All Your Queries are Belong to Us:

The Power of File-Injection Attacks on Searchable Encryption

Yupeng Zhang , Jonathan Katz, Charalampos Papamanthou
University of Maryland







Agenda

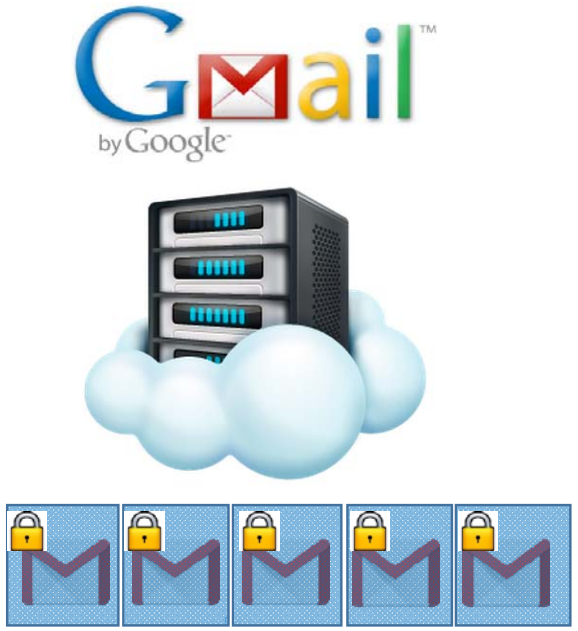
- Background on Searchable Encryption
- Attacks on Searchable Encryption
- Experimental results
- Conclusions

Email system

client



Search?



Privacy?

Searchable Encryption

What is Searchable Encryption?



