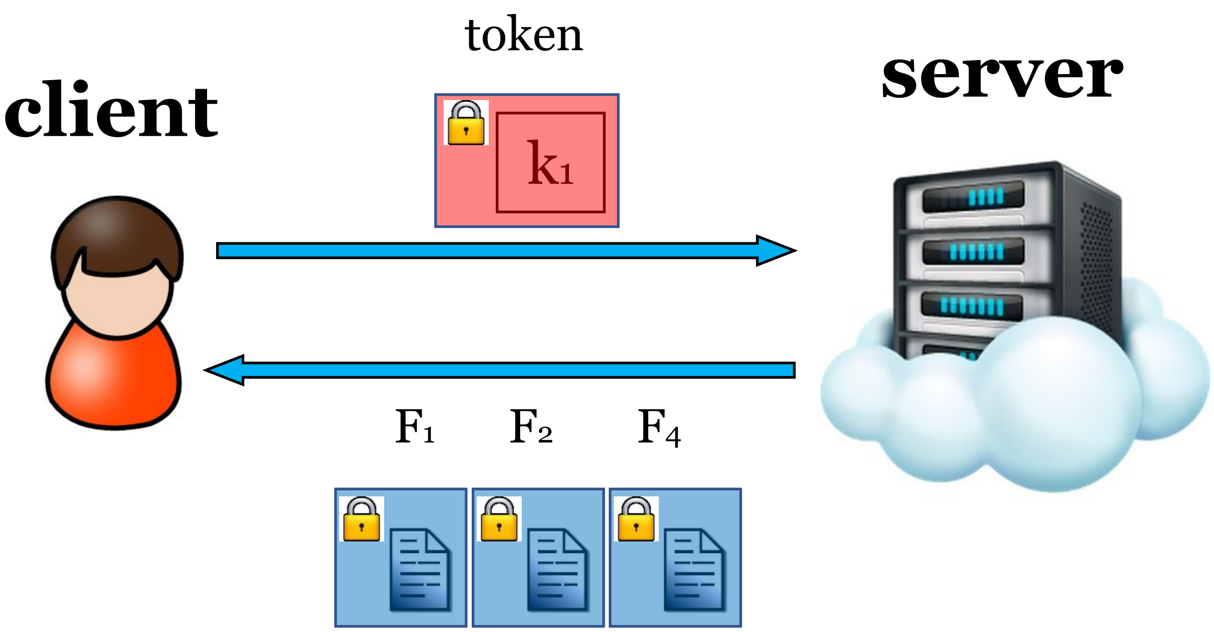
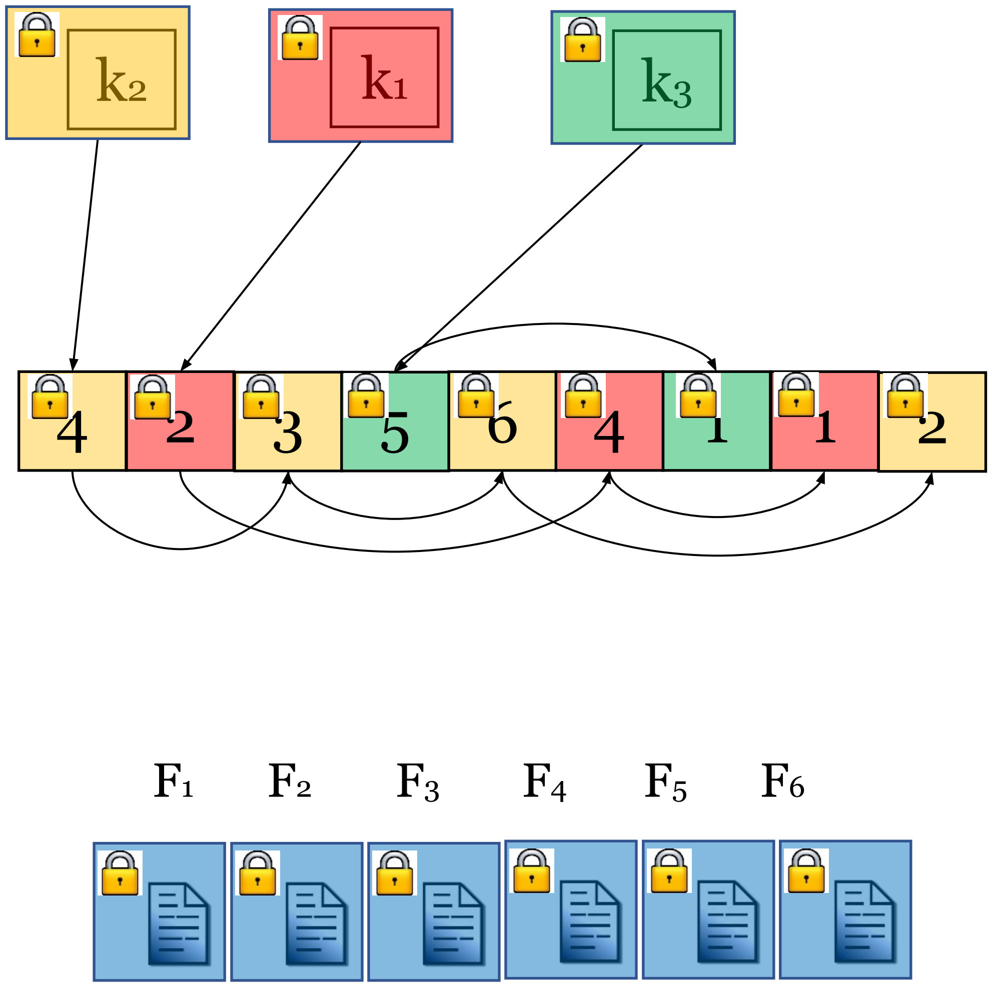
Attacks on Searchable Symmetric Encryption

