Securely Computing Clustering Coefficient for Outsourced Dynamic Encrypted Graph Data

Laltu Sardar[1] Gaurav Bansal[2] Sushmita Ruj [1,3] Kouichi Sakurai[4]

[1]**Indian Statistical Institute, Kolkata, India**
[2]Indian Institute of Technology, Jammu, India
[3]CSIRO Data61, Australia
[4]Kyushu University, Japan

ComsNets 2021, Bengaluru, India

Importance of Graph

 # Social Medias

Increasing ↑

 # Users

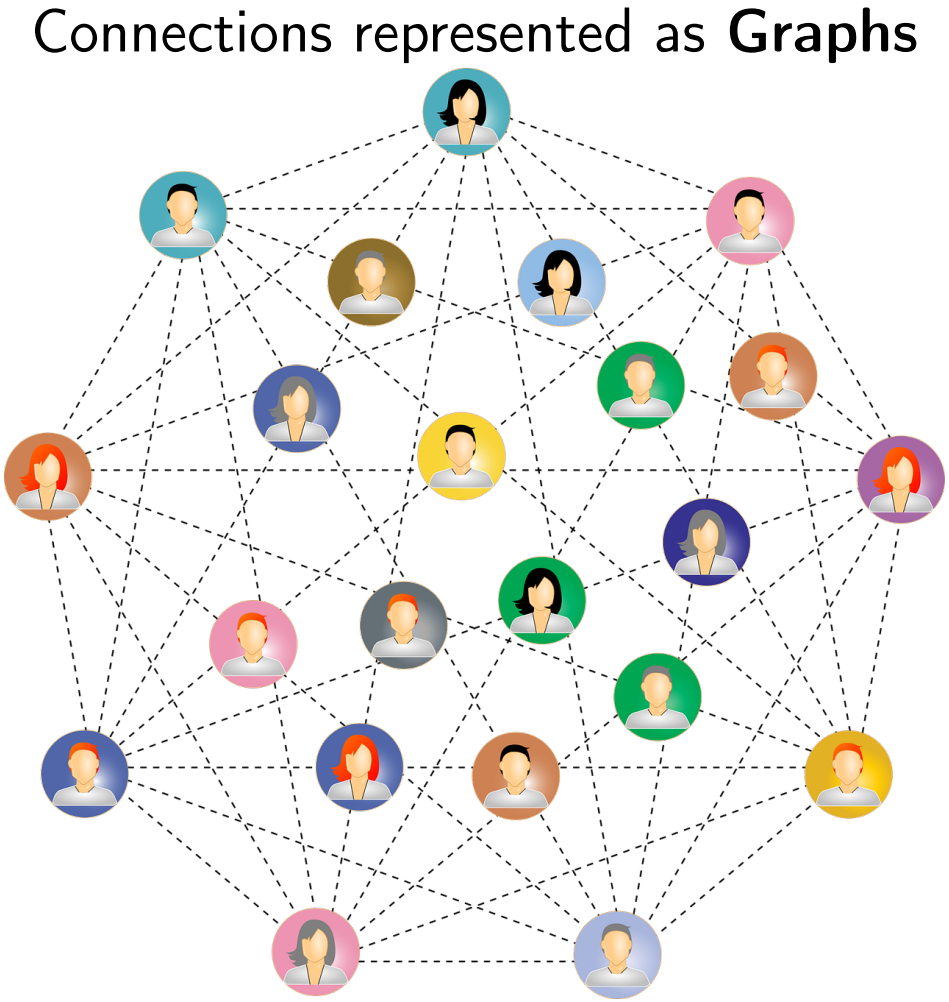
Increasing ↑

Importance of Graph

We Spending
More & More
Time

We Are getting
More & More
Connected

Importance of Graph



Dependency on Cloud Computing and Storage Services

Small and Medium Enterprises

Dependency on Cloud Computing and Storage Services

Small and Medium Enterprises

SME using Cloud Computing for *Cost & Security effectiveness*

- Easier to process requests and Faster to respond
- Data survivable probability Increases

Can we TRUST Cloud Service Providers?

TRUST?

Can we TRUST Cloud Service Providers?

TRUST?

Read Data

Sell Data

Hacked

Manipulation

Can we TRUST Cloud Service Providers?

TRUST?

Read Data

Sell Data

Hacked

Manipulation

How to be SAFE?

How to be SAFE?

Data Anonymity

Ji et al. [?] showed that all the state-of-the-art anonymization schemes are vulnerable
to several or all of the modern structure-based de-anonymization attacks.

Fails

How to be SAFE?

How to be SAFE?

Encrypt Graph

Send Encrypted Graph

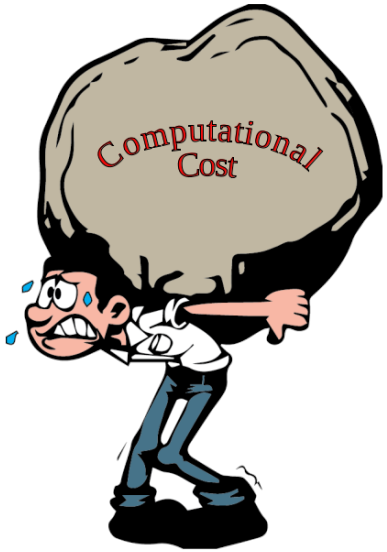
Store in cloud

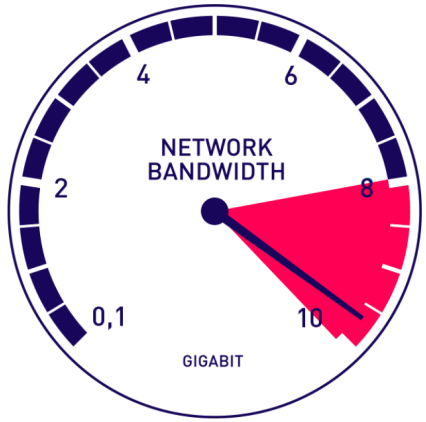
How to be SAFE?

Encrypt Graph

Send Encrypted Graph

Store in cloud





Cloud can't Search!

