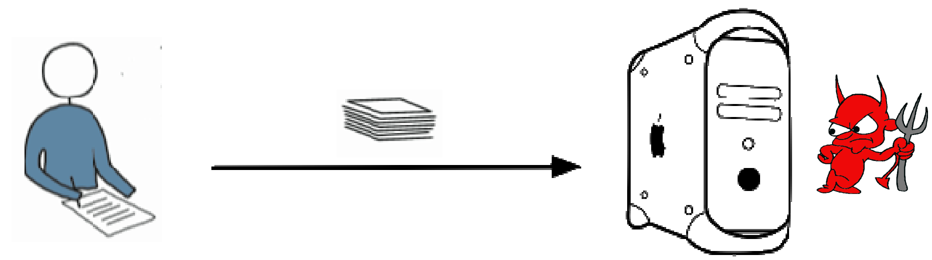
Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions

Reza Curtmola (JHU)
Juan Garay (Bell Labs)
Seny Kamara (JHU)
Rafail Ostrovsky (UCLA)

Lucent Technologies
Bell Labs Innovations

Remote Storage

- Remote storage is ubiquitous:
  - Data back-ups
  - Gmail, Yahoo Mail, …

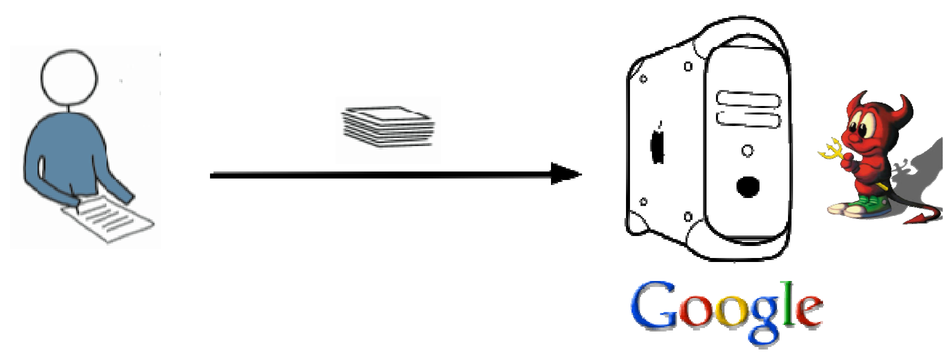


- Q: How do we store sensitive data on an untrusted server?
- A: Encryption
  - Hides all partial information about the data
  - Client must upload *all* data, decrypt and perform operations locally
- Can we enable the server to help?

Google Desktop

"If a consumer chooses to use it, the new "Search Across Computers" feature will store copies of the user's Word documents, PDFs, spreadsheets and other text-based documents on Google's own servers, to enable searching from any one of the user's computers"

02/09/06



Talk Outline

- Motivation
- Overview of Different Models for Private Searching
- Our Focus: Searchable *Symmetric* Encryption (SSE)
  - Revisiting security definitions for SSE
  - Two new notions of security for SSE: "*Adaptive,*" "*Non-adaptive*"
  - Two new constructions
- Extensions

Private Searching

- MPC: general, but inefficient [Yao82, GMW87, BGW88, CCD88]
- Searching (explicitly) – different settings
  - Public data: unencrypted (e.g., stock-quotes, news articles, patents)
    - Client wishes to hide which element is accessed
    - PIR and its variants [CGKS,KO97,...]
  - User-owned data: symmetrically encrypted
    - Client can upload additional “encrypted” data structures to help search
    - *Oblivious RAMs*, searchable symmetric encryption [O90, OG96, SWP00, Goh03, CM05]
  - Third-party data: public-key encrypted
    - Data comes encrypted to server from users other than client
    - Public-key searchable encryption (*PEKS*) [BDOP05,BW06...]

Searchable *Symmetric* Encryption

- Scenario:
  - Client has a collection of documents that consists of a set of words
  - Encrypts document collection together with additional data structure
  - Sends everything to server
- Functionality: support the following types of queries
  - Find all documents that contain a particular keyword
- Privacy: allow server to help, but reveal as little as possible

Prior Work on SSE

- SSE can be achieved using *oblivious RAMs* [O90, OG96]
  - Functionality: can simulate any data structure in a hidden way, and can support conjunctive queries, B-trees, etc...
  - Privacy: hides everything, even the access pattern
  - Efficiency: logarithmic number of rounds per each read/write
- Q: Can we search over encrypted data in single/constant rounds?
  - With absolute privacy, we don't know (great open problem!)
  - What if we relax the security requirements?

Prior Work on SSE (cont'd)

- How do we relax the security definition?
  - Leak the access pattern but nothing else
  - Defining this formally is "delicate"
- Three previous constant-round SSE proposals
  - "Practical techniques for searches on encrypted data" [SWP00]
  - "Secure Indexes" [Goh03]
  - "Privacy-preserving keyword searches on remote encrypted data" [CM05]

Talk Outline

- Motivation
- Overview of Different Models for Private Searching
- Our Focus: Searchable *Symmetric* Encryption (SSE)
  - Revisiting security definitions for SSE
  - "*Non-adaptive*" definitions and construction
  - "*Adaptive*" definitions and construction
- Extensions

Revisiting SSE Security Definitions

- [SWP00,Goh03,CM05]: "A secure SSE scheme should not leak anything beyond the outcome of a search"
  - "Search outcome:" memory addresses (identifiers) of documents that contain a hidden keyword (precise definition later)
  - **Note:** Different keyword queries may lead to same search outcome
  - "Search pattern:" whether two queries were for the same keyword or not
- A (slightly) better intuition: "A secure SSE scheme should not leak anything beyond the outcome and the pattern of a search"