Encrypted index

deterministic!





file access patterns!

Leakage

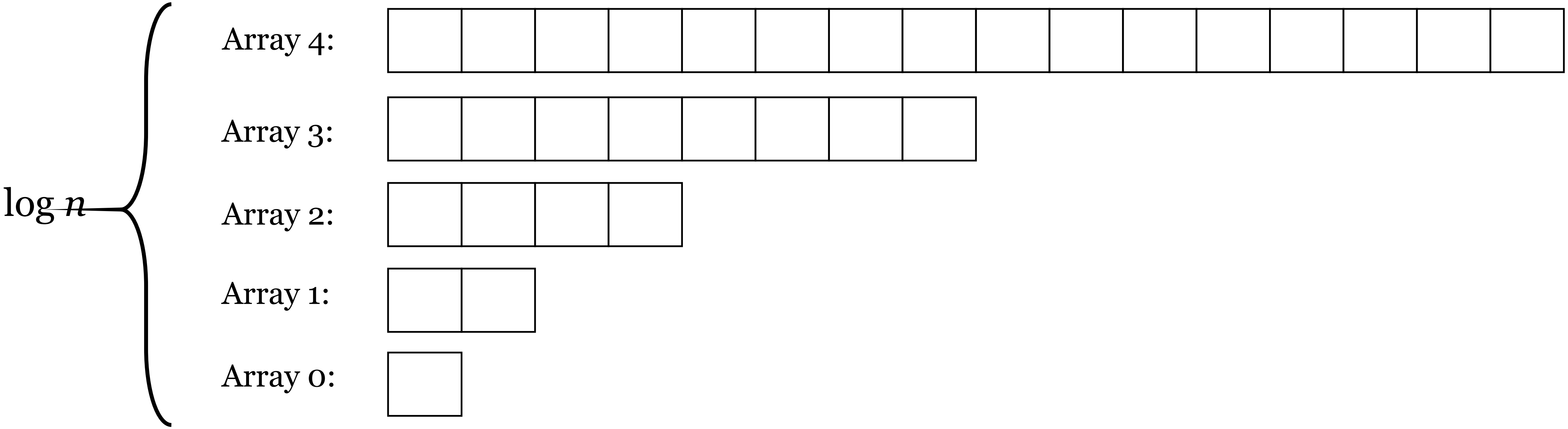
- Access pattern
- Search pattern

Forward and backward security

- Forward privacy: server cannot search on new files using old tokens

- Backward privacy: server cannot search on deleted files using new tokens

Dynamic SSE with forward privacy



log n arrays, each with 2^i elements

What are the consequences of the leakage?

