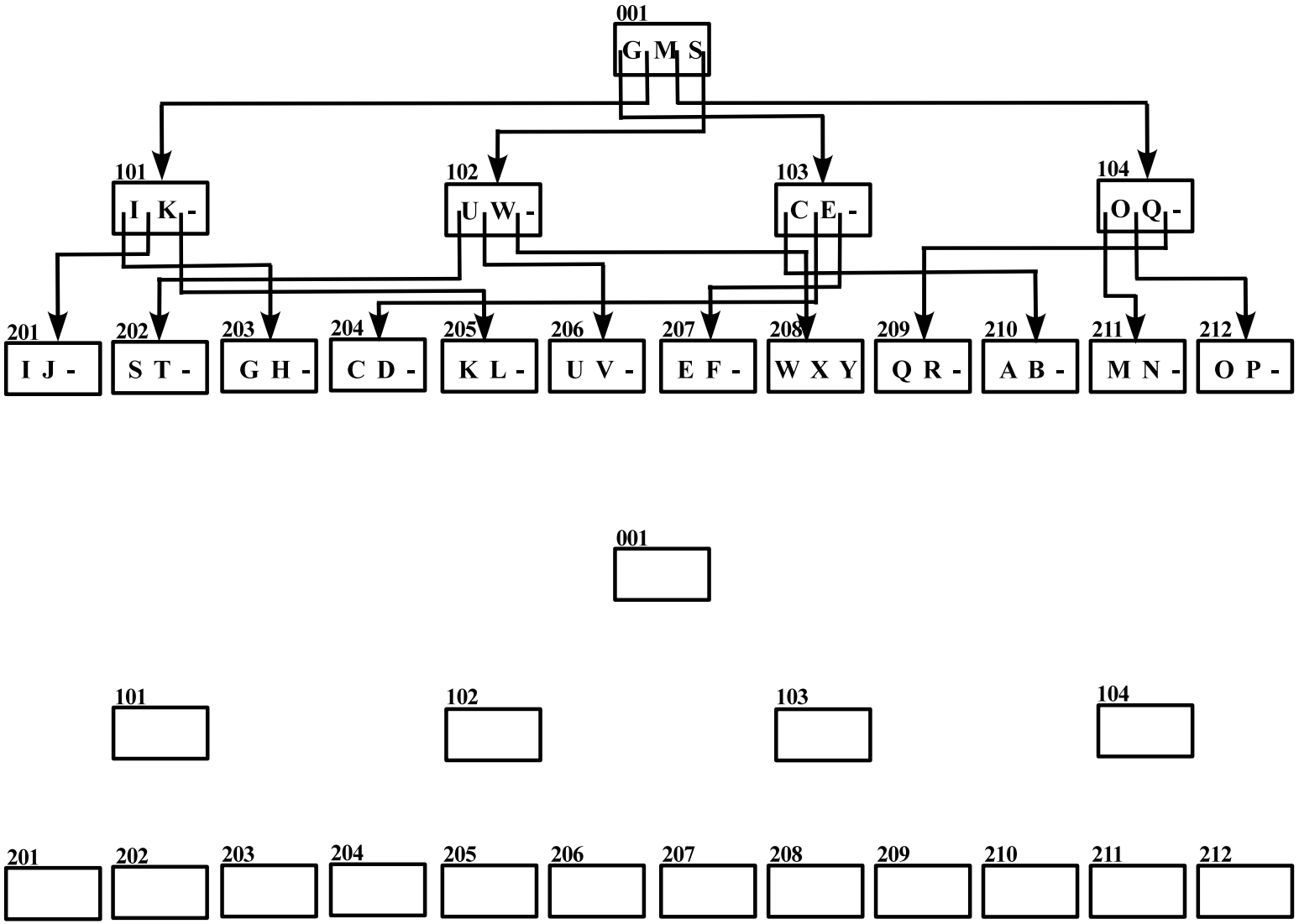
l	$Cache_l$
0	$001\,[_{103}G_{101}M_{104}S_{102}]$
1	$101\,[_{203}I_{201}K_{205}- -]$
	$103\,[_{210}C_{204}E_{207}- -]$
2	$203\,[GH-]$
	$210\,[AB-]$

num_cover=1

num_cache=2

target= F

covers= S,M



Access execution – Example

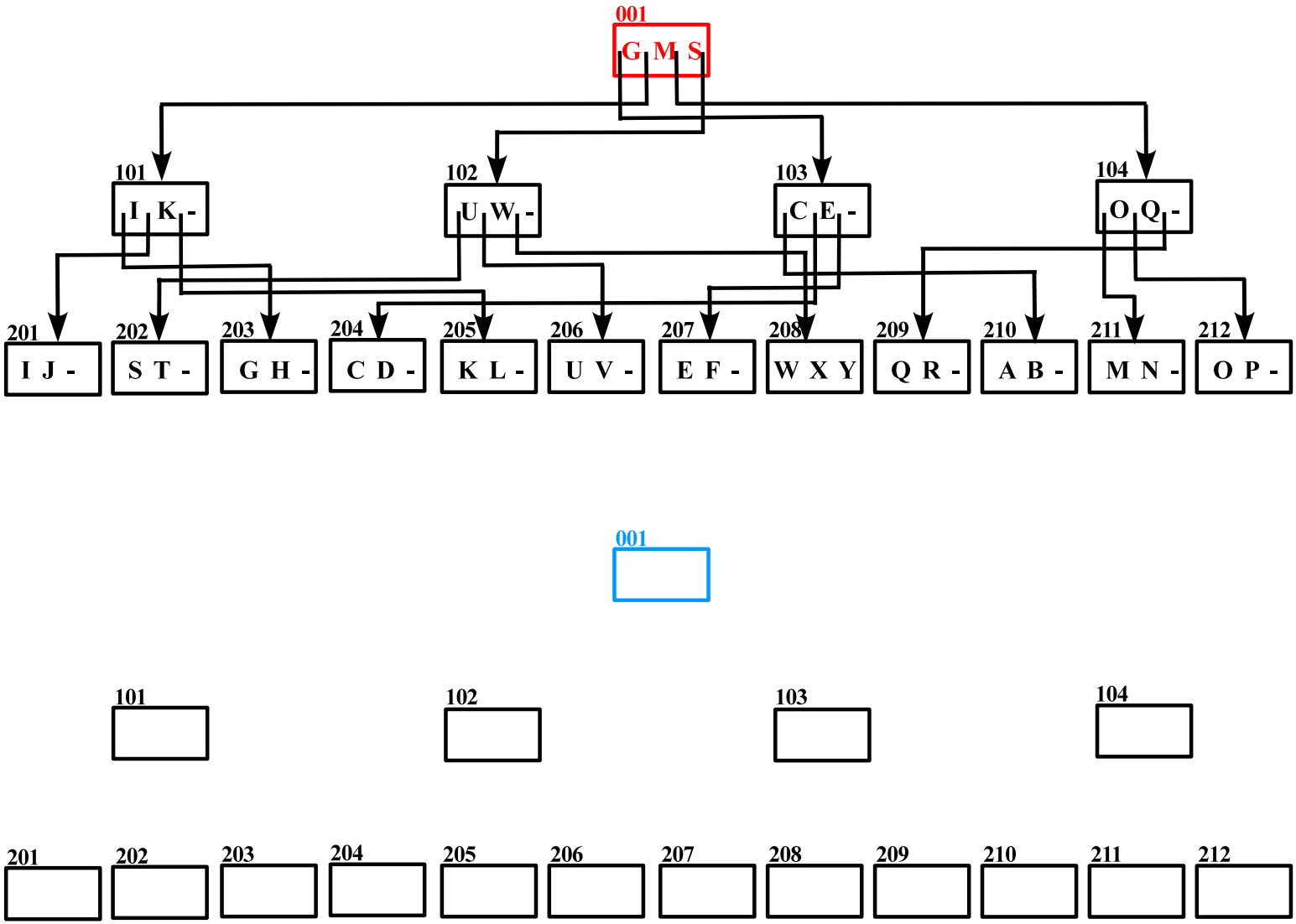
l	$Cache_l$
0	001 [$_{103}$G$_{101}$M$_{104}$S$_{102}$]
1	101 [$_{203}$I$_{201}$K$_{205}$- -]
	103 [$_{210}$C$_{204}$E$_{207}$- -]
2	203 [GH-]
	210 [AB-]

