

GRECS: GRaph Encryption for Approx. Shortest Distance Queries

Xianrui Meng (*Boston University*)

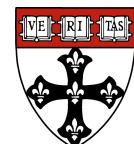
Seny Kamara (*Microsoft Research*)

Kobbi Nissim (*Ben-Gurion U. & CRCS Harvard U.*)

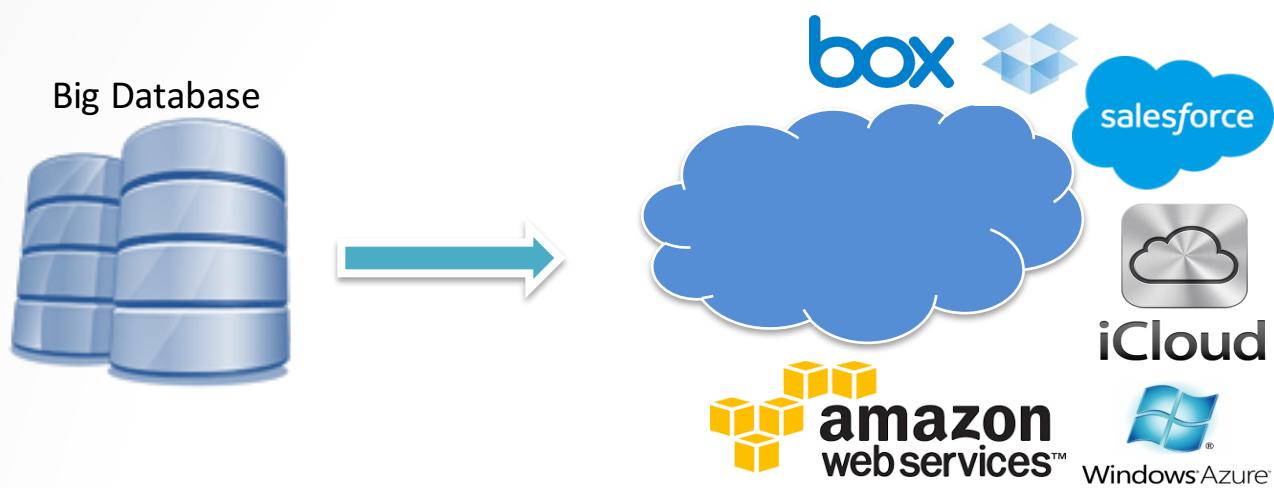
George Kollios (*Boston University*)