- [IKK12] first paper to study attacks: query recovery attack

- Assumptions:
  - The plaintext of all documents are known (strong)
  - Access pattern (search pattern implicitly)

Idea of [IKK12]

- From search queries: matrix of (token – encrypted files) R
- From known plaintext: matrix of (keyword - files) M

R is a submatrix of a permutation of M

Find the best permutation with constraints:

  - Access pattern of single keyword
  - Intersections of multiple keywords

Optimization problem

- Transfer it to (joint) probability: continuous, good for optimization

$$\operatorname*{argmin}_{\langle a_1,\cdots,a_l\rangle} \sum_{\mathcal{Q}_i,\mathcal{Q}_j\in\mathcal{Q}} \left(\frac{R_{\mathcal{Q}_i}\cdot R^T_{\mathcal{Q}_j}}{n} - \left(\mathcal{K}_{a_i}\cdot M\cdot \mathcal{K}^T_{a_j}\right)\right)^2 \tag{1}$$

- NP-complete

$$Constraints:\quad \forall j \text{ s.t. } \mathcal{Q}_j\in\mathcal{S}, a_j = x_j \text{ s.t. } \langle \mathcal{K}_{x_j}, \mathcal{Q}_j\rangle \in K_Q$$
$$\forall j, \| \mathcal{Q}_j \| = 1$$

- Heuristics to find the solution

Justification of known plaintext assumption

- Common emails like announcements and ads

Counting attack [CGPR15]

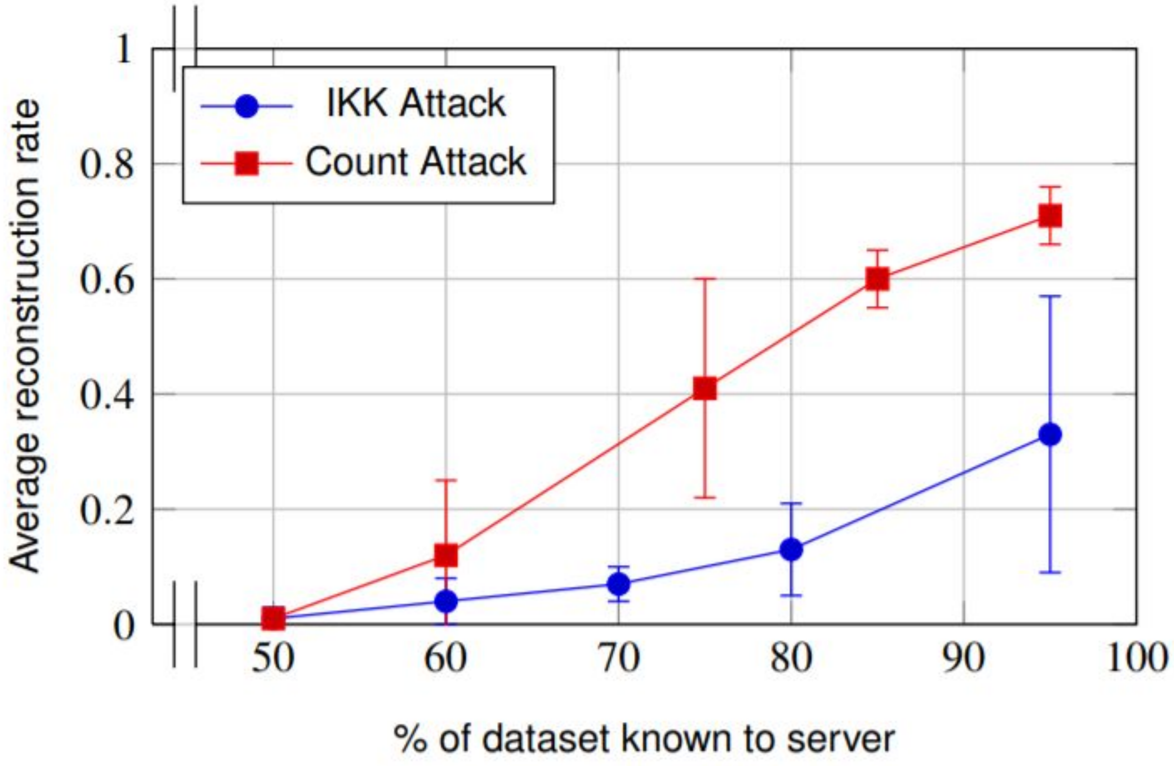
Greedy algorithm:

- when the “count” is unique, identify the keyword, remove it from the problem

- Use these keywords as references for “co-occurrence”

Counting attack [CGPR15]

