

# NSAC

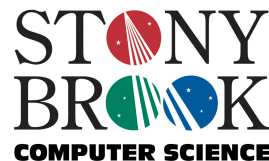
Network Security and Applied  
Cryptography Laboratory

<http://crypto.cs.stonybrook.edu>

# Secure Data Outsourcing

Tutorial @ COMAD 2006, New Delhi, India

Radu Sion  
Stony Brook NSAC Lab  
[sion@cs.stonybrook.edu](mailto:sion@cs.stonybrook.edu)



# Feynman moment



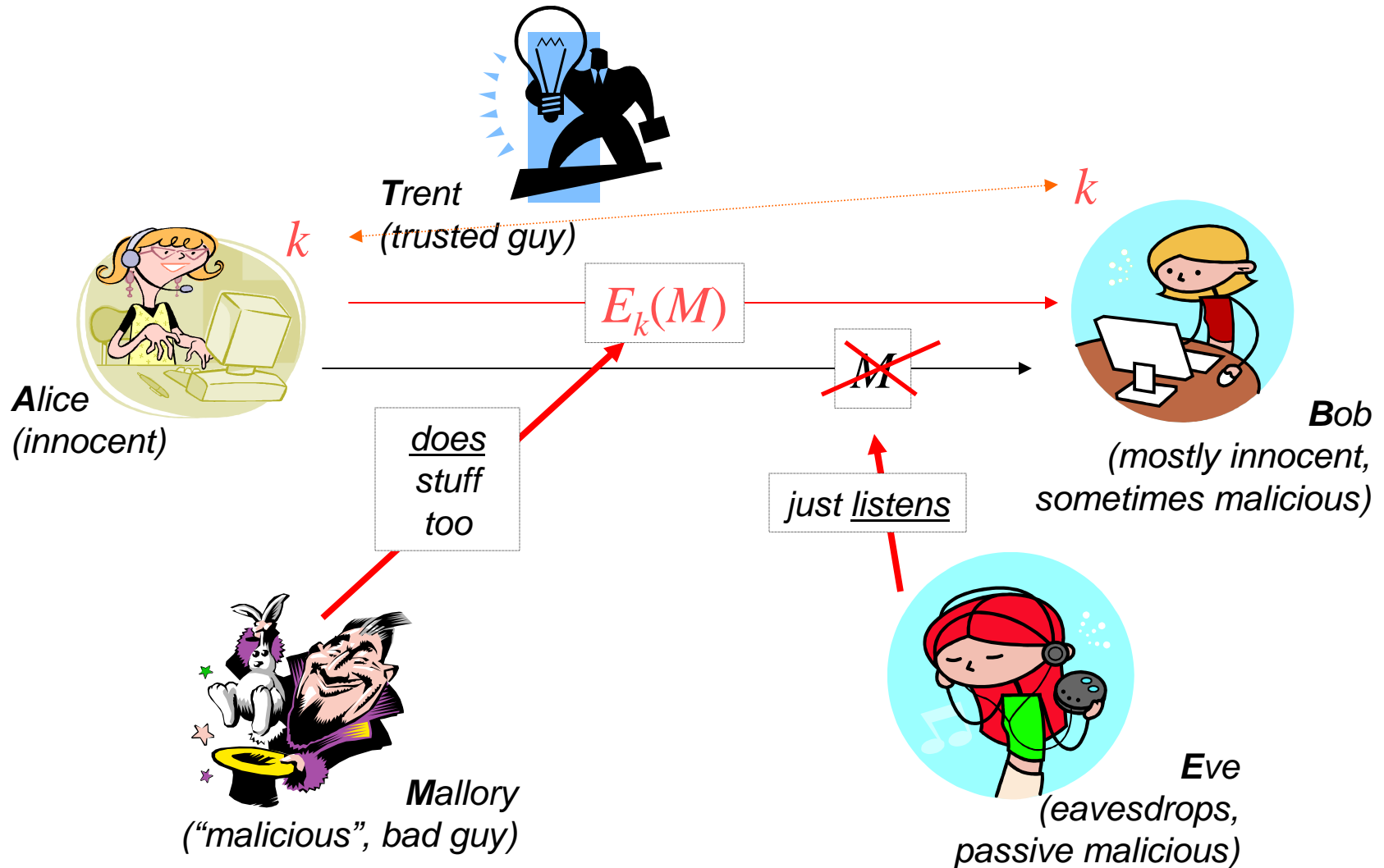
© Copyright California Institute of Technology. All rights reserved.  
Commercial use or modification of this material is prohibited.

"I have much experience only in teaching graduate students [...] and as a result [...] I know that I don't know how to teach."

- ➔ *Crypto Crash Course*
- ➔ *Data Outsourcing*
- ➔ *Query Correctness*
- ➔ *Data Confidentiality*
- ➔ *Access Privacy*
- ➔ *Searching on Encrypted Data*
- ➔ *Trusted Hardware*

- *Randomness*
- *Crypto Hashes*
- *Encryption*
- *Public key encryption*
- *Signatures*
- *Ciphers*
- *Semantic Security*
- *Forward Secrecy*
- *Performance*
- *Merkle/Hash trees*

# Crypto: Meet the cast



*Cryptographically random numbers:* a sequence of numbers  $X_1, X_2, \dots$  such that for any integer  $k > 0$ , it is **impossible** for an observer to predict  $X_k$  even if all of  $X_1, \dots, X_{k-1}$  are known.

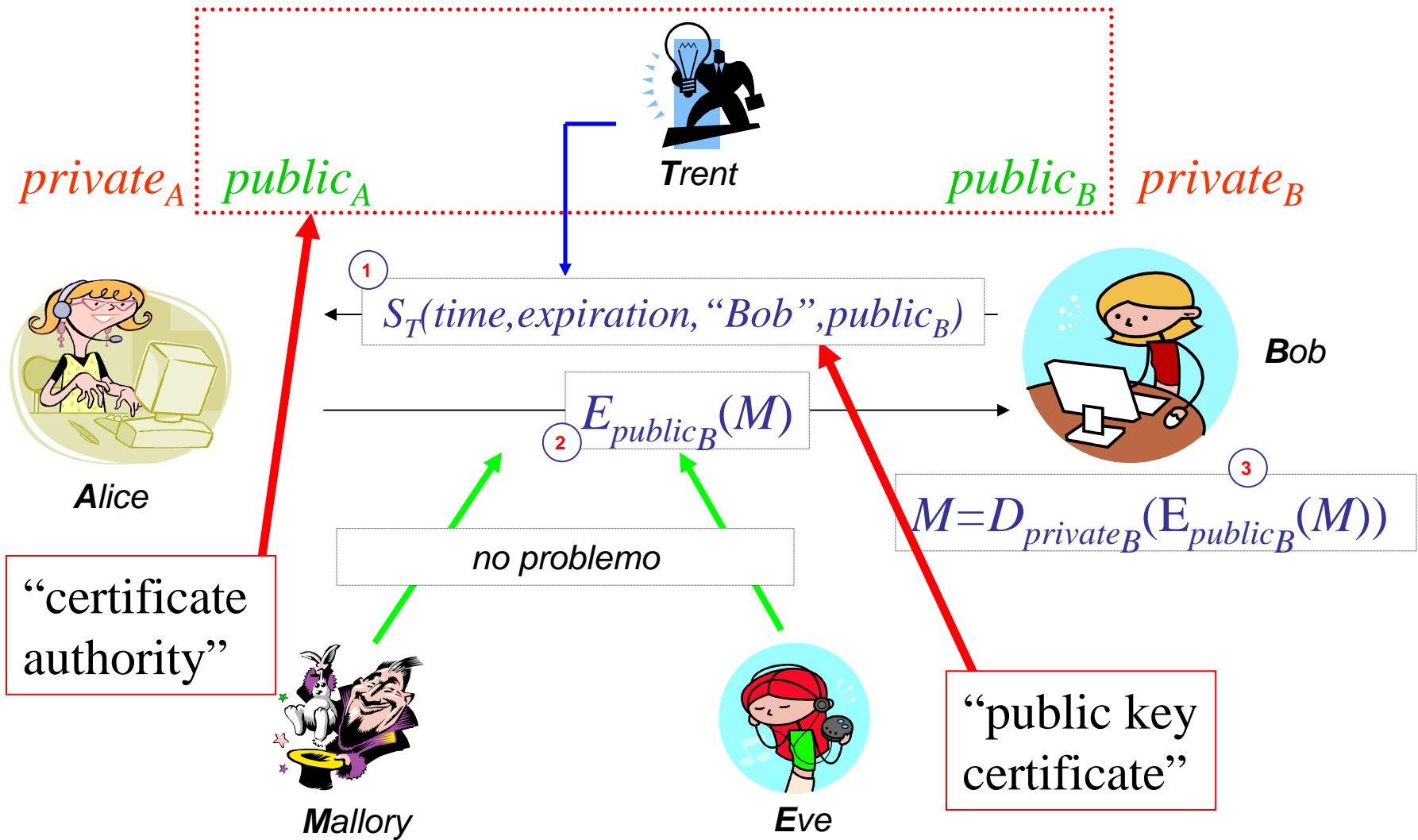
**Problem:** True RNGs cannot be deterministically algorithmic in a closed system. “Anyone who considers arithmetic methods ... is in a state of sin” (von Neuman)

**Being creative:** simulate a sequence of cryptographically random numbers but generate them by an algorithm.

*Pseudo-random numbers:* a sequence of numbers  $X_1, X_2, \dots$  such that for any integer  $k > 0$ , it is **hard** for an observer to predict  $X_k$  even if all of  $X_1, \dots, X_{k-1}$  are known.

- A hash is a **one-way, non-invertible** function of that produces **unique** (with *high likelihood*), **fixed-size** outputs for different inputs.
- The probability of any bit “flipping” in the output bit-string should be always  $\frac{1}{2}$  for any change (even one bit) in the input (“randomness”).

# Crypto: PKI

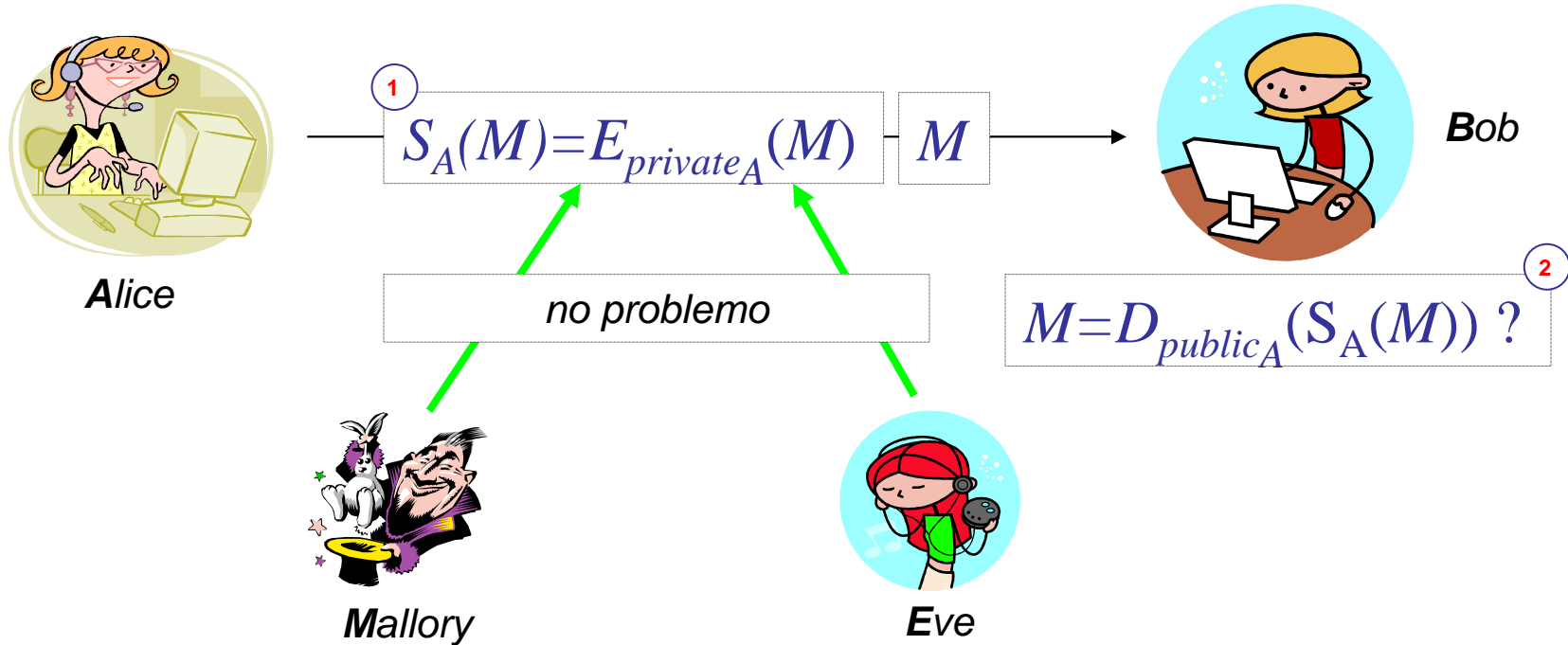


# Crypto: Signatures

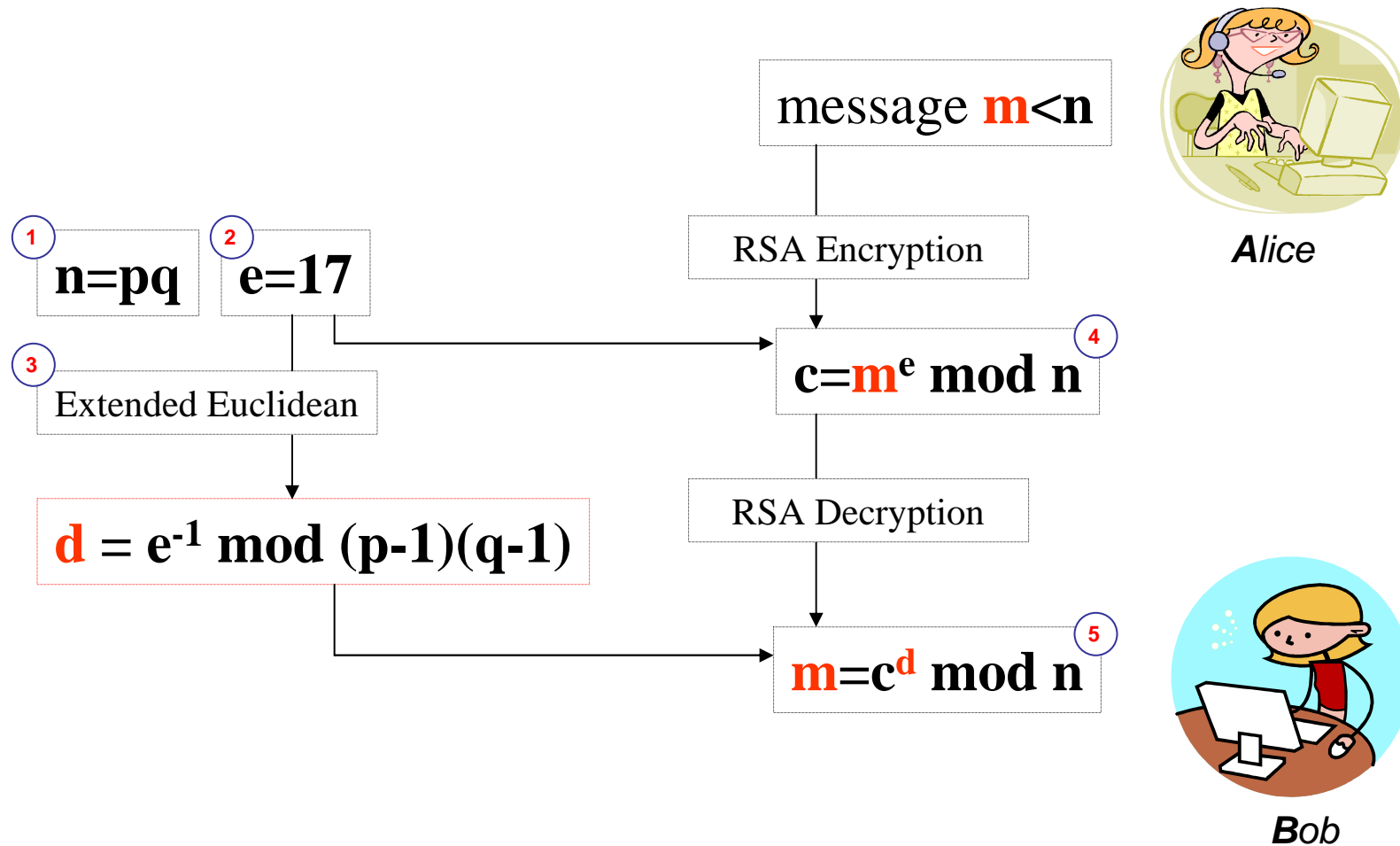
$$M = D_{private_A}(E_{public_A}(M)) = D_{public_A}(E_{private_A}(M))$$

*private<sub>A</sub>* *public<sub>A</sub>*

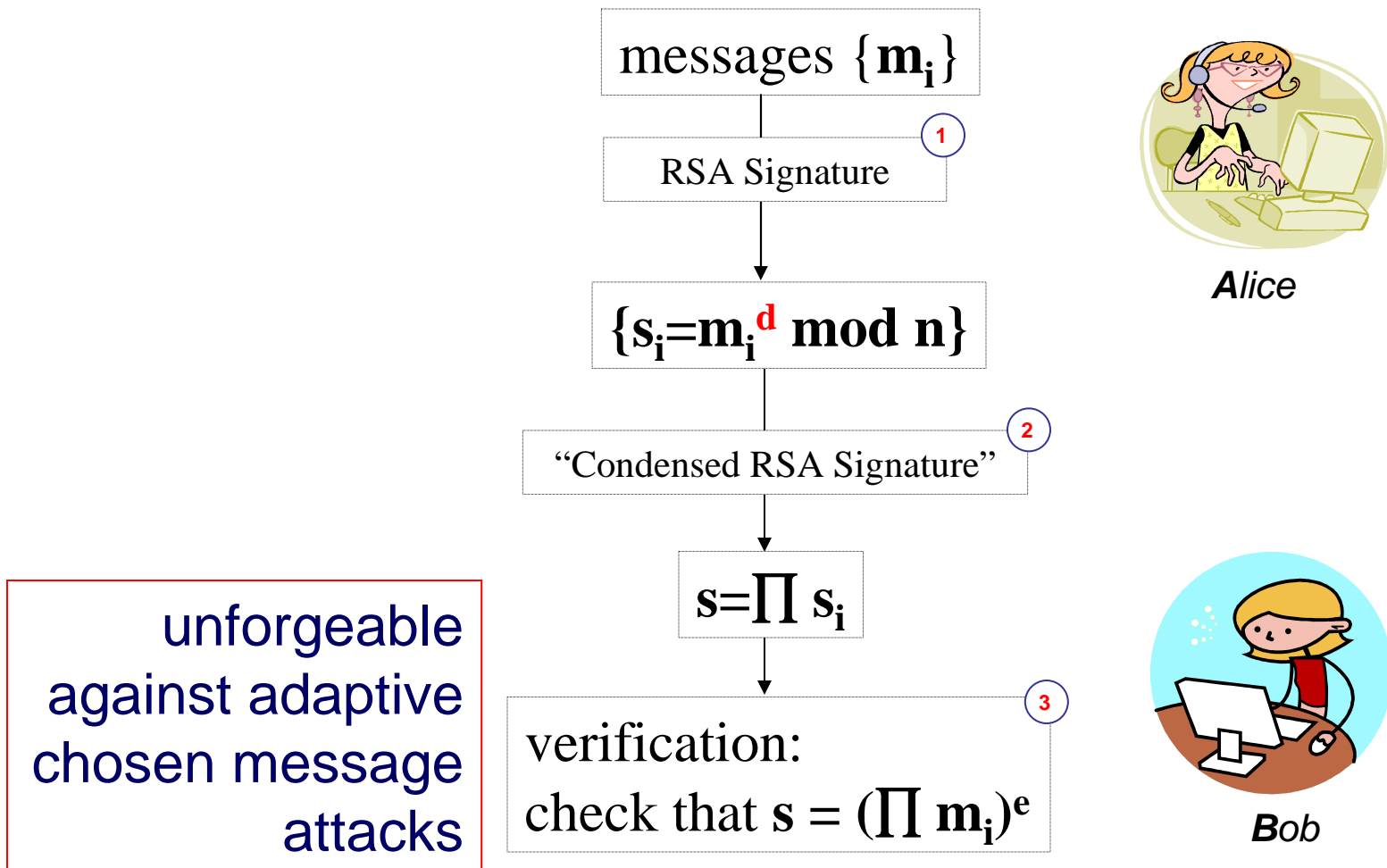
*public<sub>B</sub>* *private<sub>B</sub>*