Microsoft Research

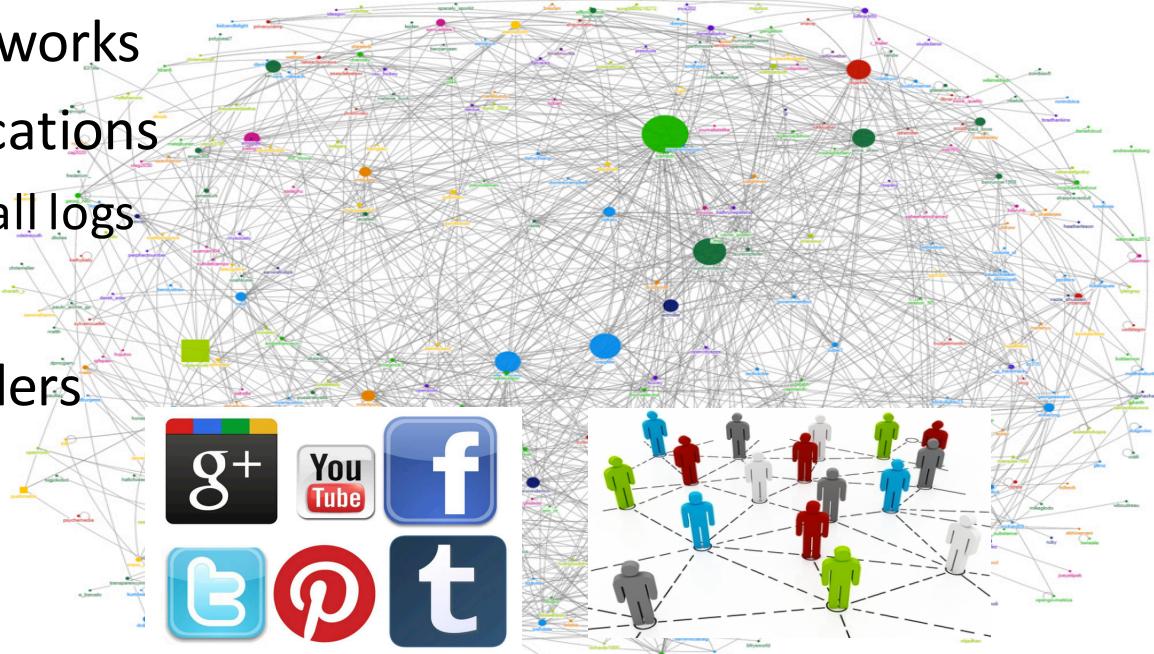


Cloud Storage

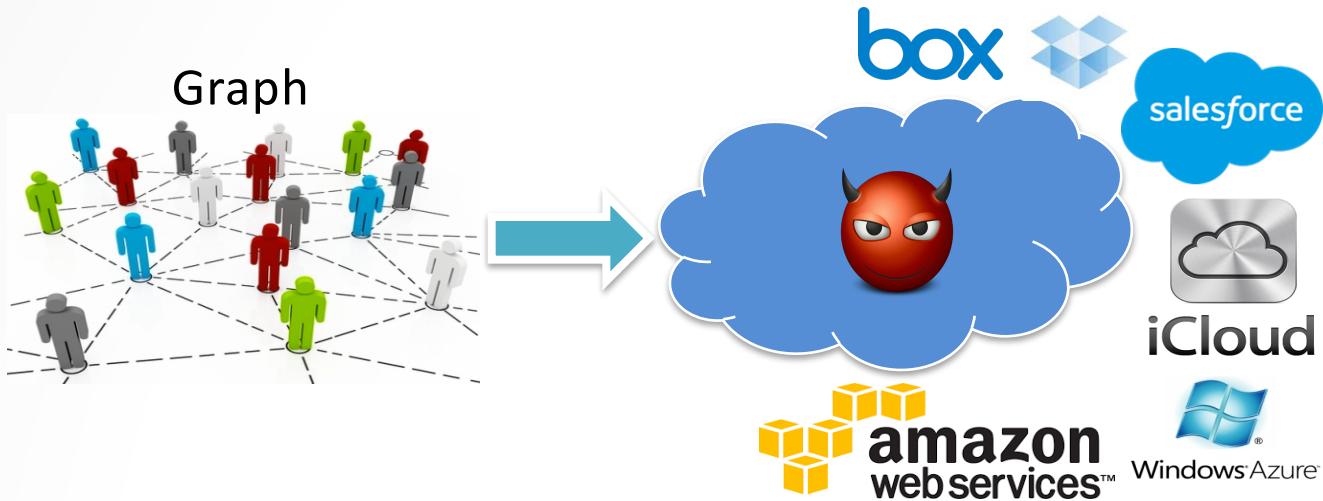


Graph Data

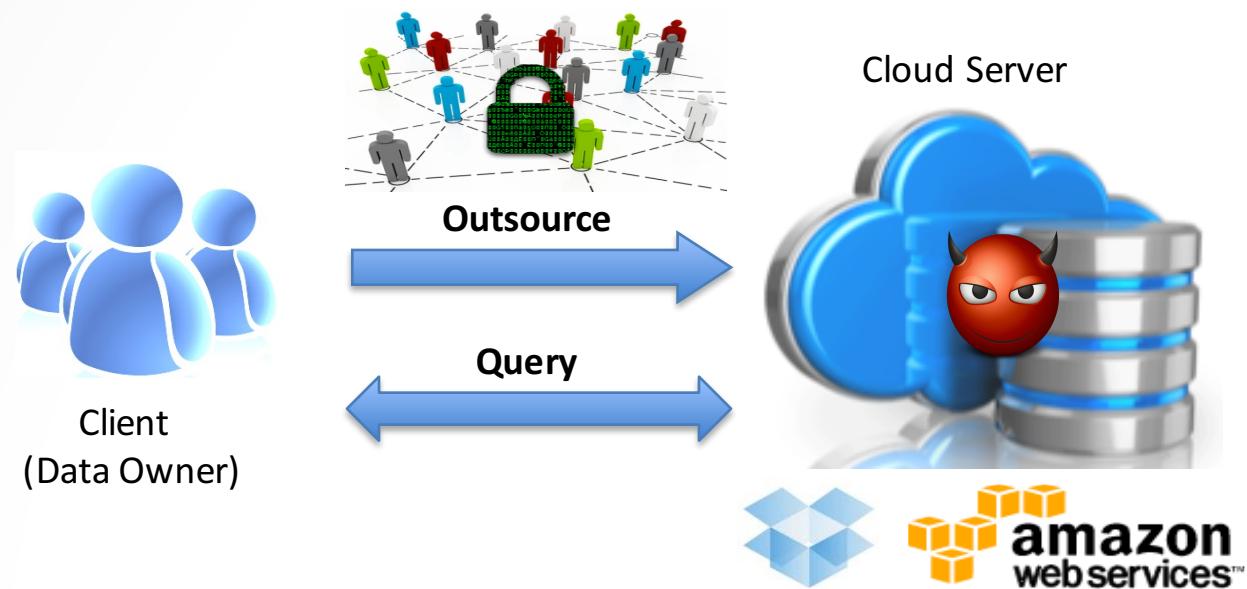
- Social Networks
- Communications
 - phone call logs
- Networks
- Web crawlers
-