SWP's SSE Definition

- Implicitly use indistinguishability [GM84] as a security definition
  - "Any function of the plaintext that can be computed from the ciphertext can be computed from the length of the plaintext"
- **Issue:** Adversary gets to see search outcomes and search pattern
- Secure encryption scheme definition does not model the fact that this additional information is revealed
- Before [Goh03, CM05]…

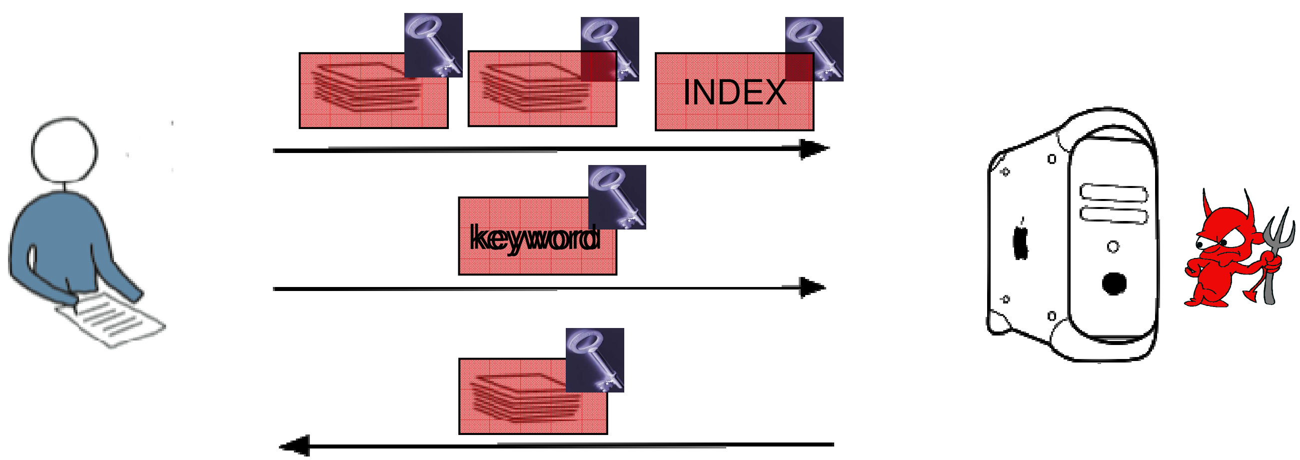
SSE Algorithms

- Keygen(1^k): outputs symmetric key K
- BuildIndex(K, $\{D_1, ..., D_n\}$): outputs secure index I
- Trapdoor(K, w): outputs a trapdoor T_w
- Search(I, T_w): outputs identifiers of documents containing w:

  ($id_1, ..., id_m$)

SSE System Operation

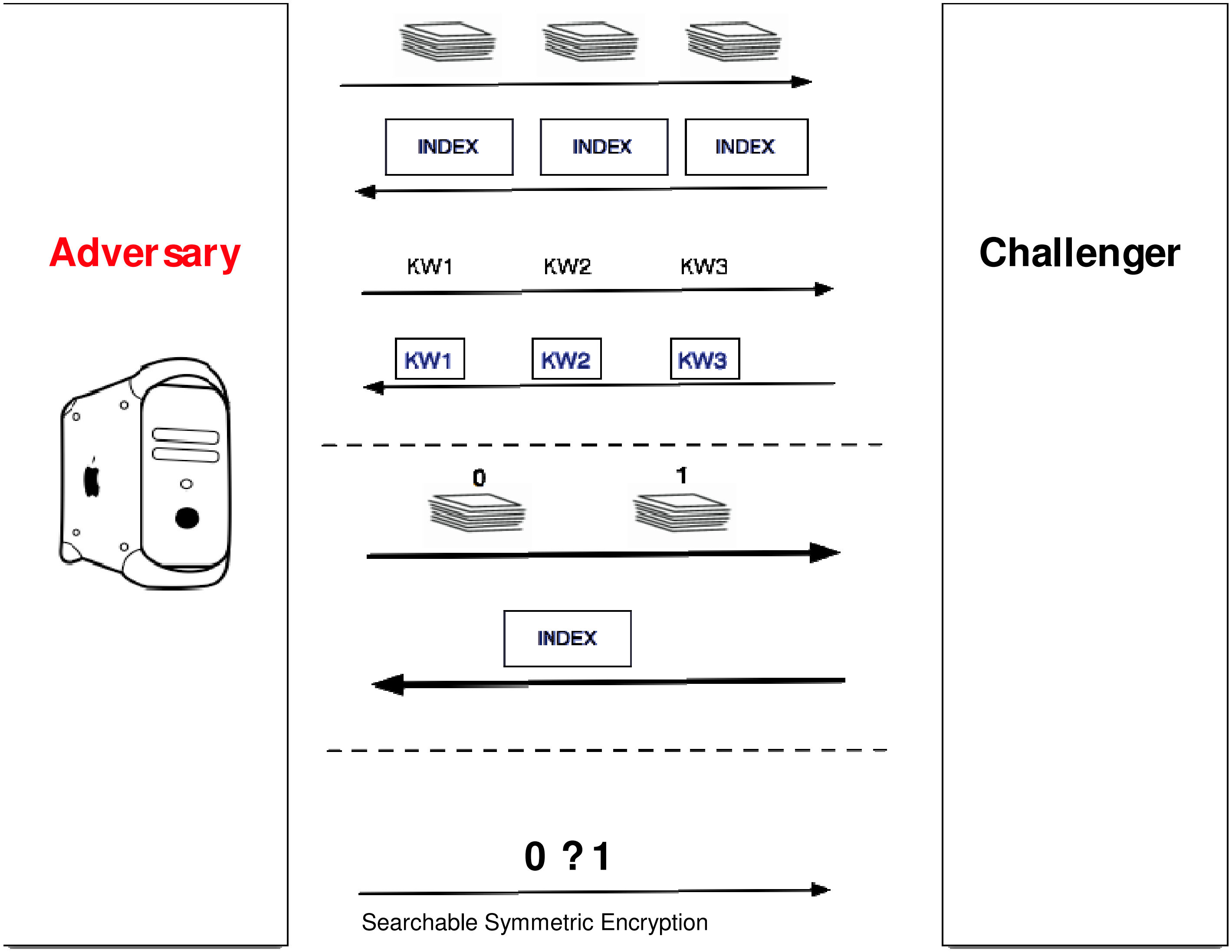
- Secure index: additional data structure that helps the server to search (following [Goh03] terminology)
- Symmetrically encrypted data: client performs encryption himself
- Trapdoors: associate a trapdoor to keywords which enables server to search while keeping keyword hidden