What we Need $\rightarrow$ Ability to Query

What we Need → Ability to Query

Encrypt Graph

Send Encrypted Graph

Store in cloud

What we Need → Ability to Query

Encrypt Graph

Send Encrypted Graph

Store in cloud

1. User Sends

Query Token

2. Cloud Search

over Encrypted Graph

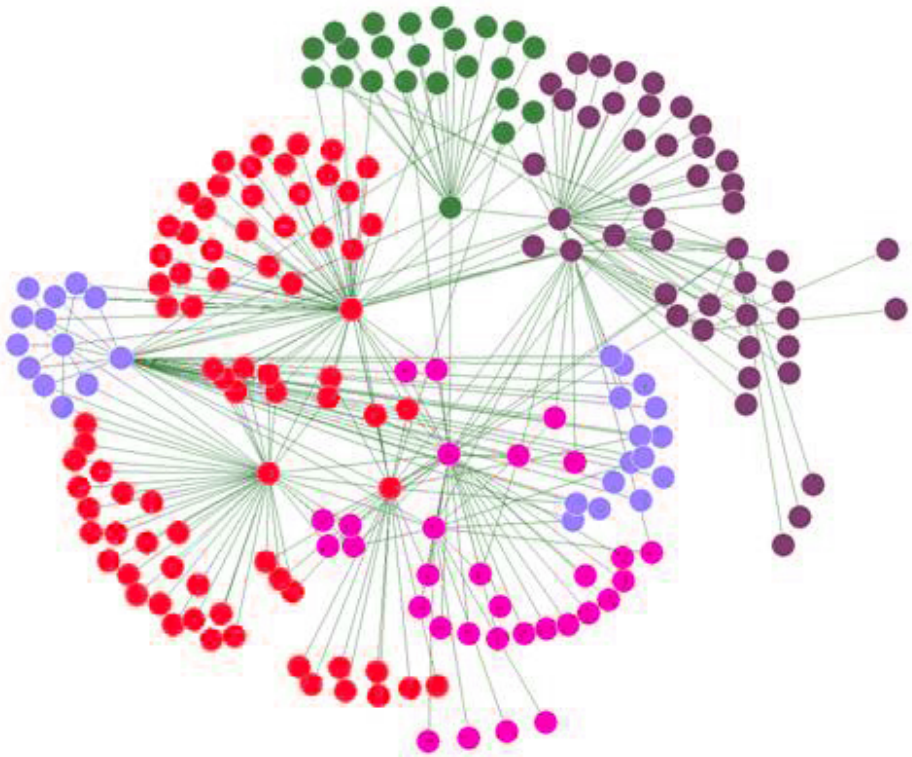
3.User received

Result

Type of Queries

- basic queries like vertex degree query, adjacency query [CK10].
- complex queries, the link prediction [SR19].
- The shortest distance query, that returns shortest distance between two given points [SMZ+18], [MKNK15], [WRD+17] etc.
- Xie and Xing [XX14] → clustering coefficient → used public key encryption scheme → makes the scheme inefficient for large datasets.

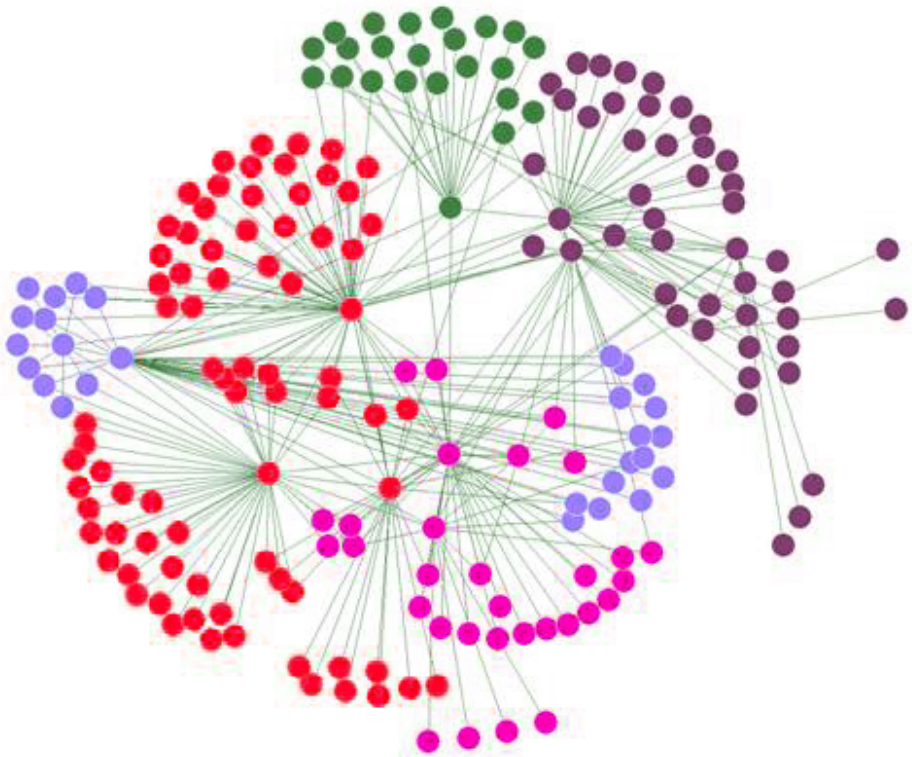
Clustering Coefficient: Concept



- In most real-world networks, nodes tend to create tightly knit groups characterized by a relatively high density of ties.
- The nodes $\in$ such a group have higher probability to be connected in the future.

Clustering Coefficient: Concept

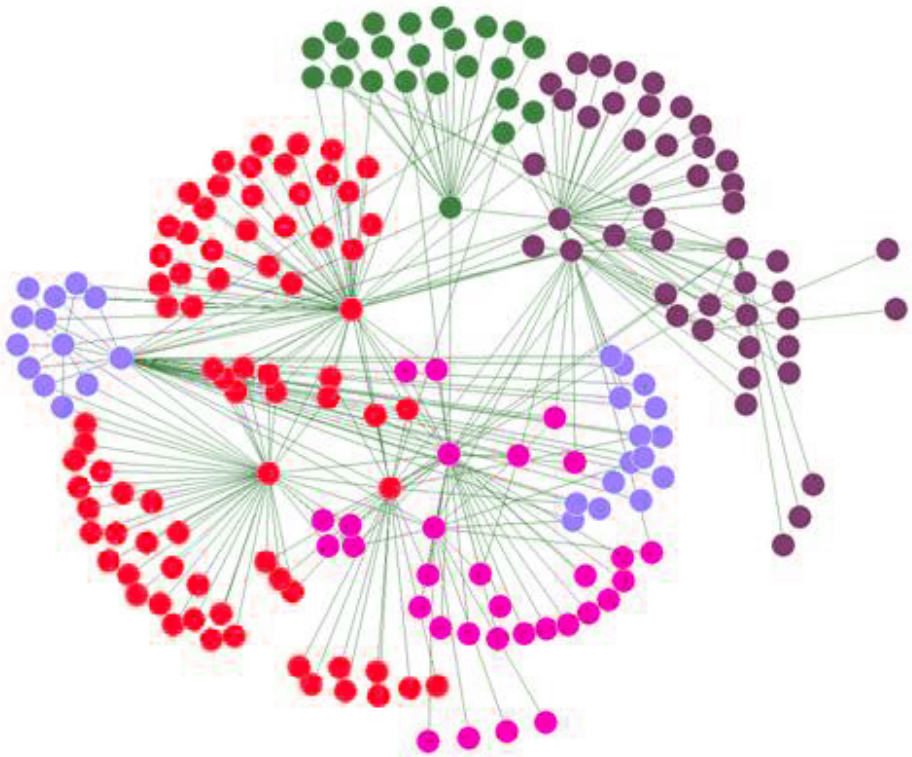
Measure of the degree to which nodes in a graph associate with one another



- In most real-world networks, nodes tend to create tightly knit groups characterized by a relatively high density of ties.
- The nodes $\in$ such a group have higher probability to be connected in the future.

