SAC Summer School 2019

Encrypted Search: Leakage Attacks

Seny Kamara

BROWN

ENCRYPTED SYSTEMS LAB

How do we Deal with Leakage?

- Our definitions allow us to prove that our schemes
  - achieve a certain leakage profile
  - but doesn’t tell us if a leakage profile is exploitable?
- We need more than proofs

The Methodology

- Leakage analysis: what is being leaked?
- Proof: prove that scheme leaks no more
- Cryptanalysis: can we exploit this leakage?

Leakage Attacks

- Target
  - *query recovery*: recovers information about query
  - *data recovery*: recovers information about data
- Adversarial model
  - *persistent*: needs EDS and tokens
  - *snapshot*: needs EDS
- Auxiliary information
  - *known sample*: needs sample from same distribution
  - *known data*: needs actual data
- Passive vs. active
  - *injection*: needs to inject data

Leakage Attacks

- Inference attacks ≈ (passive) known-sample attacks
  - [Islam-Kuzu-Kantarcioglu12]*
    - persistent query-recovery vs. SSE with baseline leakage
  - [Naveed-K.-Wright15,…]
    - snapshot data-recovery vs. PPE-based encrypted databases
  - [Kellaris-Kollios-Nissim-O'Neill,…]
    - persistent query-recovery vs. encrypted range schemes

Leakage Attacks

- Leakage-abuse attacks ≈ (passive) known-data attacks
  - [Cash-Grubbs-Perry-Ristenpart15]
    - persistent query-recovery vs. SSE with baseline leakage
- Injection attacks ≈ (active) chosen-data attacks
  - [Cash-Grubbs-Perry-Ristenpart15]
    - persistent query-recovery vs. non-SSE-based solutions
  - [Zhang-Papamanthou-Katz16]
    - persistent query-recovery vs. SSE with baseline leakage

Typical Citations

- “For example, IKK demonstrated that by observing accesses to an encrypted email repository, an adversary can infer as much as 80% of the search queries”

- “It is known that access patterns, to even encrypted data, can leak sensitive information such as encryption keys [IKK]”

- “A recent line of attacks […,Count,…] has demonstrated that such access pattern leakage can be used to recover significant information about data in encrypted indices. For example, some attacks can recover all search queries [Count,…] …”

IKK Attack

[Islam-Kantarcioglu-Kuzu12]

- Published as an inference attack
  - persistent *known-sample* query-recovery attack
  - exploits co-occurrence pattern + knowledge of **5%** of queries
    - co-occur: times each pair of documents occur together
- Highly cited but significant limitations
  - experiments only for **2500** out of **77K+** keywords
  - auxiliary and test data were not independent
- [CGPR15] re-ran IKK on independent test data
  - it achieved **0%** recovery

IKK as a Known-Data Attack

[Islam-Kantargioglu-Kuzu12, Cash-Grubbs-Perry-Ristenpart15]

- What if we just give IKK the client data; does it work then?
- Notation
  - δ: fraction of adversarially-known data
  - φ: fraction of adversarially-known queries
- [CGPR15] experiments for IKK attack
  - δ = 70% + φ = 5% recovers 5% of queries
  - δ = 95% + φ = 5% recovers 20% of queries

The Count Attack

[Cash-Grubbs-Perry-Ristenpart15]

- Known-data attack (i.e., “leakage-abuse attack”)
  - Count v.1 [2015] and Count v.2 [2019]
  - exploit co-occurrence pattern + response length
- Count v.1
  - δ = **80%** + $\varphi = 5\%$ recovers **40%** of queries
  - δ = **75%** + $\varphi = 5\%$ recovers **0%** of queries
- Count v.2
  - δ = **75%** recovers **40%** of queries

Revisiting Leakage-Abuse Attacks

- High known-data rates ($\delta \geq 75\%$)
  - how can an adversary learn **75%** of client data?
  - recall that when outsourcing, client *erases* plaintext
  - if client needs to outsource public data *it should use PIR*
- Known queries ($\varphi \geq 5\%$)

Revisiting Leakage-Abuse Attacks

- Low-vs. high selectivity keywords
  - Experiments all run on high-selectivity keywords
  - We re-ran on low-selectivity keywords and attacks failed
- Both exploit co-occurrence pattern
  - relatively easy to hide (see OPQ [Blackstone-K.-Moataz19])

Revisiting Leakage-Abuse Attacks

- Should we discount the IKK and Count attacks?
  - No! they are interesting, just not necessarily practical
- Theoretical attacks (e.g., Count, IKK)
  - rely on strong assumptions, e.g., $\delta > 20\%$ or $\varphi > 20\%$
- Practical attacks (e.g., [Naveed-K.-Wright15] vs. PPE-based)
  - weak adversarial model
  - mild assumptions (*real-world* auxiliary input)

Q: can we do better than IKK & Count?