client

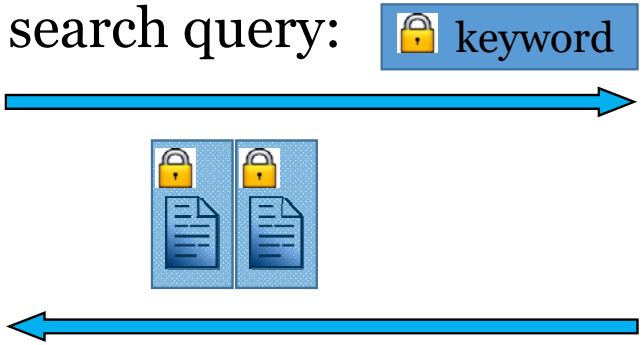
server







search query: keyword

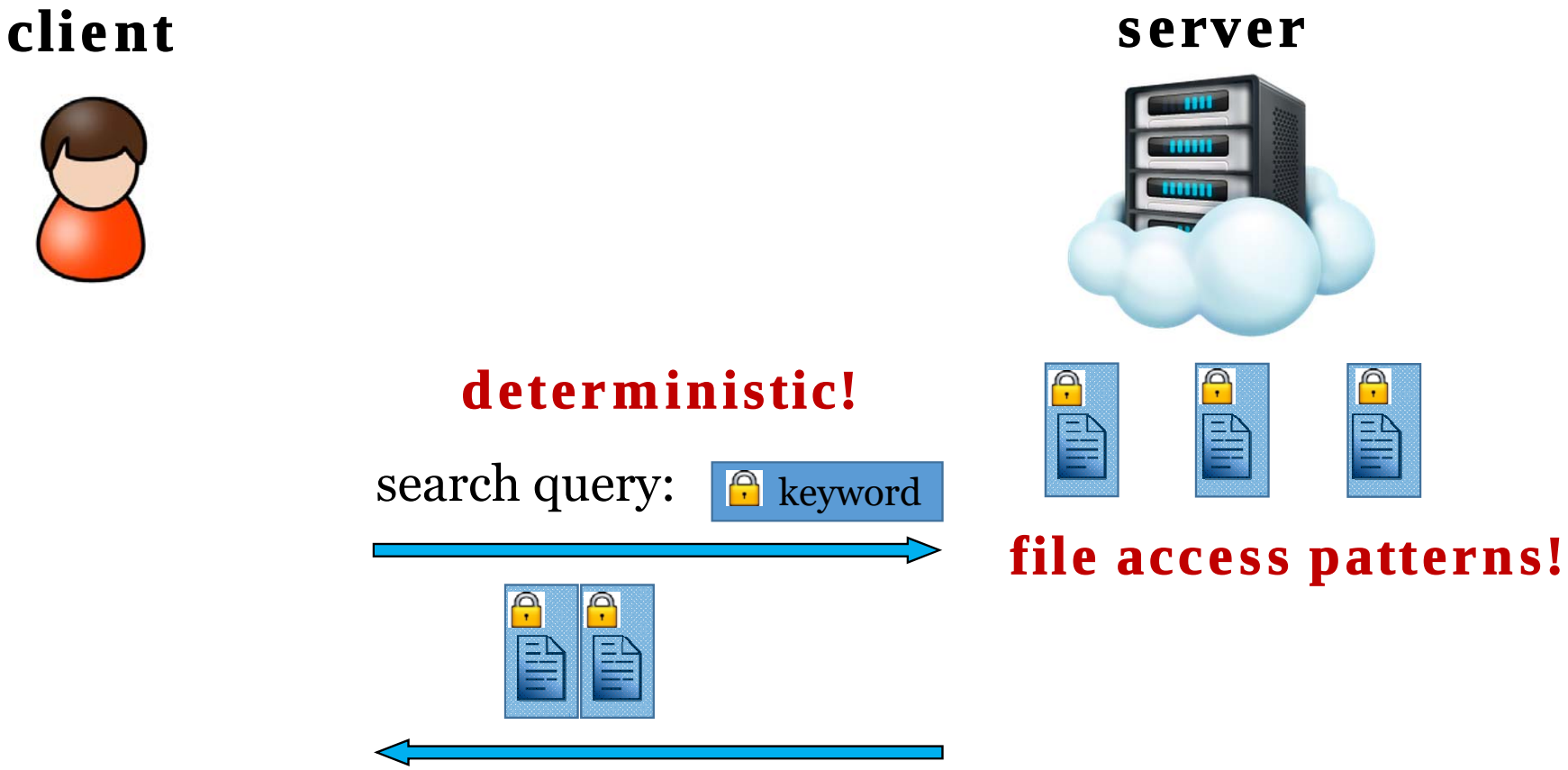


Applications

- Encrypted Storage: Skyhigh Networks, CipherCloud
- Encrypted Emails

Leakage

Leakage of Searchable Encryption

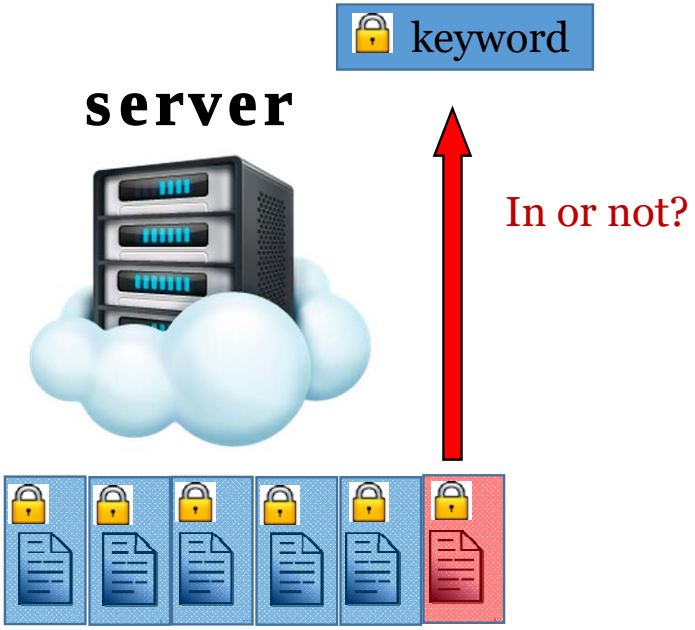


Leakage of Searchable Encryption

client



server



No forward privacy!

new file:

Leakage of Searchable Encryption

- Search pattern leakage: can tell when query repeats.
- Access pattern leakage: can tell whether a file is returned.

Leaked by all efficient searchable encryption schemes.

- No Forward Privacy: can search old tokens on new files.

All SE schemes except [CM05, SPS14, Bost16] do not have forward privacy.

What information does this leakage reveal?

Prior Attacks on Searchable Encryption

- Islam et al. (IKK12) proposed a **query recovery attack**.
- Cash et al. (CGPR15) proposed another attack with higher success probability.

These attacks assume:

the server knows all the client's files in plaintext.

Main Contributions

- We study the file-injection attacks thoroughly. (First proposed in CGPR15).
- We present attacks that significantly improve the success probability.
- Eliminate or relax the client's file leakage assumption.
- Extends to conjunctive search.

We suggest reducing or eliminating these leakages, instead of accepting them by default.

Attack Target: Query Recovery Attacks

Why query privacy is important?

Practical:

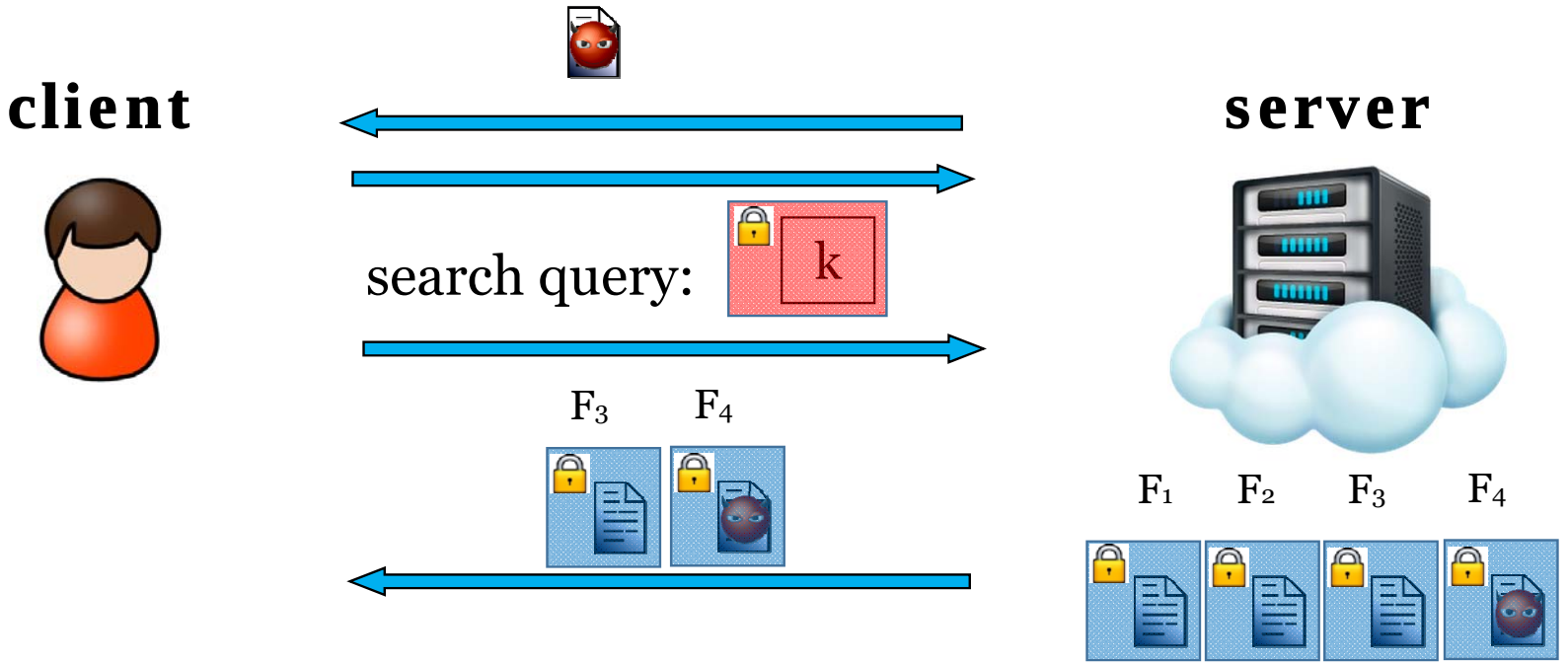
- Keywords are part of the files. File content can be recovered. (CGPR15)
- Keywords can be used to classify files and help other attacks.

Theoretical:

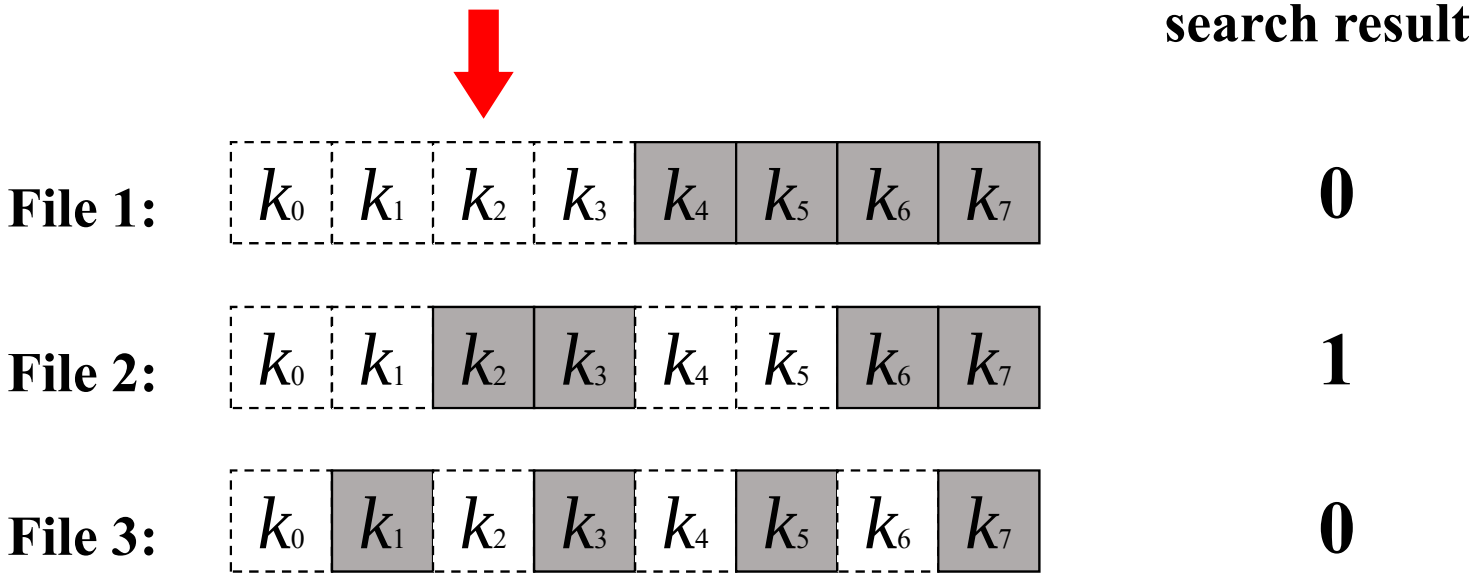
- Unexpected vulnerabilities if searchable encryption is used as a building block.