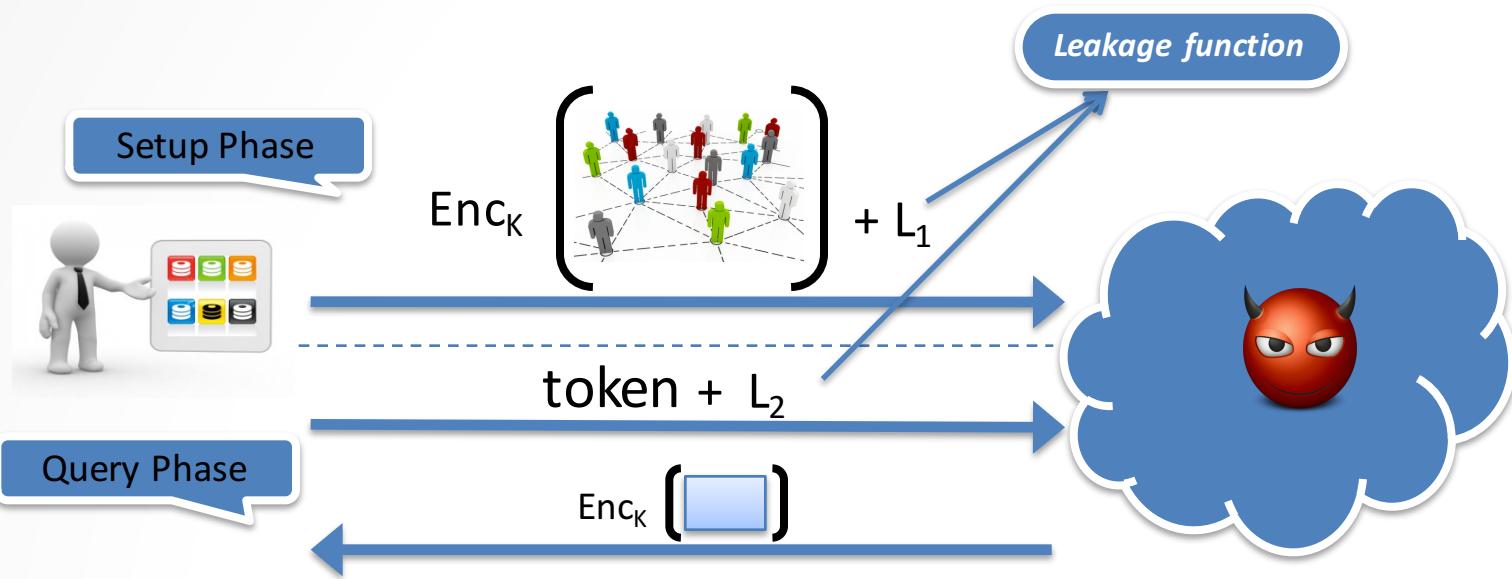
Outsource Graphs



Outsourced Graph Data



Graph Encryption



Security Definition

- Adaptive Chosen Query Attack (CQA-2)
 - Searchable Encryption
[Curtmola-Garay-Kamara-Ostrovsky06], [CK10], [CJJKRS13], etc ...
 - *Simulation-based security*

“No efficient adversary can learn any partial information about the data or the queries, beyond what is explicitly allowed by the leakage functions.”

“... even for queries that are adversarially-influenced and generated adaptively.”

Leakage

- Leakage function
 - Describe as *stateful* functions of the input data...
 - Size of the graph...
 - Query Pattern, i.e. whether the query has been repeated.
 - Access Pattern, i.e. pointer to the databases.
 - etc ...