New Known-Data Attacks

[Blackstone-K.-Moataz19]

δ needed for RR ≥ 20%

Attack	Type	Pattern	Known Queries	δ for HS	δ for PLS	δ for LS
IKK	known-data	co	Yes	≥95%	?	?
Count	known-data	rlen	Yes/No	≥80%	?	?
Injection	injection	rid	No	N/A	N/A	N/A
Subgrap^{ID}	known-data	rid	No	≥5%	≥50%	≥60%
Subgraph^{VL}	known-data	vol	No	≥5%	≥50%	δ=1 recovers<10%
VolAn	known-data	tvol	No	≥85%	≥85%	δ=1 recovers<10%
SelVolAn	known-data	tvol, rlen	No	≥80%	≥85%	δ=1 recovers<10%
Decoding	injection	tvol	No	N/A	N/A	N/A

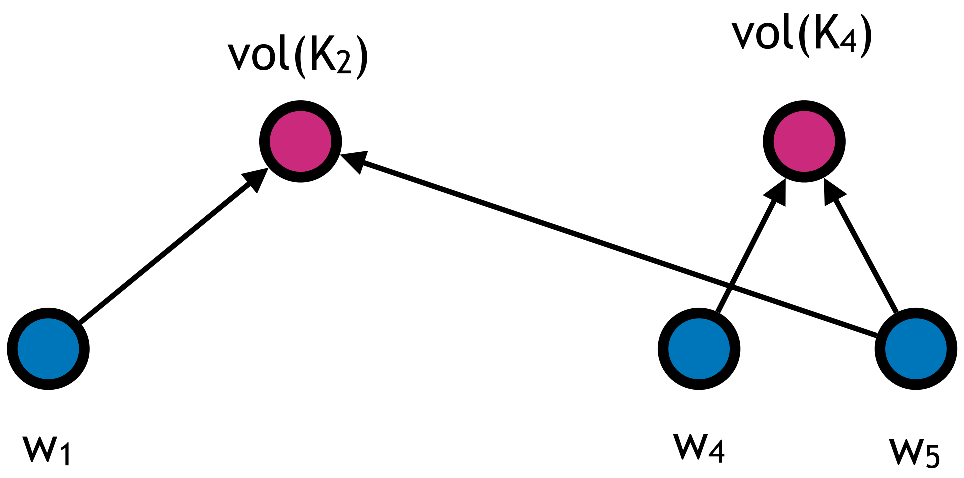
HS ≥ 13
PLS = 10-13
LS = 1-2

Apply to ORAM

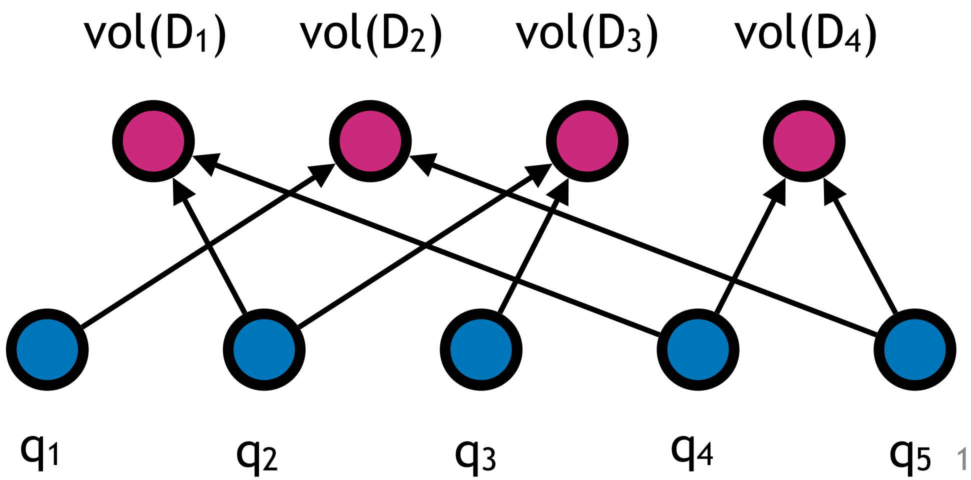
The Subgraph^{VL} Attack
[Blackstone-K.-Moataz19]

- Let $\mathbf{K} \subseteq \mathbf{D}$ be set of known documents
  - $\mathbf{K} = (K_2, K_4)$ and $\mathbf{D} = (D_1, \ldots, D_4)$