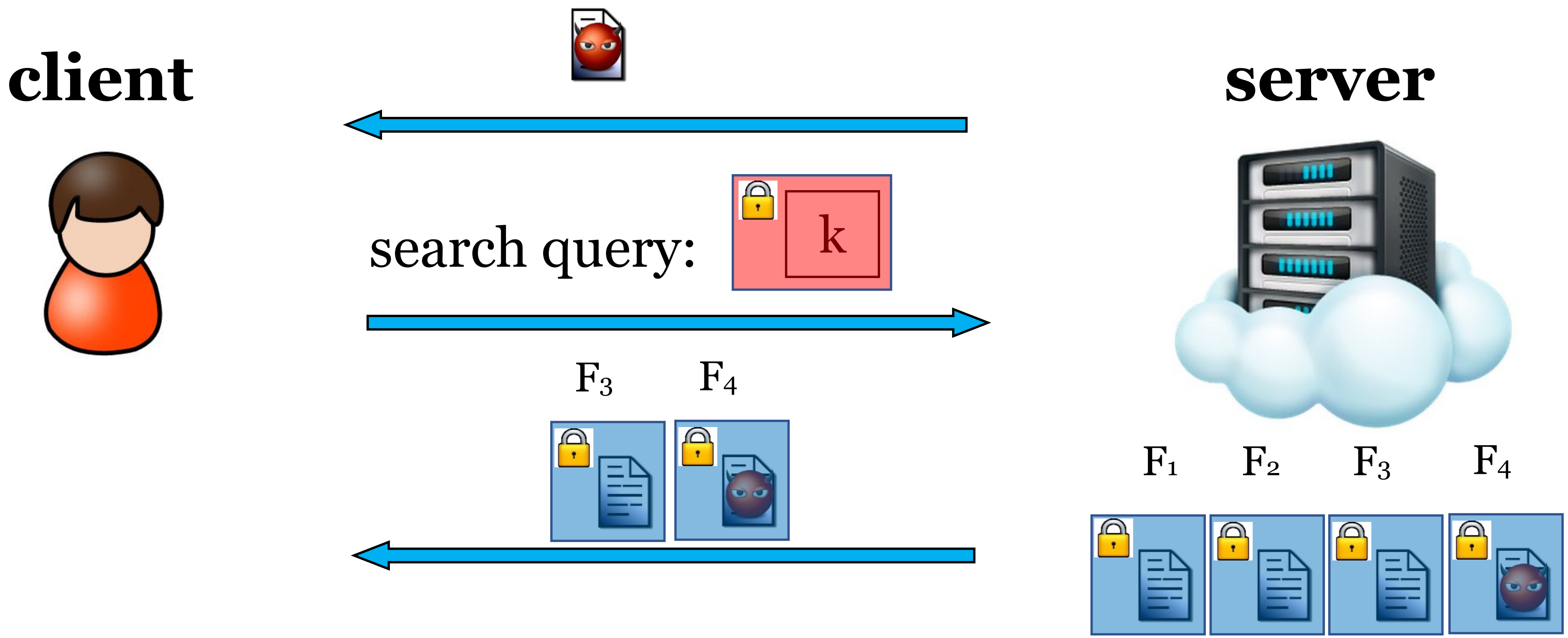
- Good performance in practice (100%)
- Still apply to partial knowledge of plaintext documents