Our Attacks

Attack Model: File-injection Attack



Binary Search Attack



- Only inject **14** files for a universe of 10,000 keywords.
- Can recover all queries with probability **1**.
- Inject before seeing the queries (**non-adaptive**).
- Only use **file access pattern** leakage.
- Universe defined by the server (small universe).

Limitation

Long injected files (**|K|/2** keywords each).

Threshold Countermeasure

Filter all files that contain more than **T** keywords.

- Index only T keywords in a file that has more than T keywords.

Enron data set: **30,109** files, universe of **5,000** keywords

Only **3%** of files have more than **T=200** keywords.

Enron email dataset. https://www.cs.cmu.edu/~./enron/. Accessed: 2015-12-14.

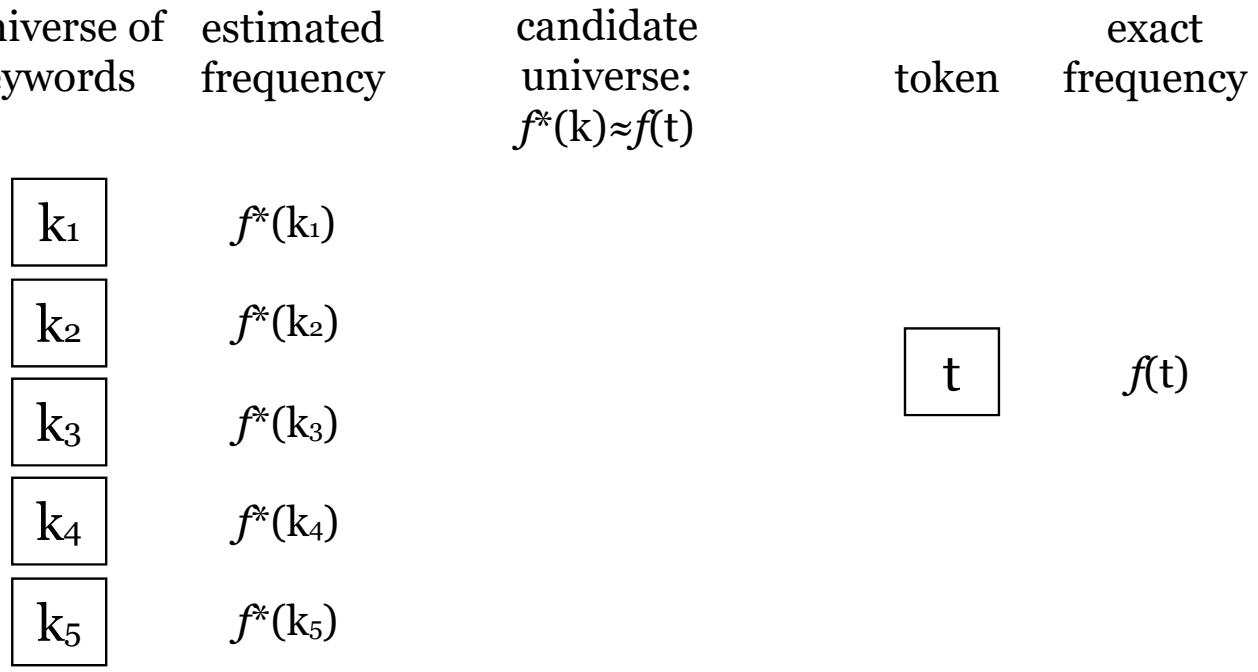
Attacks with Partial File Leakage

- The server learns a portion of client's files in plaintext.
  (Announcement and alert emails broadcasted to many people)

Attacks to Recover 1 Token

Frequency
of a token/keyword:

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



Difference from Binary Search Attack

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

Attacks to Recover Multiple Tokens

Refer to our paper for an attack to recover **multiple tokens**.