Clustering Coefficient: Concept

Measure of the degree to which nodes in a graph associate with one another



- In most real-world networks, nodes tend to create tightly knit groups characterized by a relatively high density of ties.
- The nodes $\in$ such a group have higher probability to be connected in the future.
- Important measure metric to study determine the structural properties
- High clustering coefficient $\Longrightarrow$ more prone to targeted attacks

Clustering Coefficient: Definition

Definition (Clustering Coefficient in Undirected Graph)

If the unweighted graph $G = (V, E)$ is undirected, **local** *clustering coefficient* for a node $v_i \in V$ is denoted by CC_i and defined as

$$CC_i = \frac{|\{e_{jk} : (v_j, v_k \in N_{v_i}) \wedge (e_{jk} \in E)\}|}{|N_{v_i}|(|N_{v_i}| - 1)/2}$$

$$= \frac{\text{Exising \# Connections among neighbors}}{\text{All Possible \# Connections among neighbors}}$$

Clustering Coefficient: Definition

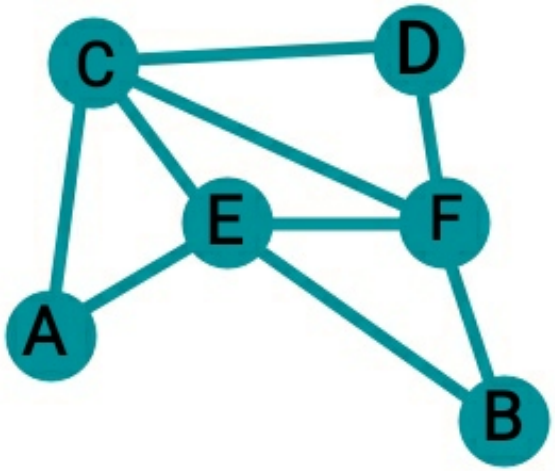
Definition (Clustering Coefficient in Undirected Graph)

If the unweighted graph $G = (V, E)$ is undirected, **local** *clustering coefficient* for a node $v_i \in V$ is denoted by CC_i and defined as

$$CC_i = \frac{|\{e_{jk} : (v_j, v_k \in N_{v_i}) \wedge (e_{jk} \in E)\}|}{|N_{v_i}|(|N_{v_i}| - 1)/2}$$

$$= \frac{\text{Exising \# Connections among neighbors}}{\text{All Possible \# Connections among neighbors}}$$

Example:



$$CC_E = \frac{3}{{}^4C_2} = \frac{3}{6} = 0.5$$

What We Contribute

- Design a novel graph encryption scheme *DCCE* that
  - performs clustering coefficient query.
  - The design is based on symmetric key encryption only.
  - Neighbor queries as well as edge queries on the same encrypted graph
  - Allows a new edge or vertex to be appended, making the scheme suitable for dynamic data.

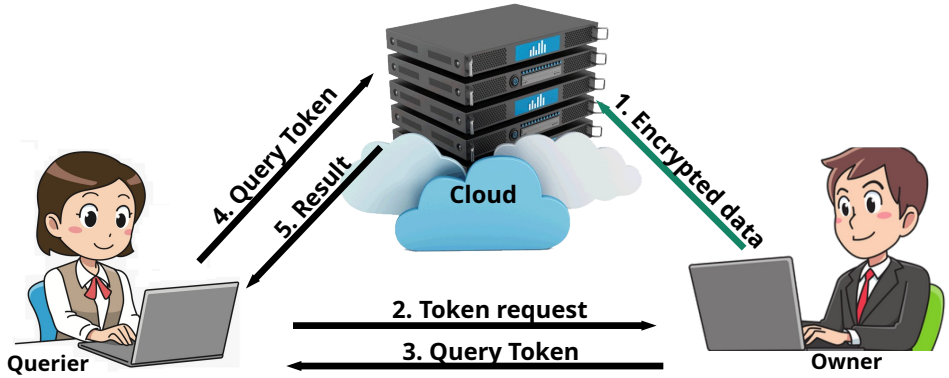
What We Contribute

- Design a novel graph encryption scheme *DCCE* that
  - performs clustering coefficient query.
  - The design is based on symmetric key encryption only.
  - Neighbor queries as well as edge queries on the same encrypted graph
  - Allows a new edge or vertex to be appended, making the scheme suitable for dynamic data.
- Define the security in the random oracle model.Show that it is provably secure under the chosen-query attack.

What We Contribute

- Design a novel graph encryption scheme *DCCE* that
  - performs clustering coefficient query.
  - The design is based on symmetric key encryption only.
  - Neighbor queries as well as edge queries on the same encrypted graph
  - Allows a new edge or vertex to be appended, making the scheme suitable for dynamic data.
- Define the security in the random oracle model.Show that it is provably secure under the chosen-query attack.
- We implement a prototype,
  - Tested with multiple real-life *SNAP* [LK14] datasets.
  - results show that the scheme is practical even for a very large database.

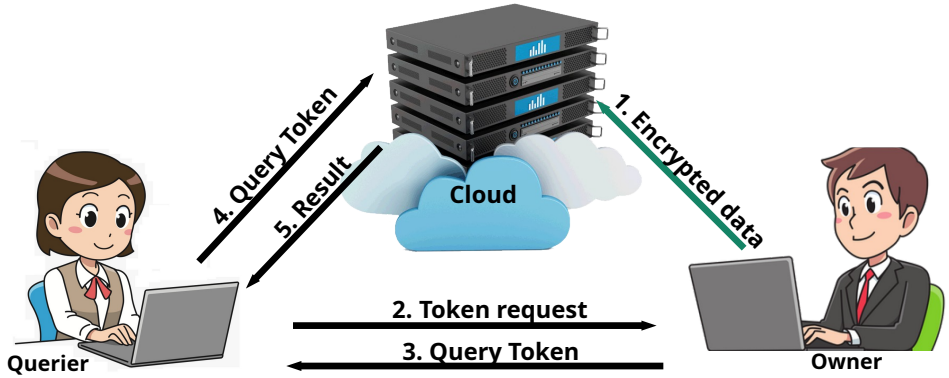
System Model: Three Entities– Owner, Cloud and User.



System Model: Three Entities– Owner, Cloud and User.