num_cover=1

num_cache=2

target= F

covers= S,M



Access execution – Example

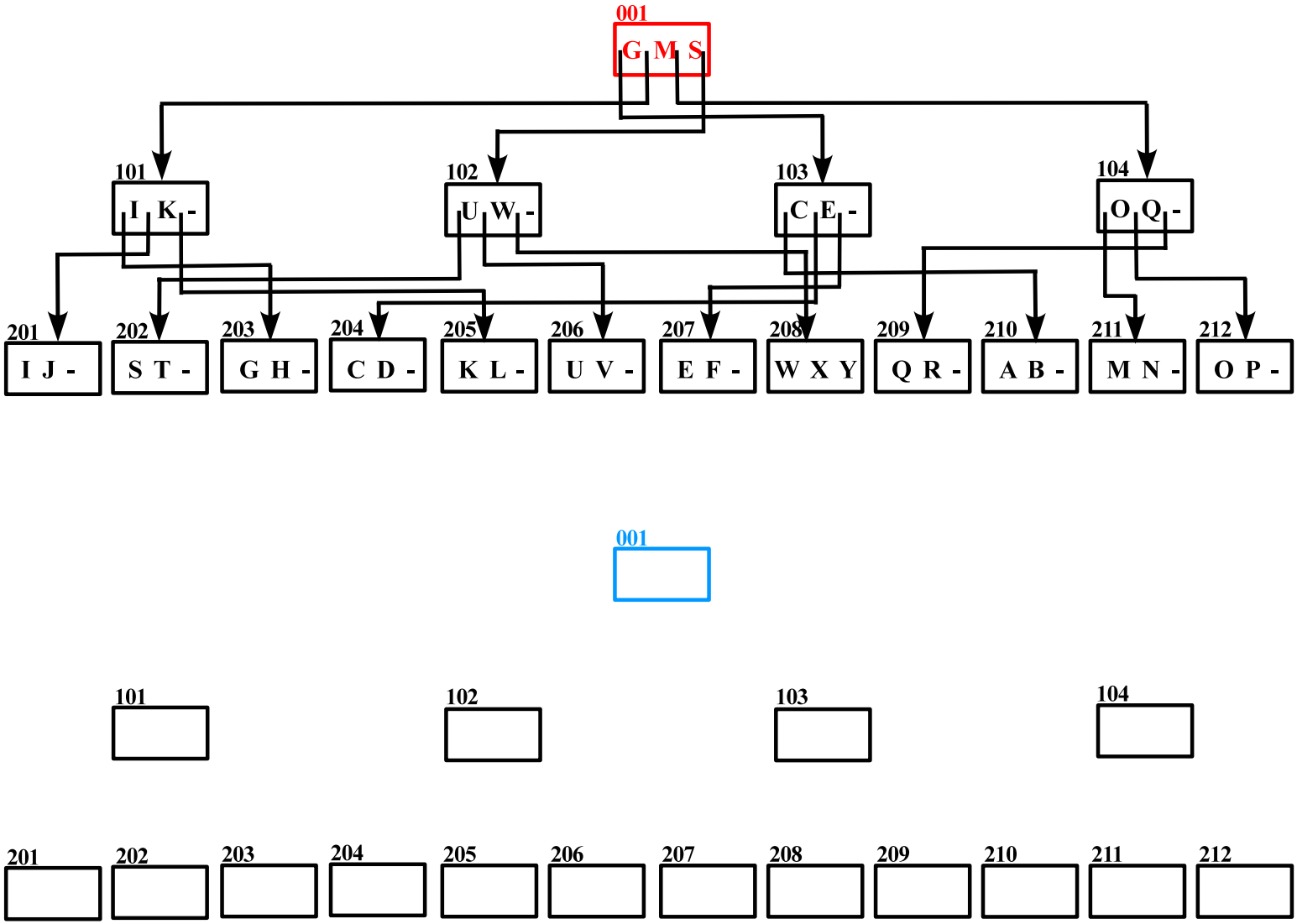
l	$Cache_l$
0	001 [$_{103}$G$_{101}$M$_{104}$S$_{102}$]
1	101 [$_{203}$I$_{201}$K$_{205}$- -]
	103 [$_{210}$C$_{204}$E$_{207}$- -]
2	203 [GH-]
	210 [AB-]

num_cover=1

num_cache=2

target= F

covers= S,M



Access execution – Example

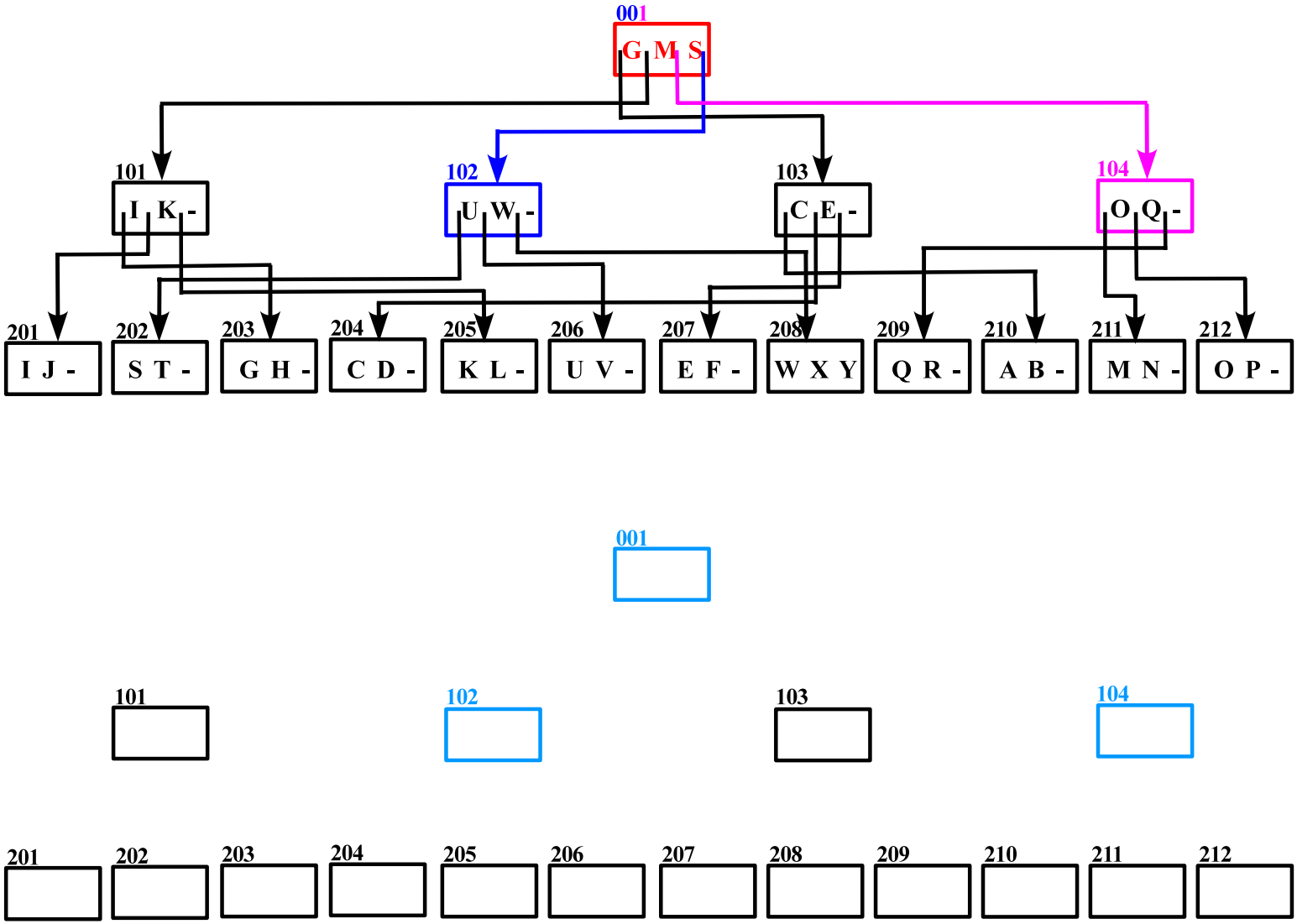
l	$Cache_l$
0	001 [$_{103}$G$_{101}$M$_{104}$S$_{102}$]
1	101 [$_{203}$I$_{201}$K$_{205}$- -]
	103 [$_{210}$C$_{204}$E$_{207}$- -]
2	203 [GH-]
	210 [AB-]