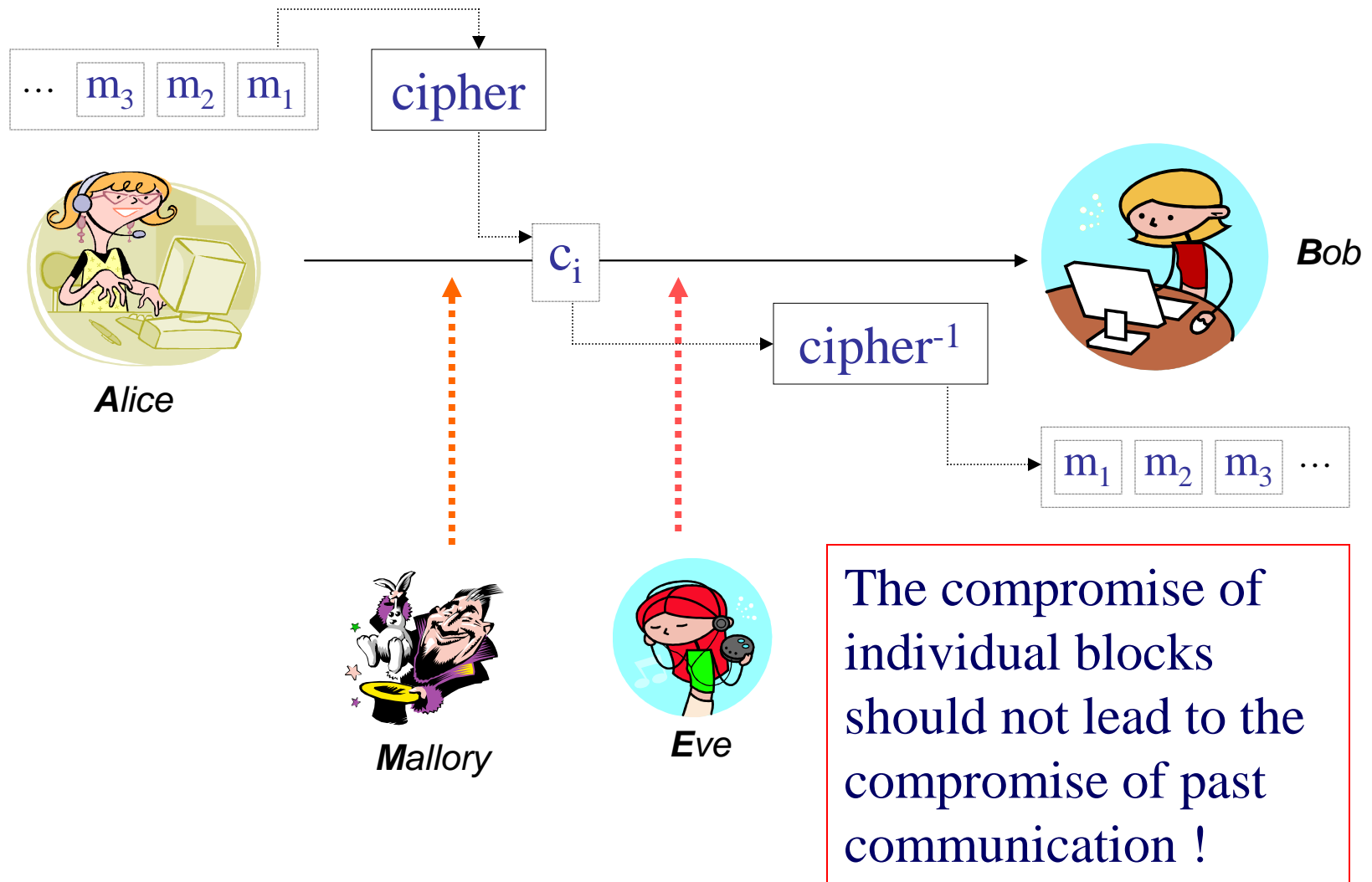
# Crypto: RSA in a nutshell



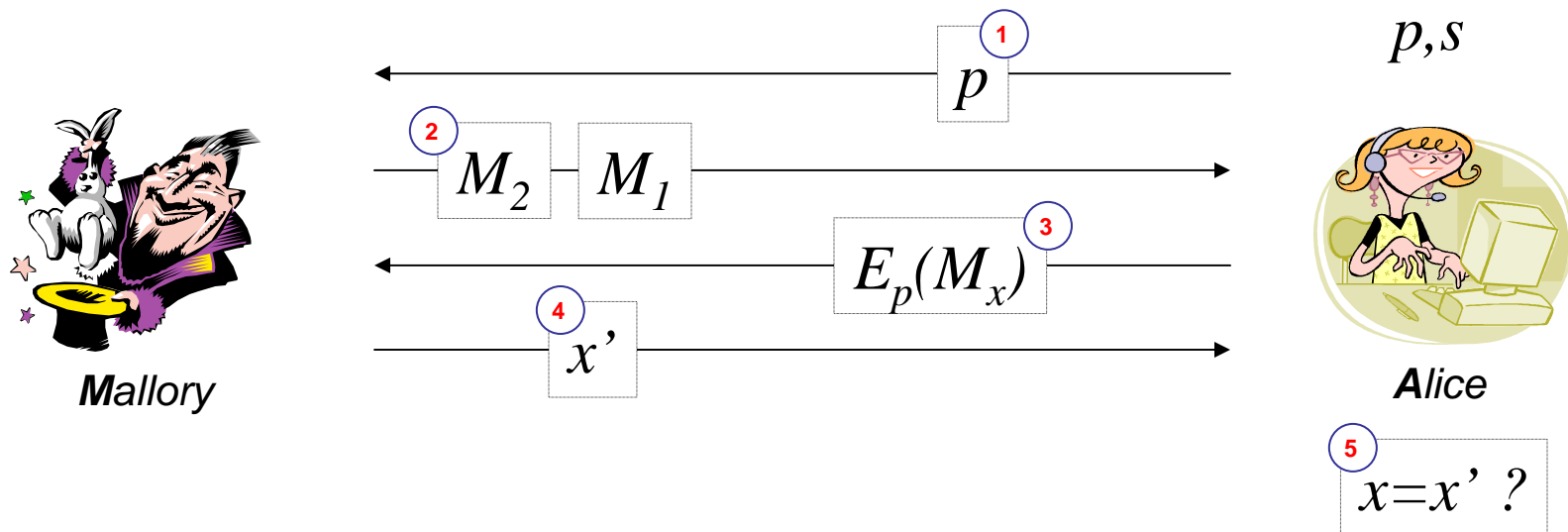
# Crypto: Condensed RSA



# Crypto: Ciphers

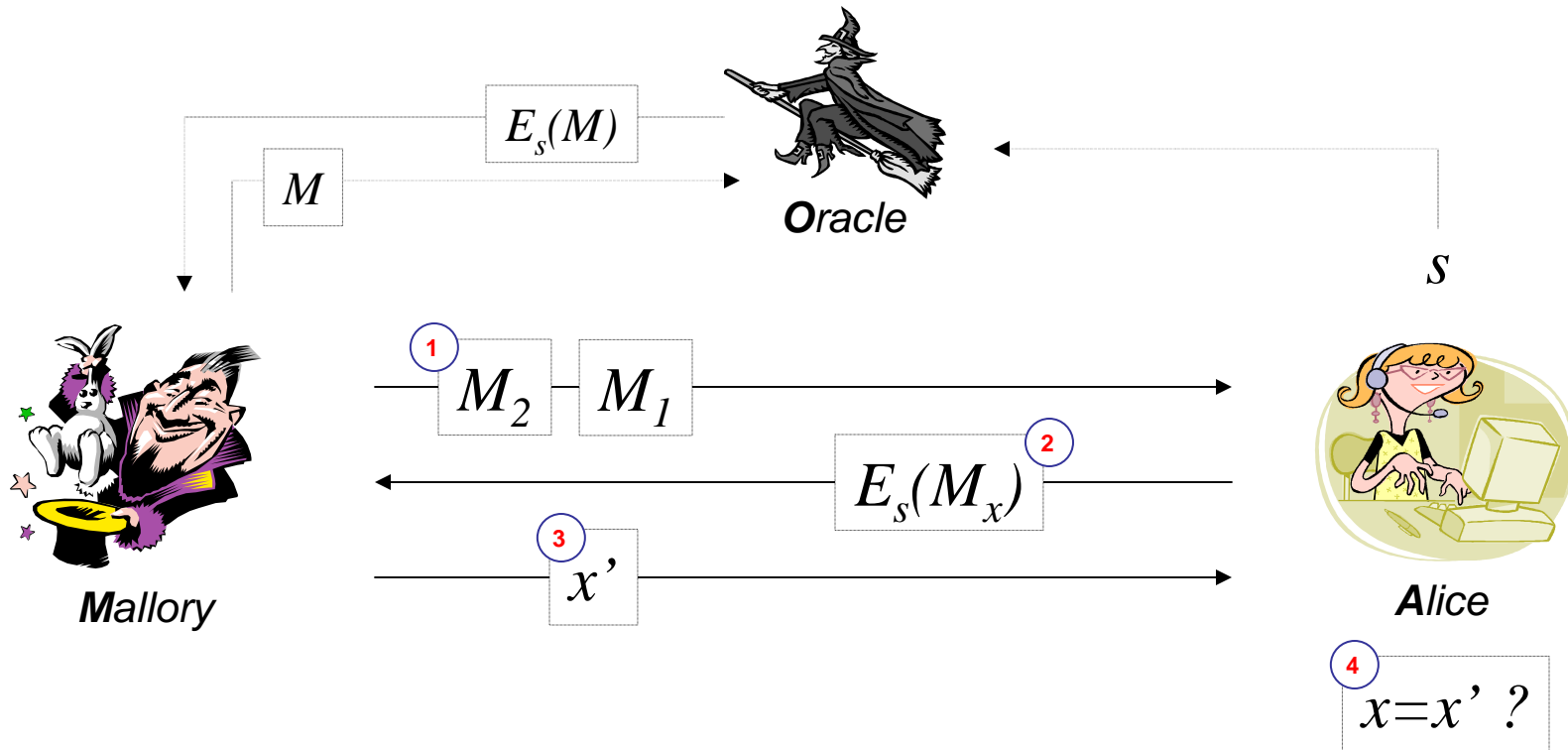


$\text{len}(M_1) = \text{len}(M_2) ?$



$E()$  is **indistinguishable under a chosen plaintext attack** (IND-CPA, “semantically secure”) if no probabilistic polynomial time-bounded Mallory can succeed in finding  $x'$ , significantly better than guessing.

# Crypto: Semantic Security



- Deterministic + stateless = insecure !
- Semantic security implies *bit security* !
- RSA : non-semantically secure ! Why ?!
- RSA + padding (e.g., RSA-OAEP): ok

Future compromise (e.g., of PK secrets) should not propagate backwards in time.

# Crypto: Performance



## Illustrative baseline.

Pentium 4. 3.6GHz.  
1GB RAM. 11000 MIPS.  
OpenSSL 0.9.7f

DES/CBC: **70MB/sec**

RC4: **138MB/sec**

MD5: **18-615MB/sec**

SHA1: **18-340MB/sec**

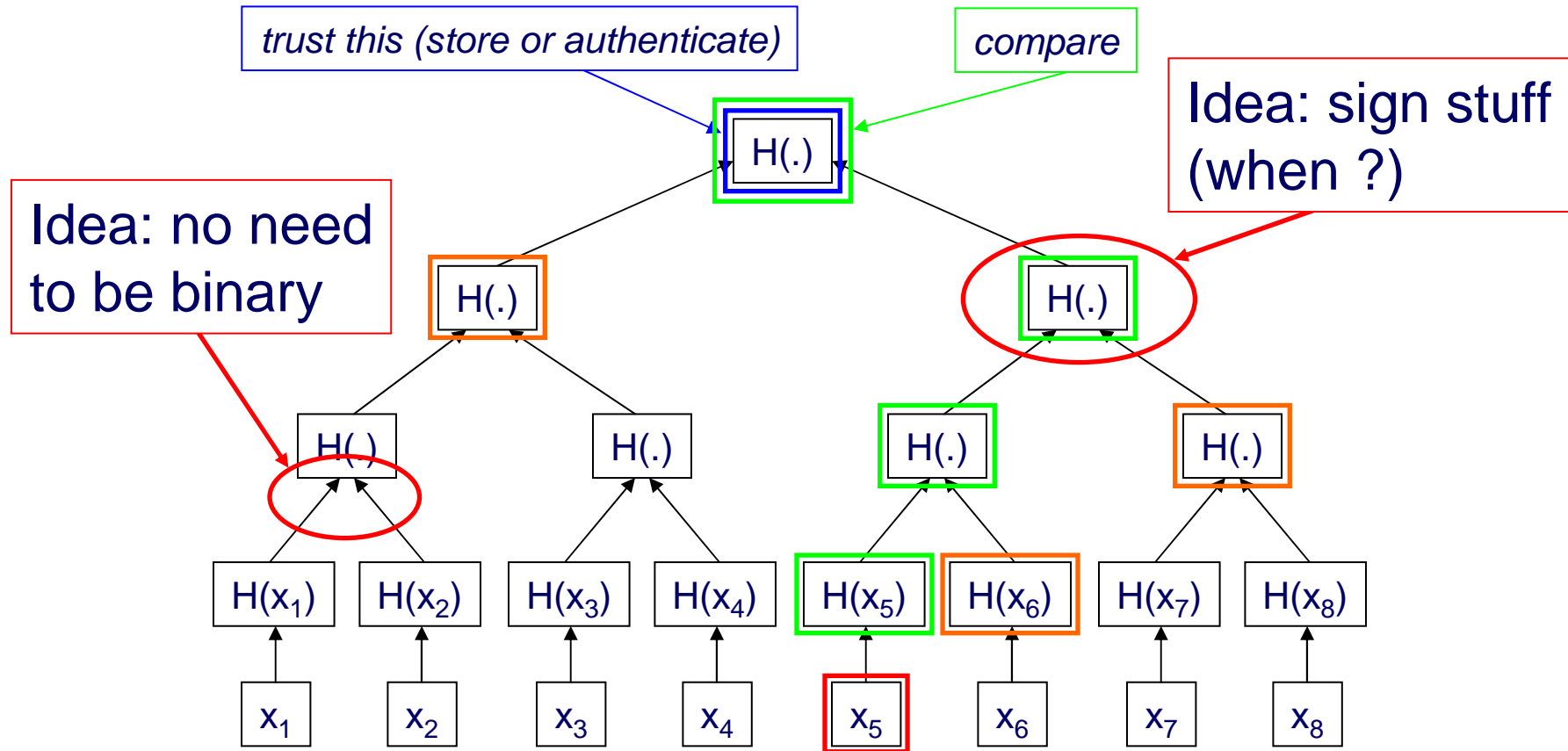
Modular MUL 1024: **273000/sec**

RSA1024 Sign: **261/sec**

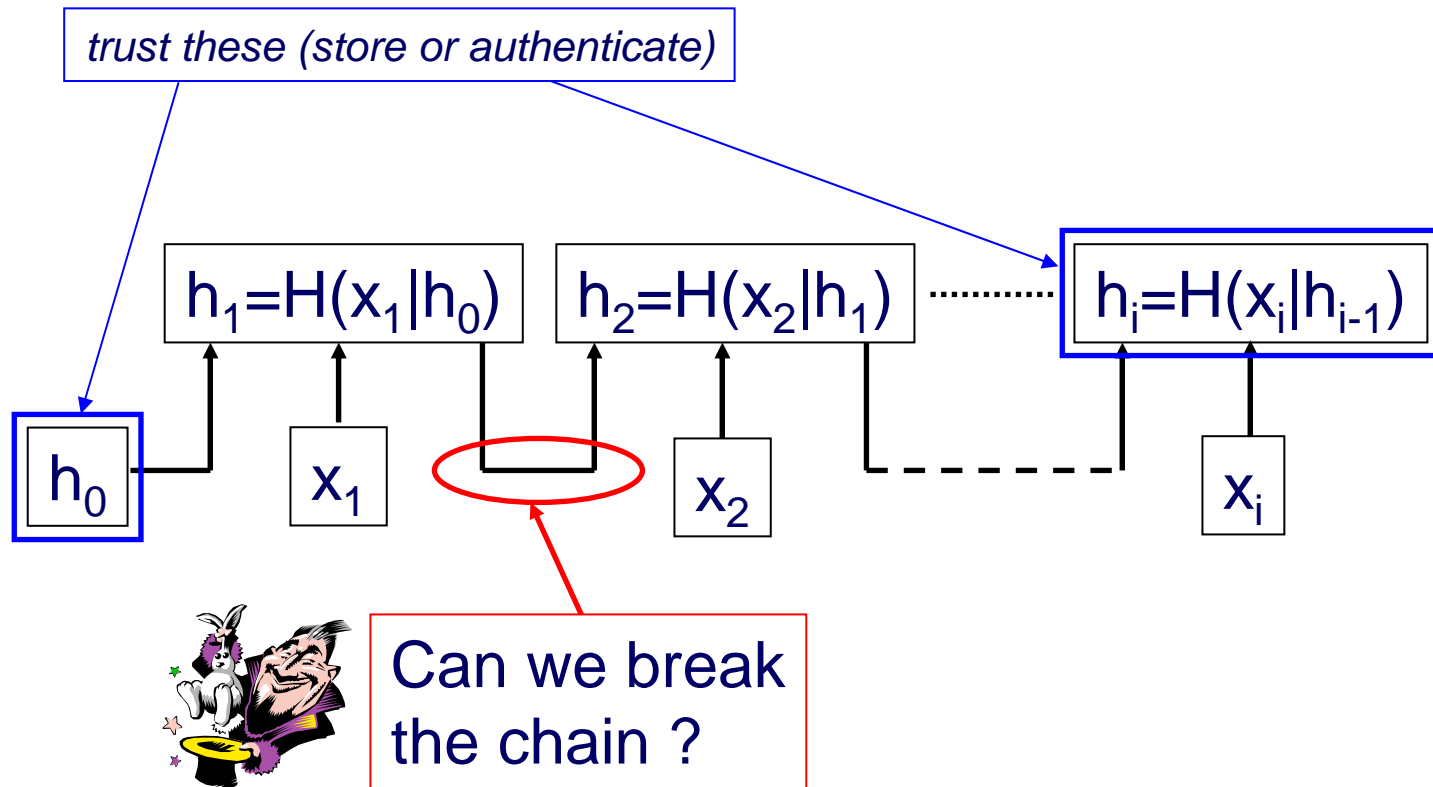
RSA1024 Verify: **5324/sec**

3DES: **26MB/sec**

# Crypto: Merkle/Hash trees



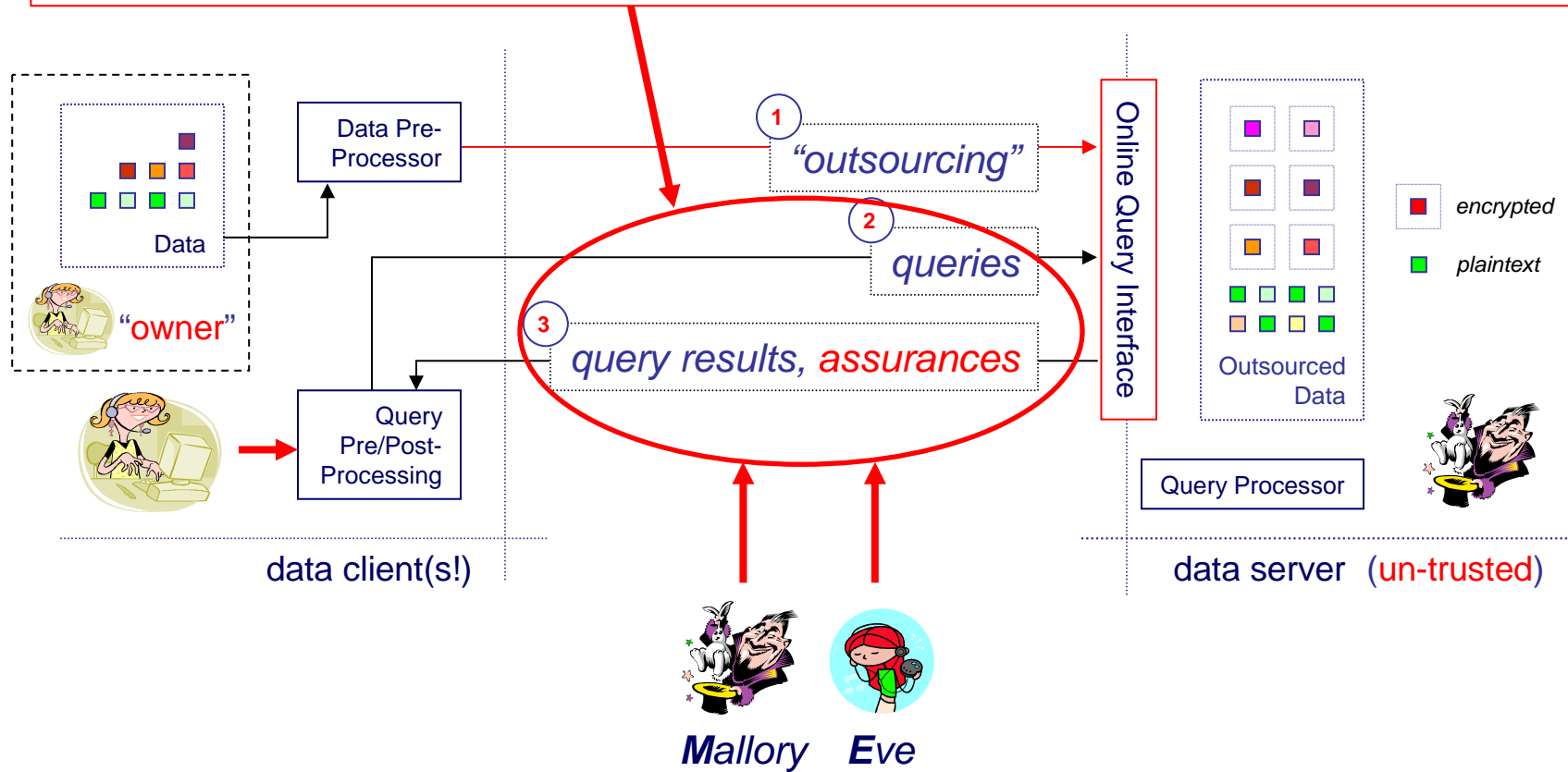
# Crypto: Hash chains



- ➔ *Crypto Crash Course*
- ➔ *Data Outsourcing*
- ➔ *Query Correctness*
- ➔ *Data Confidentiality*
- ➔ *Access Privacy*
- ➔ *Searching on Encrypted Data*
- ➔ *Trusted Hardware*

# Data Outsourcing

$assurances \subseteq \{query\ correctnes, data\ confidentiality, access\ privacy\}$



# Outsourcing Challenges

## Un-trusted server:

- lazy: incentives to perform less
- curious: incentives to acquire information
- malicious:
  - denial of service
  - incorrect results
  - possibly compromised

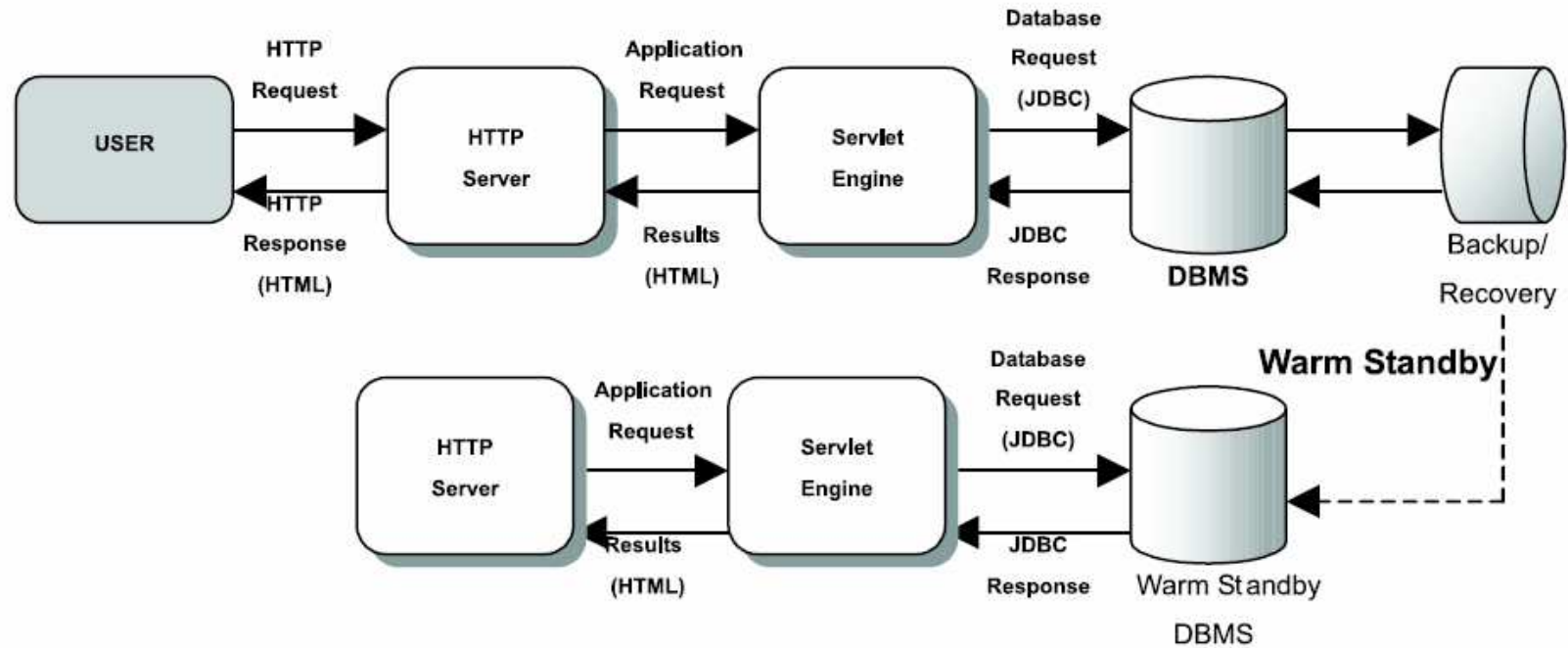
## Why is this hard ?