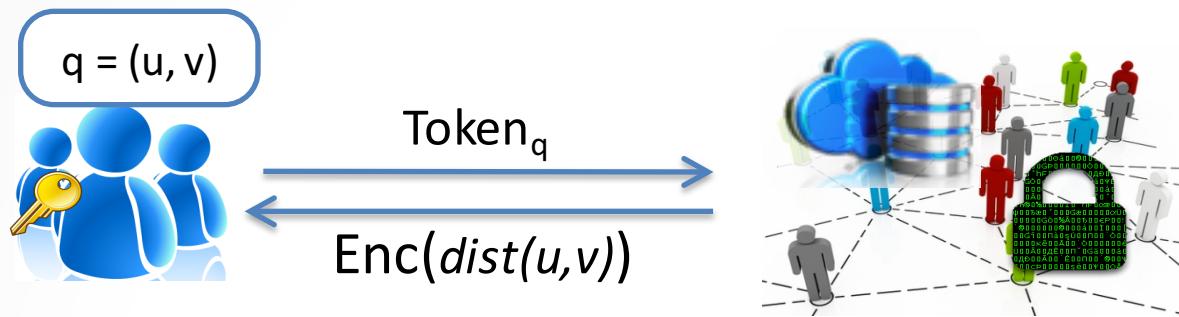
State of the Art

- Searchable Encryption (SE)
 - Keyword Search [SWP01, CM05 ,CGKO06],
 - Boolean queries [CJJKRS13]
 - Range queries [SBCSP 07, LLWB 14]
 - Dynamic SE [KPR12, KP13, SPS 14, NPG 14]
 - Structured data [CK10]
- Oblivious RAM
 - More secure: *does NOT leak access pattern* [GO92, SDSCFRYD 13, DSS 14, WNLCSSH 14, LWNHS 15, etc...]
- Fully Homomorphic Encryption
 - [Gentry09]

GRECS: GRaph EnCryption for approx. Shortest distance queries



Querying the Encrypted Graph



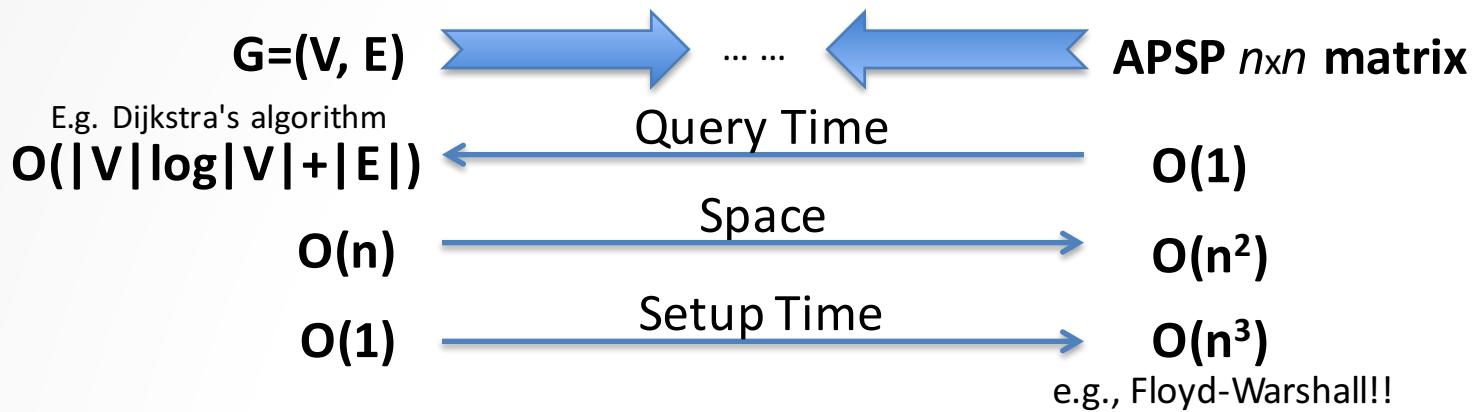
Want $\text{dist}(u, v) \approx d(u, v)^*$