num_cover=1

num_cache=2

target= F

covers= S,M



Access execution – Example

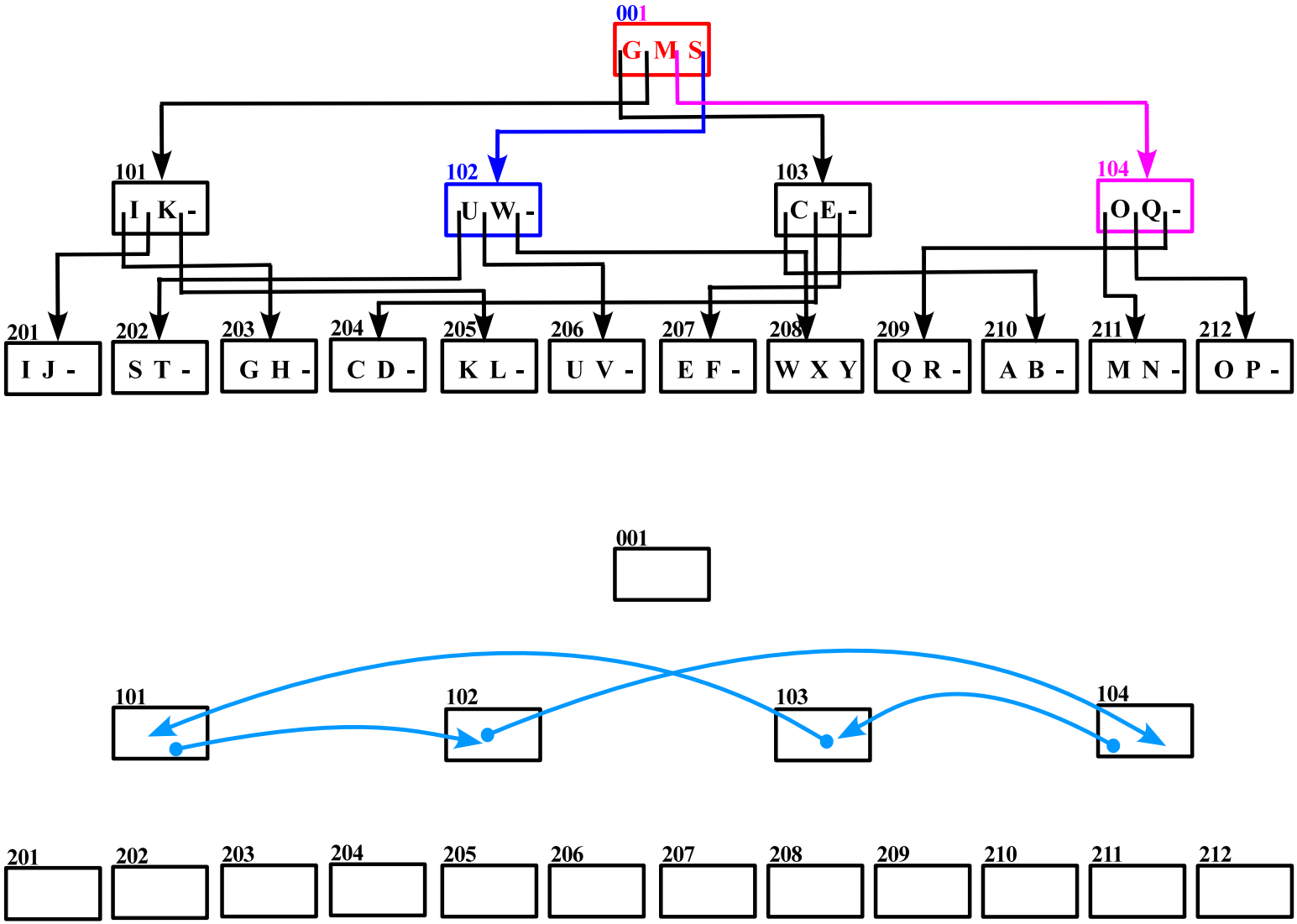
l	$Cache_l$
0	001 [$_{103}$G$_{101}$M$_{104}$S$_{102}$]
1	101 [$_{203}$I$_{201}$K$_{205}$- -]
	103 [$_{210}$C$_{204}$E$_{207}$- -]
2	203 [GH-]
	210 [AB-]

num_cover=1

num_cache=2

target= F

covers= S,M



Access execution – Example

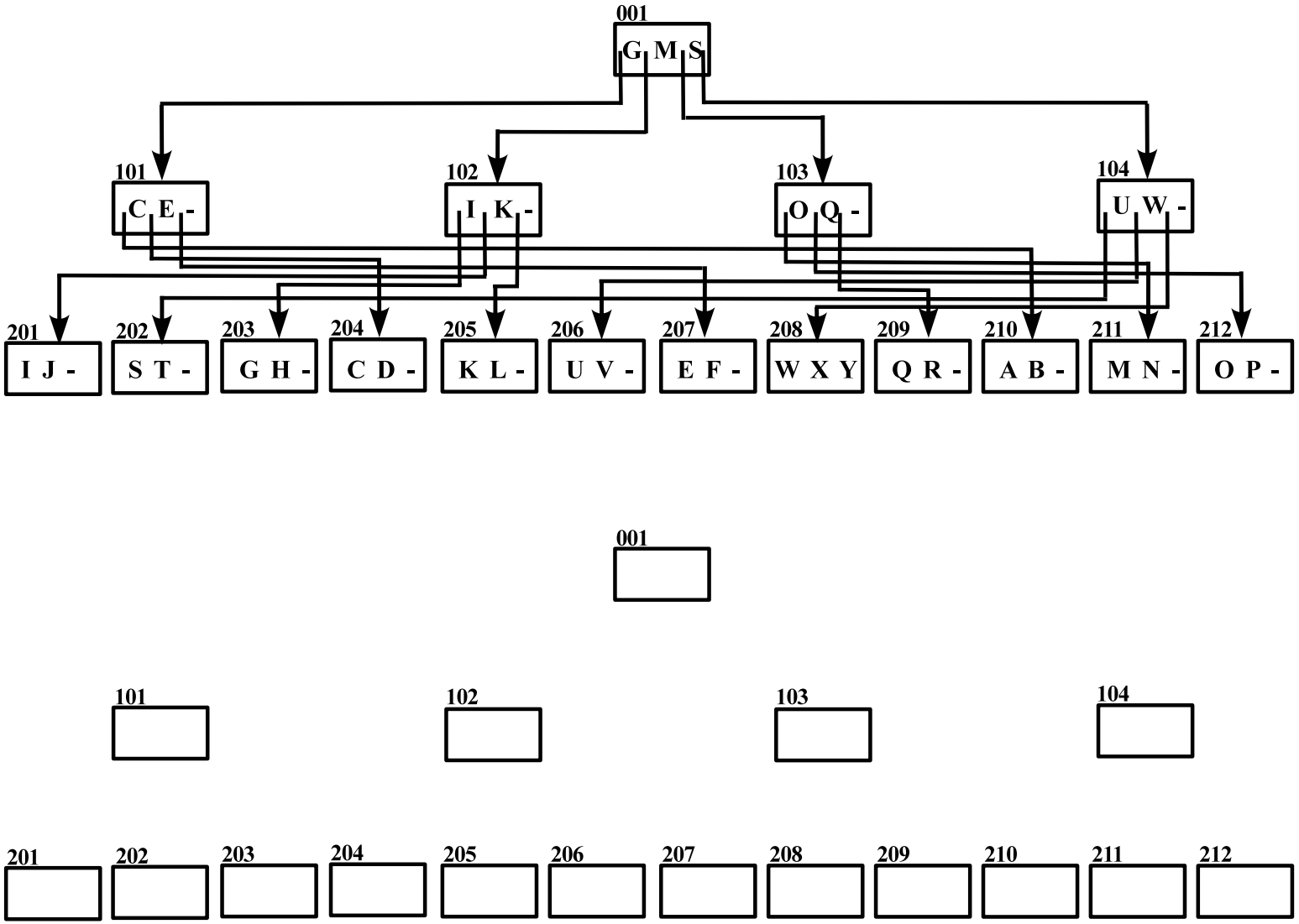
l	$Cache_l$
0	001 [$_{101}$G$_{102}$M$_{103}$S$_{104}$]
1	102 [$_{203}$I$_{201}$K$_{205}$- -]
	101 [$_{210}$C$_{204}$E$_{207}$- -]
2	203 [GH-]
	210 [AB-]