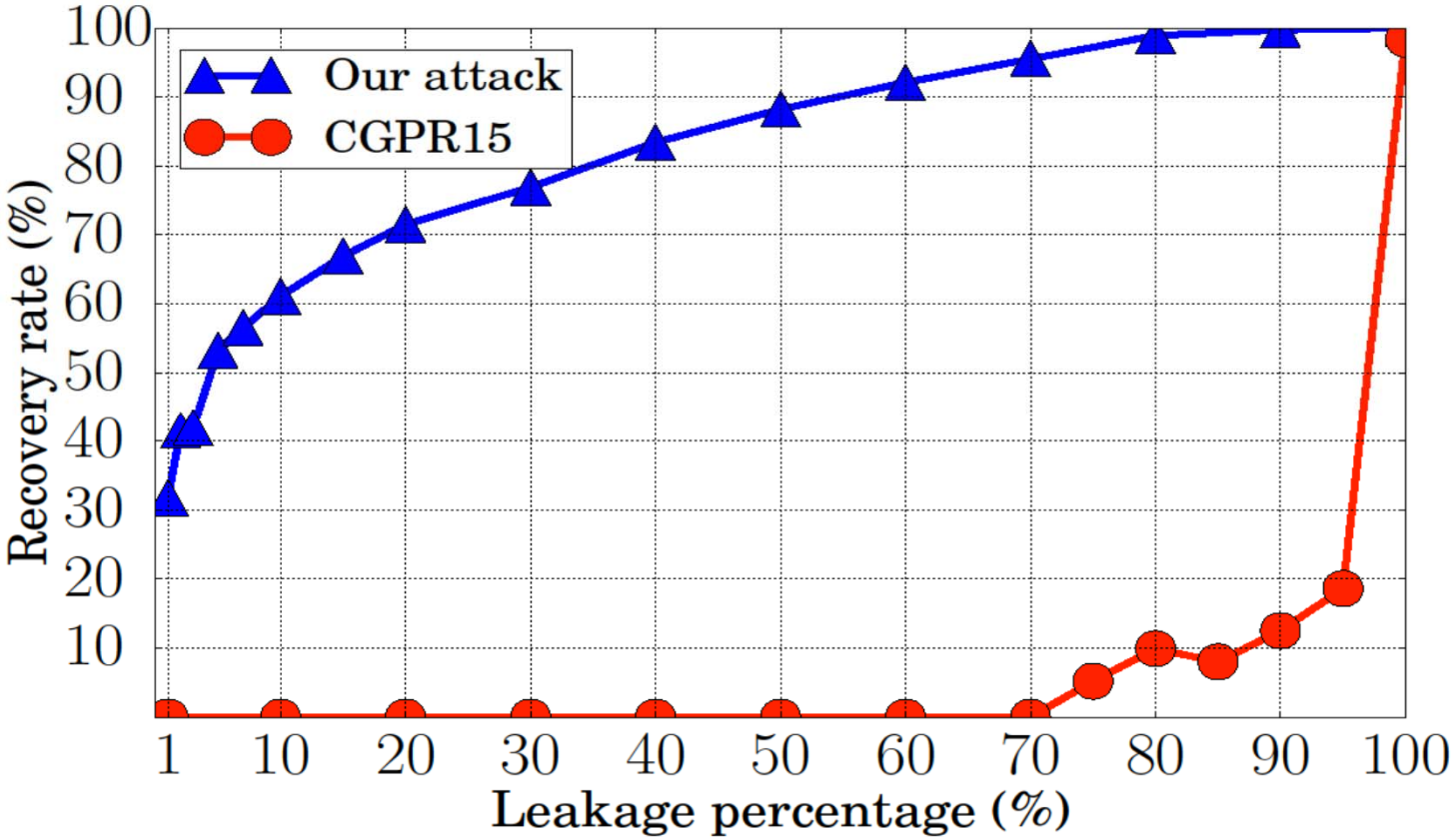
Experiments

Experimental Methodology

- Enron data set with 30,109 emails.
- Choose top 5,000 keywords with highest frequency as the universe.