Goh's SSE Definition

- IND2-CKA: *Indistinguishability against Chosen-Keyword Attacks* (semantic security against chosen-keyword attacks)
  - Intuitively, adversary cannot deduce a document's content from its index

IND2-CKA [Goh03]



Goh's SSE Definition

- IND2-CKA: *Indistinguishability against Chosen-Keyword Attacks* (semantic security against chosen-keyword attacks)
  - "Any function of the documents that can be computed from the encrypted documents and the index can be computed from the length of the documents and the search outcomes"
- **Issue:** Says nothing about keywords or trapdoors
- Why not prove index secure in the sense of IND2-CKA and trapdoors "secure" using another definition?
- We show that there exists an SSE scheme that has
  - IND2-CKA indexes and trapdoors that are semantically secure
  - but when taken together, adversary can recover keyword
- **Note:** [Goh03] other applications besides SSE; secure trapdoors is not necessary for all the applications

CM SSE Definition

- "CM Security:"
  - "Any function of the documents and keywords that can be computed from the ciphertext, the index and the trapdoors, can be computed from the length of the documents and the search outcomes"
- **Issues:**
  - Leaves out the search pattern (proofs assume unique queries)
  - Order of quantifiers implies that there will always exist a simulator that can evaluate function on documents and keywords
  - Only guarantees security against non-adaptive adversaries

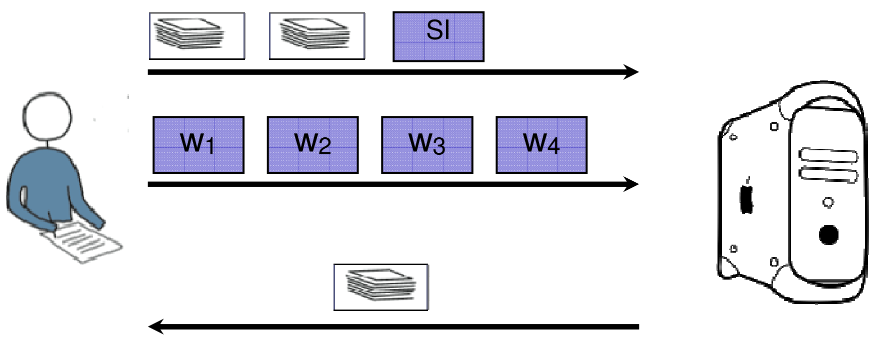
Revisiting SSE Security Definitions (cont'd)

- What is "*adaptivity*" – adaptive SSE security?
  - **Non-adaptive** adversaries make search queries *without* seeing the outcome of previous searches
  - **Adaptive** adversaries can make search queries *as a function* of the outcome of previous searches

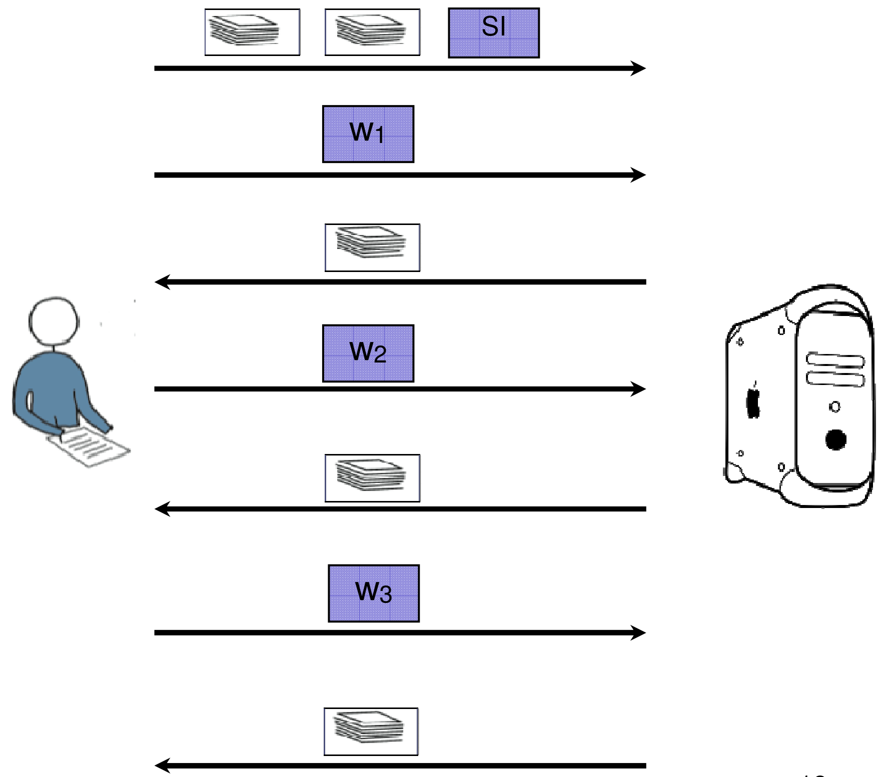
Adaptive SSE Security

Non-Adaptive SSE

[SWP00,Goh03,CM05,...]