num_cover=1

num_cache=2

target= F

covers= S,M



Access execution – Example

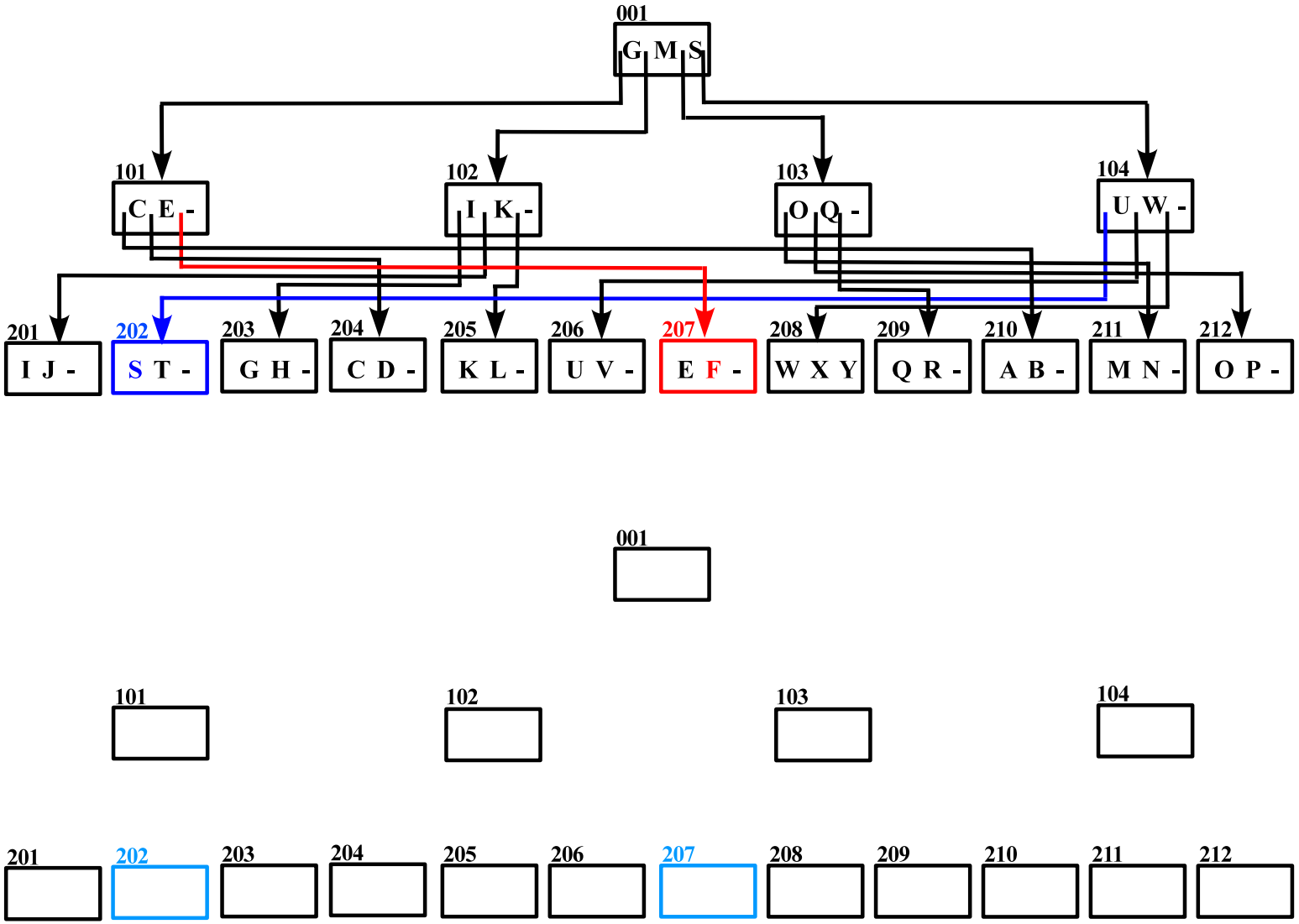
l	$Cache_l$
0	001 [$_{101}$G$_{102}$M$_{103}$S$_{104}$]
1	102 [$_{203}$I$_{201}$K$_{205}$- -]
	101 [$_{210}$C$_{204}$E$_{207}$- -]
2	203 [GH-]
	210 [AB-]

num_cover=1

num_cache=2

target= F

covers= S,M



Access execution – Example

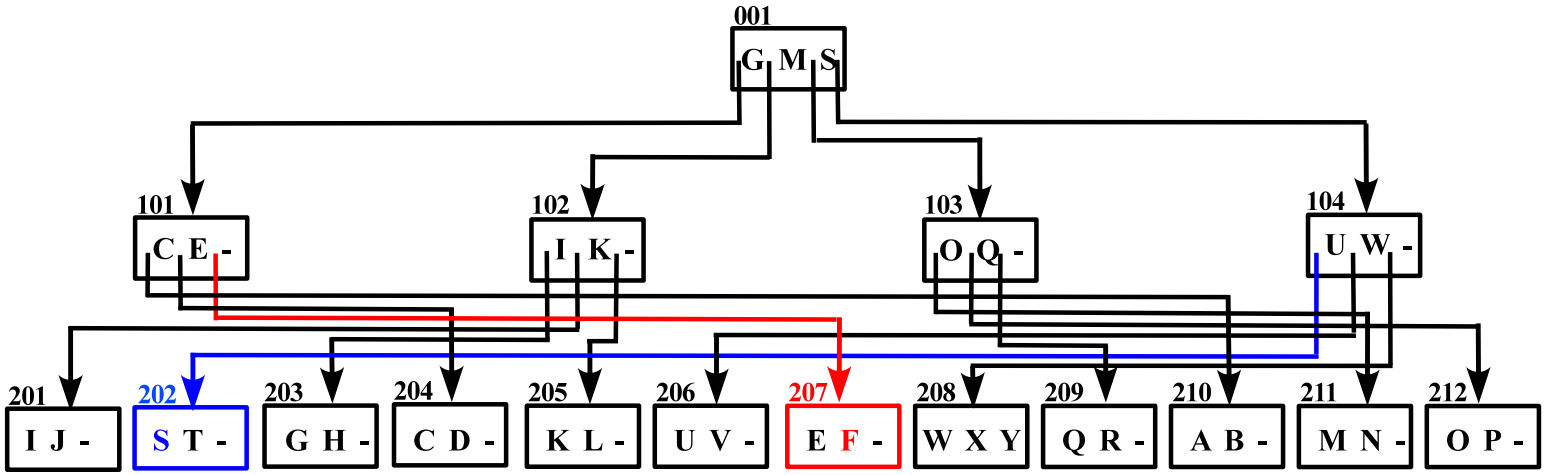
l	$Cache_l$
0	$001\,[_{101}G_{102}M_{103}S_{104}]$
1	$102\,[_{203}I_{201}K_{205}- -]$
	$101\,[_{210}C_{204}E_{207}- -]$
2	$203\,[GH-]$
	$210\,[AB-]$