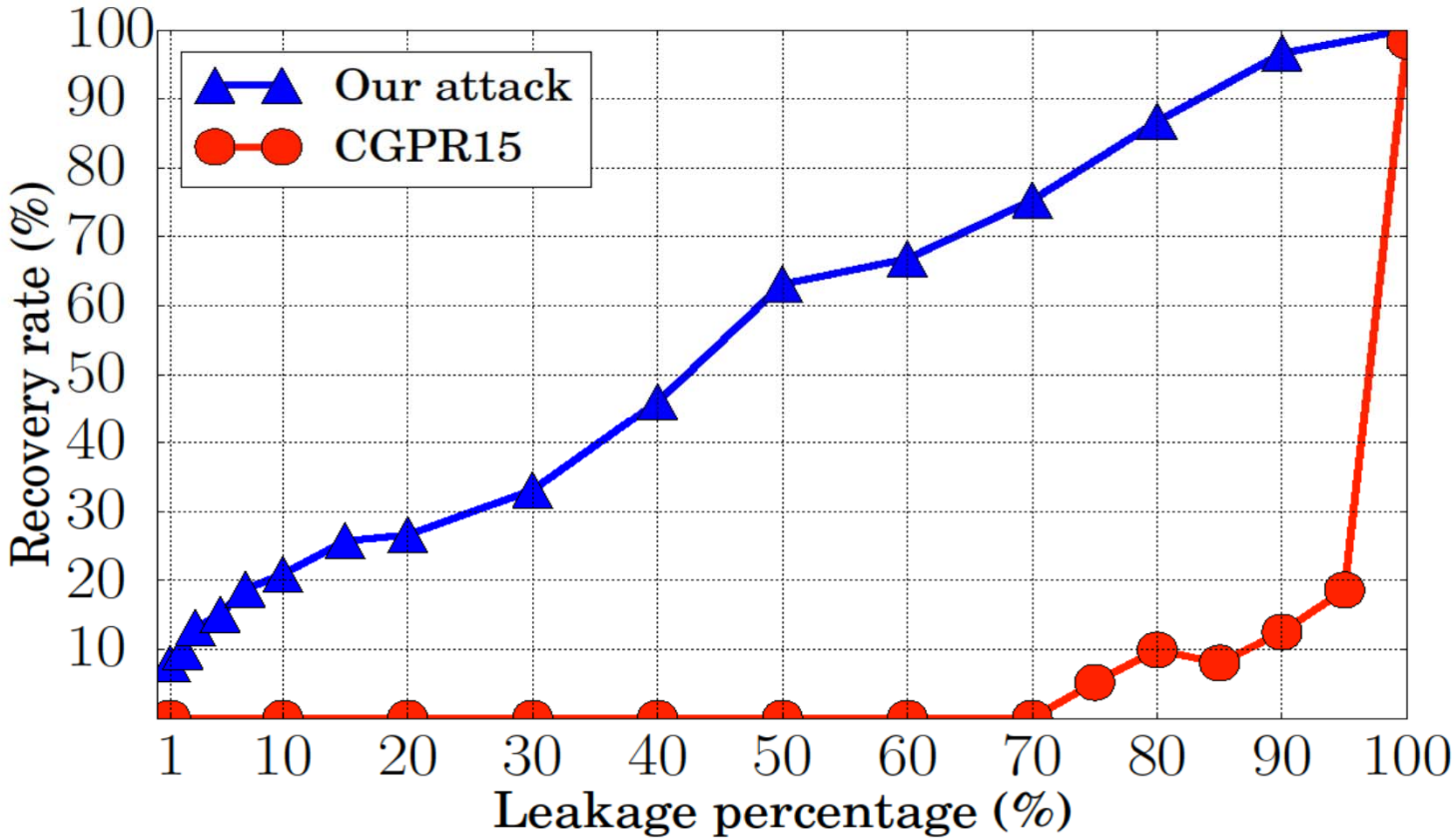
Experimental Results: Recover 1 Query

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



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 is distorted when less files are leaked.

- Our attacks: rule out bad matches, search on the remaining ones.

Conjunctive SE

Our attacks can be extended to **conjunctive** searchable encryption. Refer to our paper for details.

Countermeasures

Search Result Padding

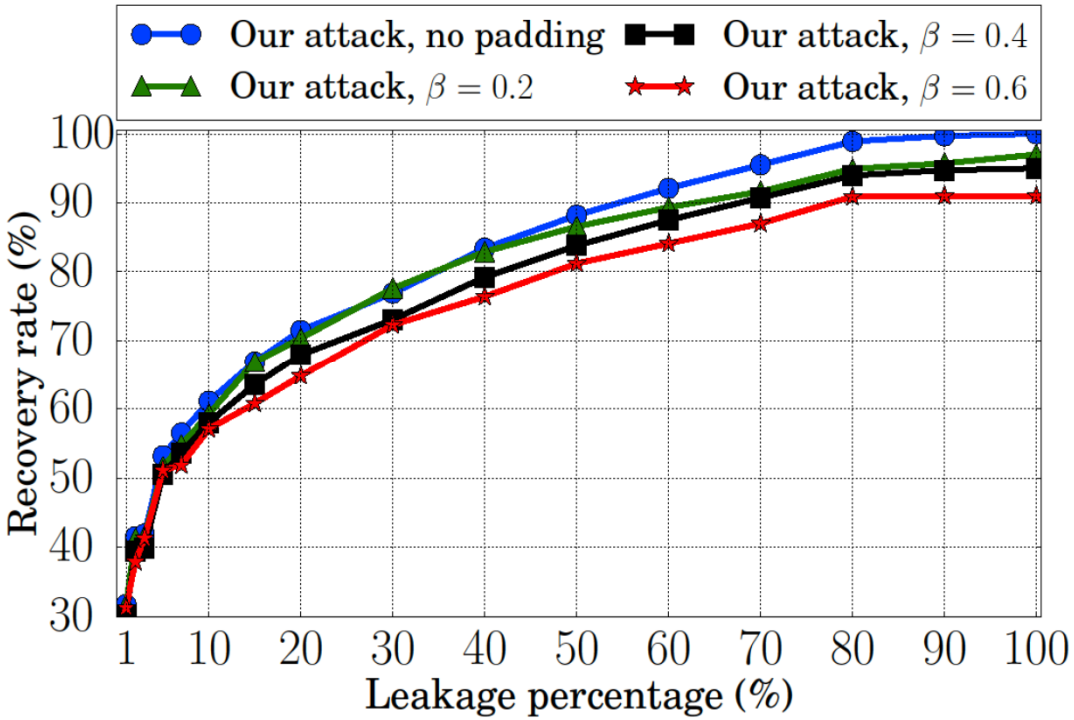
Pad the search result with random files s.t. multiple tokens have the same frequency.