Adaptive SSE

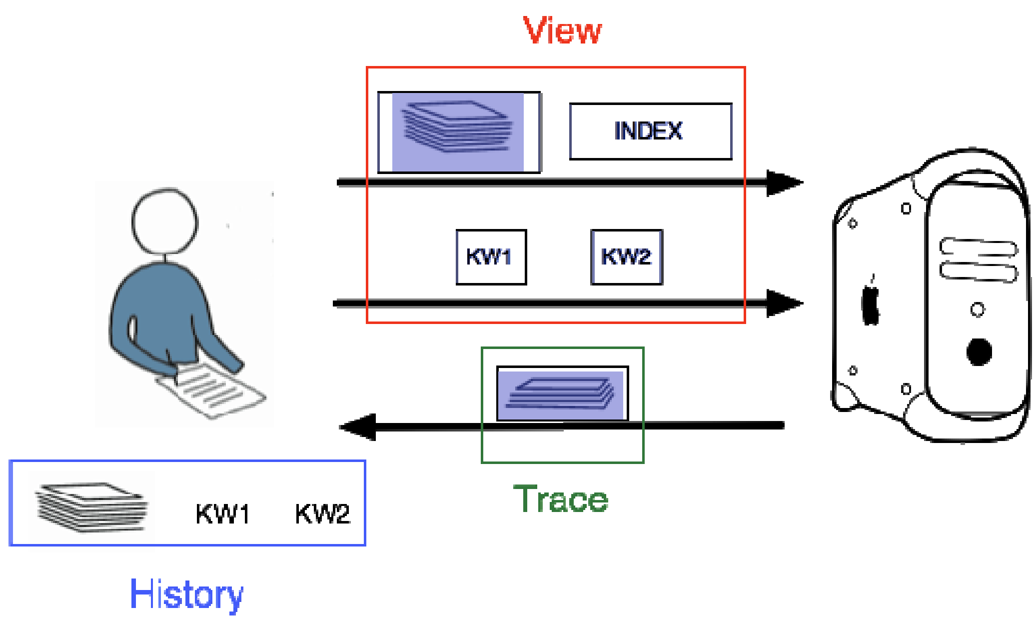


Talk Outline

- Motivation
- Overview of Different Models for Private Searching
- Our Focus: Searchable *Symmetric* Encryption (SSE)
  - Revisiting security definitions for SSE
  - "*Non-adaptive*" definitions and construction
  - "*Adaptive*" definitions and construction
- Extensions

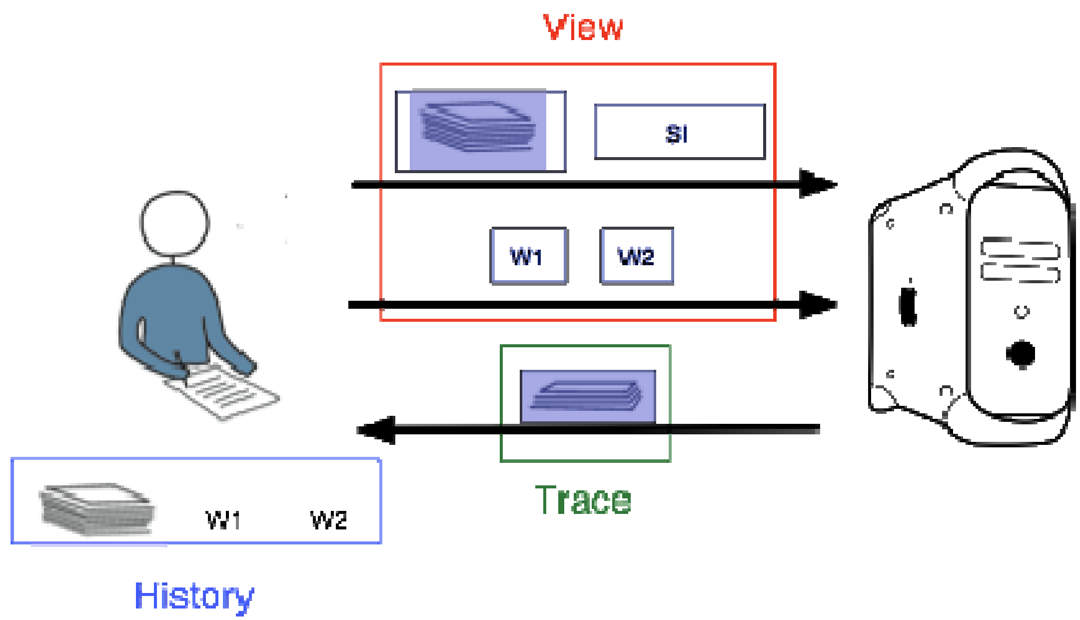
Our Model

- **History**: Documents and keywords queried
- **View**: Encrypted doc's, index, trapdoors
- **Trace**: Length of doc's, search outcomes, search pattern



Our SSE Definition (Informal)