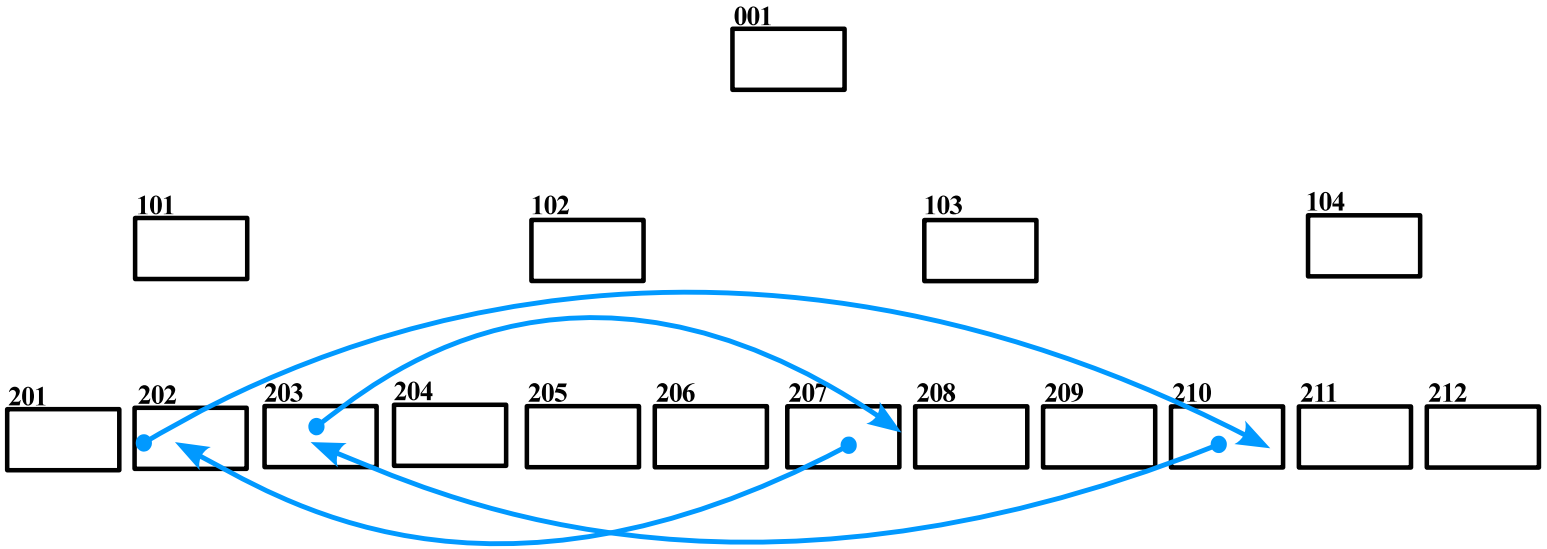
num_cover=1

num_cache=2

target= F

covers= S,M





Access execution – Example

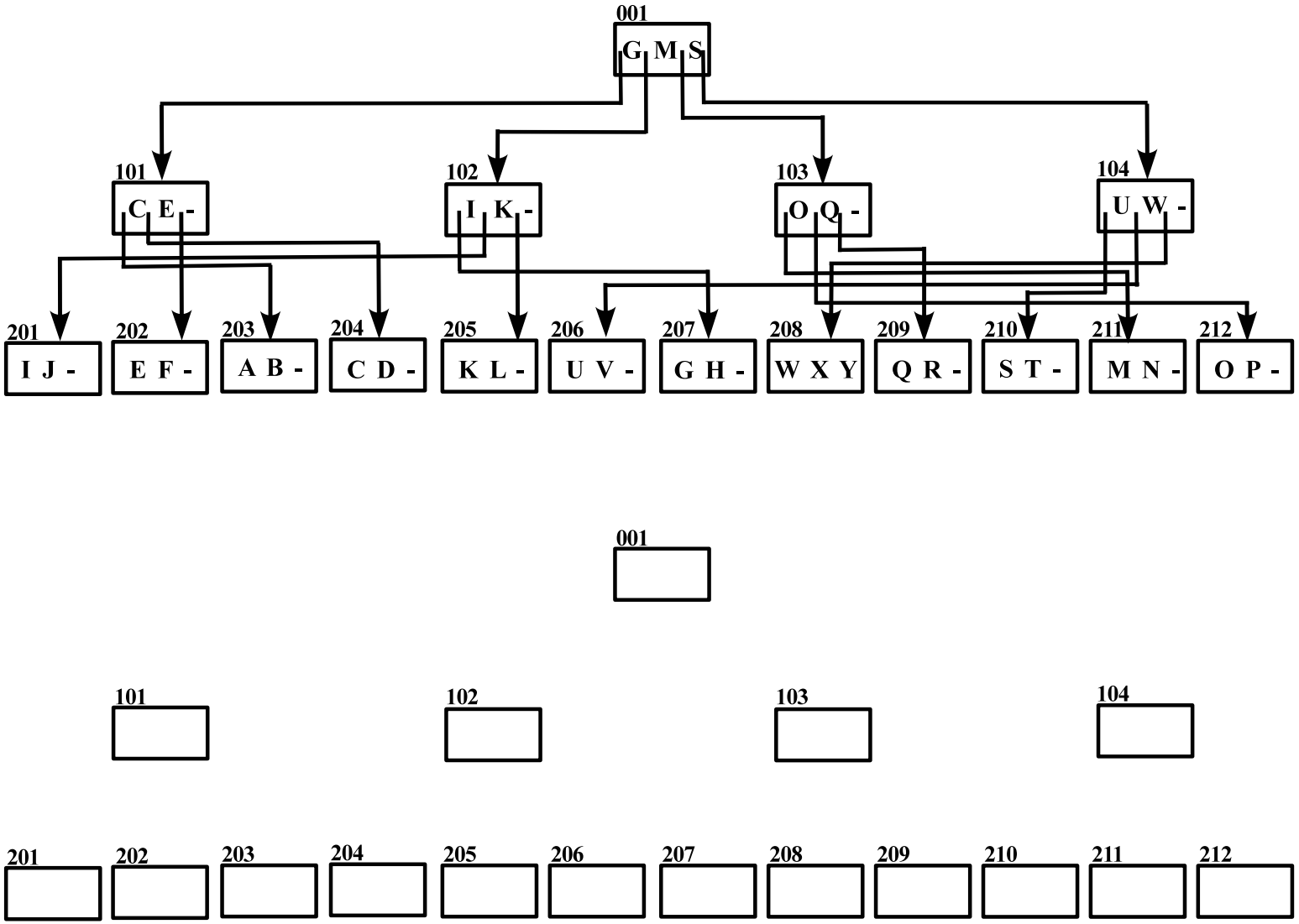
l	$Cache_l$
0	001 [$_{101}$G$_{102}$M$_{103}$S$_{104}$]
1	102 [$_{207}$I$_{201}$K$_{205}$- -]
	101 [$_{203}$C$_{204}$E$_{202}$- -]
2	207 [GH-]
	202 [EF-]

num_cover=1

num_cache=2

target= F

covers= S,M



Access execution – Example

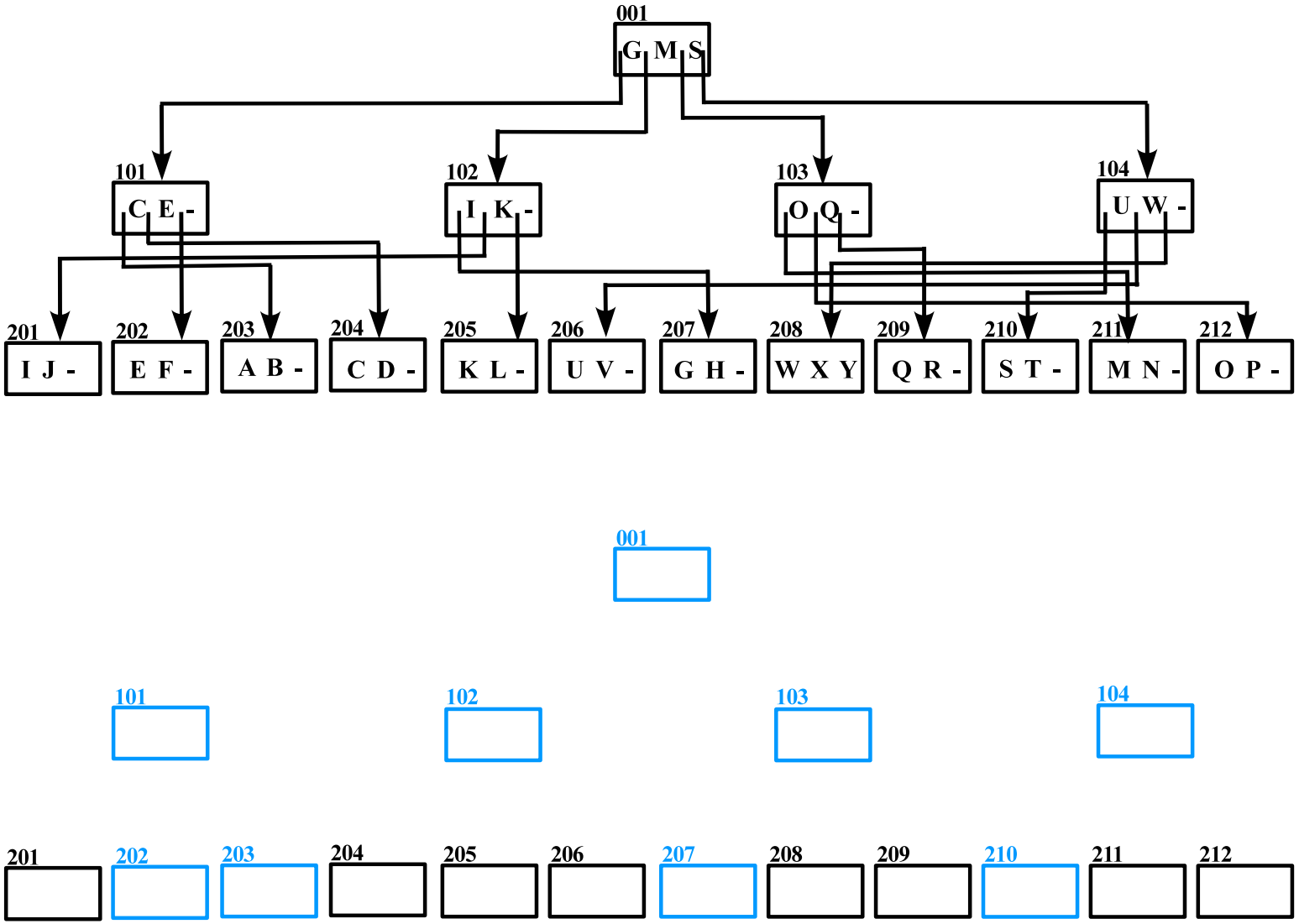
l	$Cache_l$
0	001 [$_{101}$G$_{102}$M$_{103}$S$_{104}$]
1	102 [$_{207}$I$_{201}$K$_{205}$- -]
	101 [$_{203}$C$_{204}$E$_{202}$- -]
2	207 [GH-]
	202 [EF-]