- Does not affect the binary search attack.

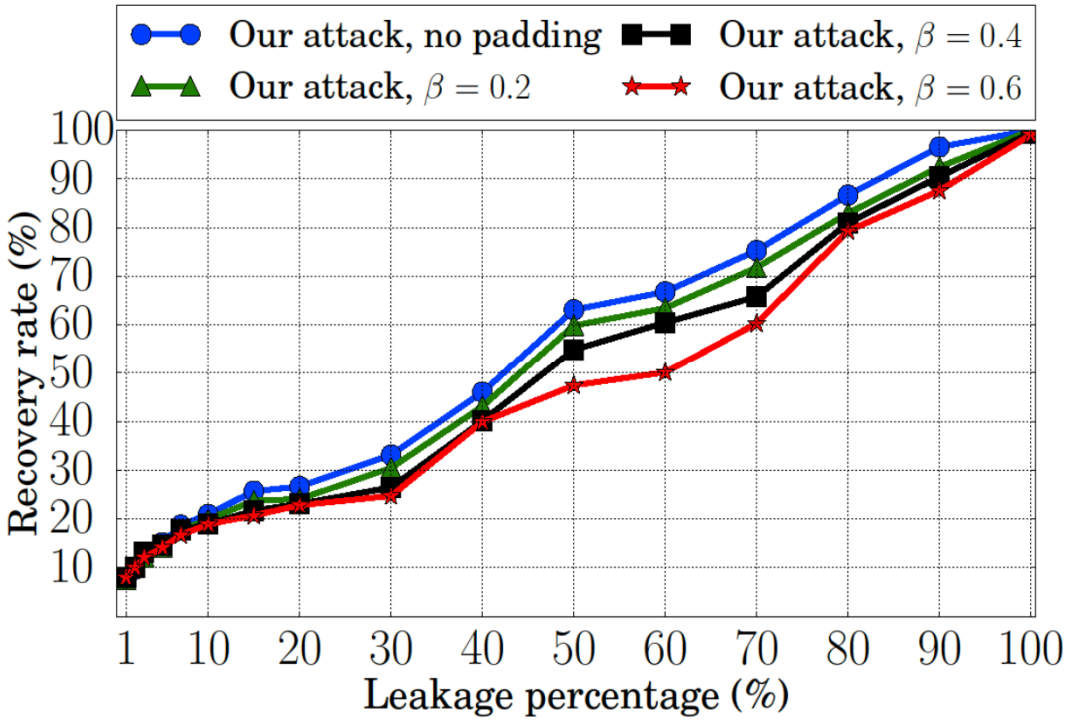
- Does not affect the advanced attacks:
  Close frequencies are still close after padding.

Search Result Padding: Experiments

β_k: # of padded files for keyword k / the original # of files containing keyword k
β: average of all β_k