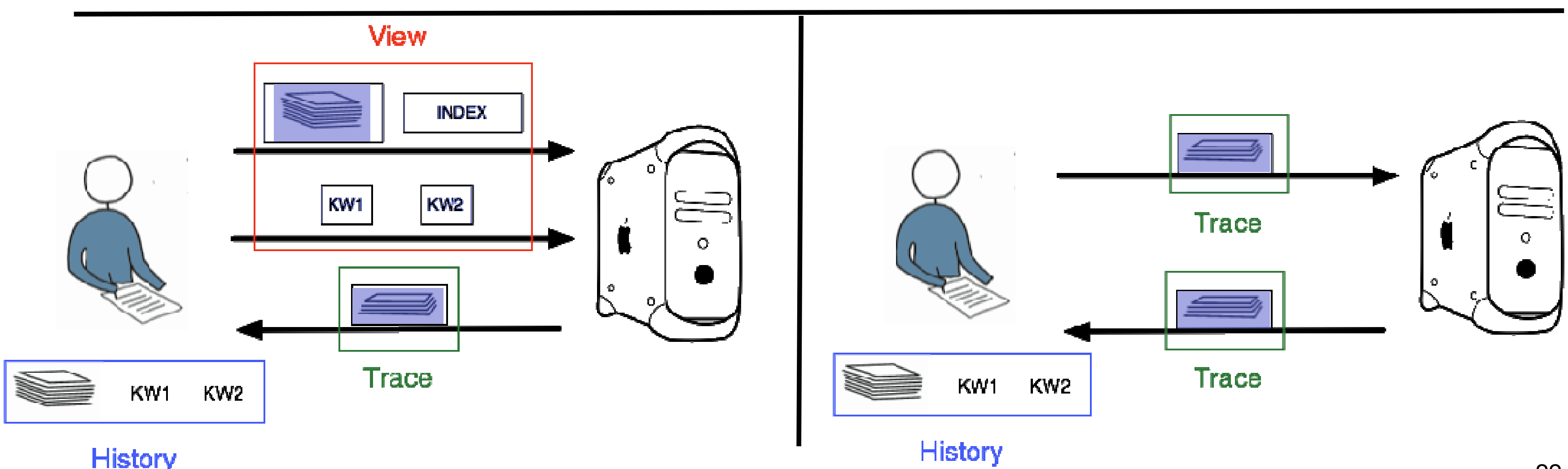
- "Any function of the documents and keywords that can be computed from the ciphertext, the index and the trapdoors, can be computed from the length of the documents, the search outcomes and the search patterns"



Non-Adaptive SSE Security (Formal)

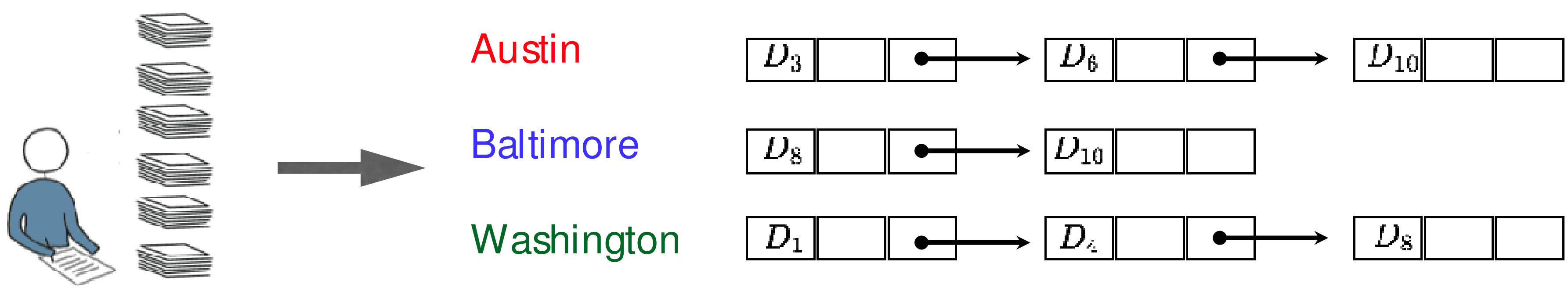
For all q, for all PPT adversaries, for all functions f, for all traces of length q (Tr_q), for all distributions over the set of histories with trace Tr_q, there exists a PPT algorithm (*simulator*) such that:

$$\left| \Pr\left[\begin{array}{c} \mathcal{A}(\text{View}_q) = \\ f(\text{History}_q) \end{array} \right] - \Pr\left[\begin{array}{c} \mathcal{S}(\text{Trace}_q) = \\ f(\text{History}_q) \end{array} \right] \right| \leq \text{negl}(k)$$