- how ?
- arbitrary expressivity
- overheads
  - network
  - computational costs

## What do we do ?

- query assurances
- full privacy
  - of queries (even encrypted)
  - of access patterns
- data confidentiality

# Hacigumus (2002)



System architecture of NetDB2

## Stored Data Confidentiality

```
SELECT decrypt(discount, key)
FROM lineitem
WHERE custid = 300
```

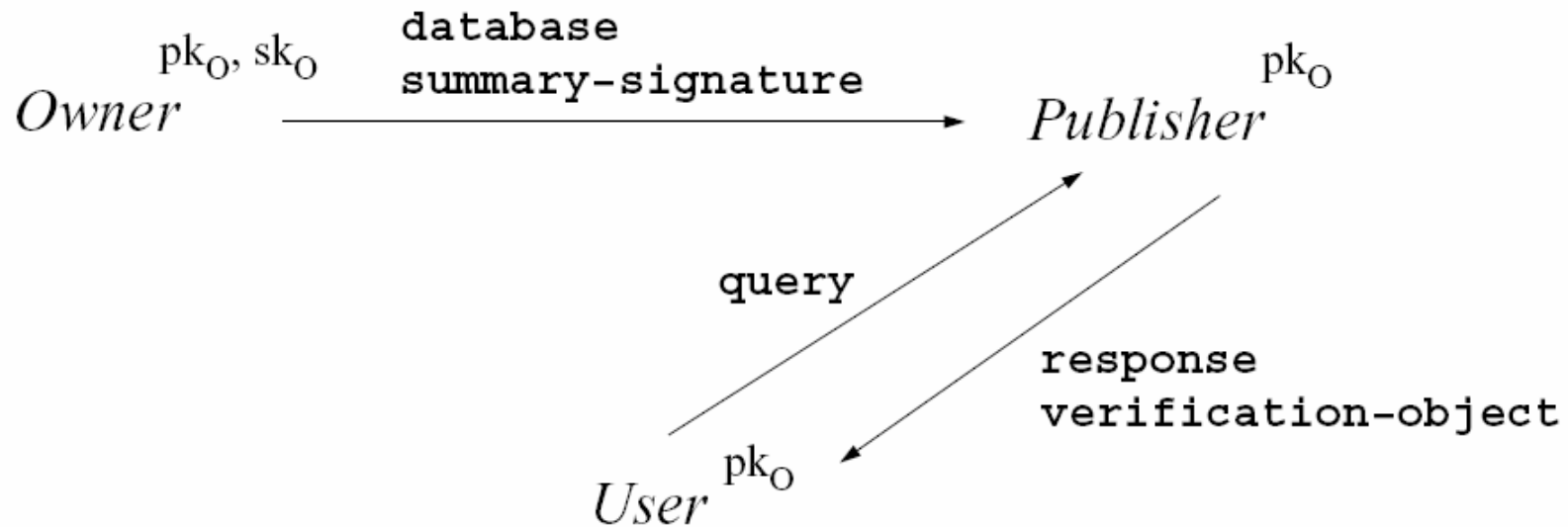
H. Hacigumus, B. R. Iyer, and S. Mehrotra.  
Providing database as a service, ICDE 2002.

- ➔ *Crypto Crash Course*
- ➔ *Data Outsourcing*
- ➔ *Query Correctness*
- ➔ *Data Confidentiality*
- ➔ *Access Privacy*
- ➔ *Searching on Encrypted Data*
- ➔ *Trusted Hardware*

Client requires quantifiable assurances that query results are correct, for arbitrary query types in the presence of a server that could be ...

... lazy

... and/or fully malicious (!)

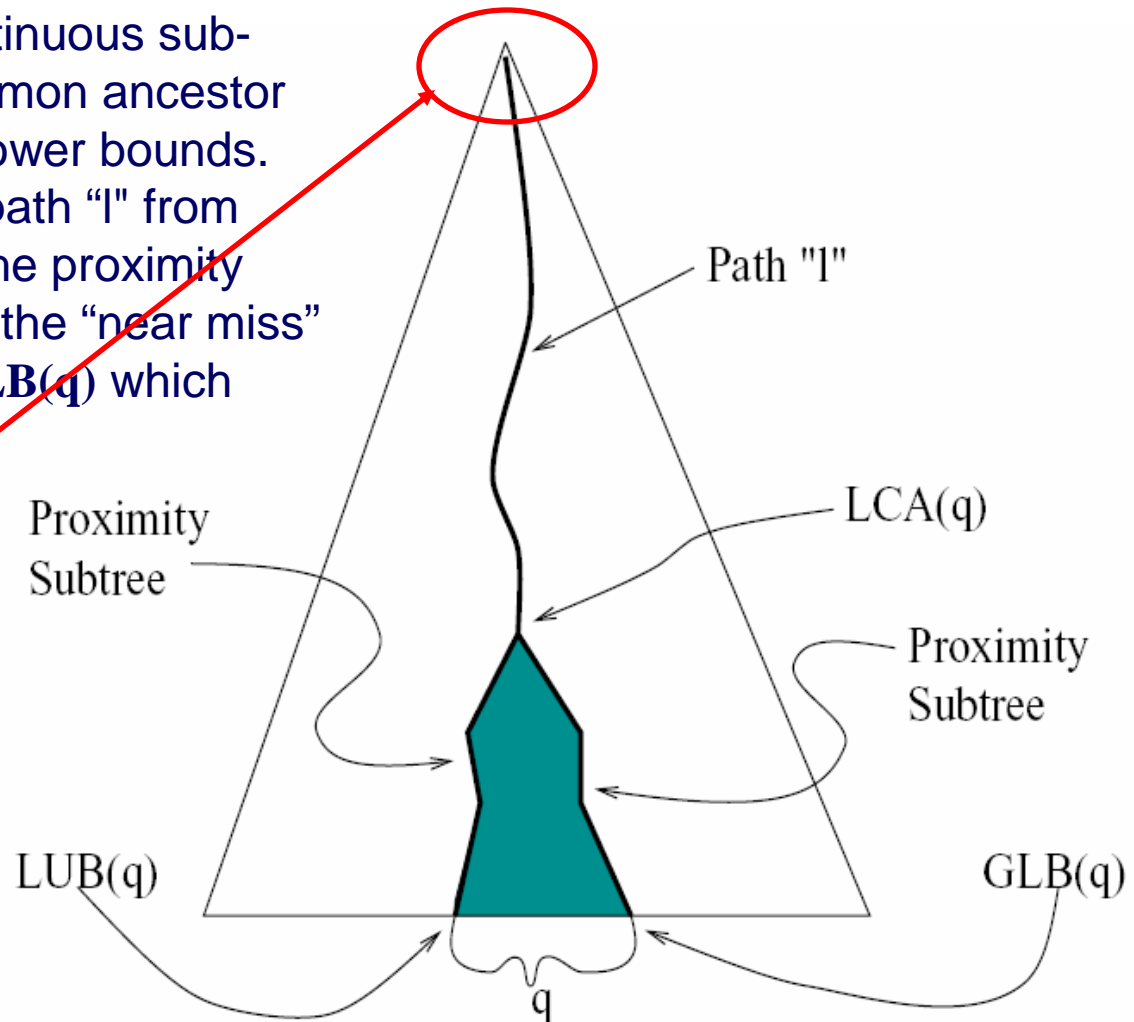


The owner provides database updates and summary signatures to the un-trusted publisher. When users make inquiries with the publisher, they get responses which can be verified using a returned verification-object. Only  $sk_O$  is secret,  $pk_O$  is authenticated.

# Devanbu et. al. (2000)

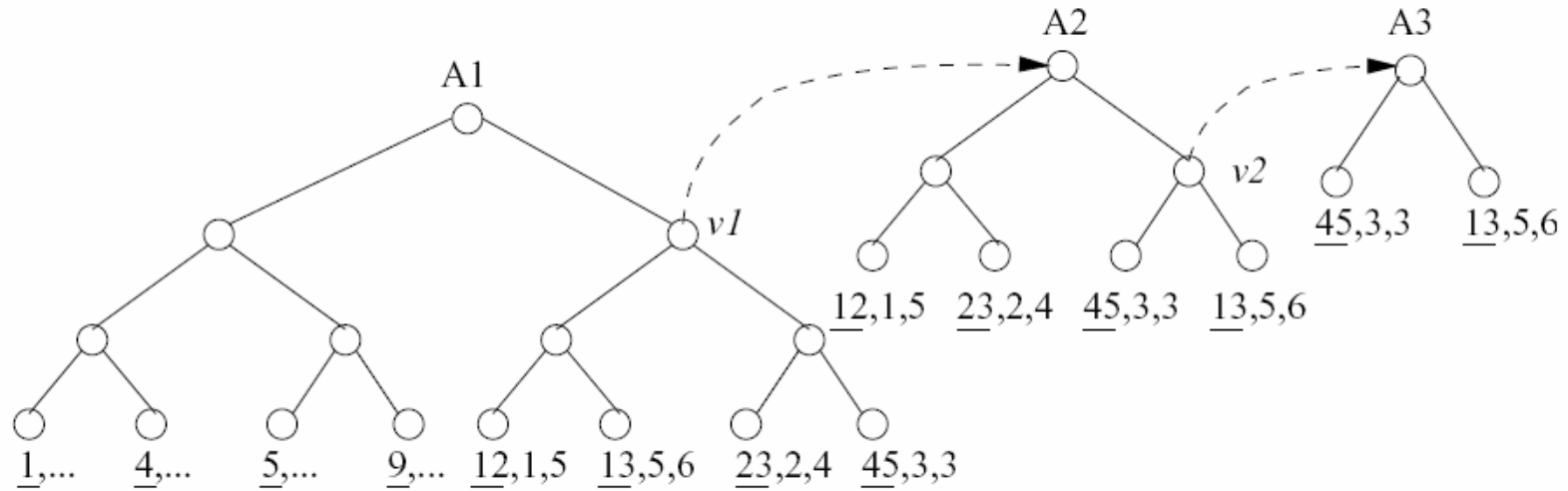
A Merkle tree, with a continuous sub-range  $q$ , with a least common ancestor  $LCA(q)$ , and upper and lower bounds. Note the verifiable hash path "I" from  $LCA(q)$  to the root, and the proximity sub-trees (thick lines) for the "near miss" tuples for  $LUB(q)$  and  $GLB(q)$  which show that  $q$  is complete.

authenticated via signature

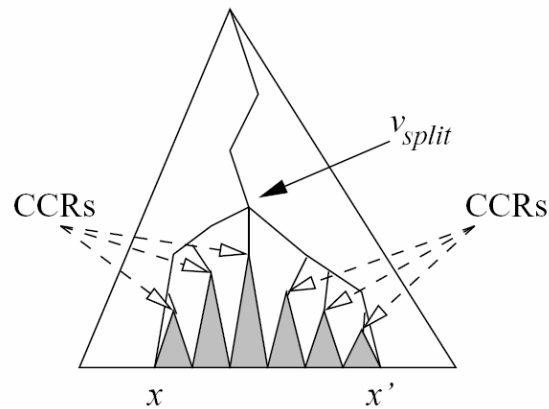


## Supported claimed operations:

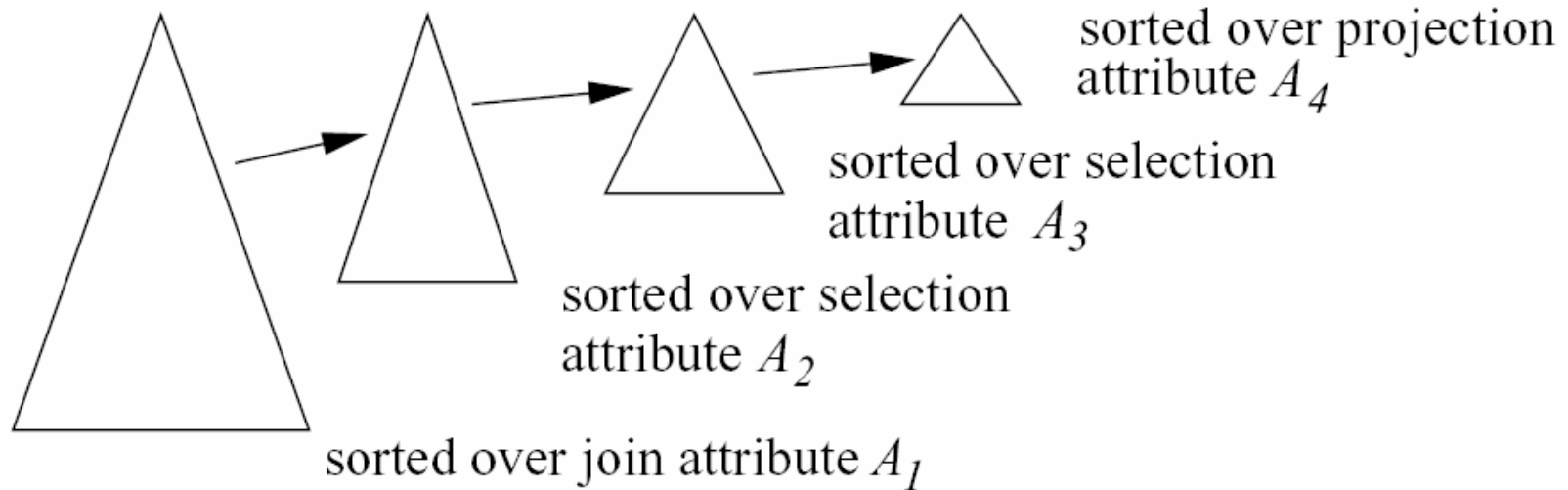
- selections
- projections
  - (1) maintaining VOs before duplicate elimination
  - (2) pre-computing VOs for common projections
- equiJOIN
  - (1) keep materialized cartesian product  $S \times R$ 
    - construct VO on sorted version of product (according to difference (S.A-R.A)) – this yields 3 types of leaf nodes (“0”, “<”, “>”) in Merkle tree
  - (2) all kinds of other tricks
- set operations
  - union (client does it and verifies VOs for input sets)
  - intersection (?)
  - multi-dimensional range queries (generalizing hash tree to “multi-dimensional range tree”)



Excerpt of a 3-dimensional range tree, sorted by attributes  $A_1, A_2$  and  $A_3$



**Covering canonical roots (CCR):**  
 roots of the canonical sub-trees  
 precisely covering the leaves  
 with values in the interval.



```
SELECT S.A4 FROM S,R  
WHERE S.A1=R.A1 AND A2<10 AND A3>17
```

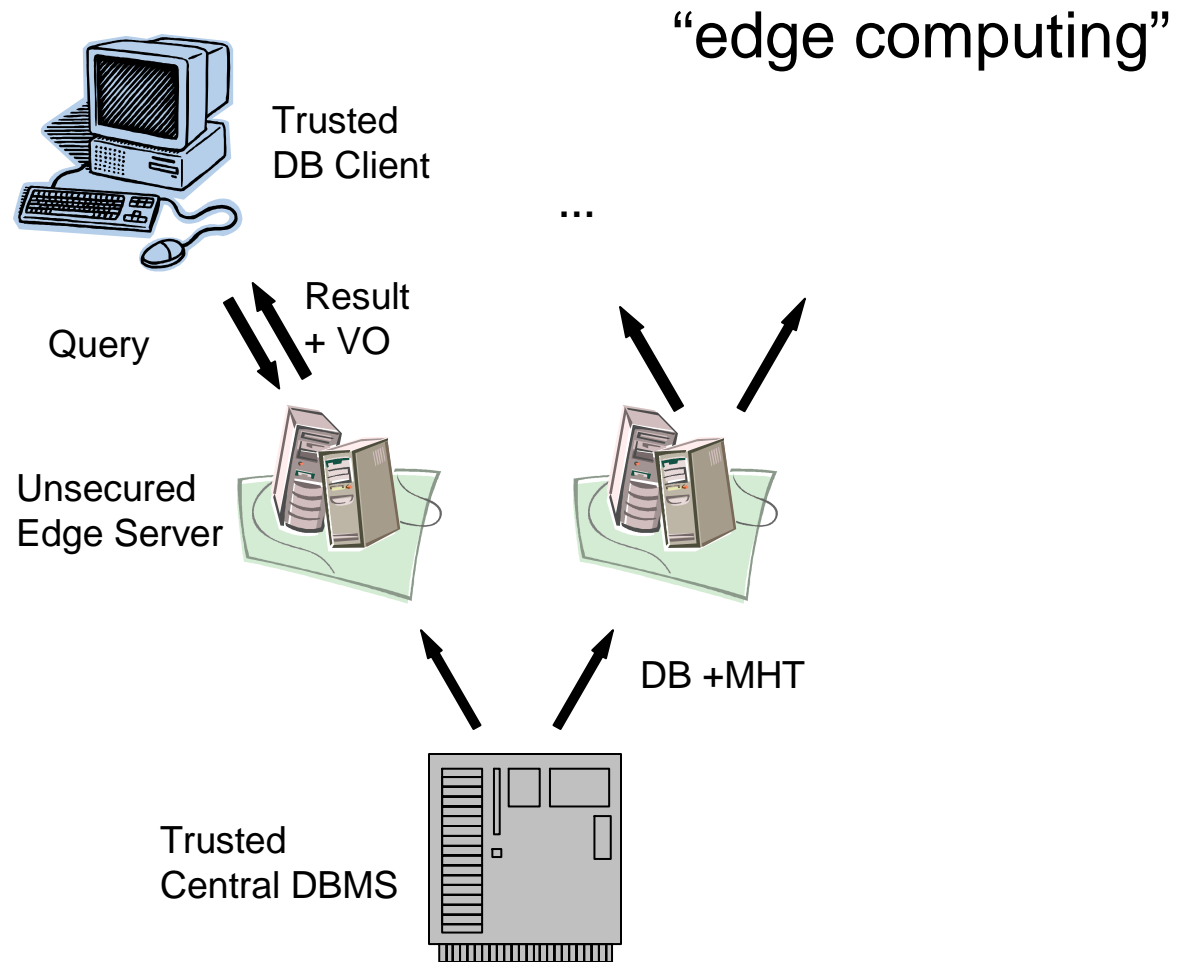
### Issues:

- query expressiveness
- query flexibility
  - works only on data with VOs
- “universe split” phenomenon
  - use timestamps, expiration times
- expensive operations (!)

Discusses the use of batch verification of signatures and similar techniques (condensed RSA) to authenticate results.

		Condensed-RSA	Batch-DSA	BGLS
Sign	1 signature	6.82	3.82	3.54
Verify	1 signature	0.16	8.52	62
	t = 1000, k = 1	44.12	1623.59	184.88
	t = 100, k = 10	45.16	1655.86	463.88
	t = 1000, k = 10	441.1	16203.5	1570.8

Cost comparison (in msec): verification and signing. Notation: t – # signatures, k – # signers



### Claimed problems with [Devanbu 2000]

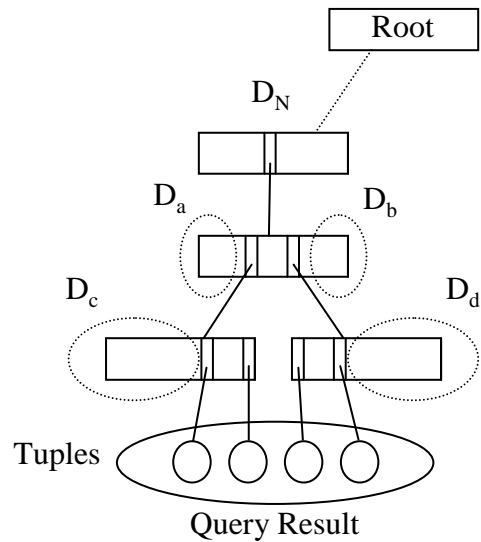
- A hash tree is needed for every sort-order
- VOs need to contain links all the way to the root,
  - VOs grow linearly to query result and logarithmic to base table size
- Projections may have to be performed by clients
- No provision for dynamic updates on the database

Aim 1: VO size just linear in query result

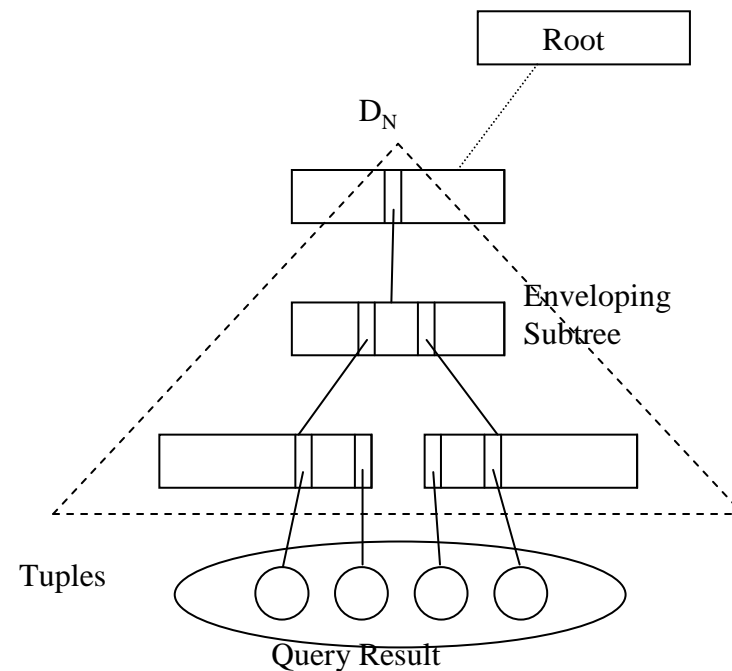
Aim 2: do not push projections to client

### Idea: use different hash function

- $h(x) = g^x \text{ mod } q$
- $h$  is commutative,  $h(x+y) = h(y+x)$ 
  - Digests can be combined arbitrarily
  - Projection can be performed at the edge servers
  - Facilitates insertion of new tuples with minimal effect on other digests
- but: significantly (1000-10000 times) slower
- trade-off: computation vs. communication



Verification object =  $D_N + D_S$   
 where  $D_S = \{D_a, D_b, D_c, D_d\}$



**Verifying Selection**  
 (no need to go up to the root  
 as everything is also signed)

---

Similar expressiveness. But ...

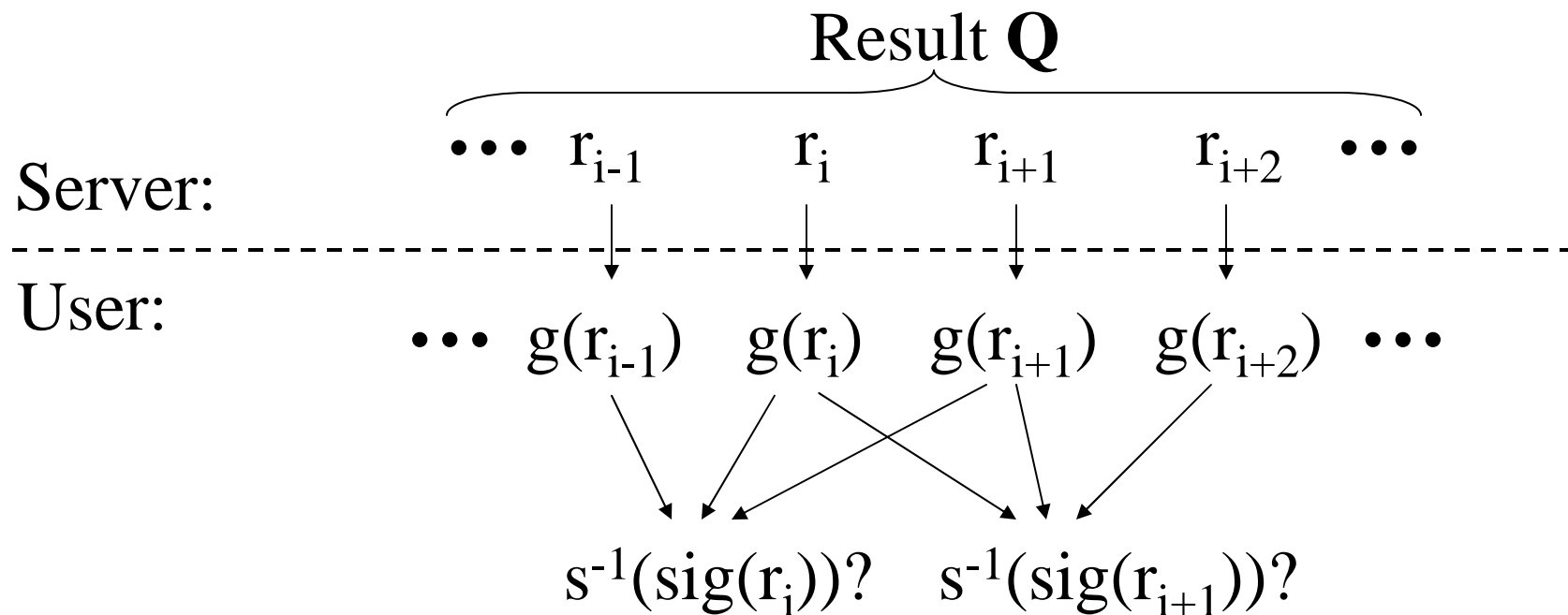
Asks: what about access control rules ?  
(Devanbu seems to reveal too much: boundary tuples)

Also claims: lower overheads for queries  
and updates.