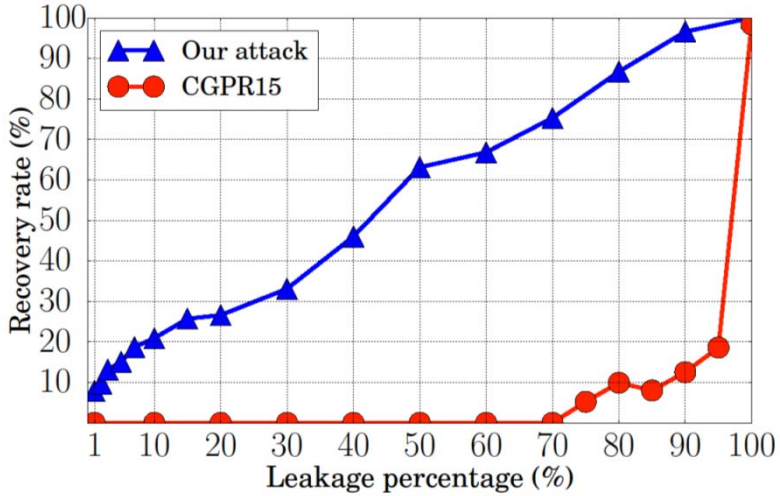
Attacking 1 token



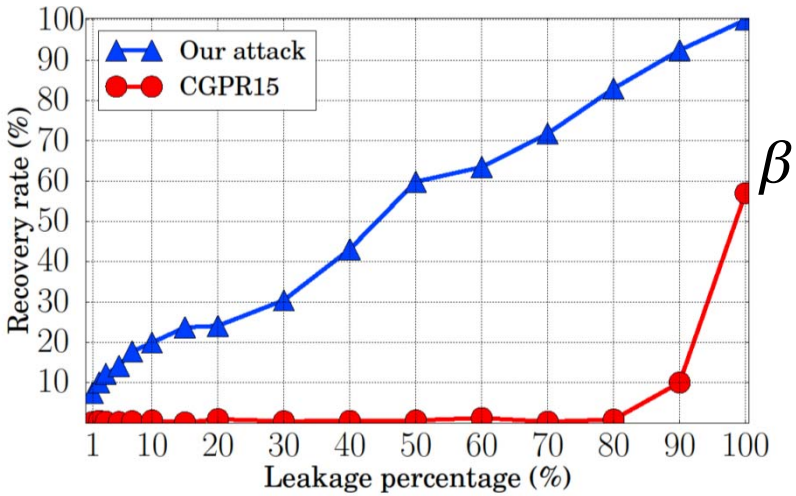
Attacking 100 tokens

Padding

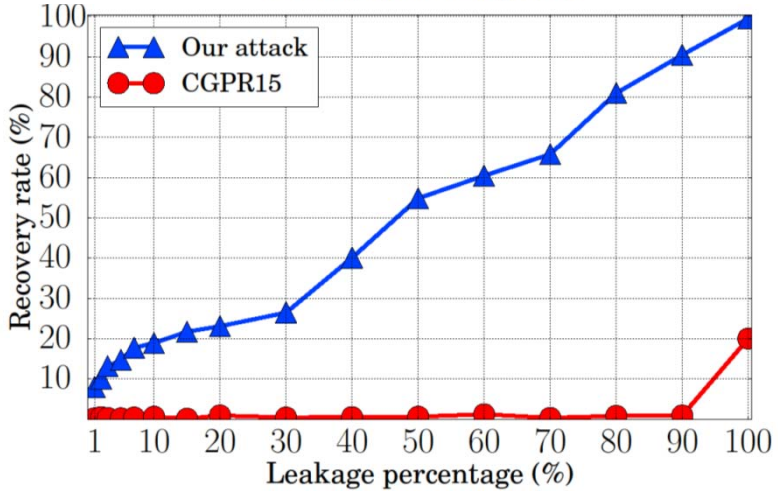
$\beta = 0$



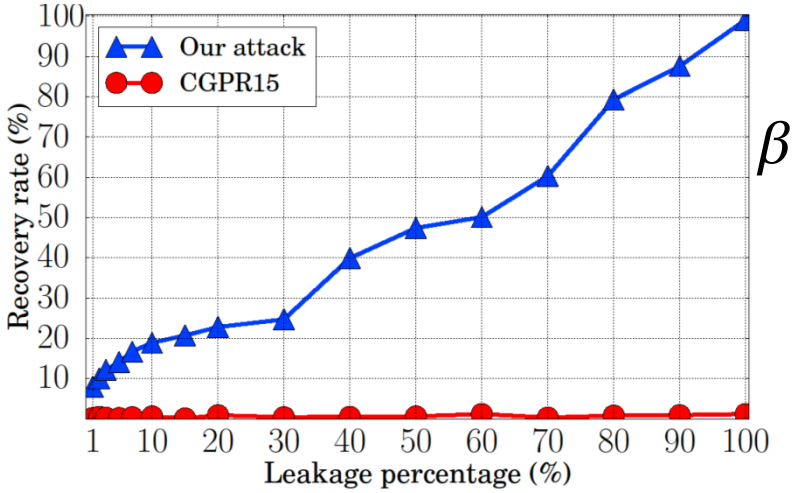
$\beta = 0.2$



$\beta = 0.4$



$\beta = 0.6$



Other Countermeasures

- File length padding.

Partially works.

1. Storage overhead. E.g. in Enron data set, **1000x** overhead.
2. Dynamic case: **timing**.

- Batched updates.

Partially works.

1. 1 injected file per batch: attacks succeed with **some** probability.
2. **Repeat** 1 injected file many times: attacks succeed with **good** probability.

Conclusions

- File-injection attacks are devastating for query privacy in SE.
- Is it a satisfactory tradeoff between efficiency and leakage for existing SE?
- Future research:
  - Reduce or eliminate access pattern leakage.
  - Exploring new directions such as multi-server schemes.
- Forward Privacy.

Thank you for listening!