SSE-1

- Building a Secure Index

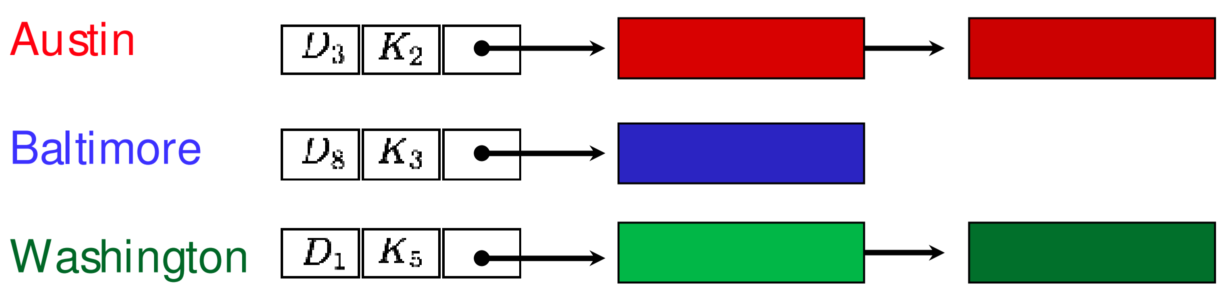


SSE-1

- Building a Secure Index

PRP $f : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$

PRF $g : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^m$

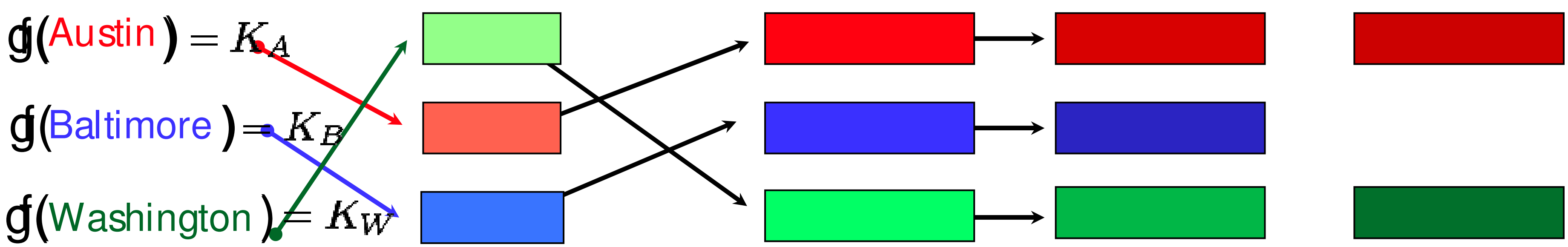


SSE-1

- Building a Secure Index

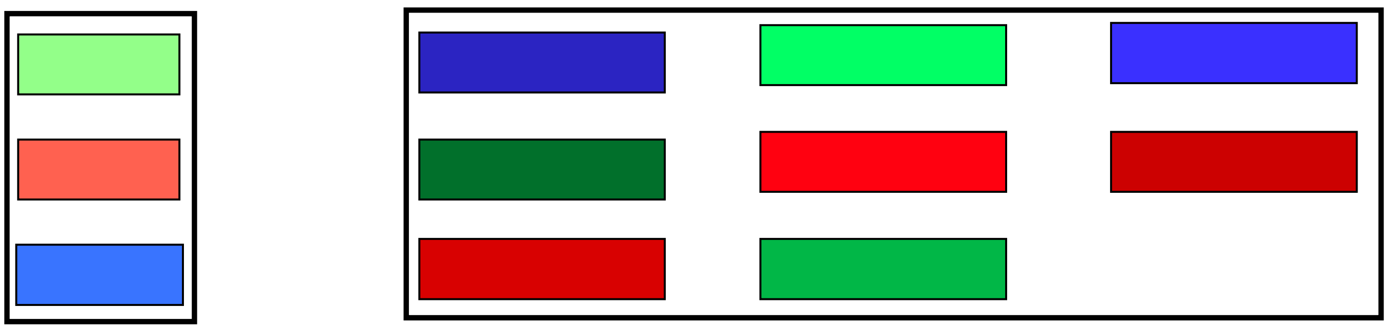
PRP $f : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$