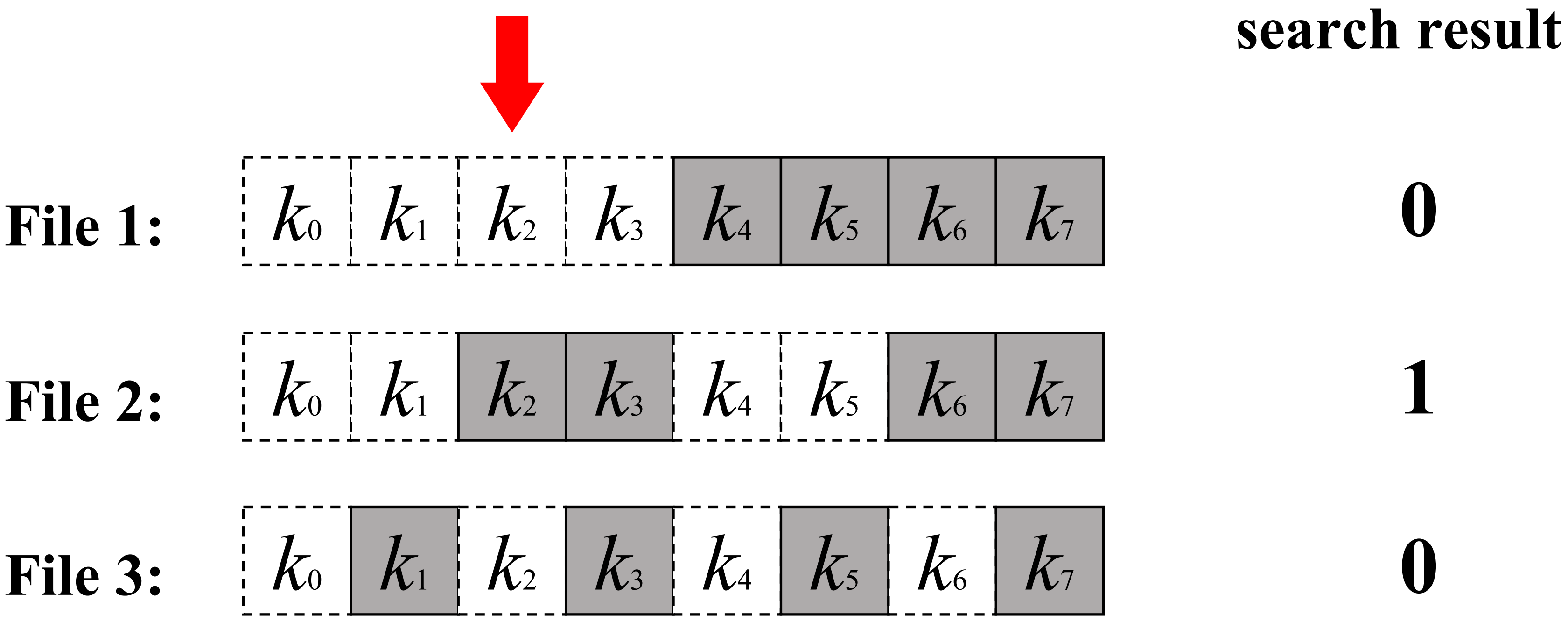
All known:
Count attack: 100%
IKK: 80%

Passive attacks vs active attacks

File-injection attacks [ZKP16]



Binary search using injected files



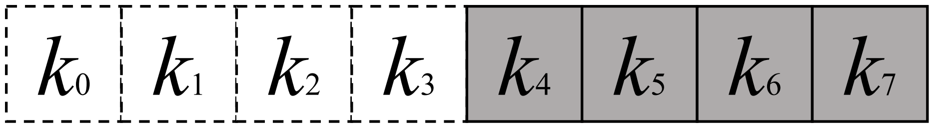
- Only inject log |K| files for a universe of |K| keywords.
- Can recover all queries with probability 1.
- Only use file access pattern leakage.
- Small universe

Limitations of the basic scheme

- Long injected files ($|K|/2$ keywords each)
- Meaningful emails?

Modifying the Attack

File 1:



File 1 **File 2**

- |K|/2T files of T keywords each to replace 1 file with |K|/2 keywords
- Hierarchical search |K|/2T ×log T
- Inject 131 files for |K|=5,000 and T=200

Properties

- Active attack: file injection
- Only access pattern, no plaintext

Better attacks with known plaintext

- Recovering target keyword(s)

- Use extra information to reduce the search base

1 Token

Frequency
of a token/keyword:

$$\frac{\text{\# of files containing it}}{\text{total \# of files}}$$

universe of keywords	estimated frequency	candidate universe: $f^*(k) \approx f(t)$	token	exact frequency
$\boxed{k_1}$	$f^*(k_1)$			
$\boxed{k_2}$	$f^*(k_2)$			
$\boxed{k_3}$	$f^*(k_3)$		$\boxed{t}$	$f(t)$
$\boxed{k_4}$	$f^*(k_4)$			
$\boxed{k_5}$	$f^*(k_5)$			

binary search attack

Multiple Tokens

1. Recover several keyword/token pairs as ground truth.

2. For a remaining token t', every keyword k',

   $f^*(k, k') \approx f(t, t')$ for all pairs (k,t) in ground truth
   → put k' into the candidate universe

3. Search.

Joint Frequency
of 2 tokens (keywords):