num_cover=1

num_cache=2

target= F

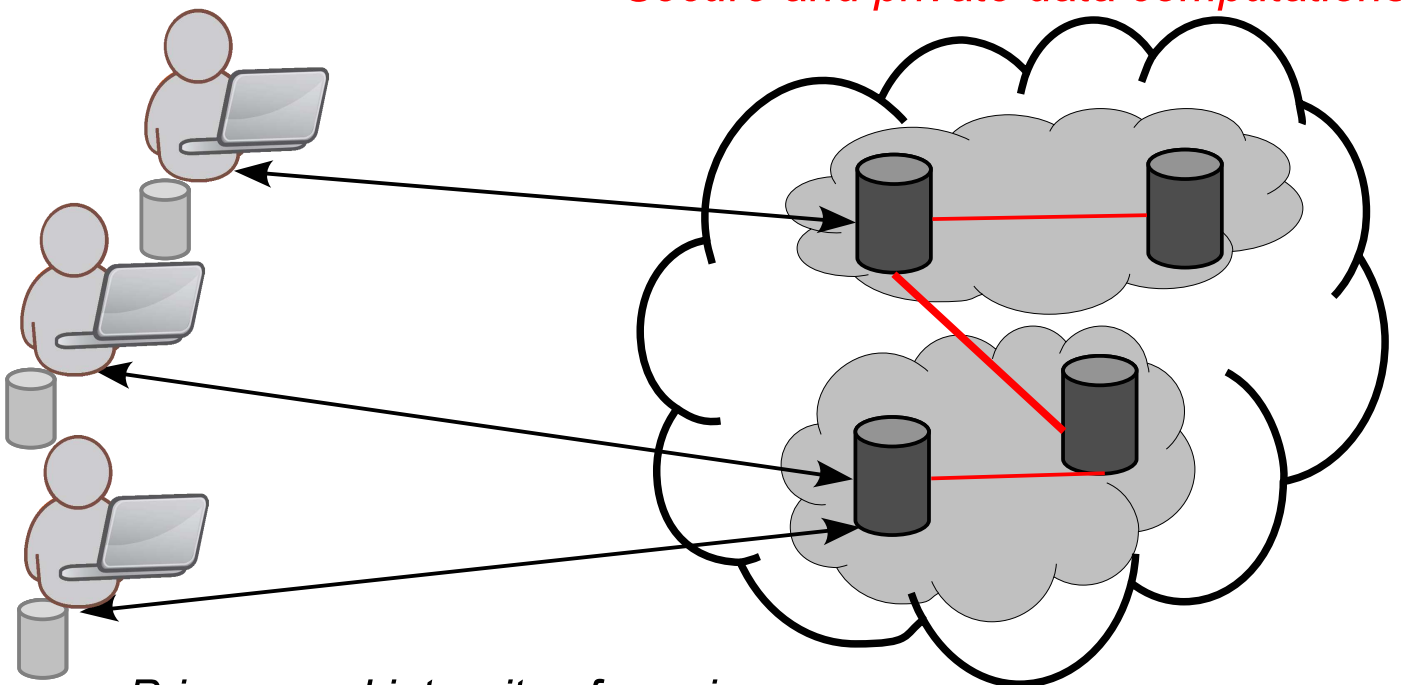
covers= S,M



Open issues...

- Data updates
- Multiple users
- Slicing and distributed storage

Secure and private data computations



Data access and query executions

Data access and query execution are more complex in emerging scenarios

- Data may be stored outside the data owner's control
- Application/query executions may entail access to data under control of different parties
- Data can move around to different locations

$\Longrightarrow$ Specification and enforcement of data sharing constraints for regulating query execution in distributed multi-authority scenarios

Some approaches

- Sovereign joins: computes a join in a way that nothing beyond the query result is revealed [Agrawal et al, ICDE 2006]

- Access patterns: specify limitations on how information sources can be accessed (e.g., [Calì et al, J.UCS 2009])

- View-based access control: provide fine-grained content-dependent access control in relational databases (e.g., [Motro, JIIS 1989; Rosenthal and Sciore, DBSec 2001; Rizvi et al., SIGMOD 2004])

- Distributed query evaluation under protection requirements

Distributed Query Evaluation under Protection Requirements

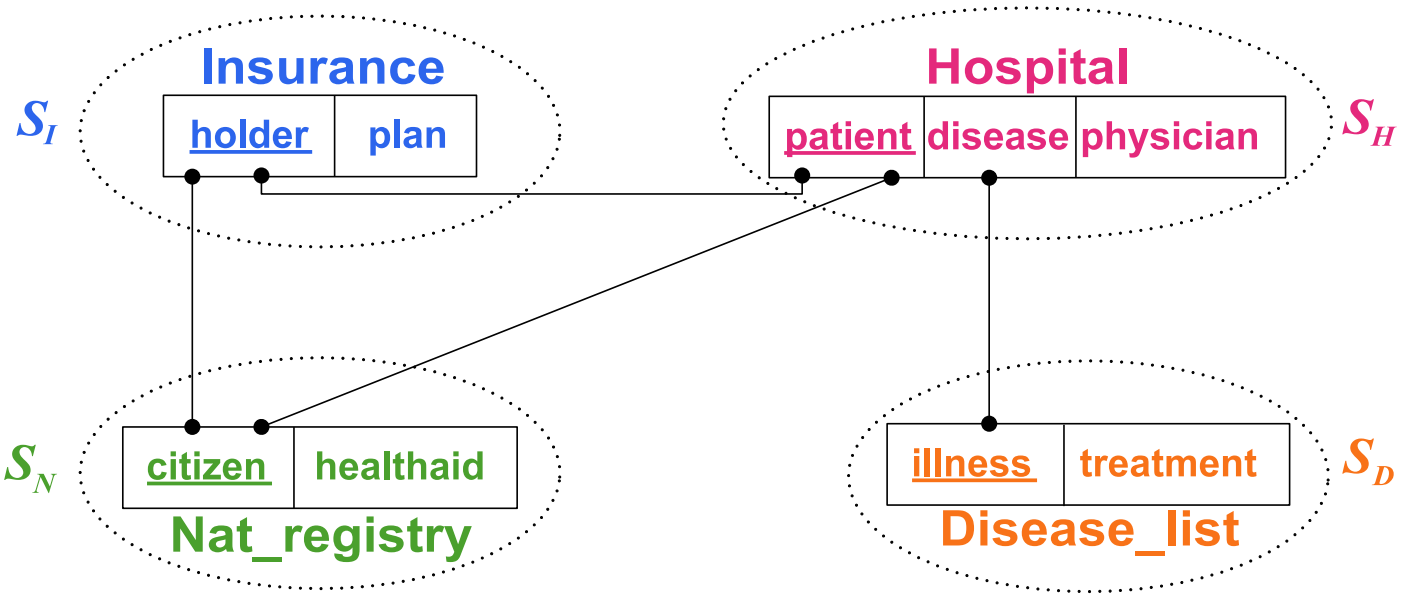
S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, P. Samarati, “Authorization Enforcement in Distributed Query Evaluation,” in *Journal of Computer Security*, 2011.