Owner

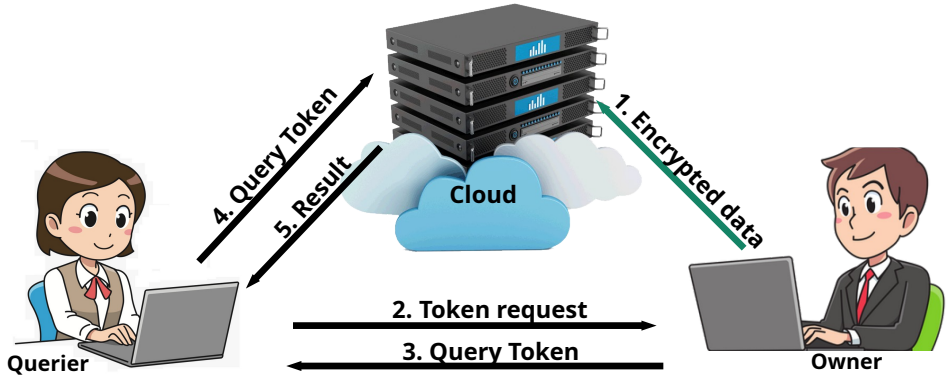
- Owns the database and Trusted
- Generates the required keys
- Encrypts graph and uploads.



System Model: Three Entities– Owner, Cloud and User.

Owner

- Owns the database and Trusted
- Generates the required keys
- Encrypts graph and uploads.



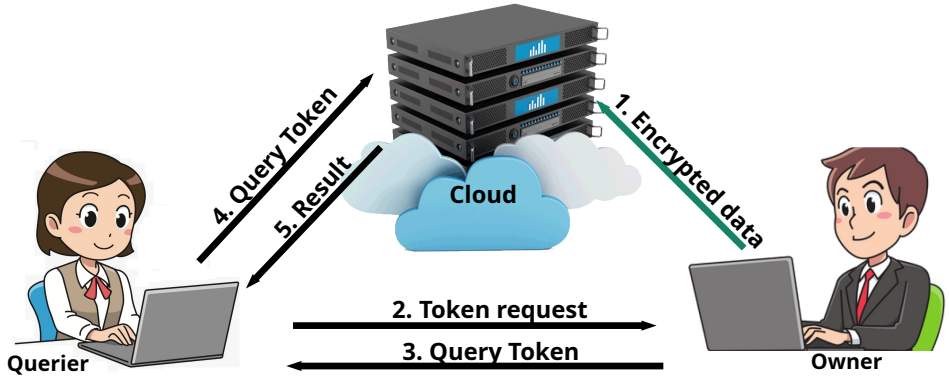
cloud

- *Honest-but-curious* cloud storage and computation service provider
- Performs a search over it on request from the user

System Model: Three Entities– Owner, Cloud and User.

Owner

- Owns the database and Trusted
- Generates the required keys
- Encrypts graph and uploads.



cloud

- *Honest-but-curious* cloud storage and computation service provider
- Performs a search over it on request from the user

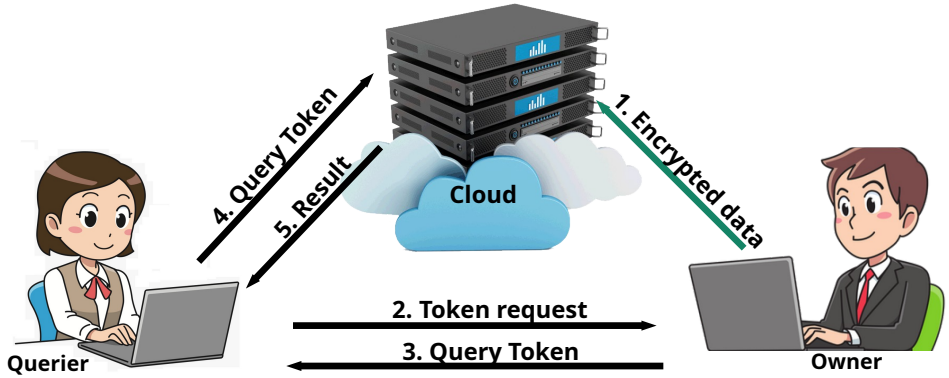
User

- Takes a token from the owner
- Requests a query to the cloud

System Model: Three Entities– Owner, Cloud and User.

Owner

- Owns the database and Trusted
- Generates the required keys
- Encrypts graph and uploads.



cloud

- *Honest-but-curious* cloud storage and computation service provider
- Performs a search over it on request from the user

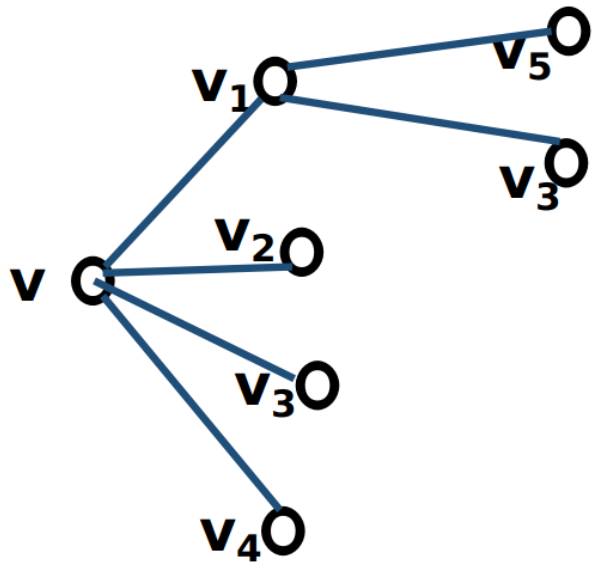
User

- Takes a token from the owner
- Requests a query to the cloud

Assumption: Communication between entities $\rightarrow$ via secure channels

Our Approach to Calculate Clustering Coefficient

- Calculating triangles
- First find neighbors,
- Then find neighbor of neighbors
- Count the matches with first set of neighbors
- Final answer = $\frac{\text{total number of } \textit{existing} \text{ matches}}{\text{total number of } \textit{possible} \text{ matches}}$