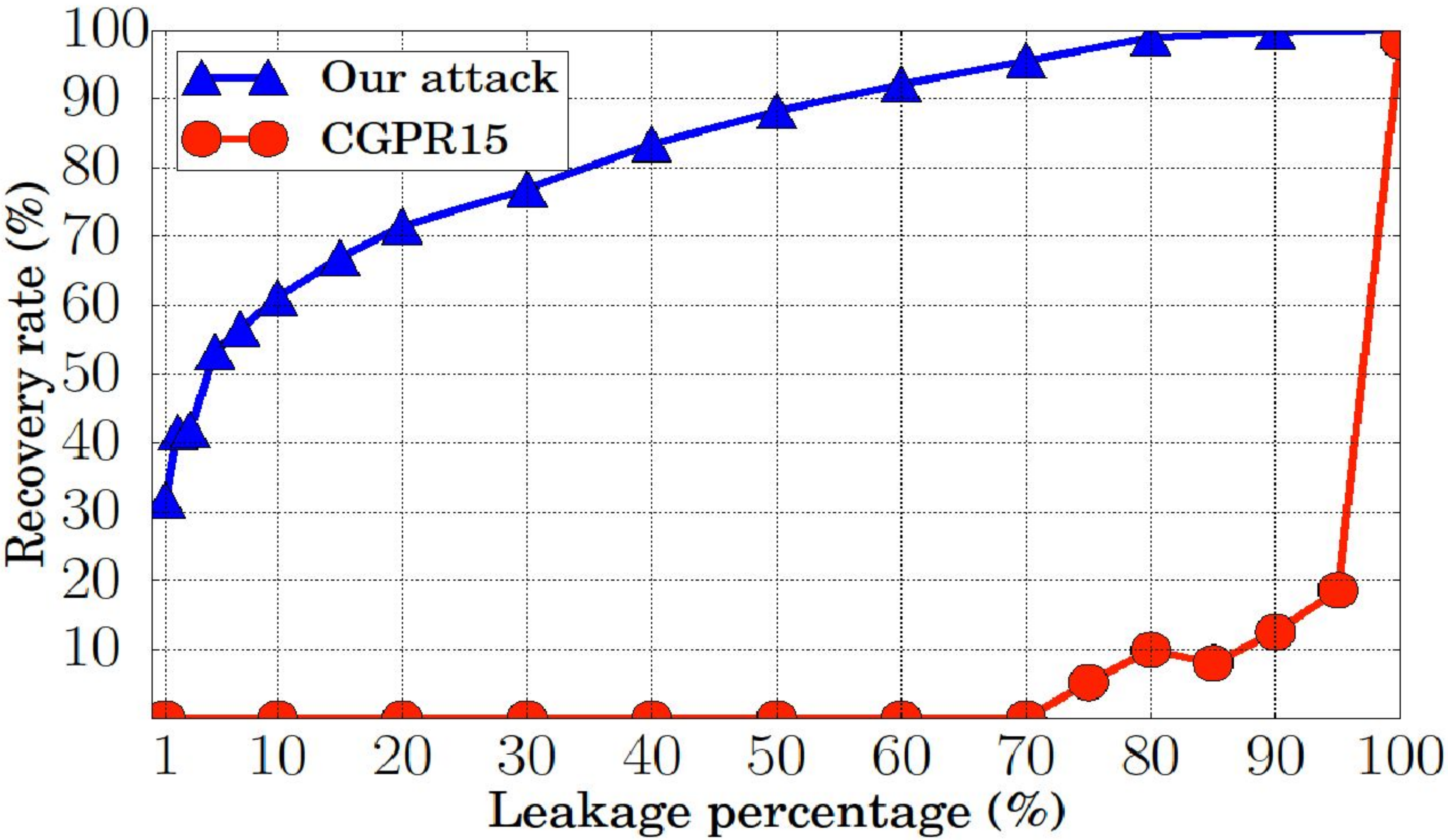
$$\frac{\text{\# of files containing them}}{\text{total \# of files}}$$

Properties

- Applies to SE schemes with no forward privacy, or token searched twice
- The server does not always succeed, but can determine whether attacks fail