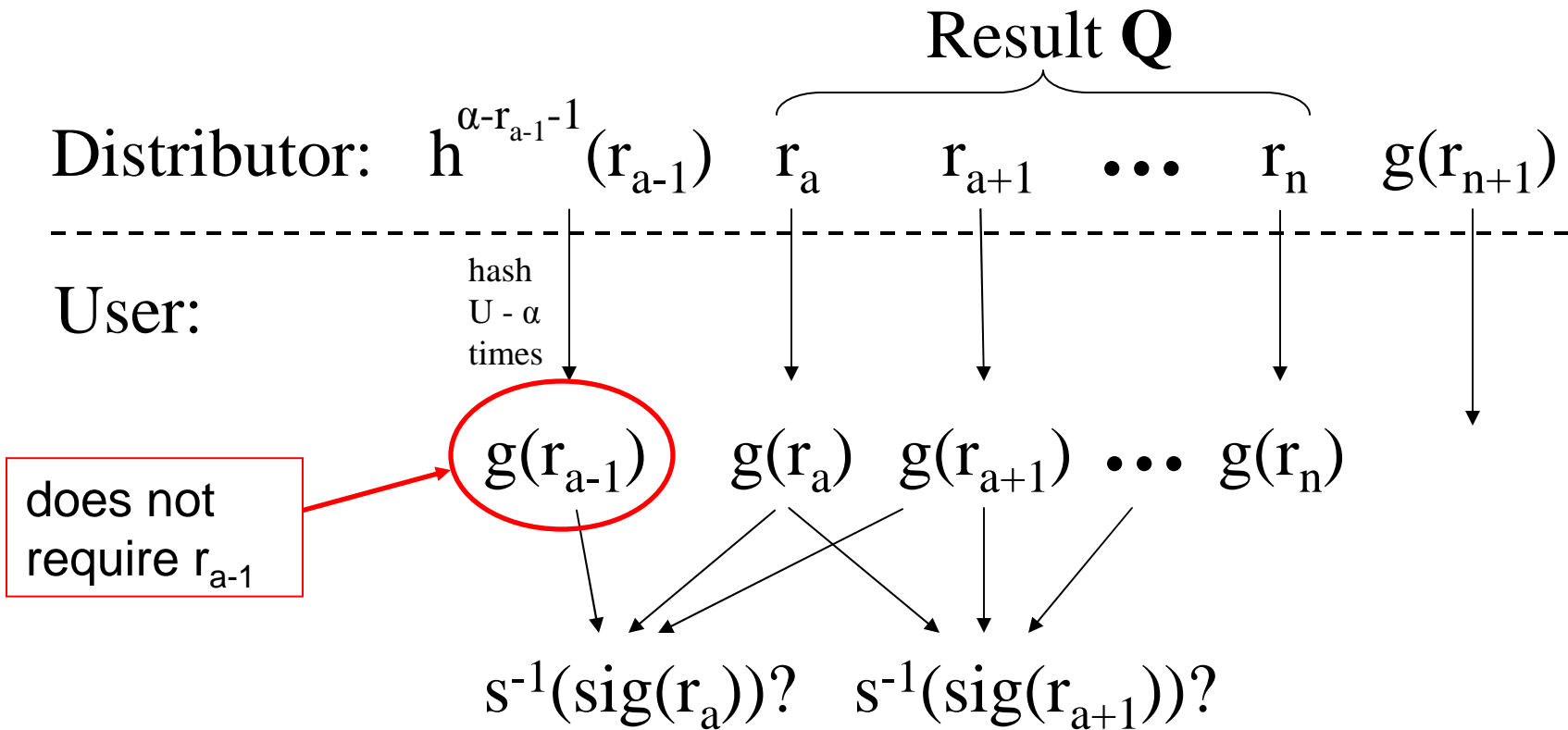
Introduces “precision” (only data  
matching the query should be returned)

Idea: use signature chains – thus no need to reveal boundary elements.

$$\text{sig}(r_i) = s(\text{h}(g(r_{i-1}) \mid g(r_i) \mid g(r_{i+1})))$$



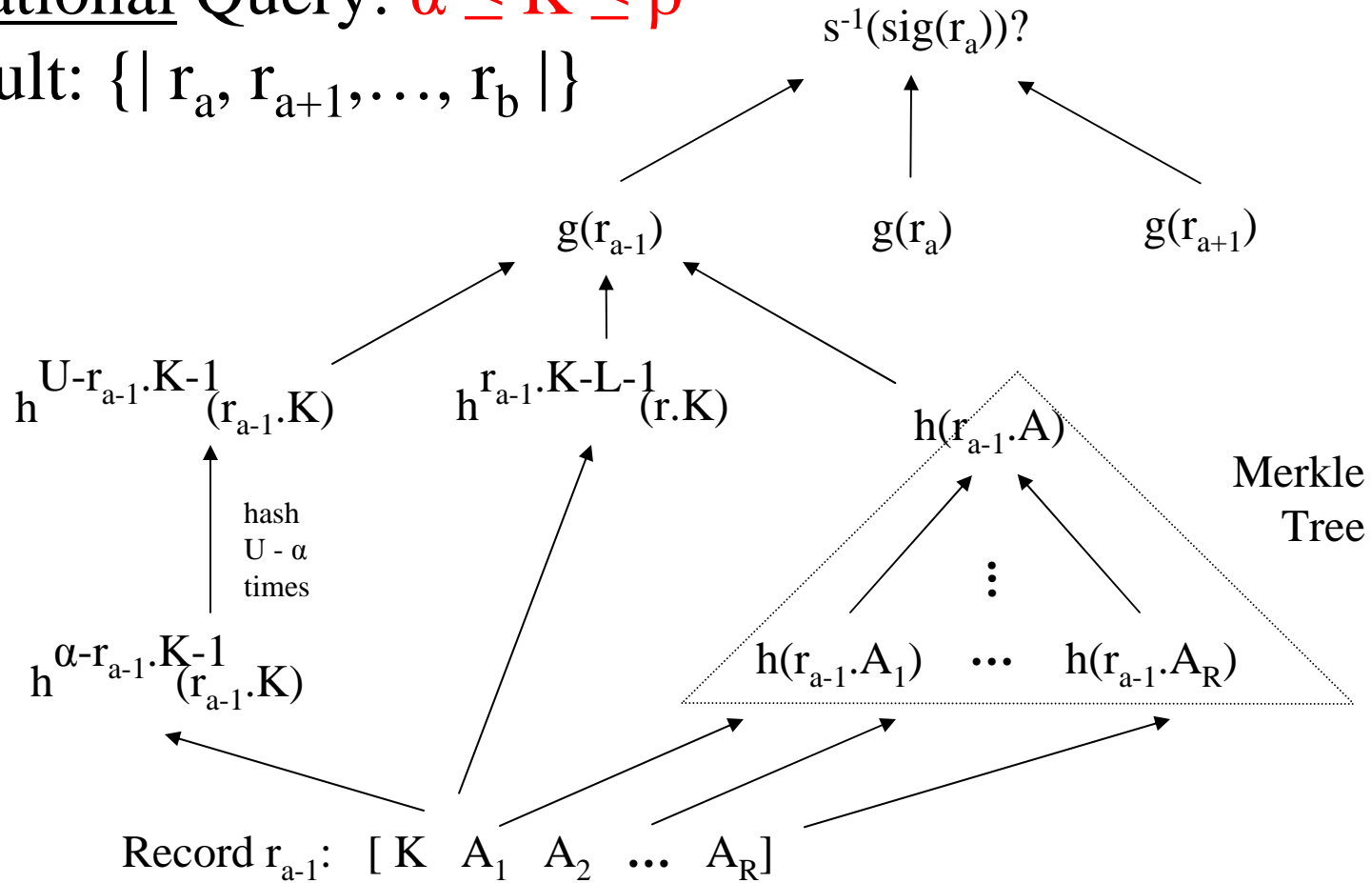
But what is  $g$ :  $g(r) = h^{U-r-1}(r)$



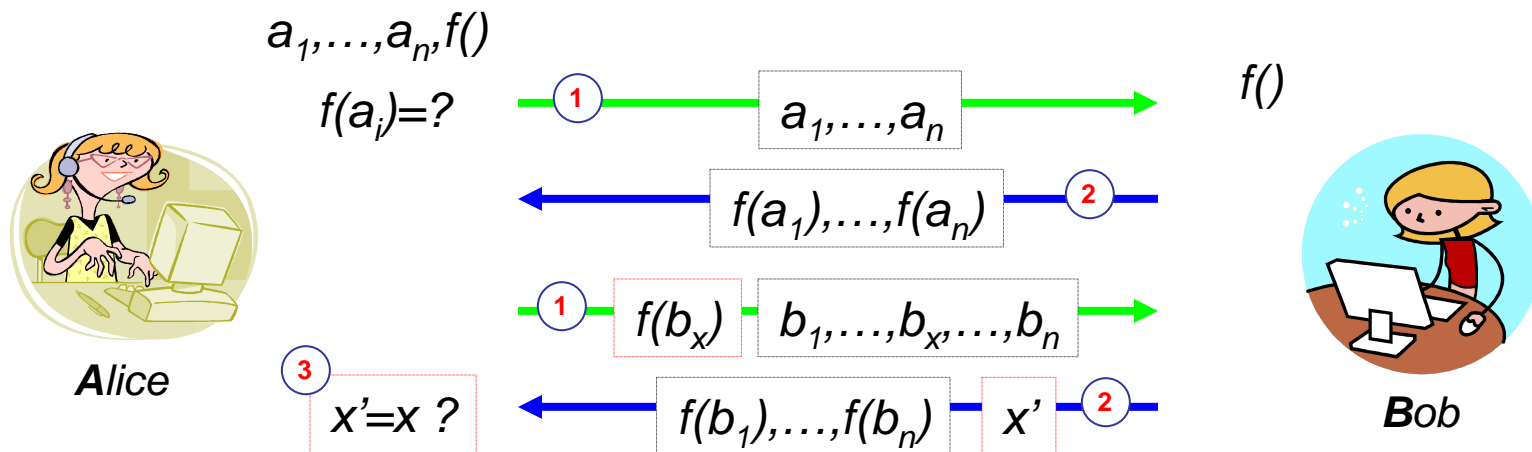
Query:  $\alpha \leq r$

Relational Query:  $\alpha \leq K \leq \beta$

Result:  $\{ | r_a, r_{a+1}, \dots, r_b | \}$

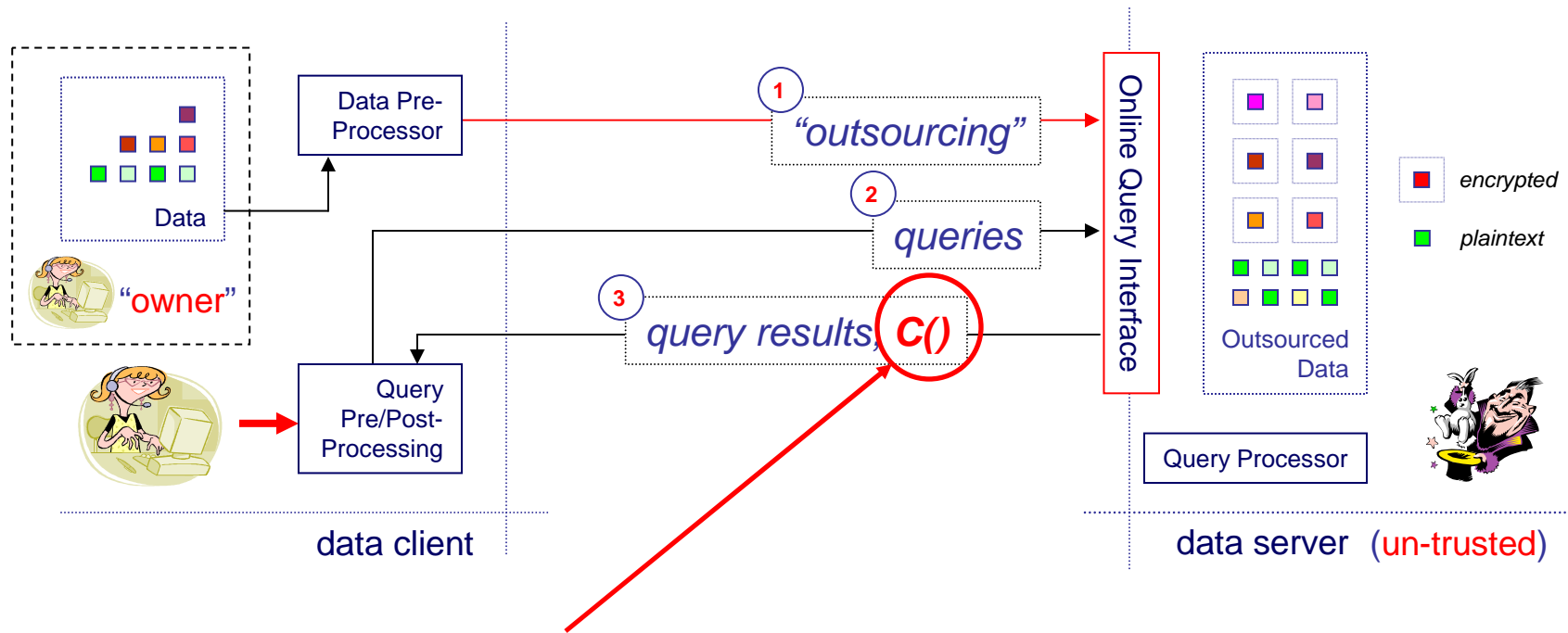


## Asks: What about arbitrary queries ?



P. Golle and I. Mironov, "Uncheatable Distributed Computations", RSA 2001 (Cryptographer's track)

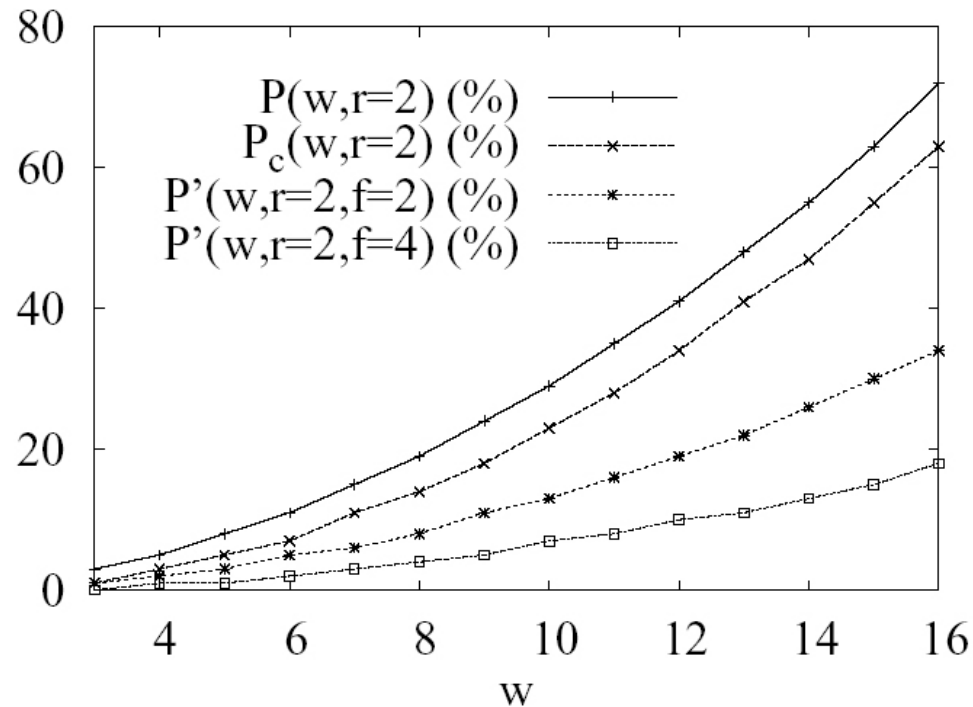
# Sion: Execution Proofs



$$C(Q, x, \epsilon) = \{H(\epsilon || \rho(Q_x)), \epsilon\}$$

A challenge token (computed by client) is sent together with the batch of queries. Upon return, batch execution is proved if  $x=x'$ .

# Sion: Cheating Probability

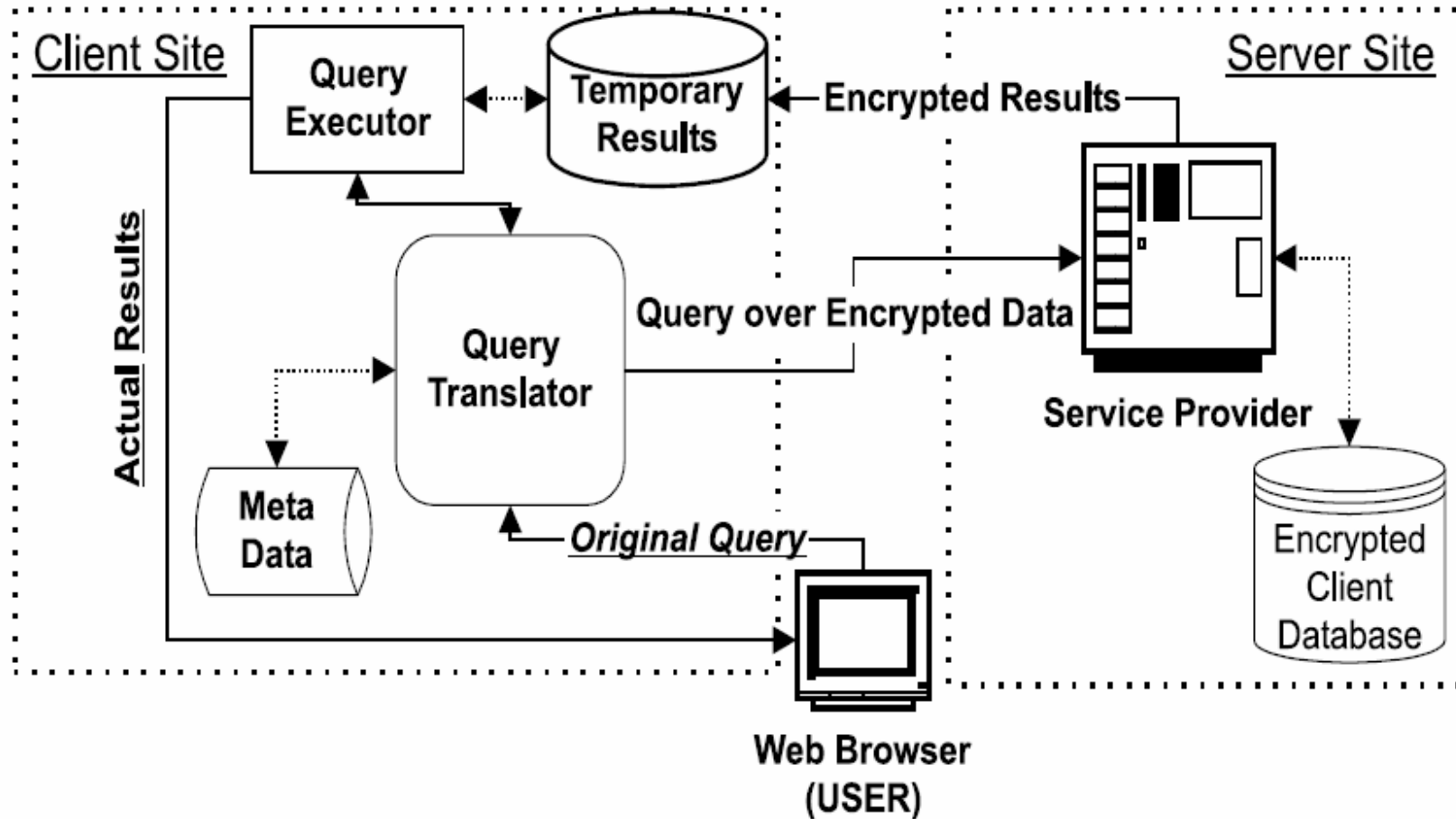


Only handles lazy server !

The behavior of  $P'(w, r, f)$  (fake tokens) plotted against  $P_c(w, r)$  (client-side result checking mechanism) showing that the query execution proof mechanism (with fake tokens) significantly decreases the ability to “get away” with less work.