Example

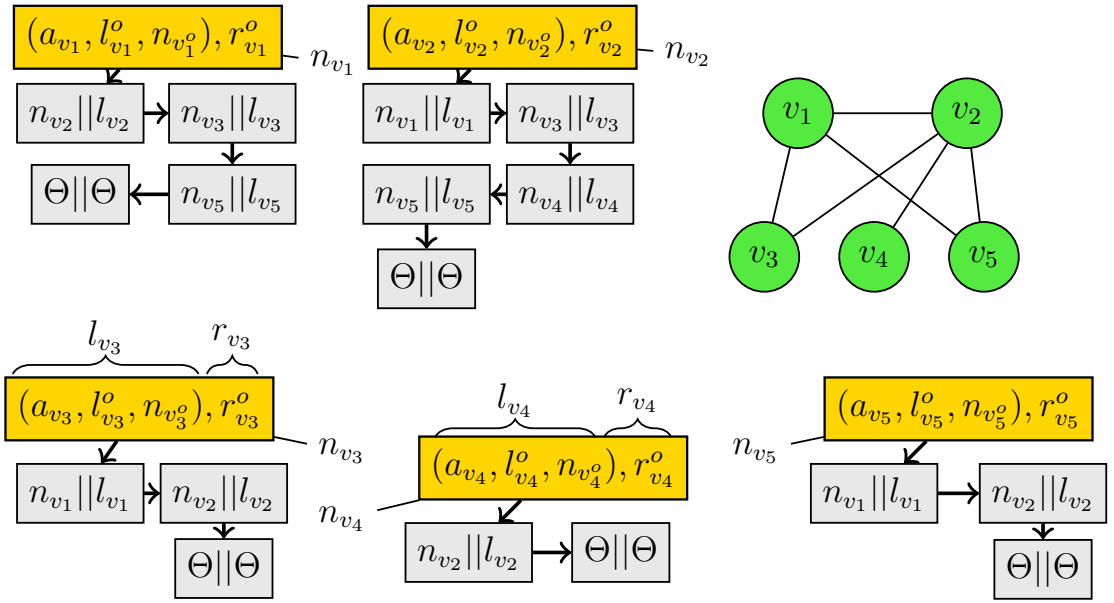
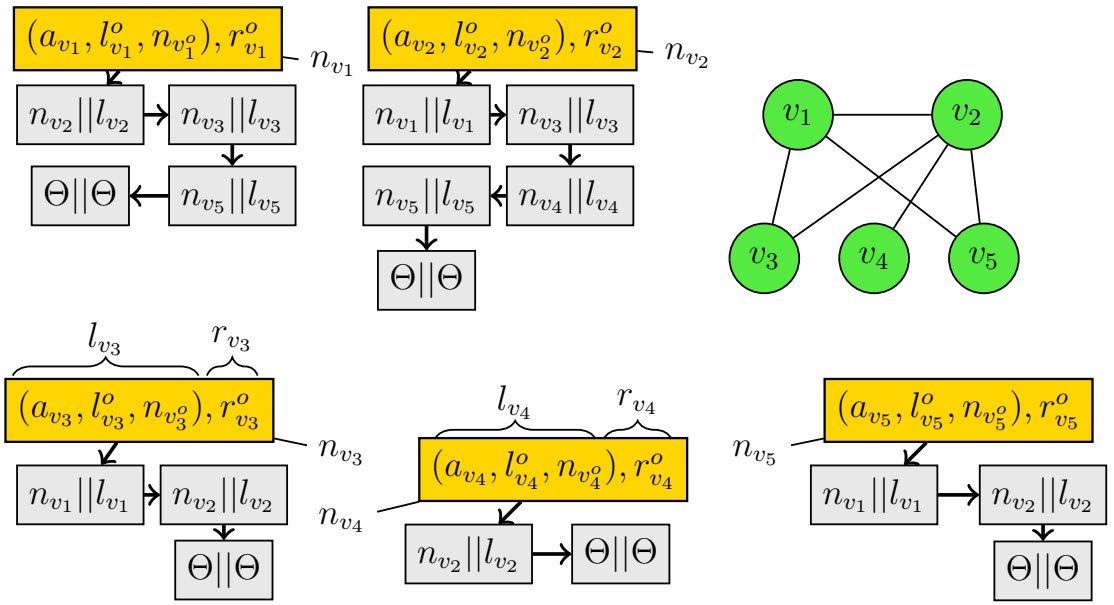


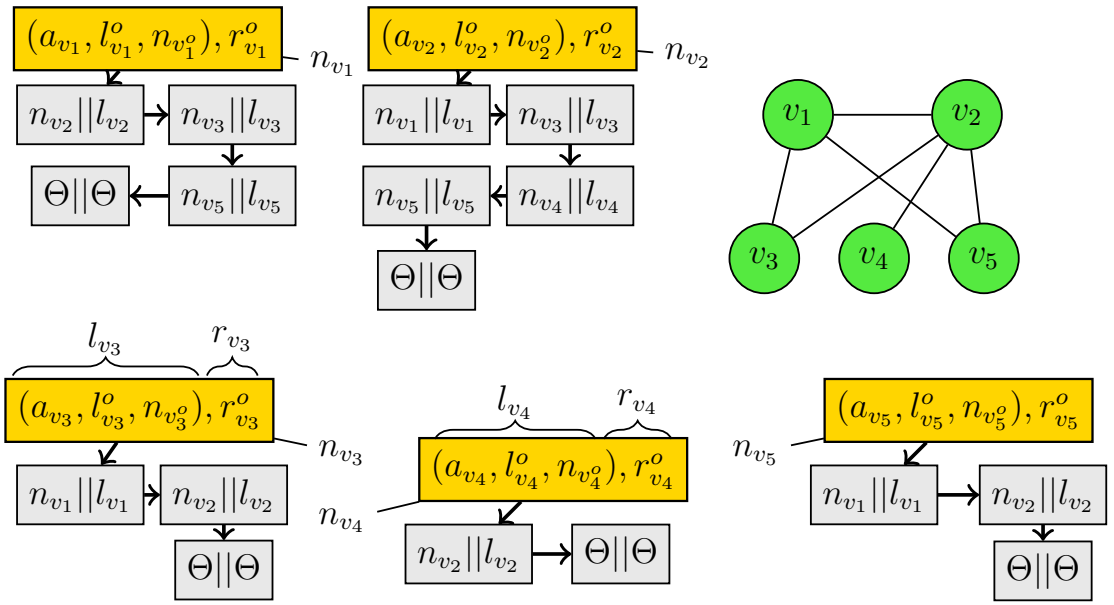
Figure: An example of what nodes store (before encryption)

- $n_v \leftarrow \mathtt{SE.Enc}(k_n, v)$;
- $T_E, T_V \leftarrow$ hash-tables corr. to edges & vertices
- $l_v \leftarrow$ to encrypt v in T_V
- $r_v \leftarrow$ to encrypt neighbors of v in t_E
- $n_v^o \leftarrow$ helps to traverse list of neighbors of v
- $\Theta \leftarrow$ null, termination symbol