Problem addressed

Regulate views and information sharing among different parties:

- support the collaboration among parties in distributed query execution on data subject to selective release
- define authorized views based on information content of a relation
- assign operations within the query to different parties in a way that is safe with respect to information that can be viewed by parties

Distributed relations – Example



Feasible query plan

Goal: given a query tree plan, determine for each operation a subject (pair of subjects in case of semi-join) responsible for the execution such that all views are authorized

- Authorization: $[\mathit{Attributes}, \mathit{JoinPath}] \rightarrow \mathit{Subject}$ authorizes release to *Subject* of set *Attributes* of attributes resulting from the *JoinPath* (sequences of equi-joins)

- Relation profile $[R^{\pi}, R^{\bowtie}, R^{\sigma}]$: capture the information content of either a base or derived (i.e., computed by a query) relation R

- Authorized view: Subject s is authorized to view a relation R iff: $\exists [\mathit{Attributes}, \mathit{JoinPath}] \rightarrow s$: $R^{\pi} \cup R^{\sigma} \subseteq \mathit{Attributes} \wedge R^{\bowtie} = \mathit{JoinPath}$

Authorized view – Example

Query from S_D:

SELECT illness
FROM Disease_list JOIN Hospital ON illness=disease
WHERE treatment = 'antihistamine'

Profile: $[R^{\pi}, R^{\bowtie}, R^{\sigma}]$
[(illness),(⟨D.illness,H.disease⟩),(treatment)]

Authorization:
[(illness,treatment),(⟨D.illness,H.disease⟩)] $\rightarrow S_D$
authorizes the query

Authorization: [(illness,treatment),_] $\rightarrow S_D$
does not authorize the query

Insurance(holder,plan)	Hospital(patient,disease,physician)
Nat_registry(citizen,healthaid)	Disease_list(illness,treatment)

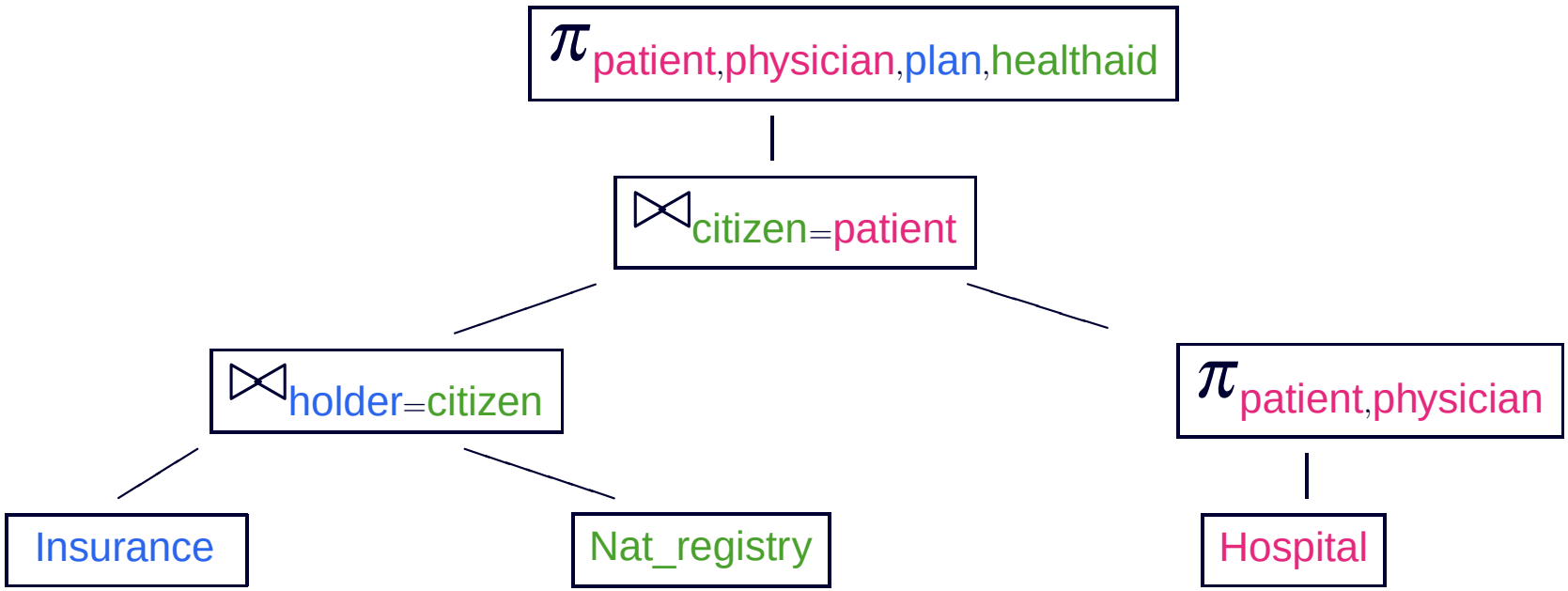
Executor assignment – Example

```
SELECT patient, physician, plan, healthaid
FROM   Insurance JOIN Nat_registry ON holder=citizen
       JOIN Hospital ON citizen=patient
```

Insurance(holder,plan)	Hospital(patient,disease,physician)
Nat_registry(citizen,healthaid)	Disease_list(illness,treatment)

Executor assignment – Example

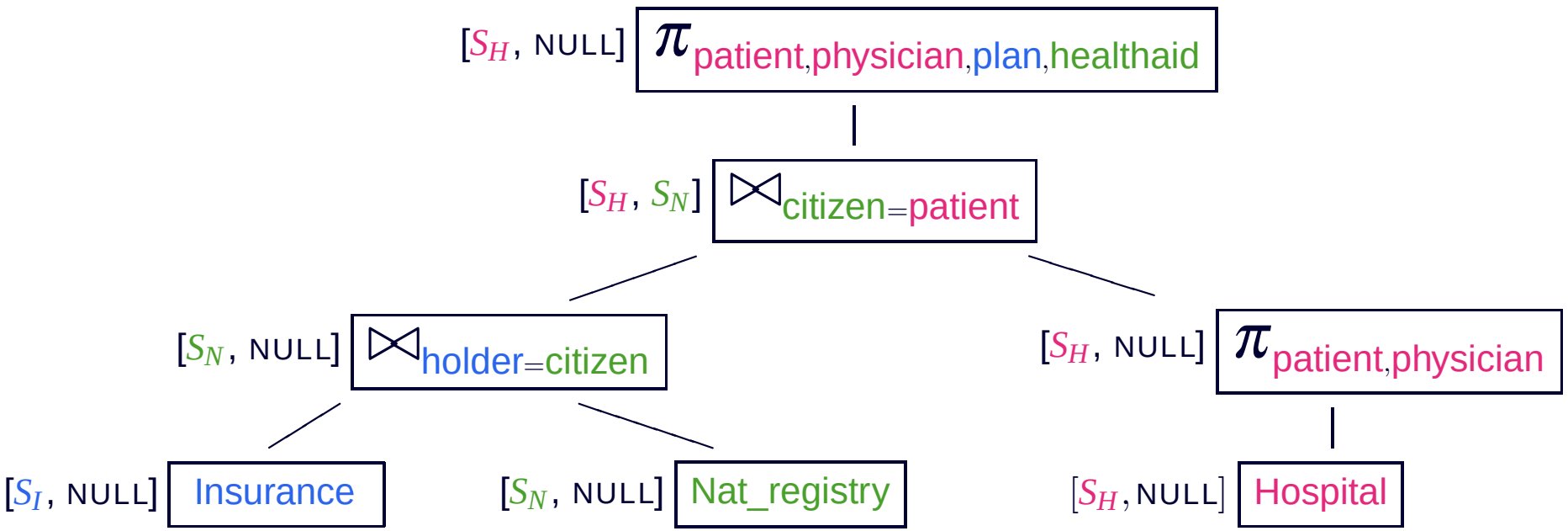
```
SELECT patient, physician, plan, healthaid
FROM   Insurance JOIN Nat_registry ON holder=citizen
       JOIN Hospital ON citizen=patient
```



Insurance(holder,plan)
Nat_registry(citizen,healthaid)
Hospital(patient,disease,physician)
Disease_list(illness,treatment)

Executor assignment – Example

```
SELECT patient, physician, plan, healthaid
FROM   Insurance JOIN Nat_registry ON holder=citizen
       JOIN Hospital ON citizen=patient
```



Insurance(holder,plan)
Nat_registry(citizen,healthaid)
Hospital(patient,disease,physician)
Disease_list(illness,treatment)

Is this enough?