PRF $g : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^m$



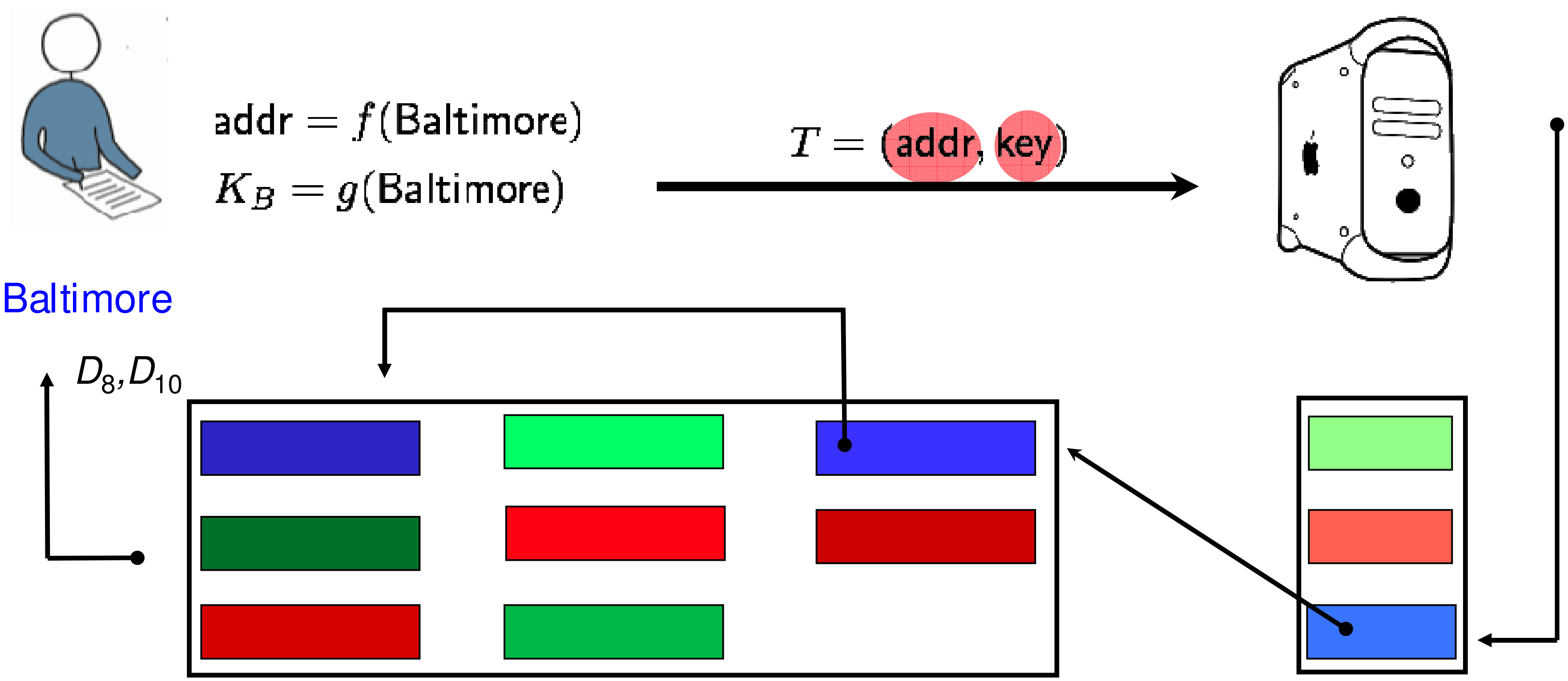
SSE-1

- Building a Secure Index



SSE-1

- Searching



SSE-1: Technical Issues

- Padding and shuffling
- Efficient storage of sparse tables
  - Large address space, small no. of entries
  - FKS dictionaries [Fredman-Komlos-Szemeredi '84]
    - Storage: O(# entries)
    - Lookup: O(1)

Performance of SSE-1

	[SWP00]	[Goh03]	[CM05]	SSE-1
server comp.	$O(n)$	$O(n)$	$O(n)$	$O(d)$

n: number of documents

d: number of documents that contain keyword

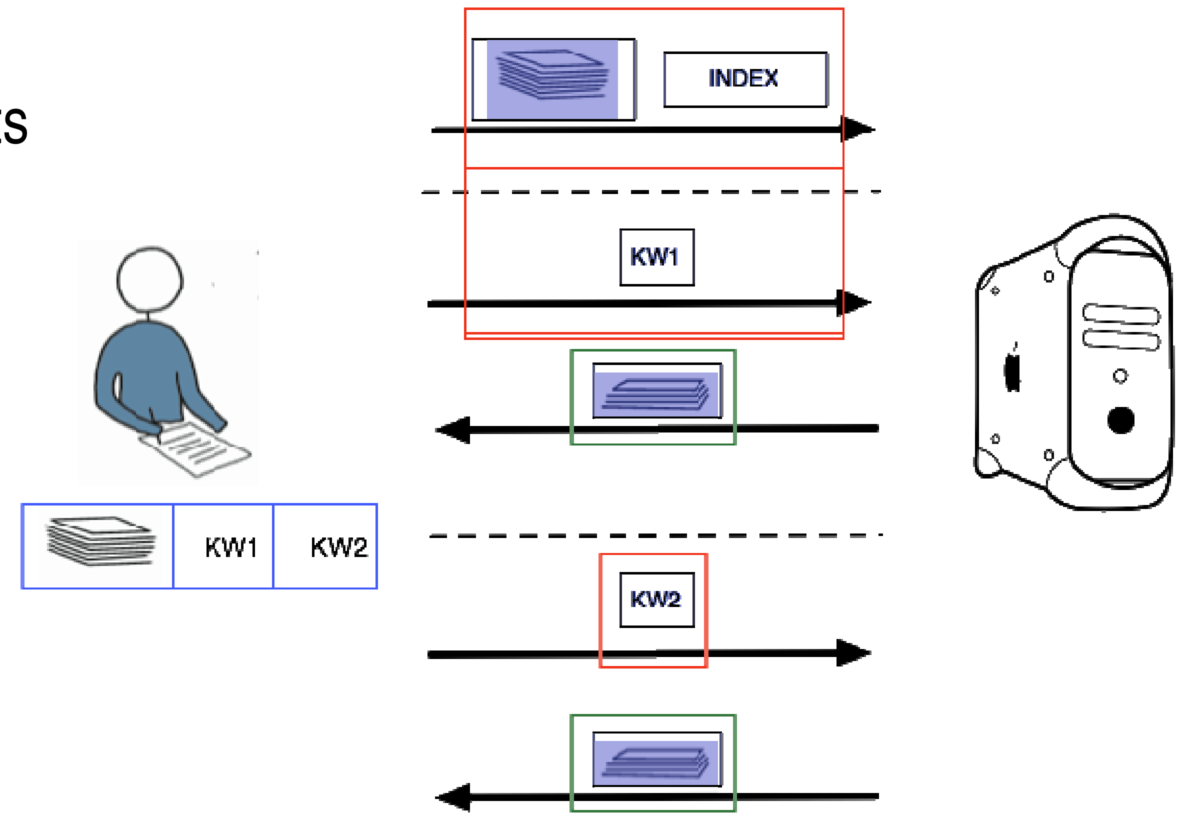
Talk Outline

- MPC vs. PPC
- Searchable Symmetric Encryption
  - Revisiting security definitions for SSE
  - “Non-adaptive” SSE definitions and construction
  - “Adaptive” SSE definitions and construction
- Extensions

Adaptive SSE Security

- "Any function about the partial history that can be computed from the partial view can be computed from the partial trace"

- **Partial History**: Documents and keywords
- **Partial View**: Encrypted doc's, index, trapdoors
- **Partial Trace**: Document lengths, search outcomes, search pattern



Adaptive SSE Security (cont'd)

For all q, for all PPT adversaries, for all functions f, for all traces of length q (Tr_q), for all distributions over the set of histories with trace Tr_q, there exists a PPT simulator such that for all $0 \leq t \leq q$:

$$\left| \Pr\left[\begin{array}{c} \mathcal{A}(\mathsf{View}_t) = \\ f(\mathsf{History}_t) \end{array} \right] - \Pr\left[\begin{array}{c} \mathcal{S}(\mathsf{Trace}_t) = \\ f(\mathsf{History}_t) \end{array} \right] \right| \leq \mathsf{negl}(\mathsf{k})$$