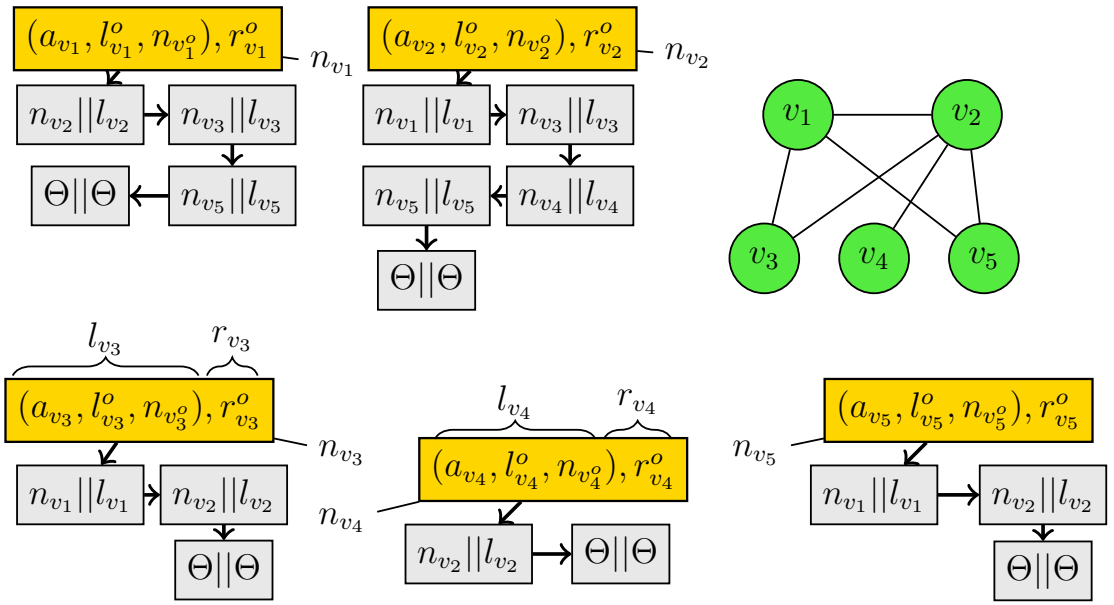
Example: Encryption



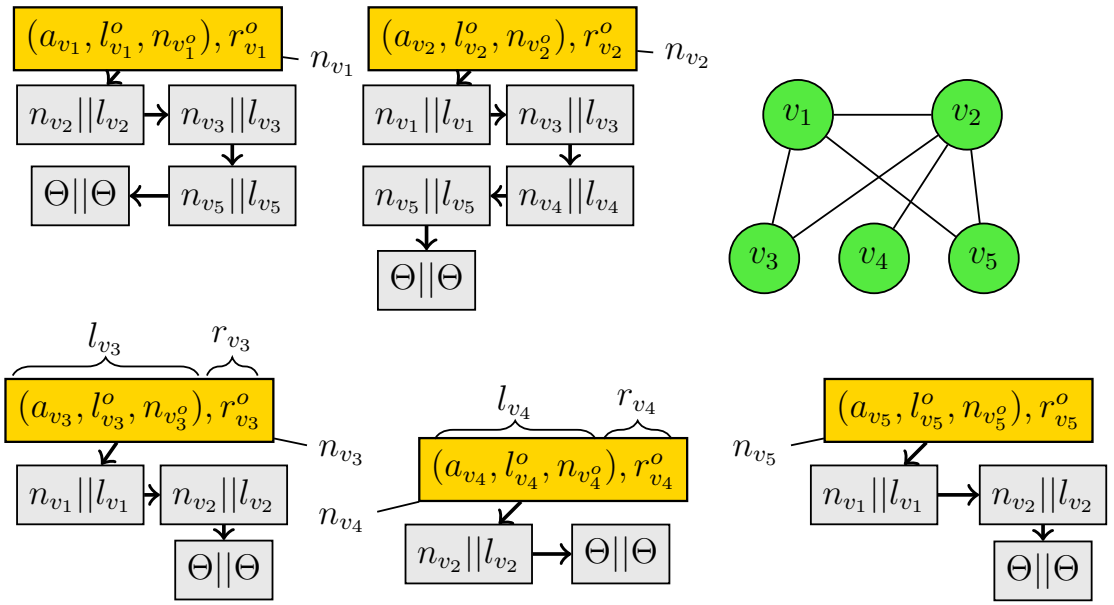
Example: Neighbor Query



Example: Clustering Coefficient Query



Example: Update



Security: Overview

Leakage Function $\mathcal{L} = \{\mathcal{L}_e, \mathcal{L}_n, \mathcal{L}_c, \mathcal{L}_{ae}, \mathcal{L}_{av}\}$.

- Initial dictionary construction:
  $\mathcal{L}_e(G) = \{|T_E|, \{n_v : v \in V\}, \{s_v : v \in V\}\}$,
- Neighbor query: $\mathcal{L}_n(v) = \{n_v, d_v, \{n_{v_i} : i = 0, \ldots, d_v\}\}$.
- Clustering coefficient query:

$$\mathcal{L}_c(v) = \{n_v, d_v, \{(n_{v_i}, d_{v_i}, \mathcal{L}_n(v_i)) : v_i \in N_v\}\}$$

- Adding a vertex: $\mathcal{L}_{av}(v) = \{key, key^o, g_v\}$
- Adding an Edge:
  $\mathcal{L}_{ae}(u, v) = \{key_u, key'_u, g_u, key_v, key'_v, g_v, \mathcal{L}_n(u), \mathcal{L}_n(v)\}$

Theorem

If F, P, G, H and $\mathtt{SE}$ is a PRG, a PRP, a PRP, a hash functions and a CPA-secure symmetric key encryption respectively, then $\mathtt{DCCE}$ is $\mathcal{L}$-secure against adaptive dynamic chosen-query attacks, in random oracle model.

Implementation Environment

System Configaration

- Language Used: C++
- Processor: Intel Core i7-4770 CPU, 8-core, 3.40GHz.
- Operating System: Ubuntu 16.04 LTS 64-bit, with 8GB RAM.

Implementation Environment

System Configaration

- Language Used: C++
- Processor: Intel Core i7-4770 CPU, 8-core, 3.40GHz.
- Operating System: Ubuntu 16.04 LTS 64-bit, with 8GB RAM.

Crypto functions

- *cryptopp* library
- $P \leftarrow$ *HMAC*; $F \leftarrow$ *Salsa20*; $H \leftarrow$*SHA-256*; $SE \leftarrow$*AES*

Implementation Environment

System Configaration

- Language Used: C++
- Processor: Intel Core i7-4770 CPU, 8-core, 3.40GHz.
- Operating System: Ubuntu 16.04 LTS 64-bit, with 8GB RAM.

Crypto functions

- *cryptopp* library
- $P \leftarrow$ *HMAC*; $F \leftarrow$ *Salsa20*; $H \leftarrow$*SHA-256*; *SE* $\leftarrow$*AES*

Dataset

- Real-world *SNAP* [LK14] datasets $\rightarrow$ a collection of large networks
- 'ca-HepPh', 'email-Enron', 'loc-Gowalla', and 'roadNet-CA'
- # nodes: 12,008, 36,692, 196,591, 1,965,206
- # edges: 118,521, 183,831, 950,327, 2,766,607