- Different servers may have different levels of trust
- Need to consider encrypted data for possibly adopting different kinds of servers in the computation
- Definition of trust boundaries
- Need to verify the integrity of the query results by exploiting the economical and functional advantages of the cloud technology …

Integrity in Query Computation

Integrity in 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:
  - authenticated data structures (e.g., signature chains, Merkle hash trees, skip lists)
  - probabilistic: exploits insertion of fake tuples in query results, replication of tuples in query results, pre-computed tokens

Integrity in query computation – 2

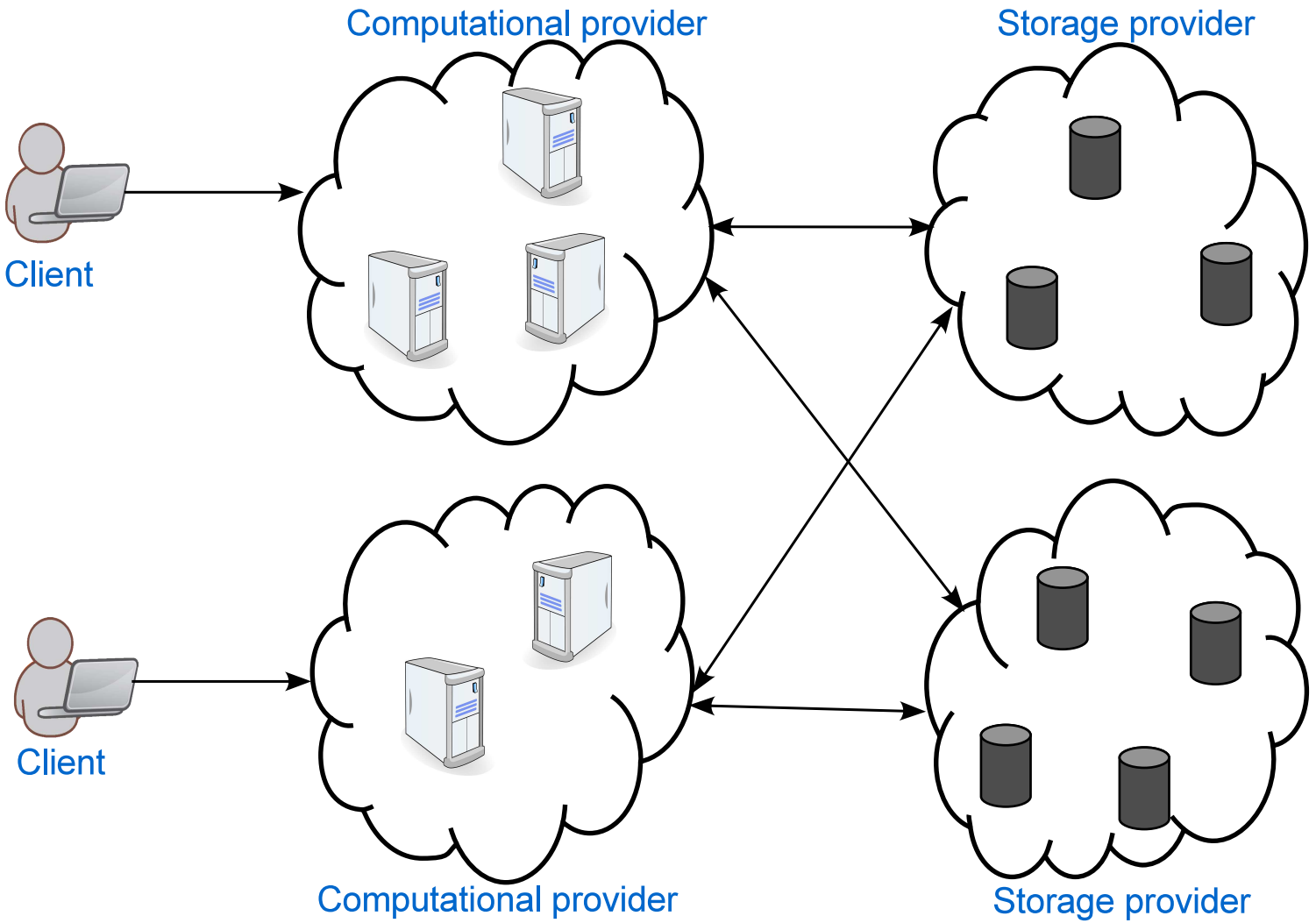
- Other approaches consider the verification of the integrity of query results of complex queries (joins):
  - fake tuples [Xie et al., VLDB 2007]
    - spurious tuples
    - high network overhead
  - Merkle hash tree or its variations [Li et al., SIGMOD 2006; Yang et al., SIGMOD 2009]
    - support only joins on which the Merkle hash tree has been constructed

Cloud opportunities

- The market shows an evolution toward of a varied ecosystem: different providers offer to the users different functional abilities
  - storage services: offer continuous availability of stored data with high bandwidth and reliability guarantees
  - computational services: offer efficient execution of computationally intensive services
- Cloud technology is used for developing applications that integrate data and function hosted by different service providers
- Not only performance but also economical costs are a key factor

  $\Longrightarrow$ exploit low-cost computational providers, while maintaining security and privacy guarantees

Scenario

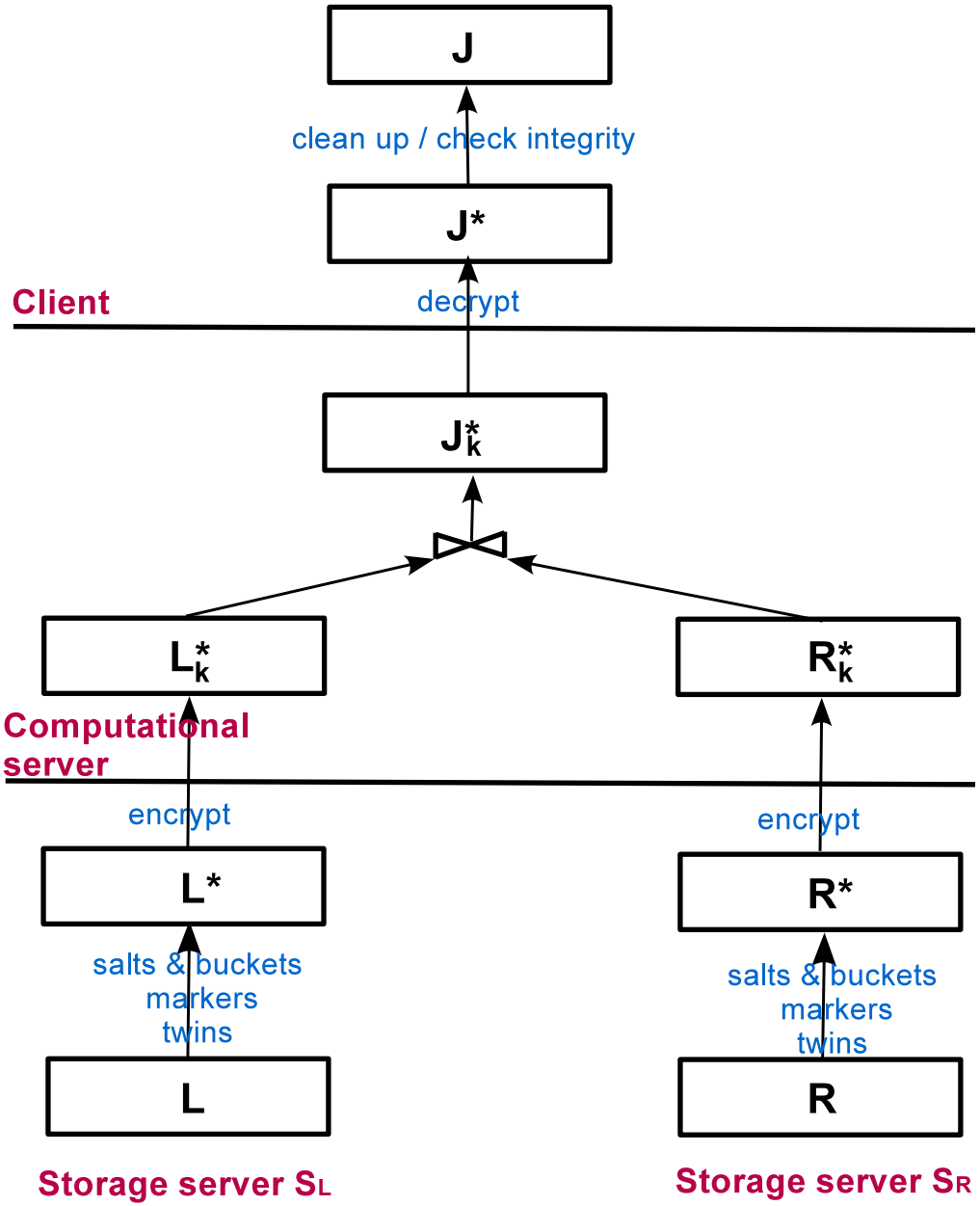

Computational provider
Storage provider
Client
Client
Computational provider
Storage provider

Probabilistic approach for join queries

- A client, with the cooperation of the storage servers, should assess the integrity of the join performed by the computational server

- Protection techniques:
  - encryption makes data unintelligible
  - markers (additional fake tuples) and twins are two complementary techniques signaling incompleteness of the query results
  - salts and buckets in the case of one-to-many joins

Probabilistic approach for join queries – Example



Open issues...

- Work distribution (e.g., join vs semi-join)
- Consideration of different trust levels
- Application of the techniques to only a portion of the data (verification object)

Conclusions

Novel Cloud scenarios:

+ provide great convenience and benefit in the management and access to the information

– introduce privacy and security risks, which require investigation and development of new techniques