* $d(u, v)$ is the *real* shortest distance between u and v

Design a Practical Scheme

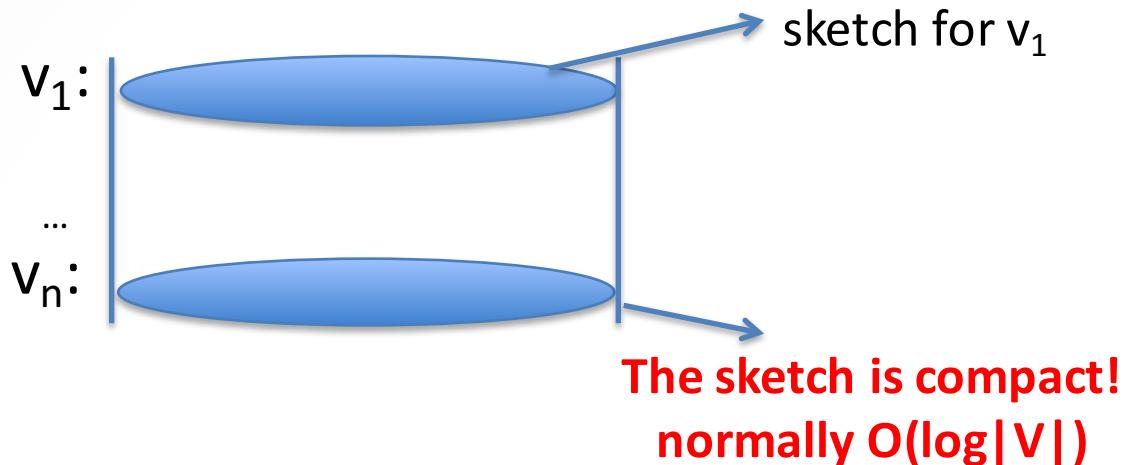
- Low Communication Complexity
- Reasonable Space Overhead
- Query Processing Overhead
 - Server: *small* computation
 - Client : *very small* computation

Shortest Distance for Graphs



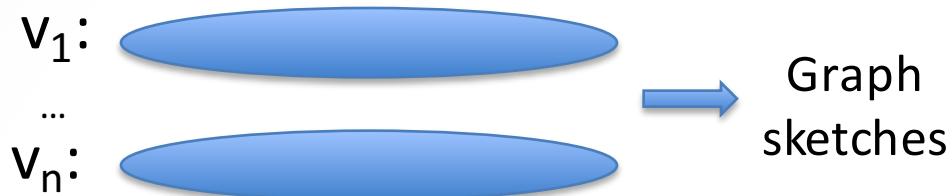
Want some efficient and compact Data Structure
for Fast Shortest Distance Queries.

Practical Distance Oracle



Data Structure that is produced by
some *Randomized* Algorithm...

Practical Distance Oracle



DO returns d s.t. $dist \leq d \leq \alpha \cdot dist$
($dist$: real shortest distance)

Most DOs have to compromise on accuracy: don't return the accurate distance but rather a constant-factor approximation of it.

Our basic idea: to encrypt the DO

$v_i:$ $(v_3, 2) (v_4, 3) (v_5, 2) (v_8, 1) (v_{11}, 3)$



$v_j:$ $(v_4, 2) (v_2, 2) (v_5, 2) (v_1, 2) (v_8, 2)$

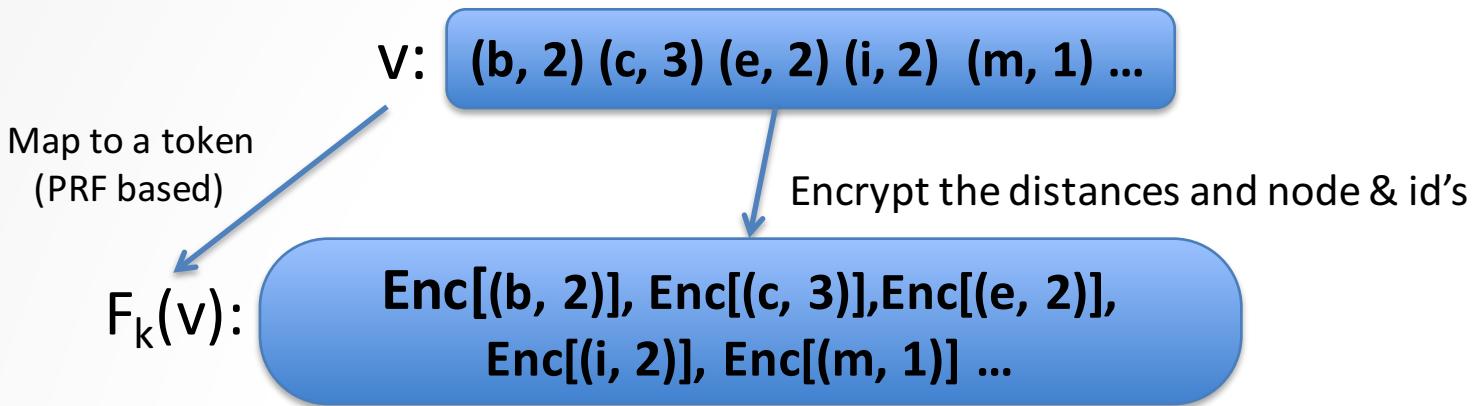
$dist(v_i, v_j) : \min\{ (v_5, v_8), (2+2), (2+1) \}$