Encryption Time

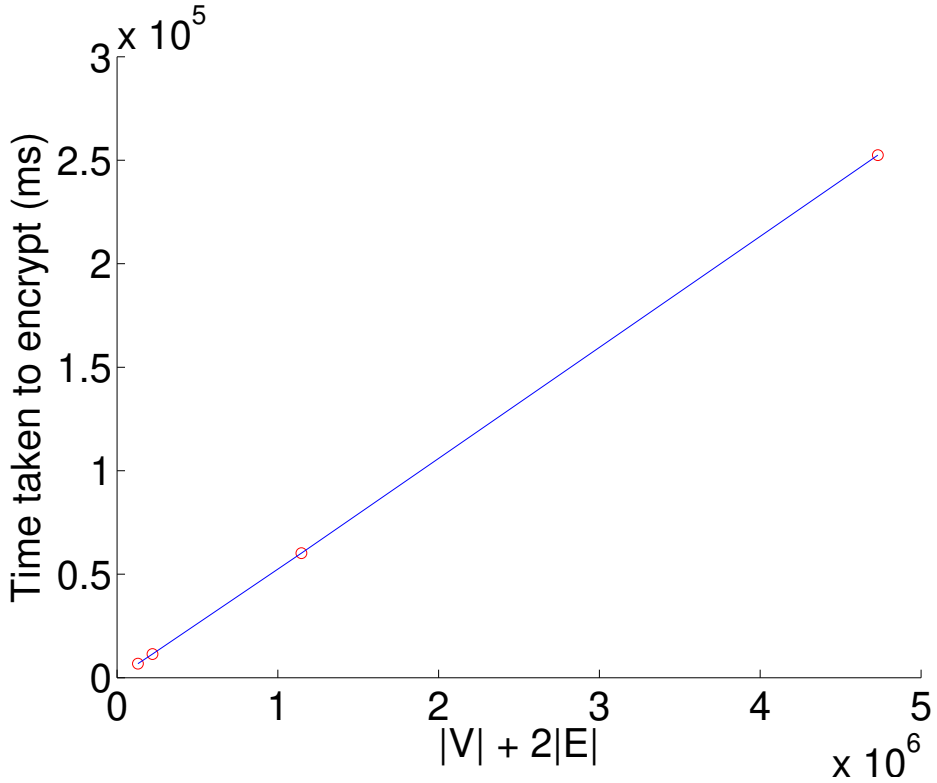
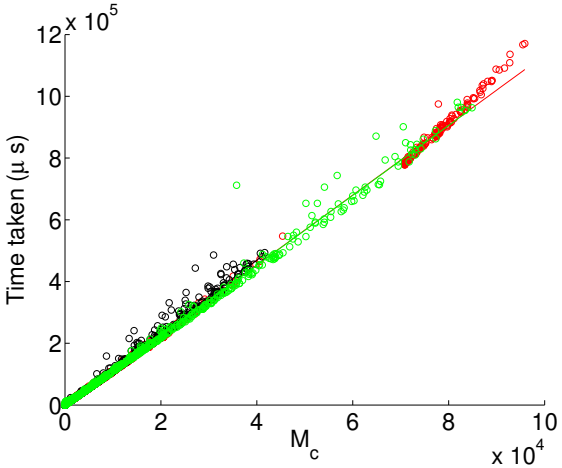


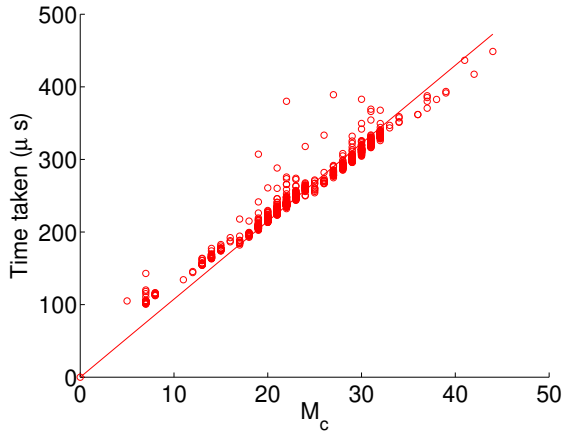
Figure: Encryption Time on different datasets

- Encrypt time $\propto M_e = (|V| + 2|E|)$.
- $M_e = 53.3\mu s$, in our case

Clustering coefficient query Time



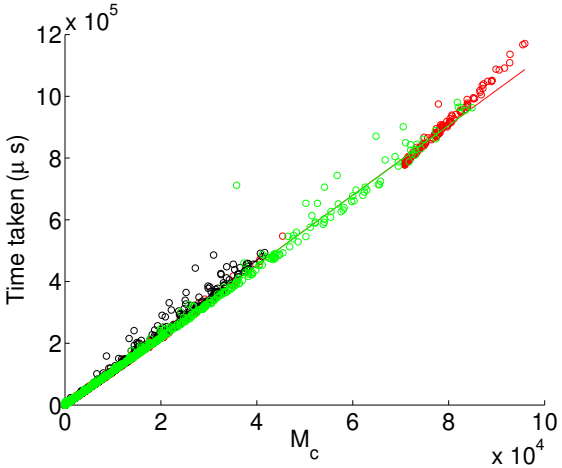
(a) HepPh, Enron, Gowalla



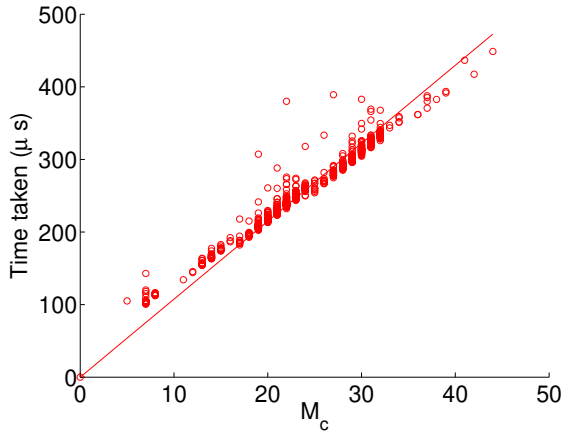
(b) roadNet-CA

Figure: CC Query time on different datasets

Clustering coefficient query Time



(a) HepPh, Enron, Gowalla

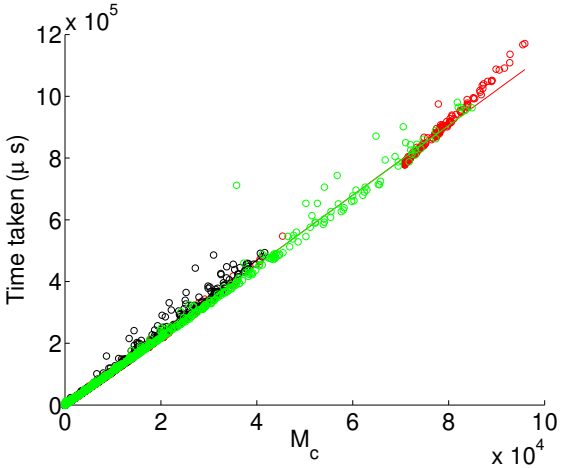


(b) roadNet-CA

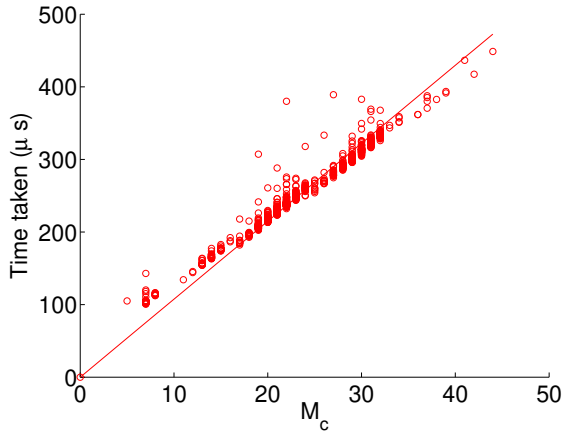
Figure: CC Query time on different datasets