Experimental Results: Recover 1 Query

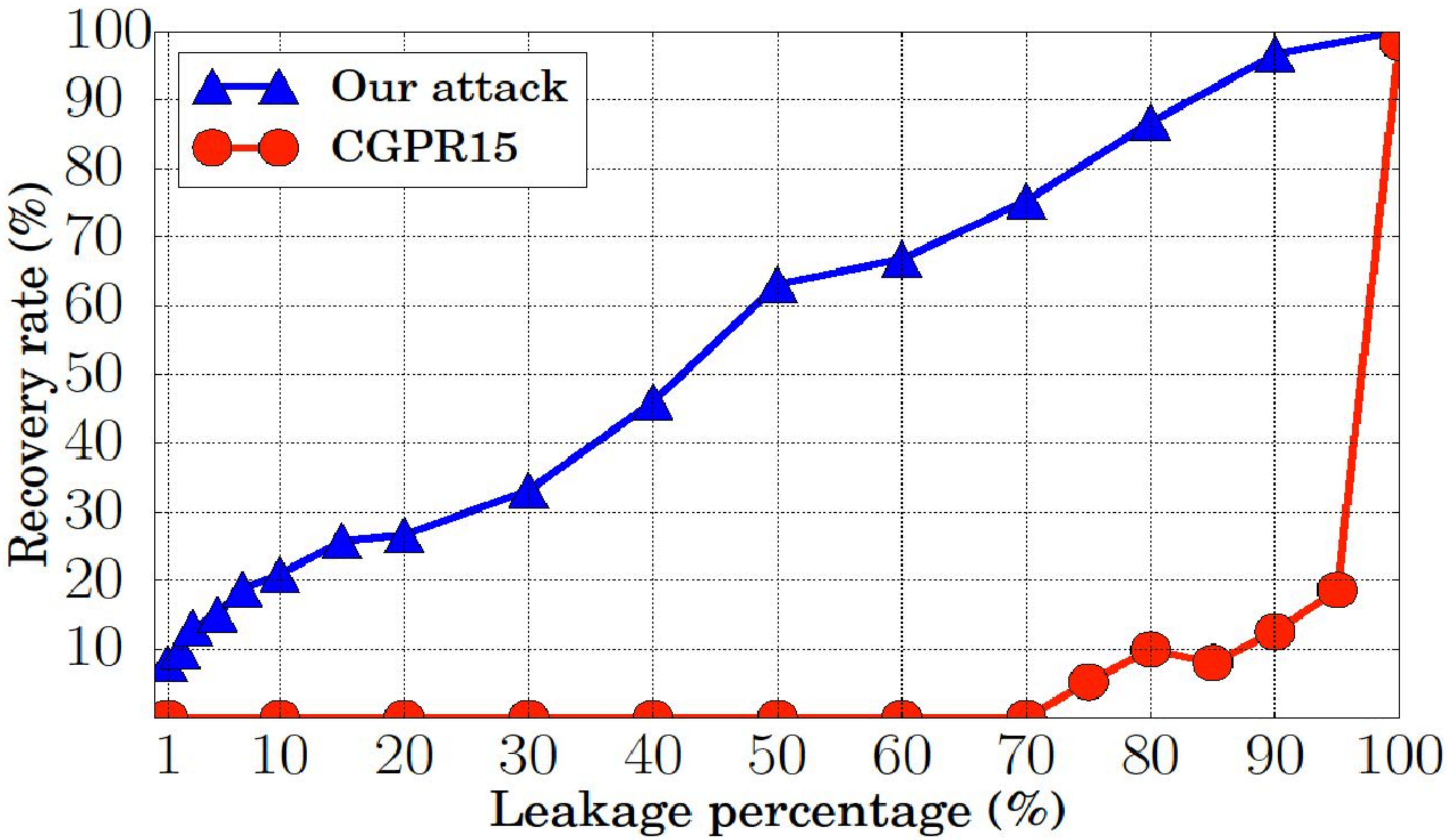
U = 5,000, T = 200, number of injected files = 9



different attack models!

Experimental Results: Recover 100 Queries

U = 5,000, T = 200, number of injected files <= 40



Insights

- Prior attacks: find the best match between keywords and tokens.
  uniqueness of the frequency: distorted when less files are leaked.
- File injection attacks: rule out bad matches, search on the remaining ones.

Conjunctive SE

- Search files with d keywords k_1, k_2, … k_d.

- Ideal leakage: only leak the intersection of their search results. (No existing scheme achieves ideal leakage.)

Countermeasures

- Padding: return fake results to change the count/frequency
  - Work for frequency analysis (IKK12, CGPR15)
  - Doesn’t work well for file injection