- ⇒ *Crypto Crash Course*
- ⇒ *Data Outsourcing*
- ⇒ *Query Correctness*
- ⇒ *Data Confidentiality*
- ⇒ *Access Privacy*
- ⇒ *Searching on Encrypted Data*
- ⇒ *Trusted Hardware*

# Hacigumus (SIGMOD 2002)



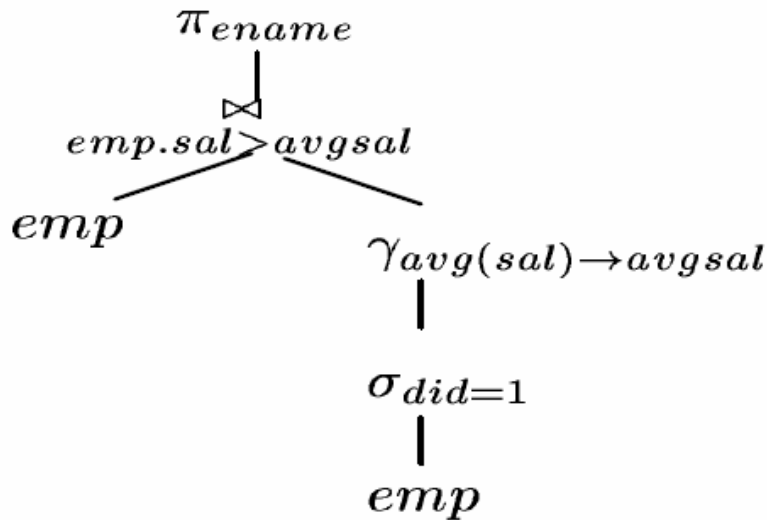
## Main Steps:

1. Partition sensitive domains
  - Order preserving: supports comparison
  - Random: query rewriting becomes hard
2. Rewrite queries to target partitions
3. Execute queries and return results
4. Prune/post-process results on client

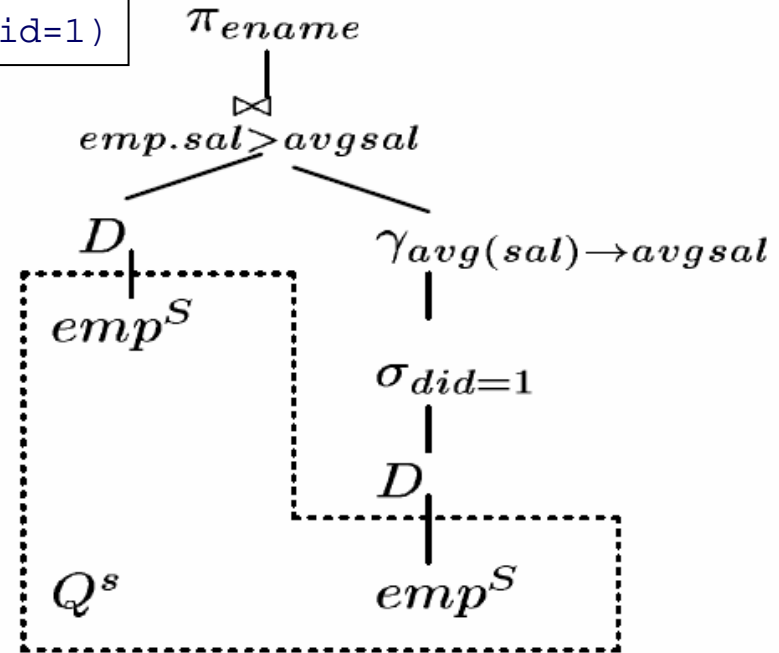
# Hacigumus (SIGMOD 2002)

```

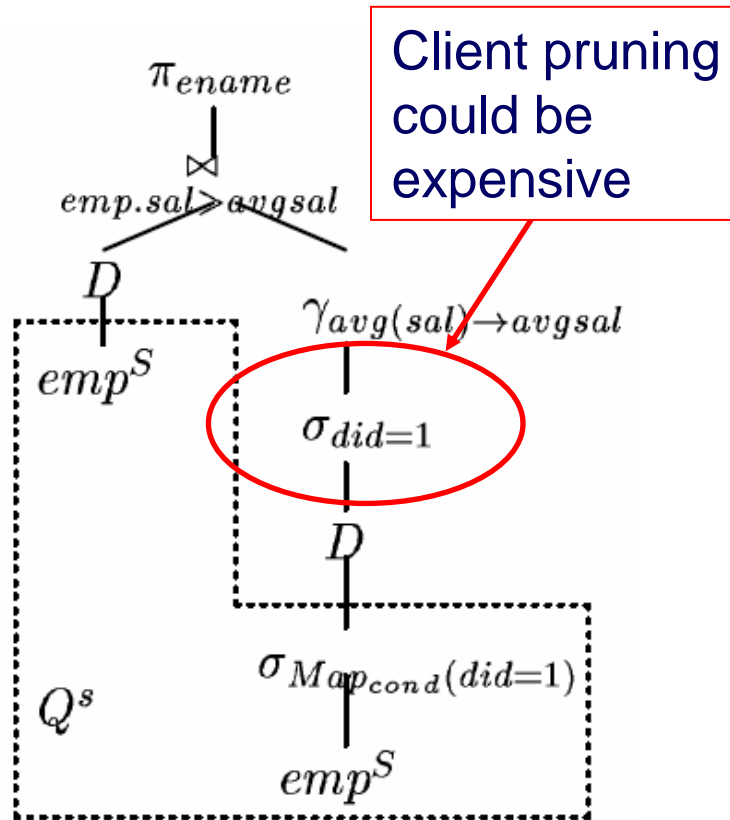
SELECT emp.name FROM emp
WHERE emp.salary >
  (SELECT AVG(salary) FROM emp WHERE did=1)
    
```



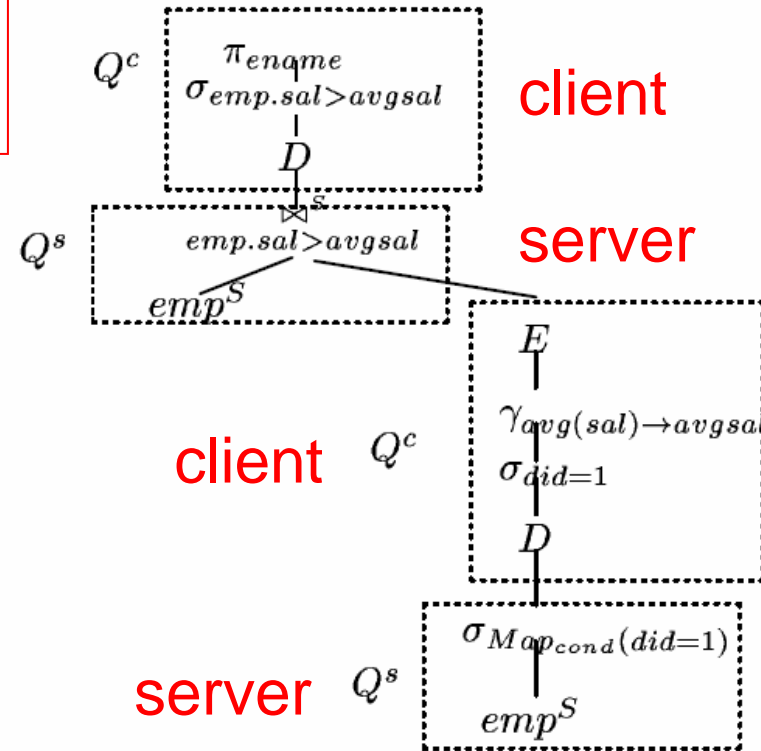
(a) Original query tree.



(b) Replacing encrypted relations.



(c) Doing selection at server.



(d) Multiple interactions between Client and Server.

## Confidentiality-Overhead Trade-off

Larger segments ==  
increased privacy ==  
increased overheads

Goal: For a uniform distribution of queries  
- minimize any leaks to any adversaries  
(even) knowing segmentation parameters.

Idea 1: Maximize variance of distribution  
of values in segment

Idea 2: Increase segment entropy

Issue: What about performance ?

### Solution: “Controlled Diffusion”

#### Idea:

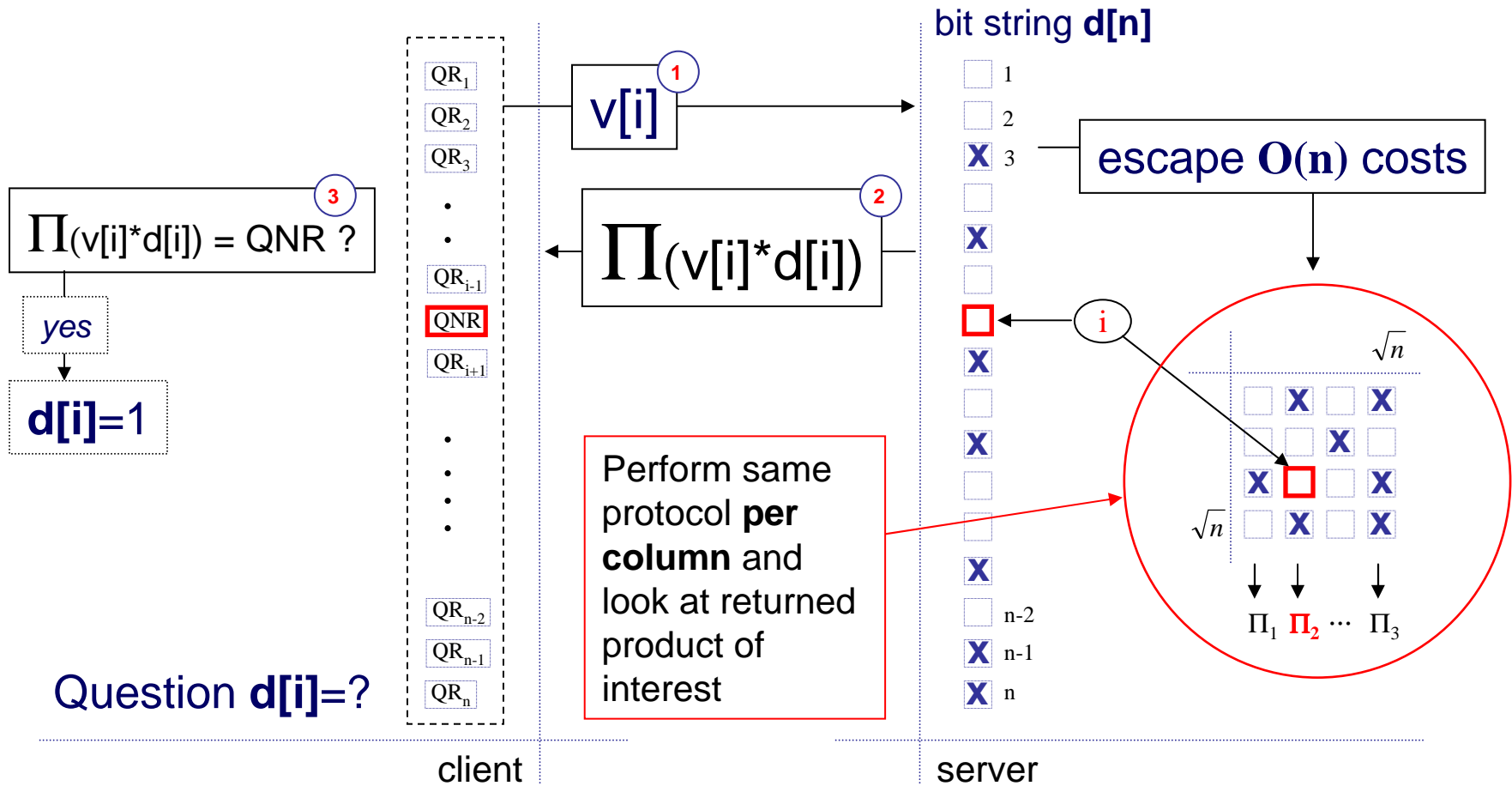
1. design for efficiency, then ...
2. ... diffuse (re-distribute) elements inside the segments to increase per-segment entropy and variance

Asks: Similarly, how to structure query trees to optimally balance the security-efficiency trade-off in [Hacigumus 2002].

Idea: client generates optimal partitioned query execution plans given statistics and metadata input from the server.

- ➔ *Crypto Crash Course*
- ➔ *Data Outsourcing*
- ➔ *Query Correctness*
- ➔ *Data Confidentiality*
- ➔ *Access Privacy*
- ➔ *Searching on Encrypted Data*
- ➔ *Trusted Hardware*

# QR PIR



The  $n$  bits of the database are organized logically at the server as a bi-dimensional matrix  $M$  of size  $\sqrt{n} \times \sqrt{n}$ . To retrieve bit  $M(x, y)$  with computational privacy, the client:

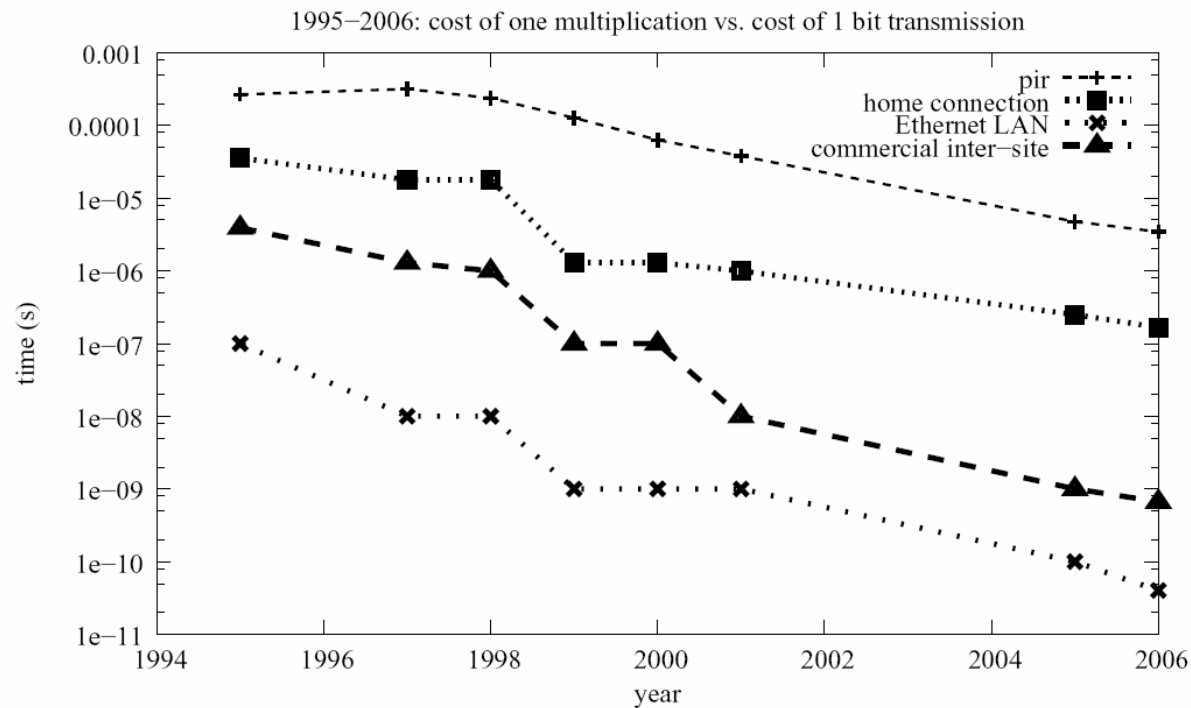
- randomly chooses two prime numbers  $p$  and  $q$  of similar bit length, computes their product,  $N = pq$  and sends it to the server.
- generates  $\sqrt{n}$  numbers  $s_1, s_2, \dots, s_{\sqrt{n}}$ , such that  $s_x$  is a quadratic non-residue (QNR) and the rest are quadratic residues (QR) in  $\mathbb{Z}_N^*$ .
- sends  $s_1, s_2, \dots, s_{\sqrt{n}}$  to the server.

For each “column”  $j \in (1, \sqrt{n})$  in the  $\sqrt{n} \times \sqrt{n}$  matrix, the server:

- computes the product  $r_j = \prod_{0 < i < \sqrt{n}} q_{ij}$  where  $q_{ij} = s_i^2$  if  $M(i, j) = 1$  and  $q_{ij} = s_i$  otherwise <sup>2</sup>.
- sends  $r_1, \dots, r_{\sqrt{n}}$  to the client

The client then simply checks if  $r_y$  is a QR in  $\mathbb{Z}_N^*$  which implies  $M(x, y) = 1$ , else  $M(x, y) = 0$ .

# PIR is (still) impractical

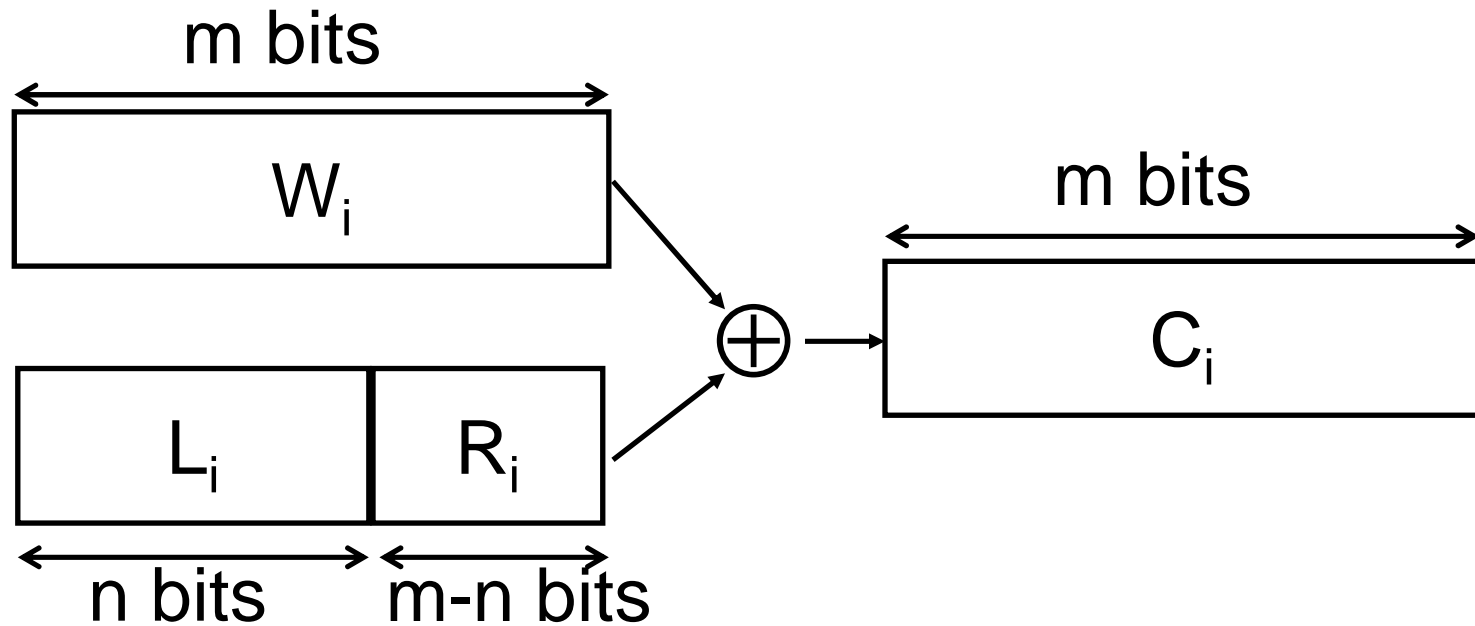


Comparison between the time required to perform PIR and the time taken to transfer the database, between 1995 and 2005. (logarithmic)

- ➔ *Crypto Crash Course*
- ➔ *Data Outsourcing*
- ➔ *Query Correctness*
- ➔ *Data Confidentiality*
- ➔ *Access Privacy*
- ➔ *Searching on Encrypted Data*
- ➔ *Trusted Hardware*

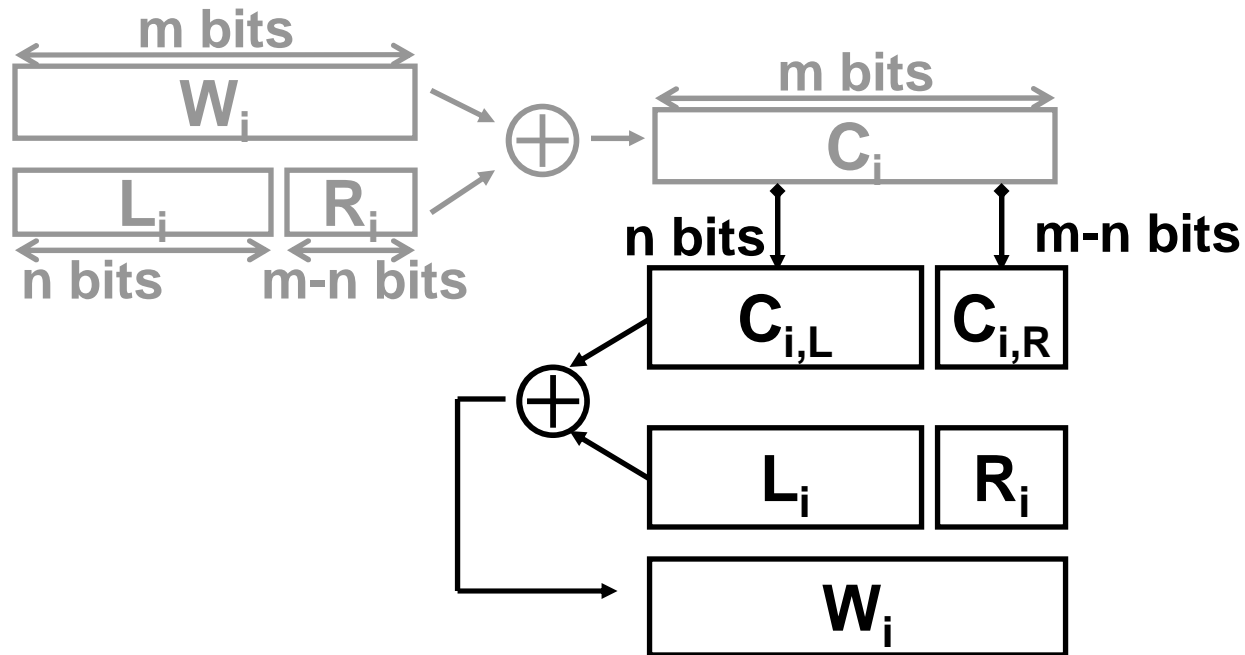
- Sequential Scan
- Index-based

## Encryption:



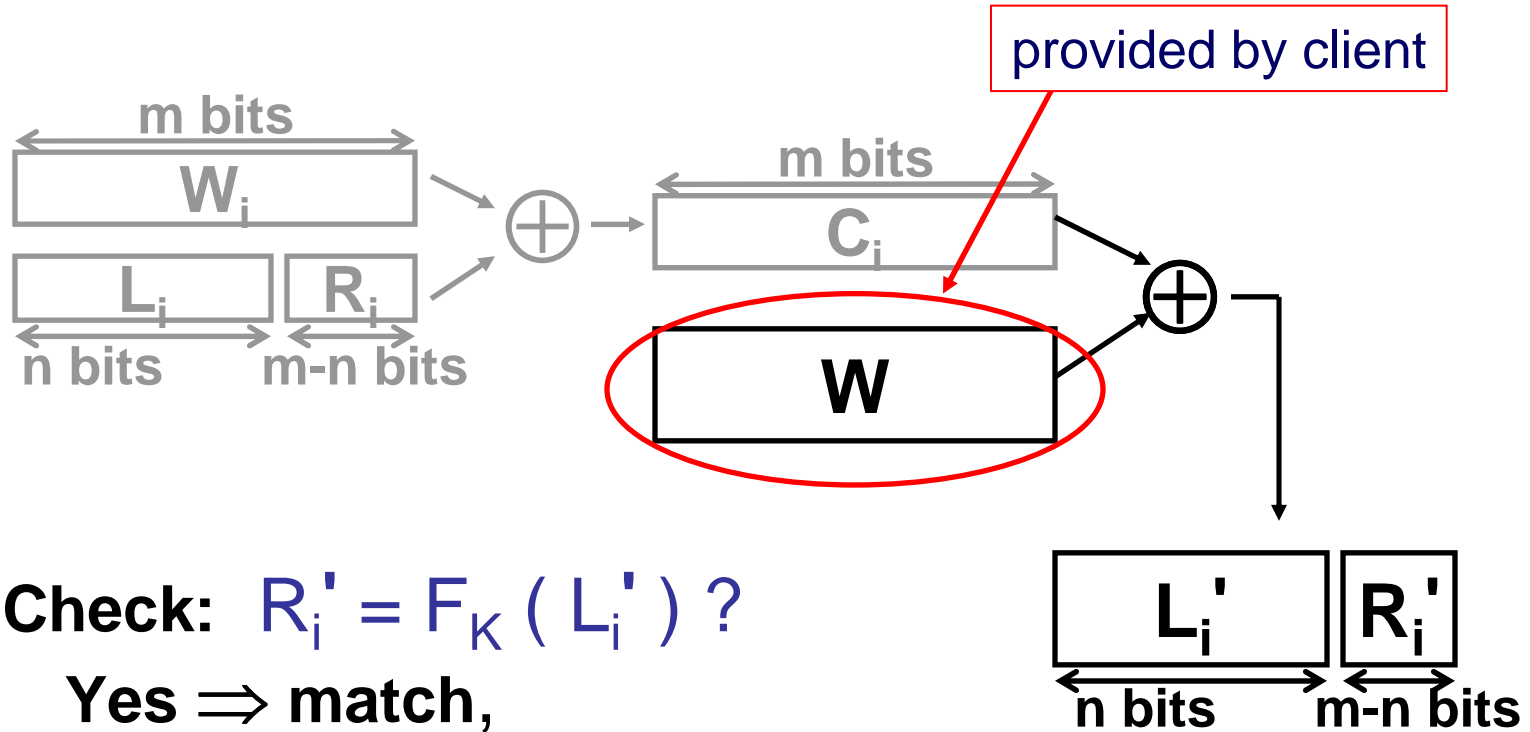
$$L_i \leftarrow G_i(\text{seed}), \quad R_i \leftarrow F_K(L_i)$$