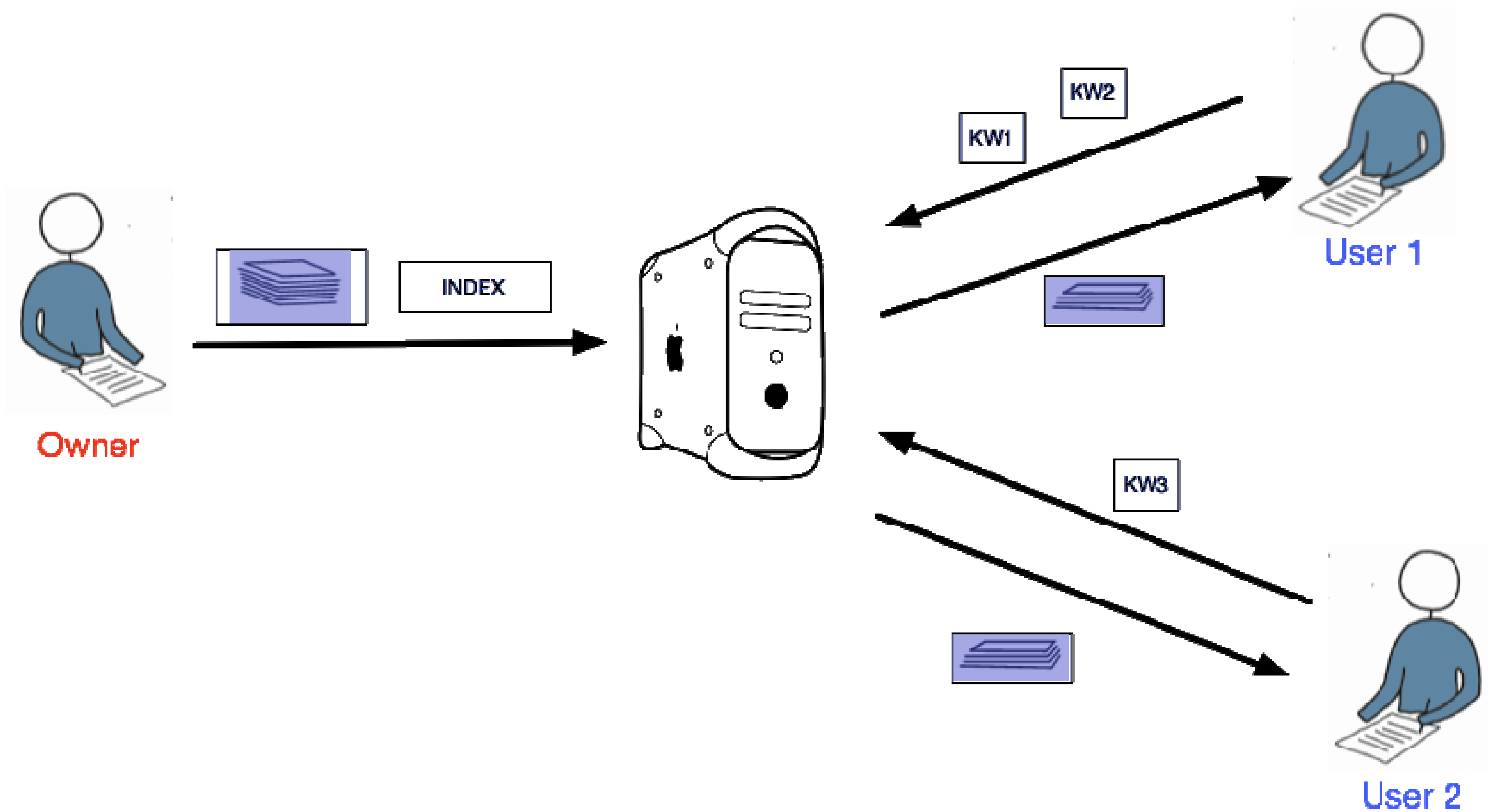
Adaptive SSE Security (cont'd)

- Simulator must be able to "*equivocate:*" "fake" trapdoors after having committed to an index
  - [Goh03, CM05] do not have this property
  - Unfortunately equivocation is
    - hard to achieve
    - expensive
- SSE-2:
  - Similar to SSE-1, with pre-processing and padding
  - Each occurrence of a word (in the collection) is treated as a unique word; this enables the simulator to equivocate
  - Constant blow-up in
    - size of trapdoors
    - size of index
    - server search time

Talk Outline

- MPC vs. PPC
- Searchable Symmetric Encryption
  - Revisiting security definitions for SSE
  - “Non-adaptive” SSE definitions and construction
  - “Adaptive” SSE definitions and construction
- Extensions

Multi-User SSE



Multi-User SSE (cont'd)

- Similar security notions to single-user SSE's
  - Secure indexes and trapdoors
- Revocation: owner can revoke searching privileges
  - Robust against user collusions
- Anonymity: server should not know who initiated search
- Simple construction that transforms single-user SSE scheme into multi-user SSE scheme
  - *Broadcast Encryption* (revocation)
  - PRPs

Open Questions

- Constant-round schemes that hide everything, even the access pattern
- Searching for Boolean combinations of words
  - Conjunctive searchable encryption [GSW04, PKL04, BW06]
  - Disjunctive searches?

References

- R. Curtmola, J. Garay, S. Kamara and R. Ostrovsky, "Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions," *ACM CCS 2006*.

Available from Cryptology ePrint archive:

`http://eprint.iacr.org/2006/210`

Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions

Reza Curtmola (JHU)
Juan Garay (Bell Labs)
Seny Kamara (JHU)
Rafail Ostrovsky (UCLA)

Lucent Technologies
Bell Labs Innovations