GRECS

	GraphEnc₁	GraphEnc₂	GraphEnc₃
Space Complexity	$O(n \log n)$	$O(n \log^2 n / \epsilon)$	$O(n \log n)$
Communication	$O(\log n)$	$O(1)$	$O(1)$
Server's Query Complexity	$O(1)$	$O(\log^2 n / \epsilon)$	$O(\log n)$
Client's Query Complexity	$O(\log n)$	$O(\text{diameter})$	$O(\text{diameter})$

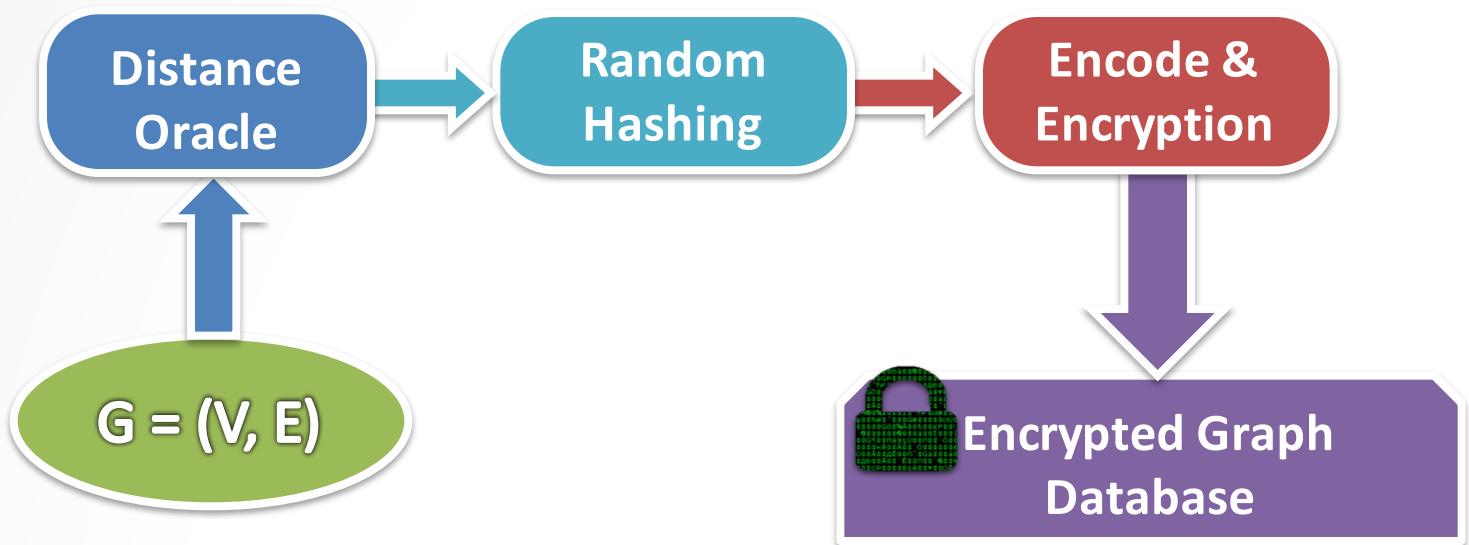
- $n = |V|$ for $G = (V, E)$
- Sketch size is $\sim O(\log n)$
- GraphEnc_3 leaks a bit of more ...

GraphEnc₁ : An Encrypted Storage Approach



**Problem: Query has high
Communication Complexity!!**

GraphEnc₂: Communication-Efficient O(1)



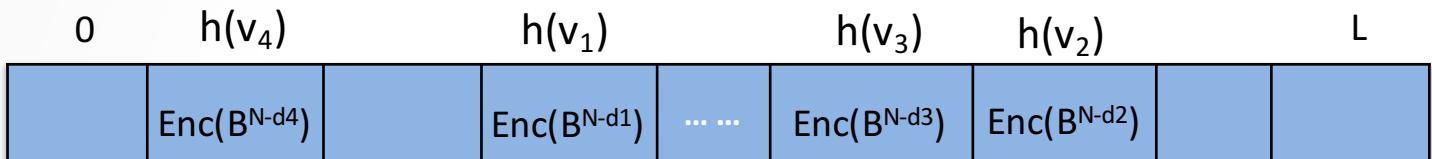
GraphEnc₂ : our basic idea

- Setup:

1. Map v to token: $F_k(v)$ $v: \boxed{(v_1, d_1) \ (v_2, d_2) \ (v_3, d_3) \ (v_4, d_4)}$

2. Random hashing :

$h(\text{node_id})$



3. Encode & Encrypt
using SWHE
(BGN encryption):

Enc($B^{N-\text{dist}}$)

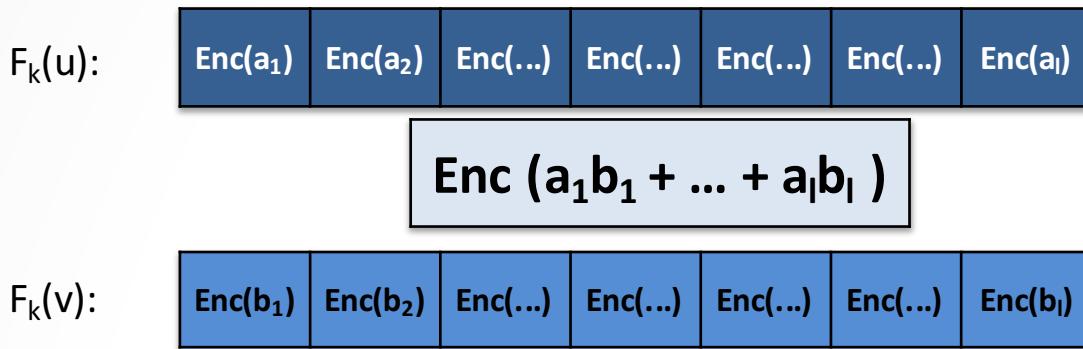
* N is max dist in DO

* B is some positive integer

4. Encrypt the rest Enc(0)

GraphEnc₂ : Query Overview

- Query: $\text{query} = (u, v) \rightarrow \text{Token: } F_k(u), F_k(v)$



- **homomorphic multiplication**: bilinear pairing on ciphertext
- **homomorphic addition**: multiplication on ciphertext

GRECS:GraphEnc₂

- Theorem:

with high probability,

$$d(u, v) - e \leq dist \leq \alpha \cdot d(u, v)$$

e : related # of common nodes in Sketch(u) and Sketch(v)

α : approximation ratio from Dist. Oracle

$$dist = \mathbf{Dec}\left(Enc(a_1b_1 + \dots + a_lb_l)\right)$$

Server only returns
only one Enc(.)!

GRECS:GraphEnc₂

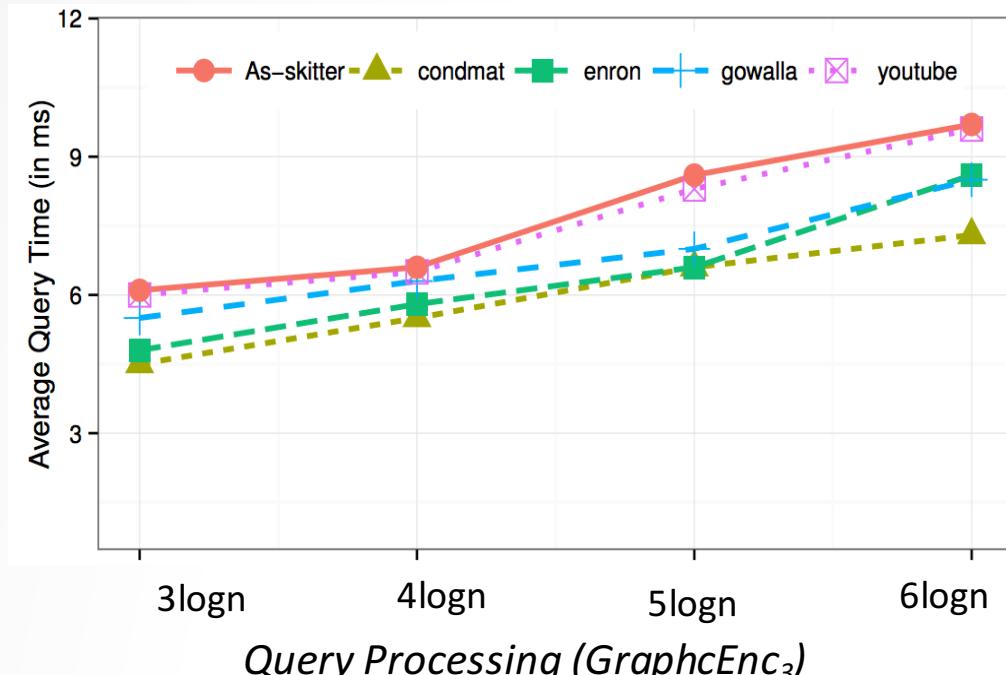
- Theorem:
with high probability,
$$d(u, v) - e \leq dist \leq \alpha \cdot d(u, v)$$