## Decryption:



$$L_i \leftarrow G_i(\text{seed}), \quad R_i \leftarrow F_K(L_i)$$

## Search:

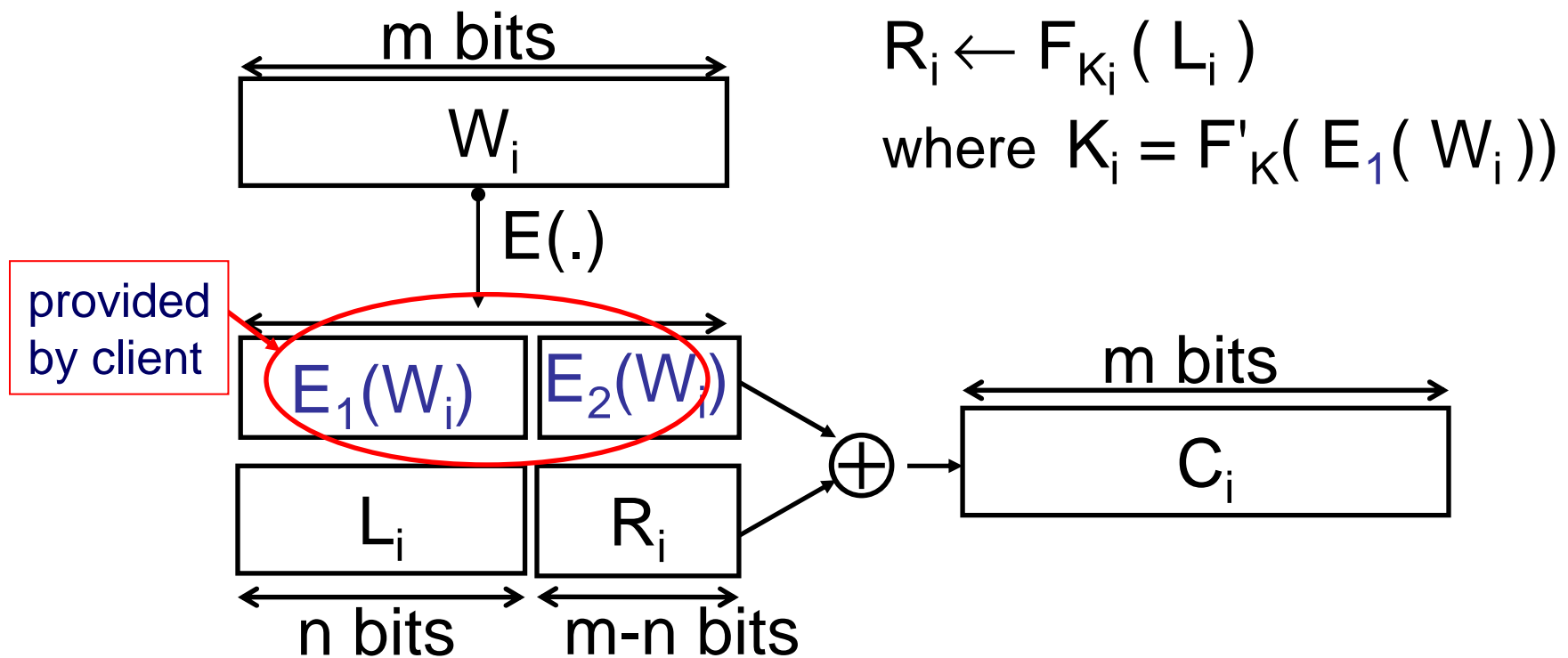


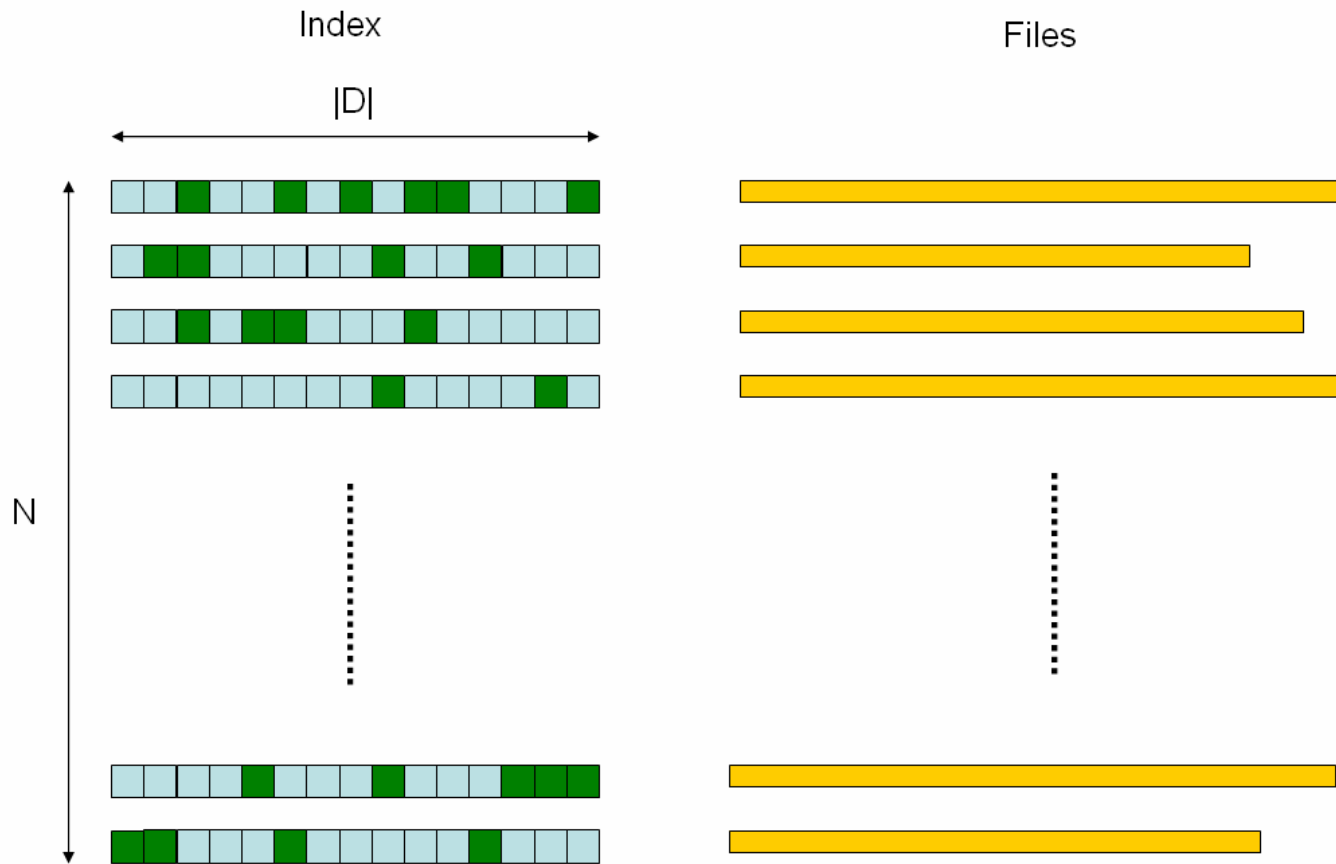
Check:  $R'_i = F_K(L'_i)$  ?

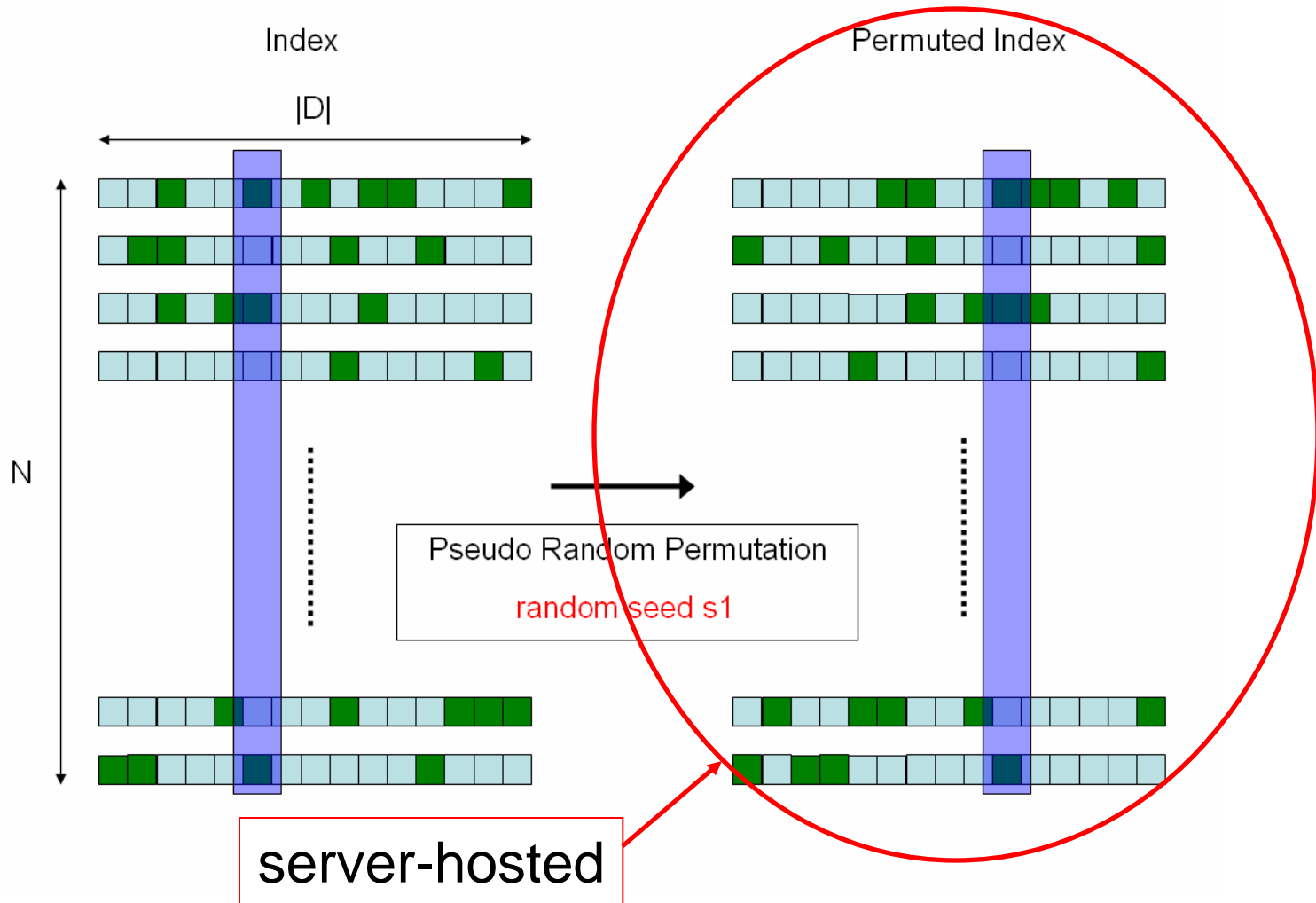
Yes  $\Rightarrow$  match,

( false positive rate =  $1 / 2^{m-n}$  )

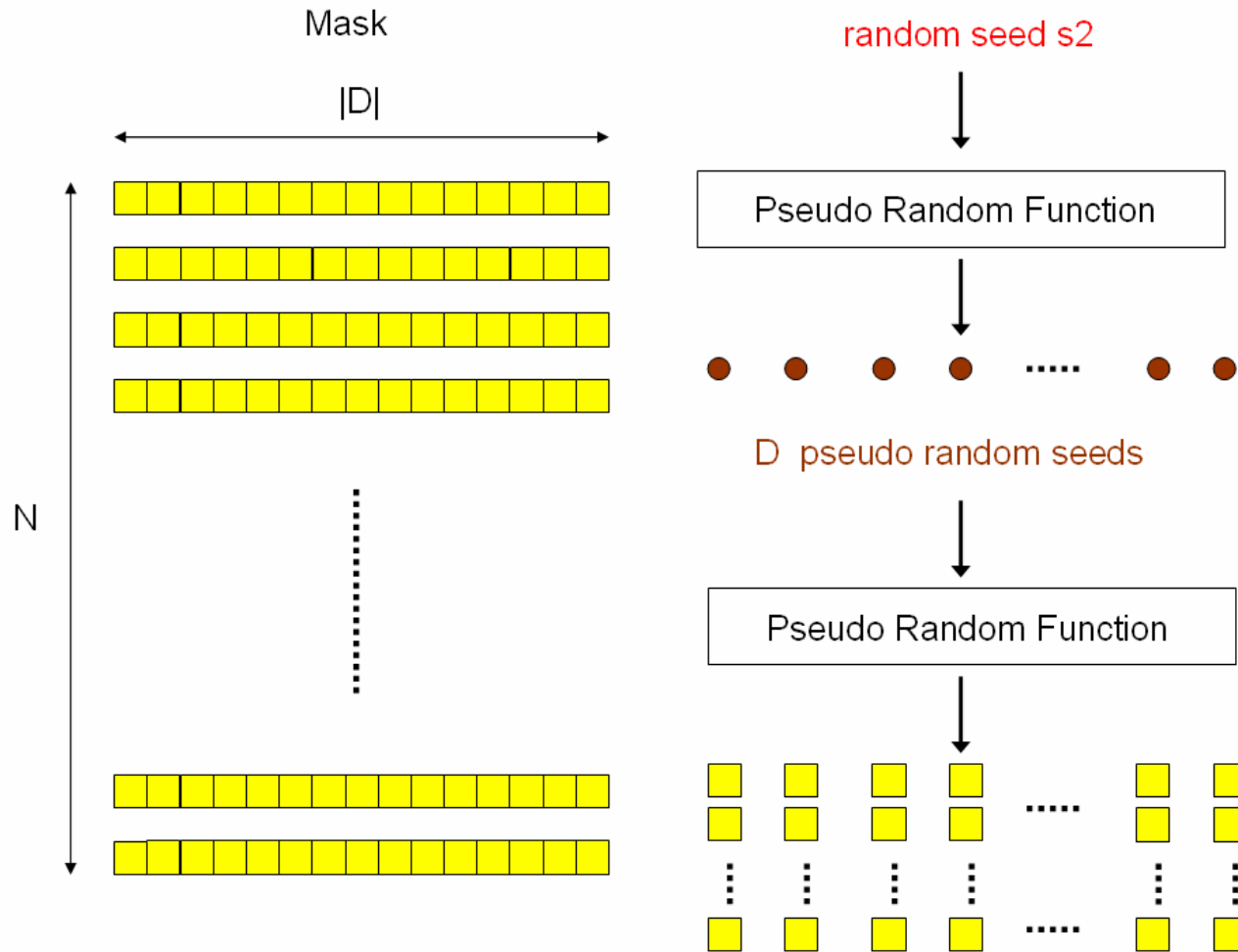
## “Hidden” Search:

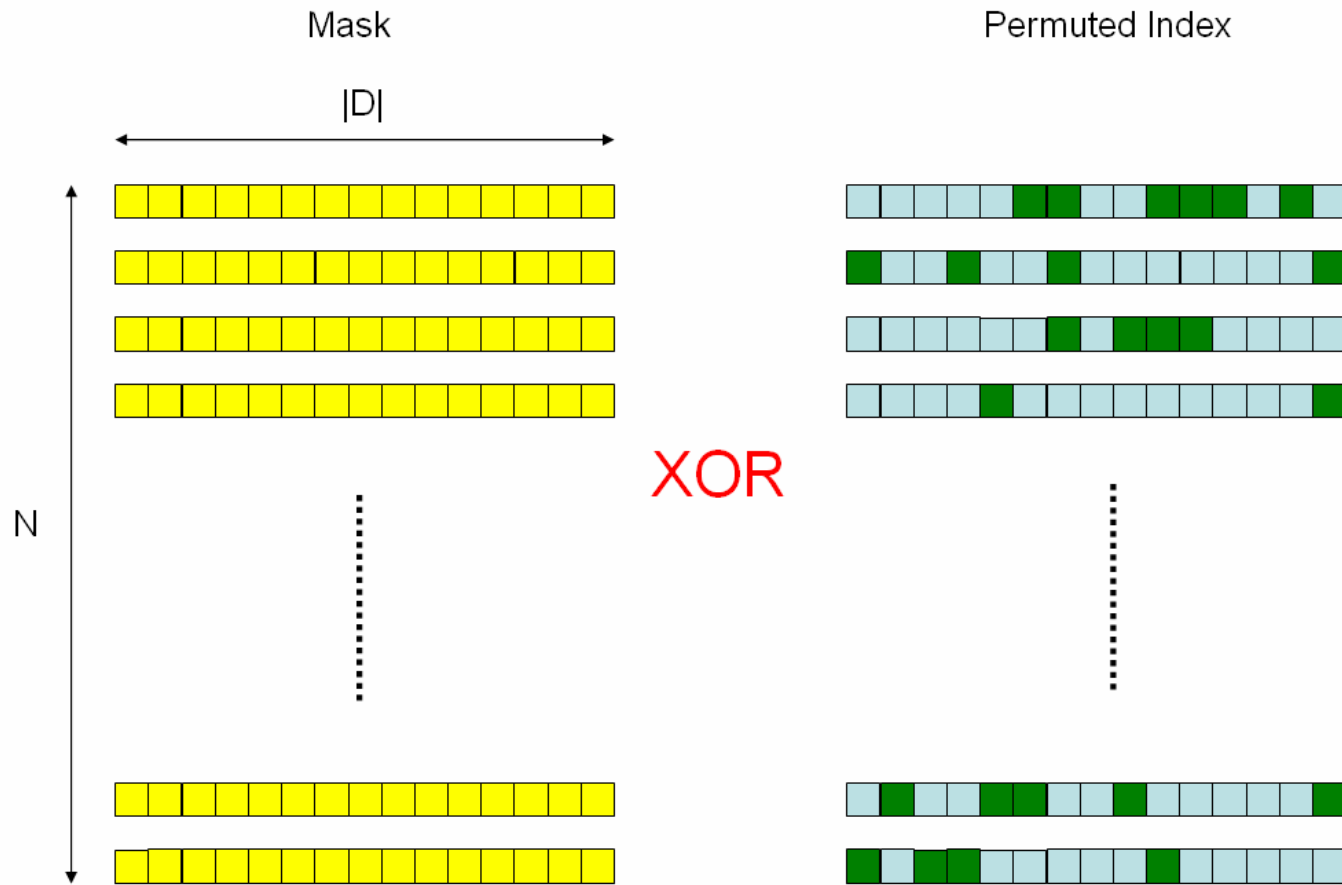


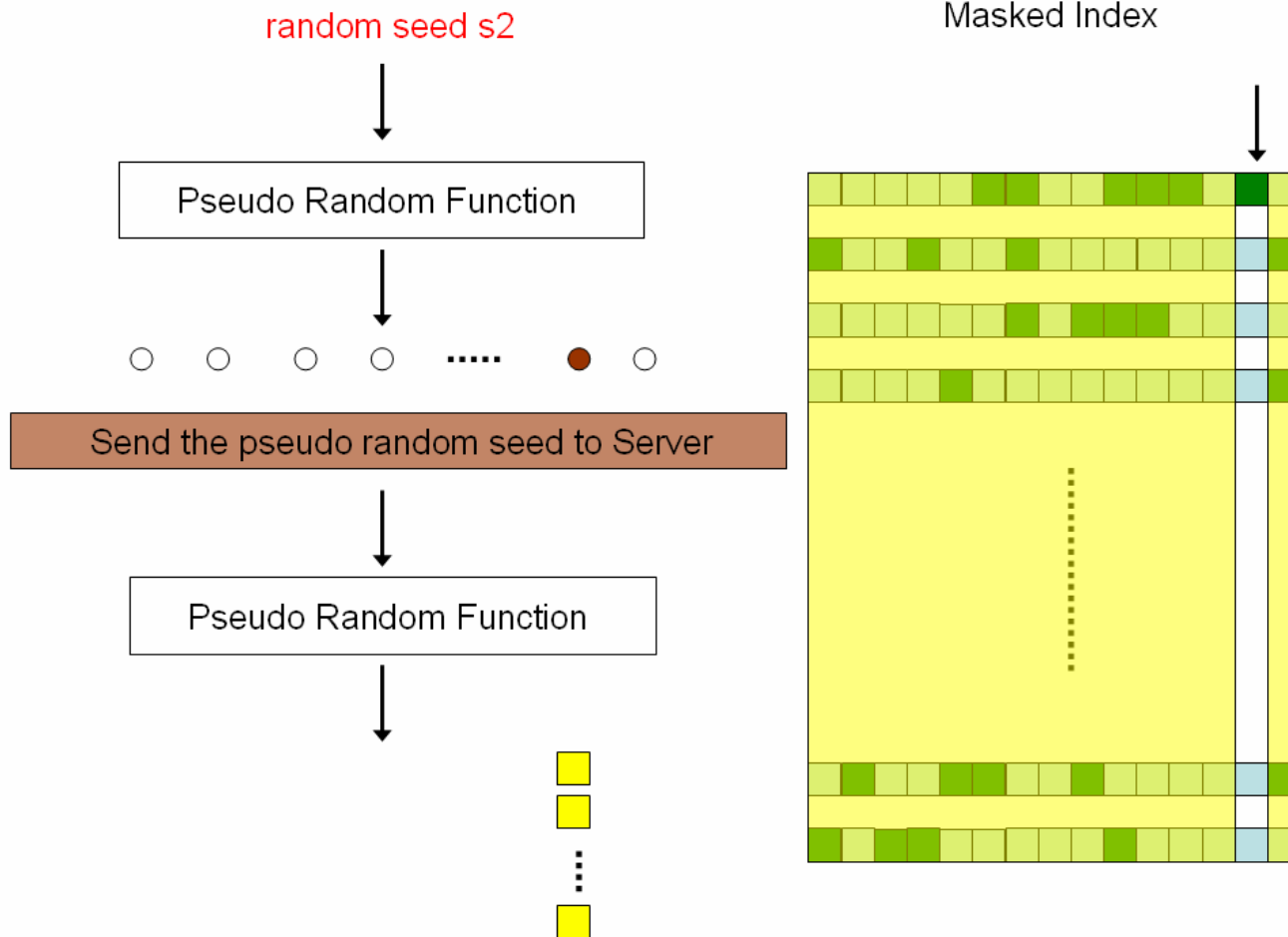




# Chang (2004)







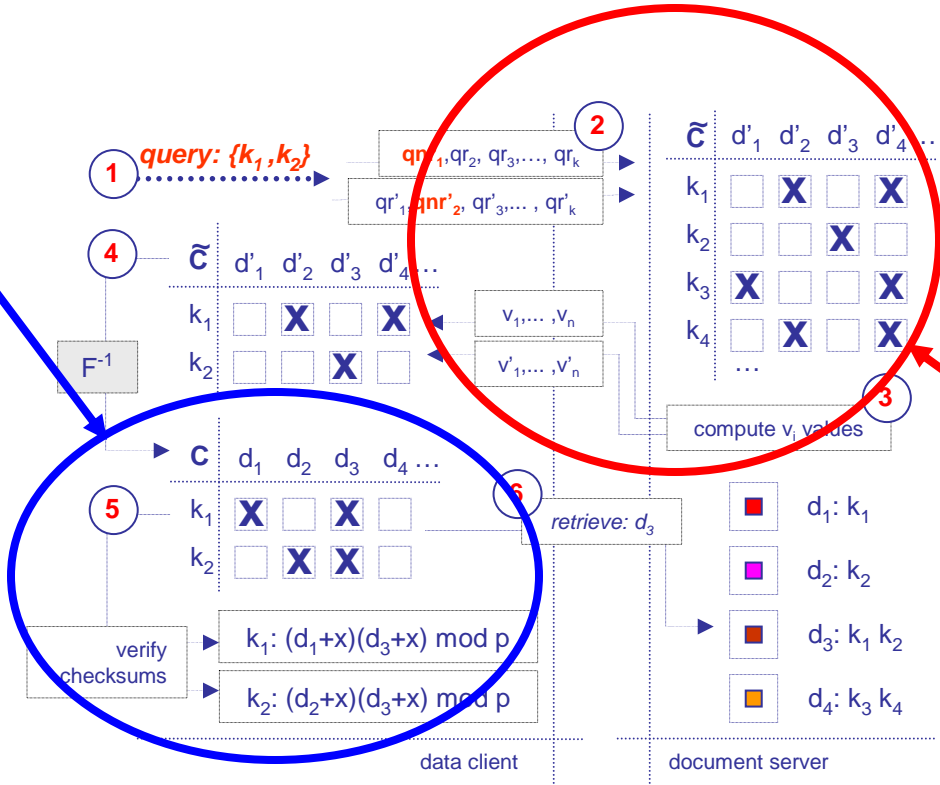
Server stores **capabilities** for conjunctive queries (linear in the total number of documents). These can be transferred offline.

The client is required to know before-hand future conjunctive queries.