Known Graph



Observed Graph

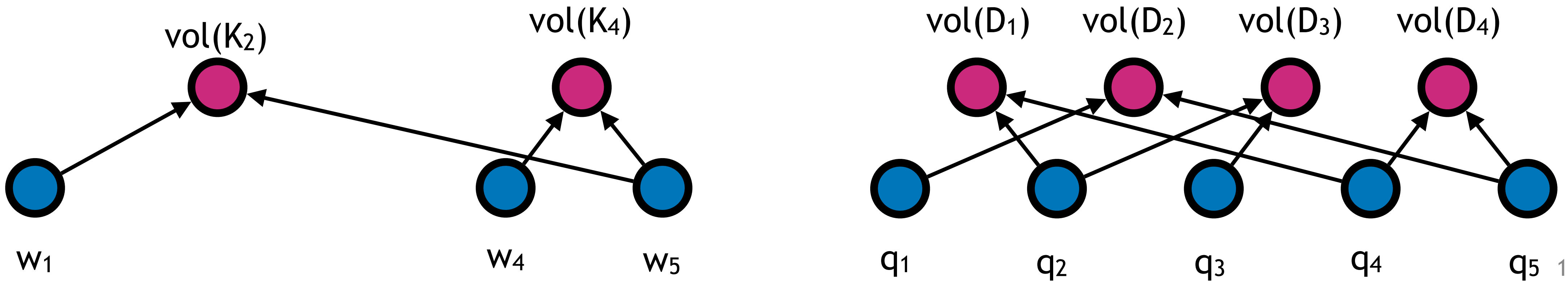


The Subgraph^{VL} Attack
[Blackstone-K.-Moataz19]

- We need to match $\mathbf{q_i}$ to some $\mathbf{w_j}$
- Observations: if $\mathbf{q_i} = \mathbf{w_j}$ then
  - $N(w_j) \subseteq N(q_i)$ and $\#N(w_j) \approx \delta N(q_i)$
  - $\mathbf{w_j}$ cannot be a match for $\mathbf{q_z}$ for $\mathbf{z \neq i}$

Known Graph

Observed Graph



The Subgraph^{VL} Attack
[Blackstone-K.-Moataz19]

- Each query **q** starts with a candidate set $\mathbf{C_q} = \mathbb{W}$
  - remove all words that have been matched to other queries
  - remove all words s.t. either $\mathbf{N(w_j)} \not\subseteq \mathbf{N(q_i)}$ or $\#\mathbf{N(w_j)} \not\approx \delta\mathbf{N(q_i)}$
  - if a single word is left that's the match
    - remove it from other queries' candidate sets

Revisiting Leakage-Abuse Attacks

[Blackstone-K.-Moataz19]

- ORAM-based search is also vulnerable to known-data attacks
- Subgraph attacks are practical for high-selectivity queries
  - can exploit **rid** or **vol**
  - need only $\boldsymbol{\delta \geq 5\%}$
- Countermeasures
  - for $\boldsymbol{\delta < 80\%}$ use OPQ [Blackstone-K.-Moataz19]
  - for $\boldsymbol{\delta \geq 80\%}$ use PBS [K.-Moataz-Ohrimenko18]
  - or use VLH or AVLH [K-Moataz19]

File Injection Attacks

[Zhang-Katz-Papamanthou16]

- Adversary tricks client into adding files
- For **i** = 1 to **log(#𝕎)**
  - inject document **D_i** = {all keywords with **i**th bit equal to **1**}
- Observation
  - if **D_i** is returned then adversary knows **i**th bit of keyword is **1**
  - otherwise **i**th bit of keyword is **0**
- When client makes a query,
  - if **D_4, D_8, D_{10}** are returned then **w** = **0001000101**

File Injection Attacks

[Zhang-Katz-Papamanthou16]

- Requires injecting documents of size
  - $2^{\log(\#\mathbb{W}) - 1} = \#\mathbb{W}/2$ keywords
- What if client refuses to add documents of size $\geq \#\mathbb{W}/2$?
  - just target a smaller set of queries $\mathbb{Q}$ s.t. $\#\mathbb{Q} = \#\mathbb{W}-2$
- Hierarchical injection attack
  - more sophisticated attack recovers sets larger than $\#\mathbb{W}/2$…
    - …even when client uses threshold

Attacks on Encrypted Range Search

- [Kellaris-Kollios-Nissim-O'Neill16]
  - recovers values by exploiting response id + volume
  - requires $O(N^4 \cdot \log N)$ queries
  - assumes uniform queries
- [Grubbs-Lacharite-Minaud-Paterson19]
  - recovers εN-approximation by exploiting response identity
  - requires $O(\varepsilon^{-2} \log \varepsilon^{-1})$ queries
- [Grubbs-Lacharite-Minaud-Paterson19]
  - recovers εN-approximate order by exploiting response identity
  - requires $O(\varepsilon^{-1} \log \varepsilon^{-1})$ queries