e : related # of common nodes in Sketch(u) and Sketch(v)
 α : approximation ratio from Dist. Oracle
- Security:
 - CQA2-secure against semi-honest adversarial server.
 - Leakage: query pattern, access pattern, $|V|$

GRECS: GraphEnc₃

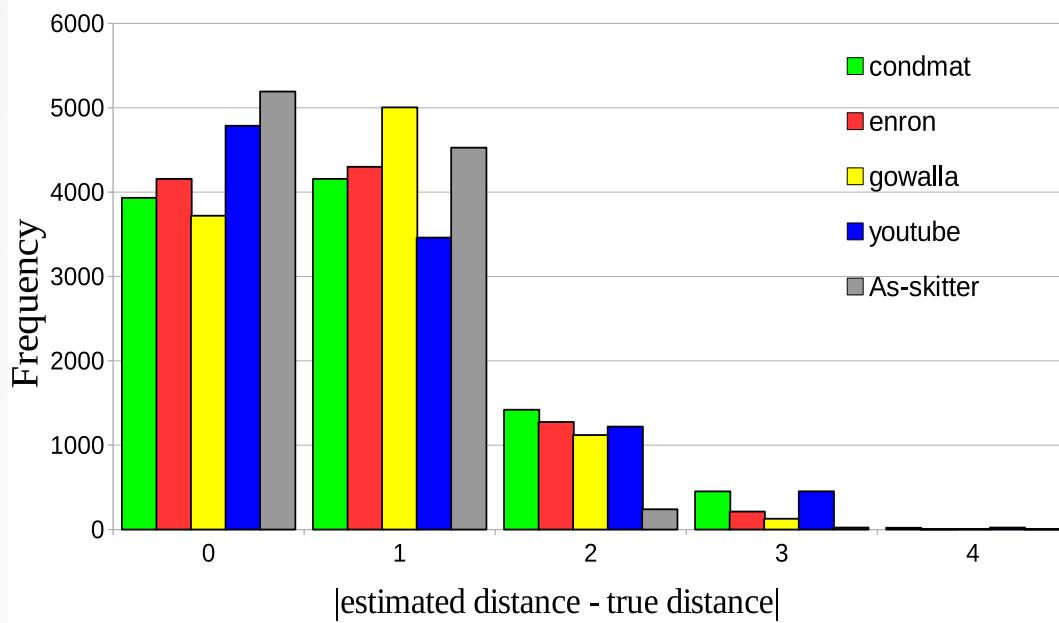
- GraphEnc₃
 - Constant Communication Complexity
 - Much Lower Space Overhead
 - Much Faster Query Time
 - Leaks a bit more than GraphEnc₂
 - Standard leakage similar to SE

Query Performance



	V	E
as-skitter:	1.6M	11M
youtube:	1.1M	2.9M
gowalla:	0.20M	0.95M
enron:	36K	0.37M
condmat:	23K	0.19M

Distance Accuracy



$|V|$ $|E|$

as-skitter: 1.6M 11M
youtube: 1.1M 2.9M
gowalla: 0.20M 0.95M
enron: 36K 0.37M
condmat: 23K 0.19M

Subsequent/Ongoing work

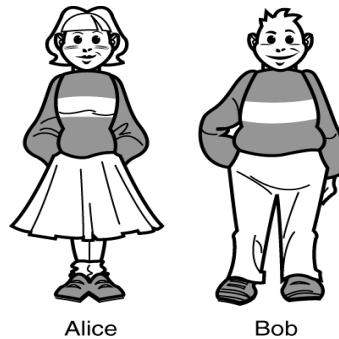
- To support more complex graph queries/graph mining tasks ...
- More efficient searchable encrypted graph database ...

Challenge

- Leakage: how to *minimize* and *control* the leakage
 - Trade-off: privacy/performance/space
- Design schemes that scale to *massive* data
- General Queries on Encrypted Graph Structure

More details see: <http://eprint.iacr.org/2015/266.pdf>

Thank you very much!



Questions?