**Query** part is sent online at the time of search. It is of constant size (number of keyword fields per documents).

## Asks: What about correctness + privacy ?

Query Correctness



Computational Privacy

Idea: Deploy modified version of computational PIR targeted at a server-side index. Augment with “multiplicative checksums”.

- ➔ *Crypto Crash Course*
- ➔ *Data Outsourcing*
- ➔ *Query Correctness*
- ➔ *Data Confidentiality*
- ➔ *Access Privacy*
- ➔ *Searching on Encrypted Data*
- ➔ *Trusted Hardware*

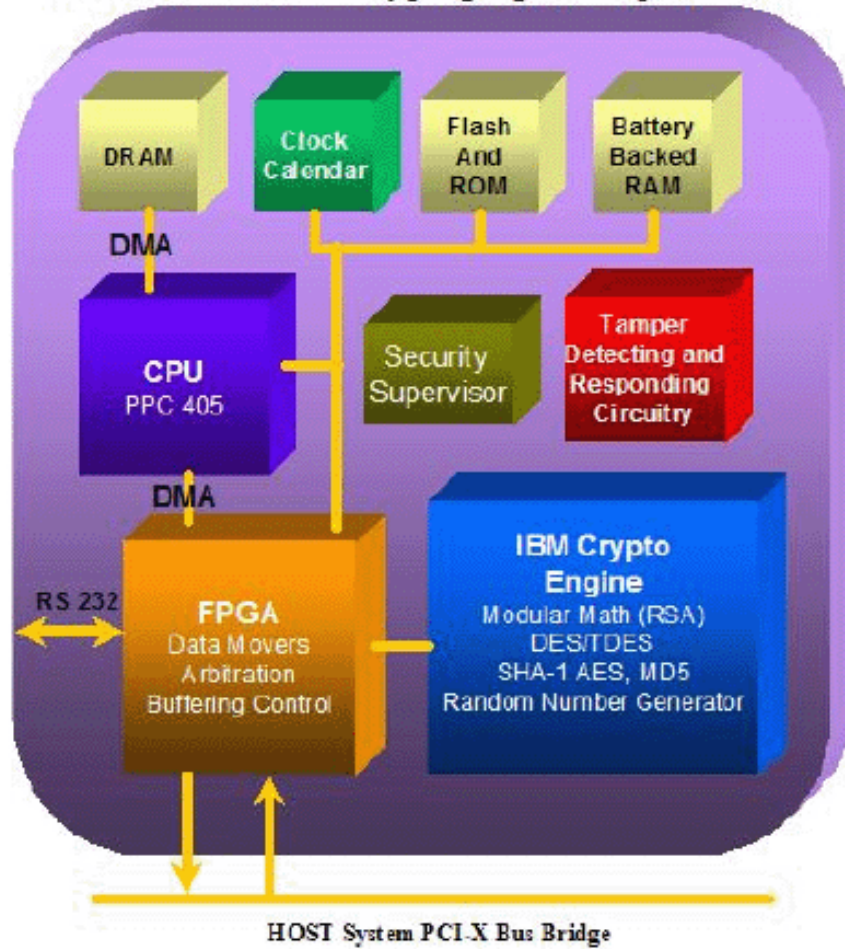
# Trusted Hardware



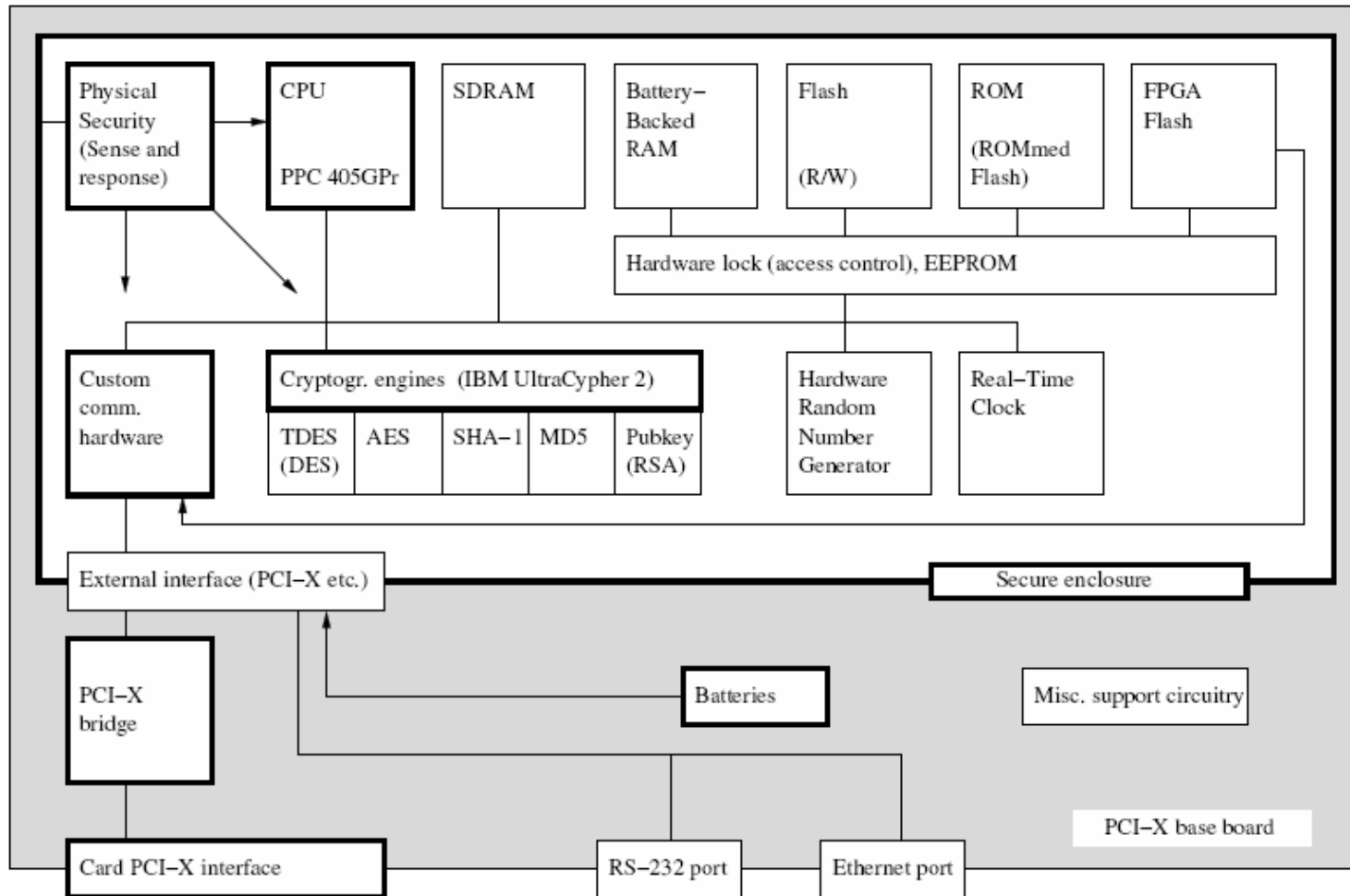
**IBM 47xx**

# IBM 4764 Architecture

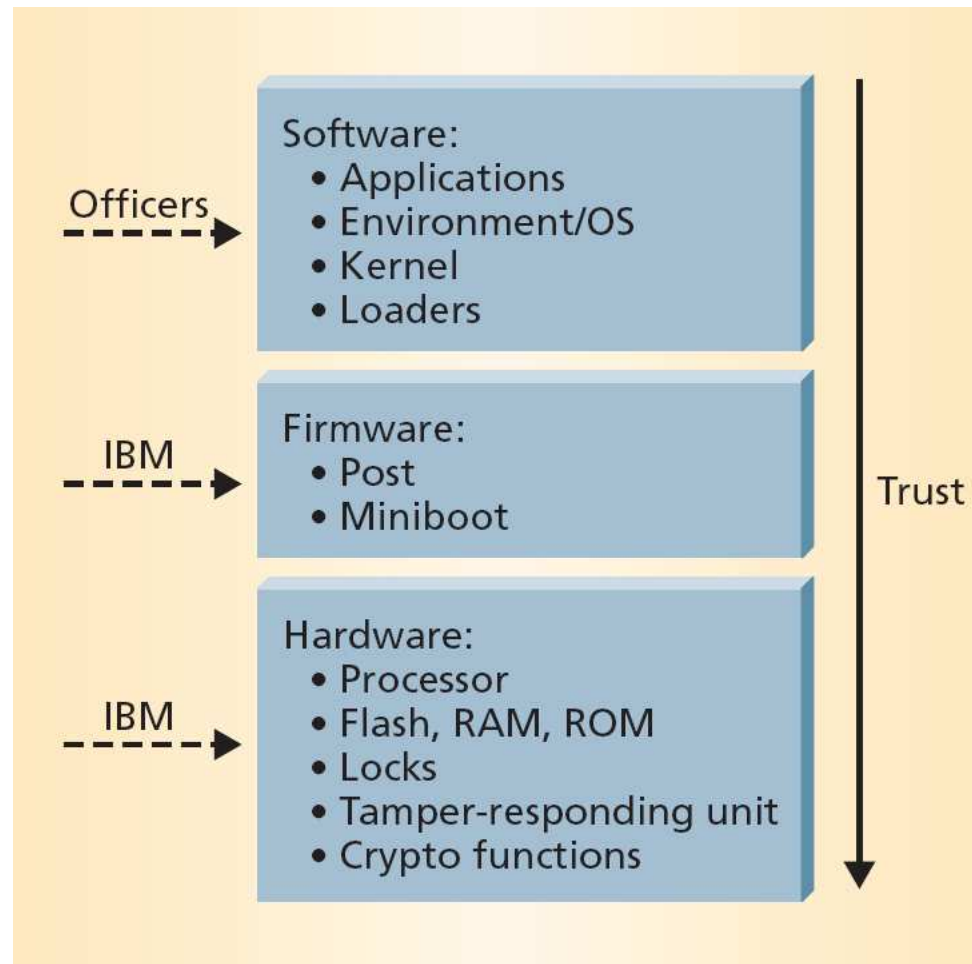
### IBM 4764 PCI-X Cryptographic Coprocessor



# IBM 4764 Architecture



# Trust Propagation



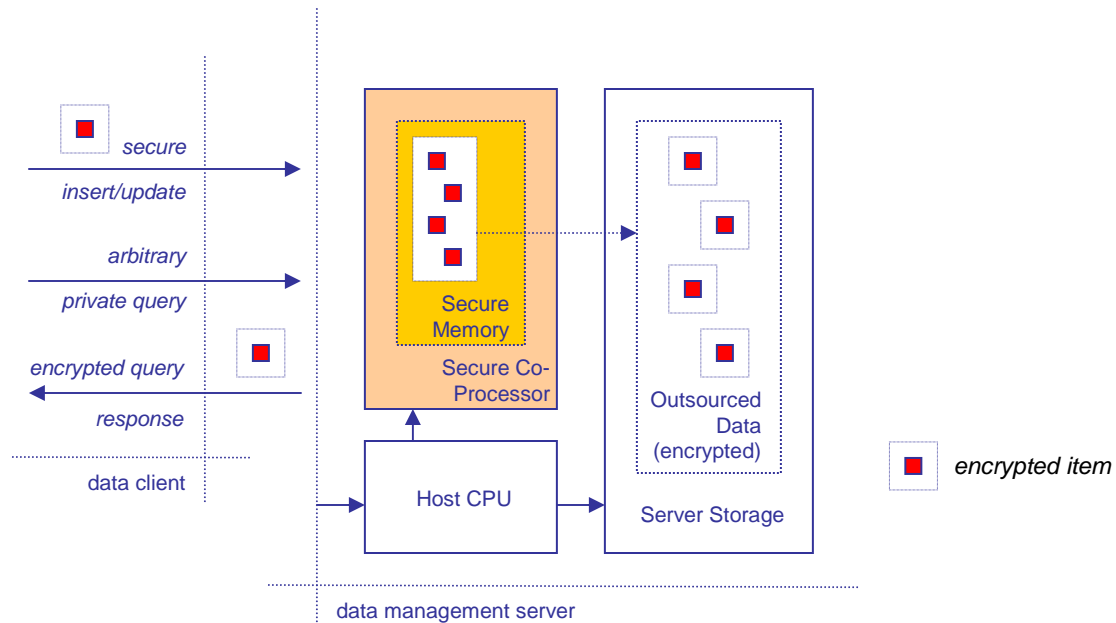
# SCPU Performance



**RSA1024 Sign: 848/sec**  
**RSA1024 Verify: 1157/sec**  
**3DES: 1-8MB/sec**  
**DES: 1-8MB/sec**  
**SHA1: 1-21MB/sec**

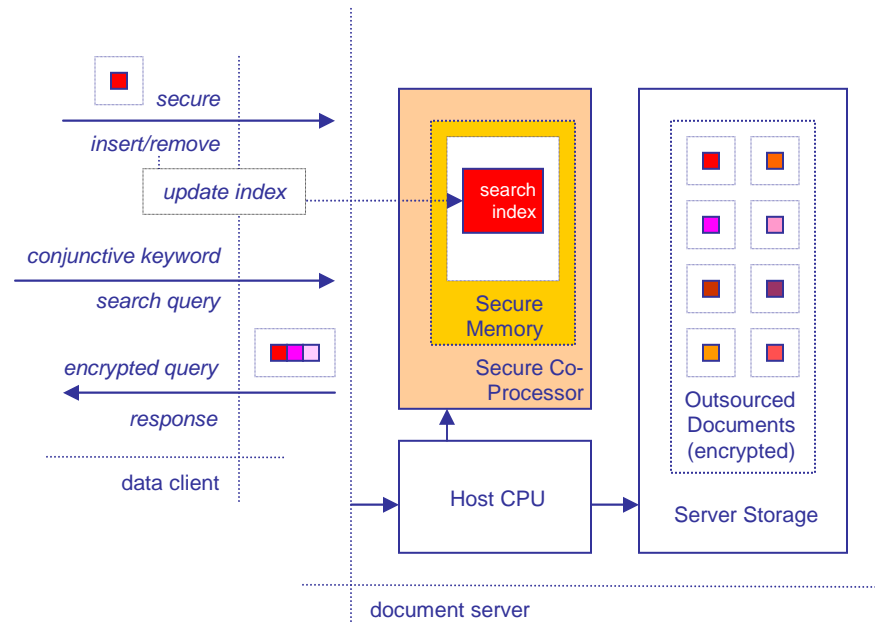
IBM 4764-001: 266MHz PowerPC. 64KB battery-backed SRAM storage. Crypto hardware engines: AES256, DES, TDES, DSS, SHA-1, MD5, RSA. FIPS 140-2 Level 4 certified.

# Possible Benefits



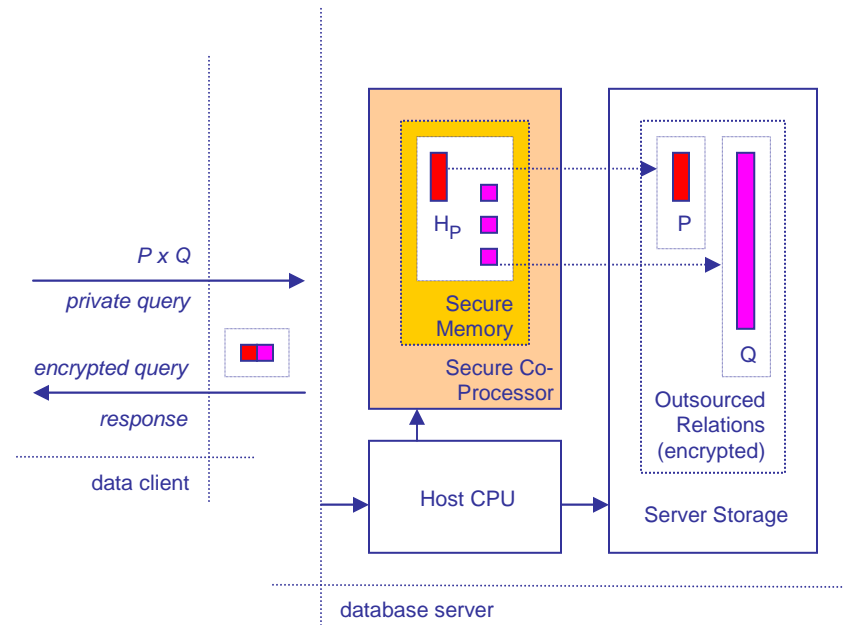
**A secure co-processor on the data management side may allow for significant leaps in expressivity for queries where privacy and completeness assurance are important.**

# Searching



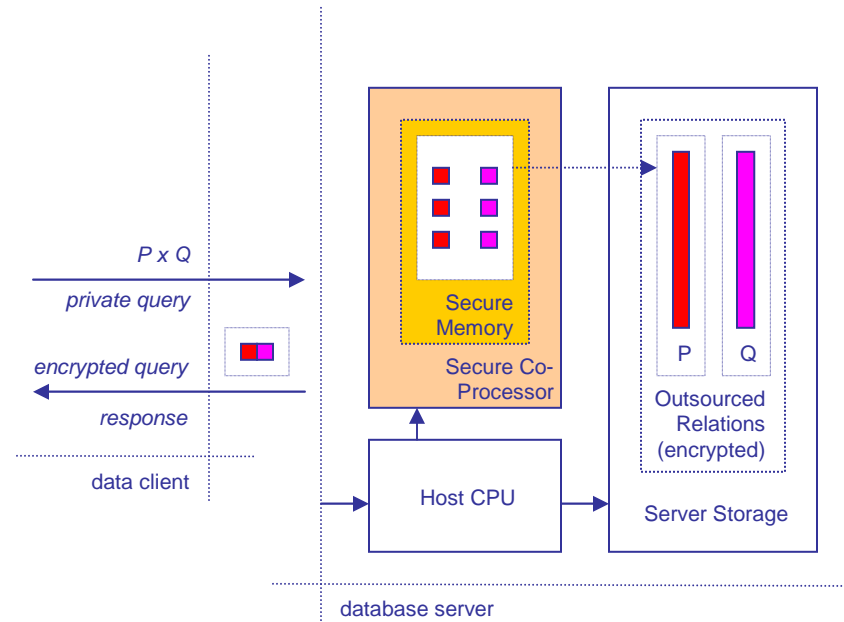
**For conjunctive keyword searches on document (email, files) servers, oblivious search index structures could be queried in secure memory achieving a novel zero-leak query model.**

# Hash-JOIN



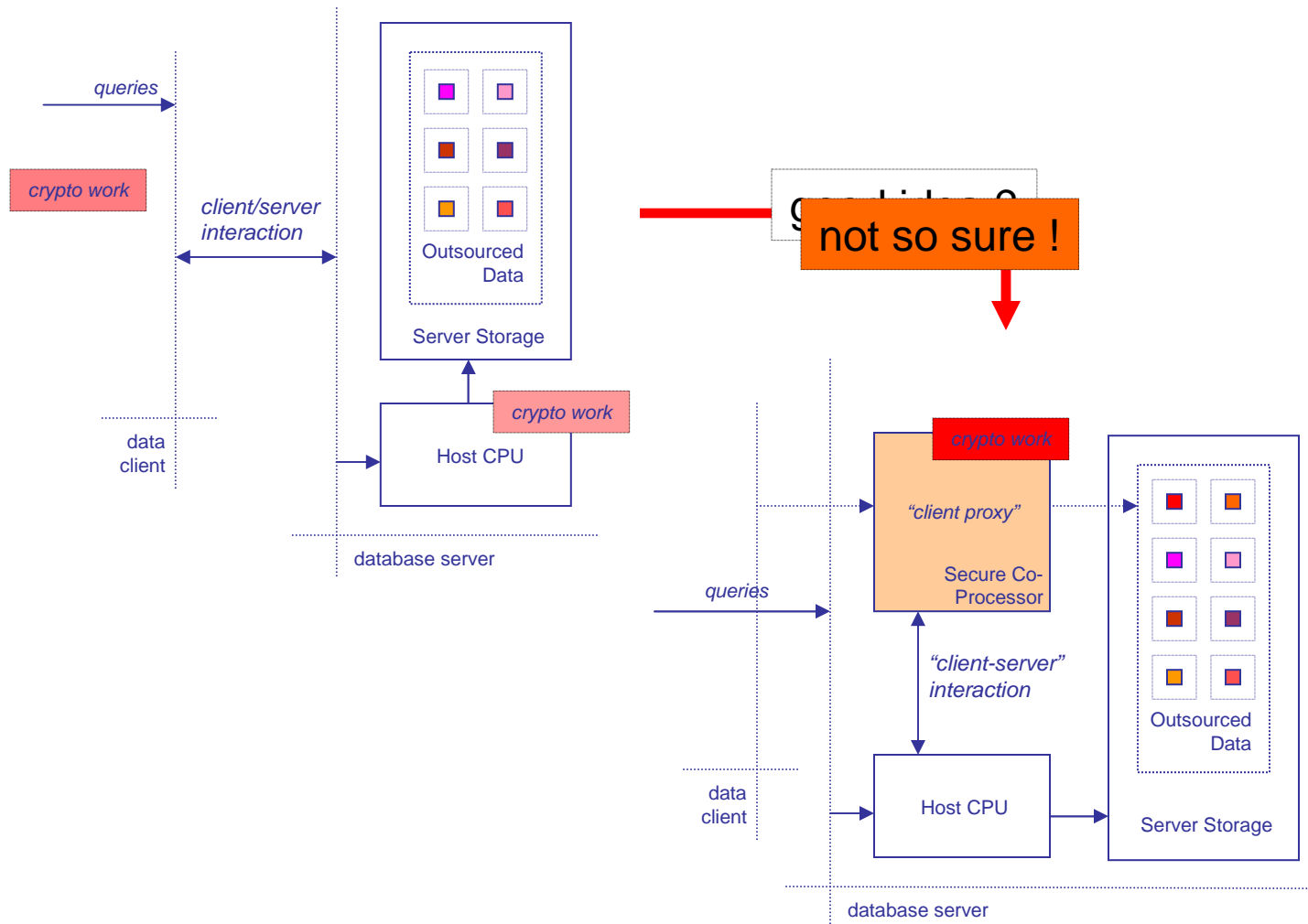
**Hash-JOIN could be naturally accommodated.**

# Merge-JOIN

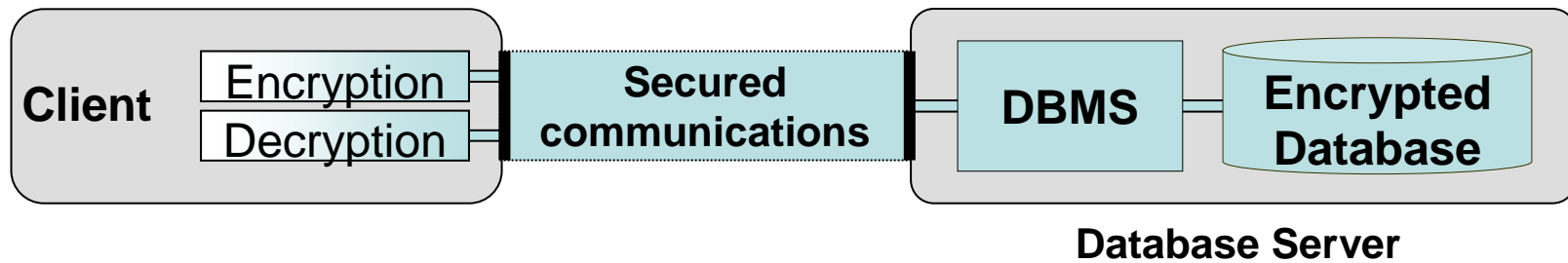
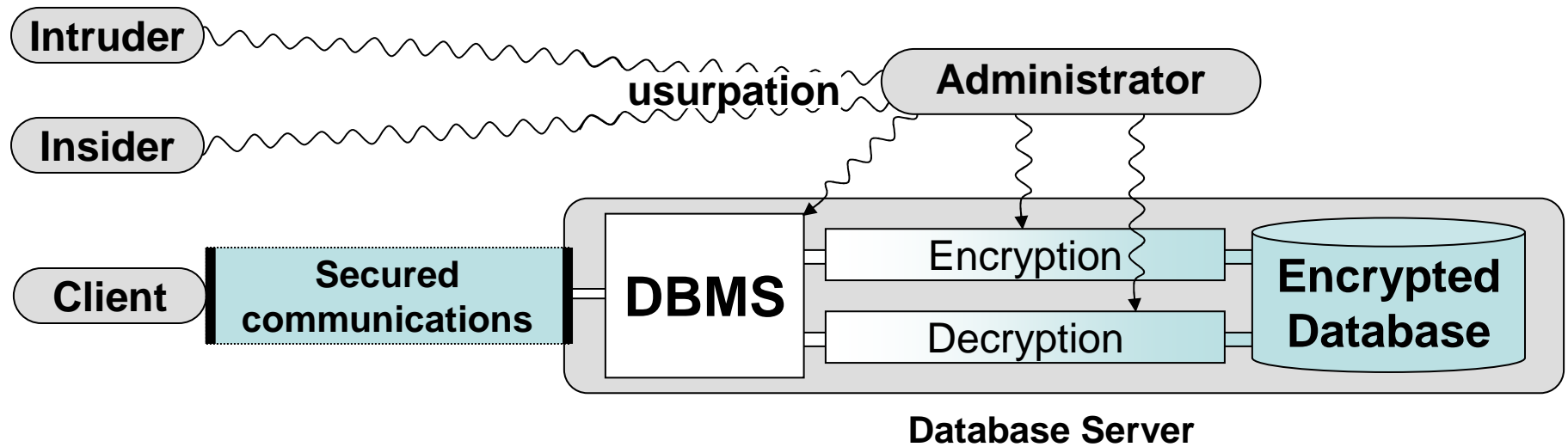


**For Merge-JOIN, order-preserving encryption primitives could be deployed to minimize the amount of data parsing required in the sorting phase.**

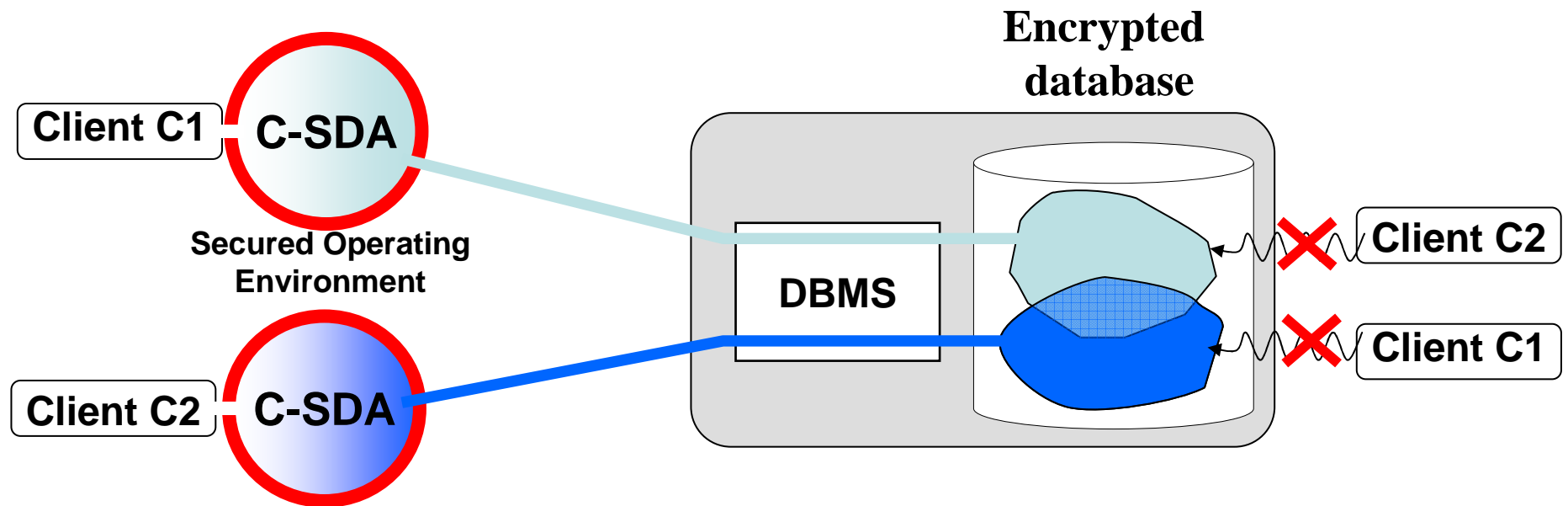
# Sample DON'T



- Process entire queries on SCPU (!)
- Dedicate (one) SCPU per query or equivalent
  - e.g., limit TPS by SCPU TPS
- Synchronize CPU with SCPU
  - e.g., block main CPU until SCPU completes
- Transfer  $\geq O(n)$  on SCPU-CPU bus (!)
- Anything else un-smart 😊

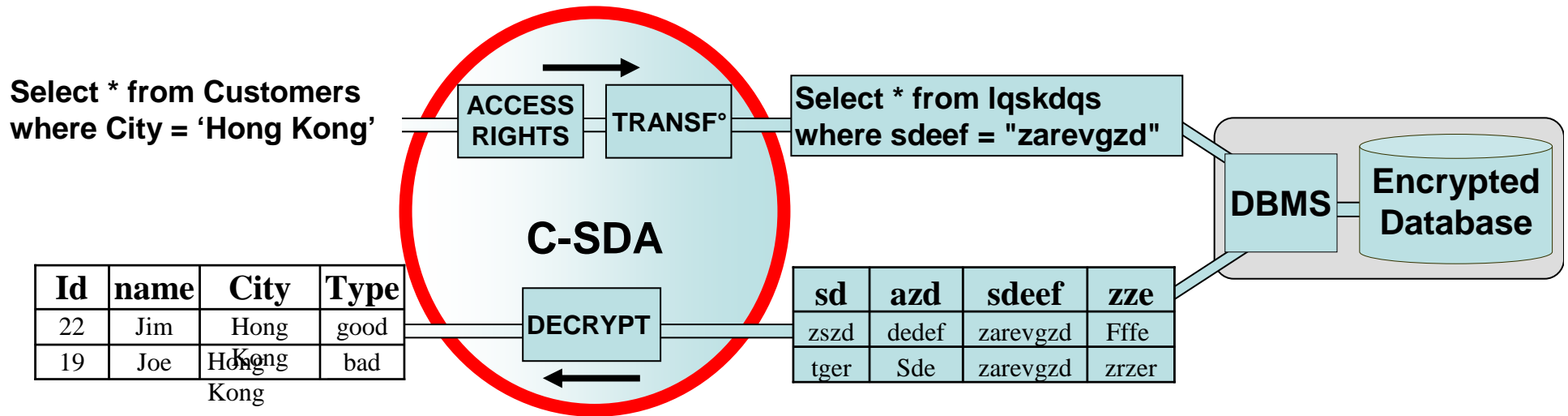


# Chip-Secured Data Access:

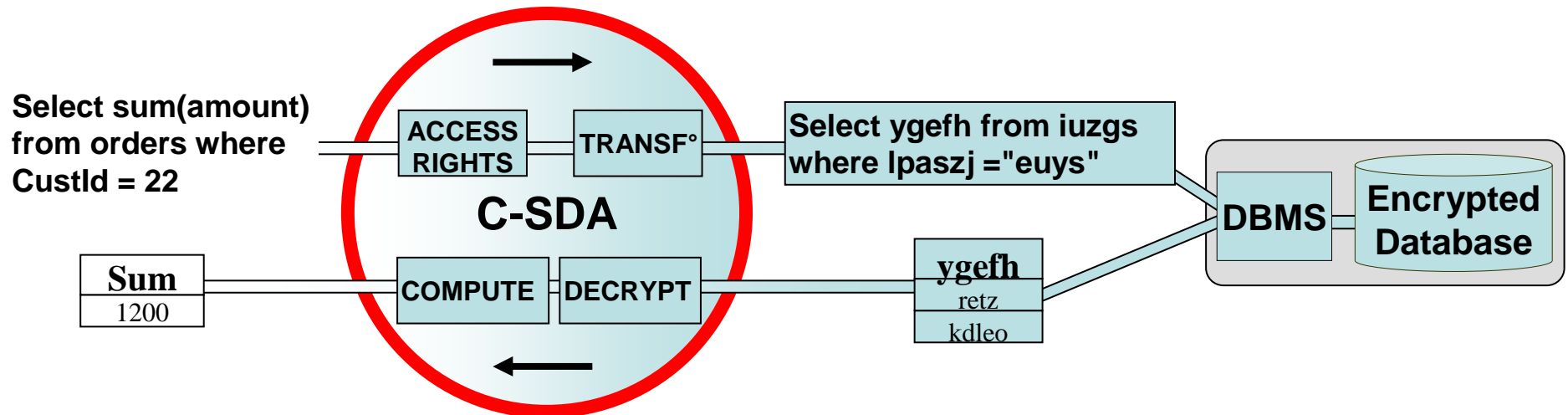


Smartcard: 32 bit RISC processor ( $\approx 40$ Mips), limited communication bandwidth (10 to 100 Kbps), tiny RAM, writes in EEPROM very costly.

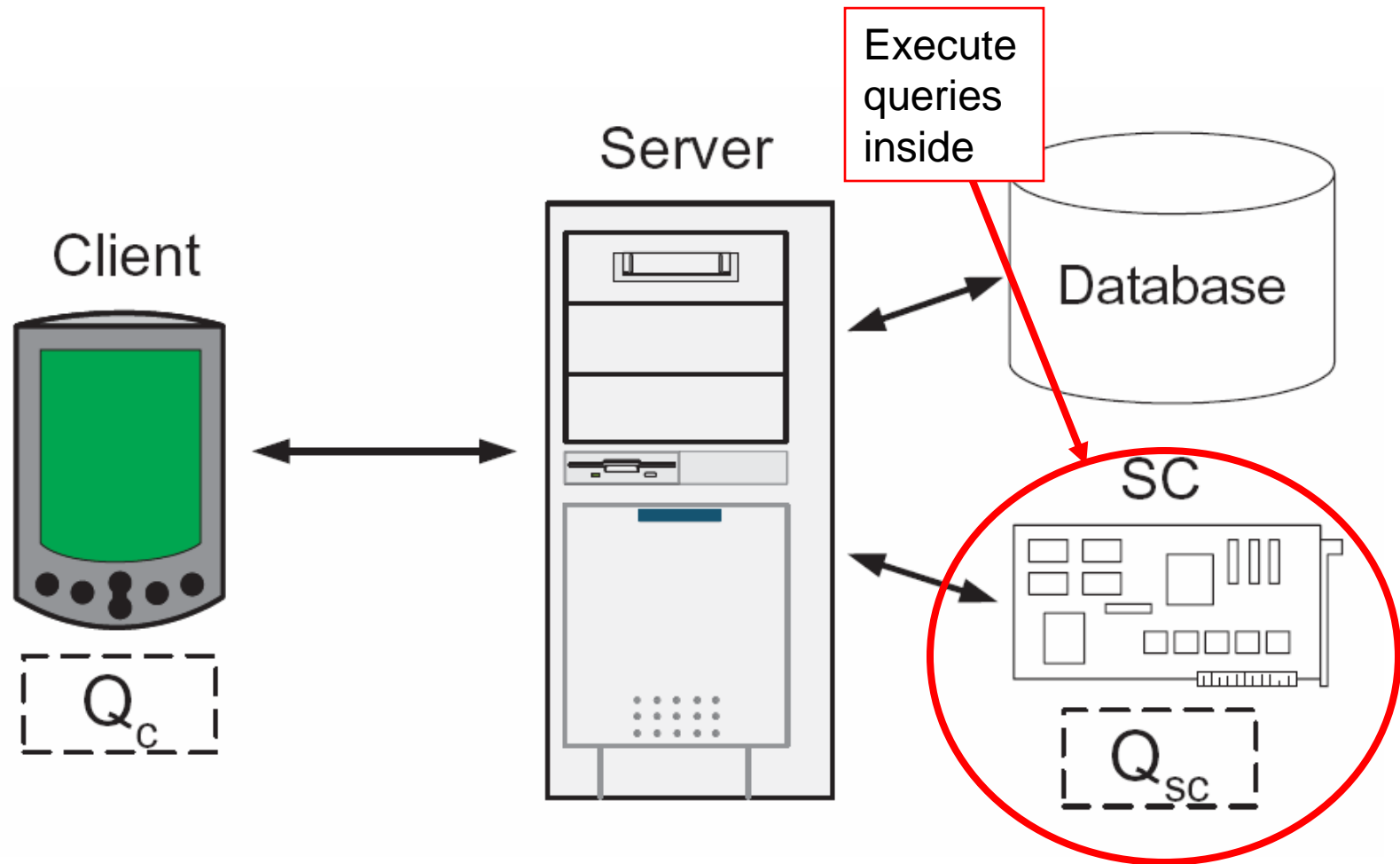
## Equi-predicate-only Queries:



## General queries:



# Tsudik (2005)



Practical maturity: in infancy, barely crawling.

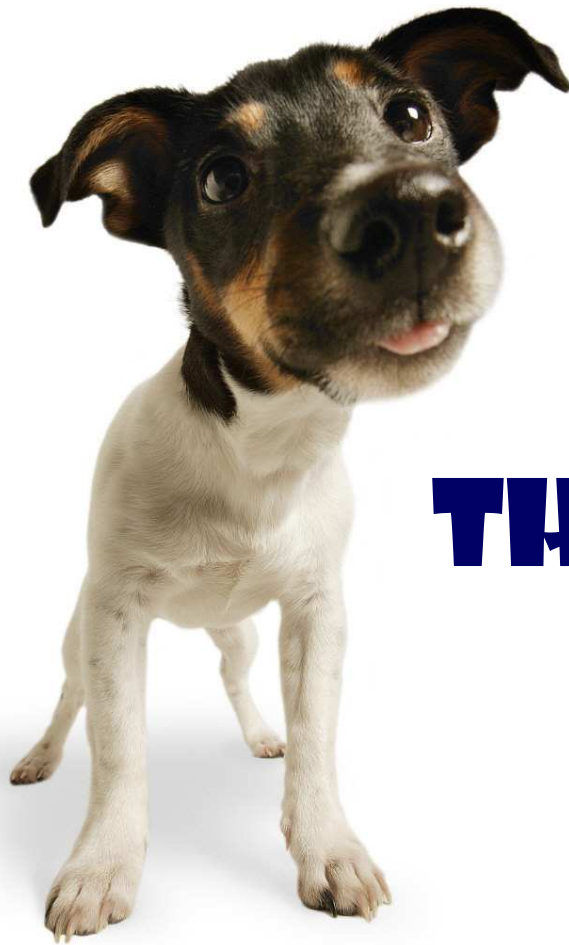
Very hard problems remain to be tackled:

- operators with integrated assurances
  - confidentiality
  - privacy of access
  - correctness
- scalable protocols for secure hardware
  - massive data
  - good utilization of host CPUs
- areas
  - relational data
  - file systems
  - streaming data

**/bin/yes > /dev/lunchtime**

---

Stony Brook Network Security and Applied Cryptography Lab



**THANK YOU !**

D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In Proceedings of Eurocrypt 2004, pages 506–522. LNCS 3027, 2004.

R. Brinkman, J. Doumen, and W. Jonker. Using secret sharing for searching in encrypted data. In Secure Data Management, 2004.

Y. Chang and M. Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. Cryptology ePrint Archive, Report 2004/051, 2004. <http://eprint.iacr.org/>.

E. Goh. Secure indexes. Cryptology ePrint Archive, Report 2003/216, 2003. <http://eprint.iacr.org/2003/216/>.

P. Golle, J. Staddon, and B. Waters. Secure conjunctive keyword search over encrypted data. In Proceedings of ACNS, pages 31–45. Springer-Verlag; Lecture Notes in Computer Science 3089, 2004.

D. Xiaodong Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In SP '00: Proceedings of the 2000 IEEE Symposium on Security and Privacy (S&P 2000). IEEE Computer Society, 2000.

Premkumar T. Devanbu, Michael Gertz, Chip Martel, and Stuart G. Stubblebine. Authentic third-party data publication. In IFIP Workshop on Database Security, pages 101–112, 2000, also in Journal of Computer Security, Vol. 11, No. 3, pages 291-314, 2003.

W. Du and M. J. Atallah. Protocols for secure remote database access with approximate matching. In Proceedings of the 1st ACM Workshop on Security and Privacy in E-Commerce, 2000.

H. Hacigumus, B. R. Iyer, and S. Mehrotra. Providing database as a service. In IEEE International Conference on Data Engineering (ICDE), 2002.

HweeHwa Pang and Arpit Jain and Krithi Ramamritham and Kian-Lee Tan. Verifying Completeness of Relational Query Results in Data Publishing. In Proceedings of ACM SIGMOD, 2005.

J. Li, M. Krohn, D. Mazières, and D. Shasha. Secure Untrusted Data Repository (SUNDR). In Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI 2004), pages 121–136, San Francisco, CA, December 2004. ACM SIGOPS.

Maithili Narasimha and Gene Tsudik. Authentication of Outsourced Databases using Signature Aggregation and Chaining. In Proceedings of DASFAA, 2006.

Charles Martel, Glen Nuckolls, Premkumar Devanbu, Michael Gertz, April Kwong, and Stuart G. Stubblebine. A general model for authenticated data structures. Algorithmica, 39(1):21–41, 2004.

E. Mykletun, M. Narasimha, and G. Tsudik. Authentication and integrity in outsourced databases. In ISOC Symposium on Network and Distributed Systems Security NDSS, 2004.

HweeHwa Pang and Kian-Lee Tan. Authenticating query results in edge computing. In ICDE '04: Proceedings of the 20th International Conference on Data Engineering, page 560, Washington, DC, USA, 2004. IEEE Computer Society.

Radu Sion. Query execution assurance for outsourced databases. In Proceedings of the Very Large Databases Conference VLDB, 2005.

F. Li, M. Hadjieleftheriou, G. Kollios, L. Reyzin, “Dynamic authenticated index structures for outsourced databases“, SIGMOD 2006

A. Buldas, M. Roos, and J. Willemson. Undeniable replies for database queries. In Proceedings of the Fifth International Baltic Conference on DB and IS, volume 2, pages 215-226, 2002.

H. Hacigumus, B. Iyer, C. Li, and S. Mehrotra. Executing SQL over encrypted data in the database service-provider model. In Proceedings of the ACM SIGMOD international conference on Management of data, pages 216–227. ACM Press, 2002.

B. Hore, S. Mehrotra, and G. Tsudik. A privacy-preserving index for range queries. In Proceedings of VLDB 2004.

Nan Zhang and Wei Zhao, Distributed Privacy Preserving Information Sharing. In Proceedings of the International Conference on Very Large Databases VLDB 2005.

Tran Khanh Dang, Privacy-Preserving Search and Updates for Outsourced Tree-Structured Data on Untrusted Servers, Springer LNCS 3477/2005, pages 338-354

Hakan Hacigumus, Bala Iyer, Sharad Mehrotra, Query Optimization in Encrypted Database Systems, DASFAA 2005

Hakan Hacigumus, Bala Iyer, Sharad Mehrotra, Efficient Execution of Aggregation Queries over Encrypted Relational Databases, DASFAA 2004

Hakan Hacigumus, Sharad Mehrotra, Williams Jonker, Milan Petkovic, Efficient key updates in encrypted database systems, SDM 2005

Hakan Hacigumus, Bala Iyer, Sharad Mehrotra, Ensuring the Integrity of Encrypted Databases in the Database-as-a-Service Model. DBSEC 2003

Hakan Hacigumus, Bala Iyer, Sharad Mehrotra, Encrypted Database Integrity in Database Service Provider Model. Certification and Security in E-Services 2002

# refs: service providers

Activehost.com Internet Services. Online at <http://www.activehost.com>.

Adhost.com MySQL Hosting. Online at <http://www.adhost.com>.

Alentus.com Database Hosting. Online at <http://www.alentus.com>.

Datapipe.com Managed Hosting Services. Online at <http://www.datapipe.com>.

Discountasp.net Microsoft SQL Hosting. Online at <http://www.discountasp.net>.

Gate.com Database Hosting Services. Online at <http://www.gate.com>.

Hostchart.com Web Hosting Resource Center. Online at <http://www.hostchart.com>.

Hostdepartment.com MySQL Database Hosting. Online at <http://www.hostdepartment.com/mysqlwebhosting/>.

IBM Data Center Outsourcing Services. Online at <http://www-1.ibm.com/services/>.

IBM Data Encryption for DB2. Online at <http://www.ibm.com/software/data/db2>.

Inetu.net Managed Database Hosting. Online at <http://www.inetu.net>.

Mercurytechnology.com Managed Services for Oracle Systems. Online at <http://www.mercurytechnology.com>.

Neospire.net Managed Hosting for Corporate E-business. Online at <http://www.neospire.net>.

Netnation.com Microsoft SQL Hosting. Online at <http://www.netnation.com>.

Opendb.com Web Database Hosting. Online at <http://www.opendb.com>.

Bouganim L., Pucheral P., "Chip-Secured Data Access: Confidential Data on Untrusted Servers", Int. Conf. on Very Large Data Bases VLDB 2002.

Ernesto Damiani, S.De Capitani di Vimercati, Sushil Jajodia, Stefano Paraboschi, Pierangela Samarati, "Balancing Confidentiality and Efficiency in Untrusted Relation DBMS", ACM CCS 2003

E. Mykletun and G. Tsudik, On using Secure Hardware in Outsourced Databases, International Workshop on Innovative Architecture for Future Generation High Performance Processors and Systems IWIA 2005.

IBM 4764 PCI-X Cryptographic Coprocessor (PCIXCC). Online at <http://www-03.ibm.com/security/cryptocards/pcixcc/overview.shtml>.

B. Bhattacharjee, N. Abe, K. Goldman, B. Zadrozny, V. Reddy, M. del Carpio, C. Apte, "Using secure coprocessors for privacy preserving collaborative data mining and analysis", Second ACM International Workshop on Data Management On New Hardware (DAMON) 2006

Kenneth Goldman, Enriquillo Valdez: "Matchbox: Secure Data Sharing", IEEE Internet Computing 2004

"Practical server privacy with secure coprocessors", IBM Systems Journal 2001, S. W. Smith, D. Safford

J. Marchesini, S.W. Smith, "SHEMP: Secure Hardware Enhanced MyProxy", Technical Report TR2005-532, Department of Computer Science, Dartmouth College, February 2005.

A. Iliev, S.W. Smith, "Protecting Client Privacy with Trusted Computing at the Server", IEEE Security and Privacy, March/April 2005

A. Iliev, S.W. Smith, "Private Information Storage with Logarithmic-space Secure Hardware.", 3rd Working Conference on Privacy and Anonymity in Networked and Distributed Systems.

A. Iliev, S.W. Smith, "Prototyping an Armored Data Vault: Rights Management on Big Brother's Computer.", Privacy-Enhancing Technology 2002

E. Mykletun and G. Tsudik, "On using Secure Hardware in Outsourced Databases", International Workshop on Innovative Architecture for Future Generation High-Performance Processors and Systems, January 2005

Maheshwari, Vingralek, and Shapiro, How to build a trusted database system on untrusted storage, OSDI 2000

E. Kushilevitz and R. Ostrovsky, “Replication is not needed: single database, computationally-private information retrieval”, FOCS 1997

Radu Sion and Bogdan Carbunar, On the Practicality of Private Information Retrieval, NDSS 2006

William Gasarch, A WebPage on Private Information Retrieval, <http://www.cs.umd.edu/~gasarch/pir/pir.html>

Josep Domingo-Ferrer, Ricardo X. Sanchez del Castillo, An Implementable Scheme for Secure Delegation of Computing and Data, 1st International Information and Communications Security Conference ICICS 1997