- query time for $v \propto M_c = (4|N_v| + \sum_{u \in N_v} |N_u|)$
- average values of M_cs are $11.33\mu s$, $11.72\mu s$, $11.32\mu s$, and $10.76\mu s$

Clustering coefficient query Time



(a) HepPh, Enron, Gowalla



(b) roadNet-CA

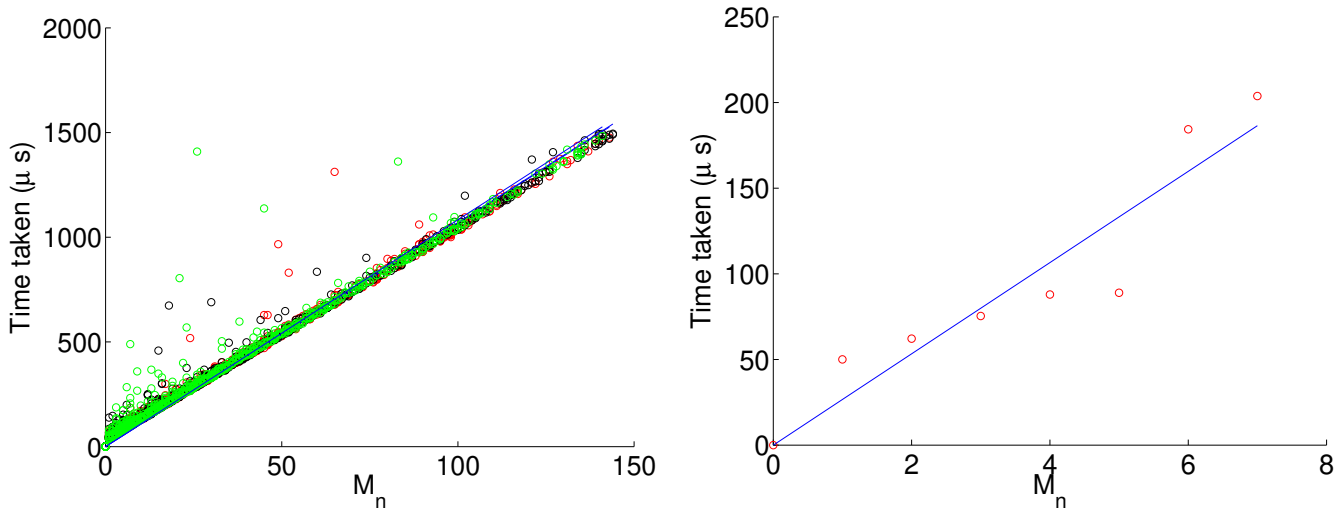
Figure: CC Query time on different datasets

- query time for $v \propto M_c = (4|N_v| + \sum_{u \in N_v} |N_u|)$
- average values of M_cs are $11.33\mu s$, $11.72\mu s$, $11.32\mu s$, and $10.76\mu s$

Improvement over PKE based Scheme

- **Our scheme**: 196,591 vertices and 950,327 edges, $\rightarrow$ $1.2s$
- **[XX14]**: 62 vertices and 159 edges, $18.77s$.
- Huge improvement $\rightarrow$ use of symmetric key encryption and efficient data structure.

Neighbor Query time



(a) HepPh, Enron, Gowalla (b) roadNet-CA

Figure: Neighbor Query time on different datasets

- Time taken by the cloud $\propto M_n = |N_v|$ for v.
- M_n is 10.40μs, 10.46μs, 10.54μs, and 23.68μs respectively.

Update time

- Addition of a vertex $\rightarrow$ constant for any vertex.
- The time taken to add a vertex = $69\mu s$.
- Adding edge $\rightarrow$ constant time.
- The time taken to add an edge = $124\mu s$ for the client.

Raphael Bost, Pierre-Alain Fouque, and David Pointcheval.

Verifiable dynamic symmetric searchable encryption: Optimality and forward security.

IACR Cryptology ePrint Archive, 2016:62, 2016.

Qi Chai and Guang Gong.

Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers.

In *Proceedings of IEEE International Conference on Communications, ICC 2012, Ottawa, ON, Canada, June 10-15, 2012*, pages 917–922, 2012.

Melissa Chase and Seny Kamara.

Structured encryption and controlled disclosure.

In *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings*, pages 577–594, 2010.

Rong Cheng, Jingbo Yan, Chaowen Guan, Fangguo Zhang, and Kui Ren.

Verifiable searchable symmetric encryption from indistinguishability obfuscation.

In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '15, Singapore, April 14-17, 2015*, pages 621–626, 2015.

Shunrong Jiang, Xiaoyan Zhu, Linke Guo, and Jianqing Liu.

Publicly verifiable boolean query over outsourced encrypted data.

In *2015 IEEE Global Communications Conference, GLOBECOM 2015, San Diego, CA, USA, December 6-10, 2015*, pages 1–6, 2015.

Jure Leskovec and Andrej Krevl.

SNAP Datasets: Stanford large network dataset collection.

`http://snap.stanford.edu/data`, June 2014.

Zheli Liu, Tong Li, Ping Li, Chunfu Jia, and Jin Li.

Verifiable searchable encryption with aggregate keys for data sharing system.

Future Generation Comp. Syst., 78:778–788, 2018.

Yuxi Li, Fucai Zhou, Yuhai Qin, Muqing Lin, and Zifeng Xu.

Integrity-verifiable conjunctive keyword searchable encryption in cloud storage.

Int. J. Inf. Sec., 17(5):549–568, 2018.

Xianrui Meng, Seny Kamara, Kobbi Nissim, and George Kollios.

GRECS: graph encryption for approximate shortest distance queries.

In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, pages 504–517, 2015.

Meixia Miao, Jianfeng Wang, Sheng Wen, and Jianfeng Ma.

Publicly verifiable database scheme with efficient keyword search.

Inf. Sci., 475:18–28, 2019.

Wakaha Ogata and Kaoru Kurosawa.

Efficient no-dictionary verifiable searchable symmetric encryption.

In *Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers*, pages 498–516, 2017.

Azam Soleimanian and Shahram Khazaei.

Publicly verifiable searchable symmetric encryption based on efficient cryptographic components.

Des. Codes Cryptography, 87(1):123–147, 2019.

Meng Shen, Baoli Ma, Liehuang Zhu, Rashid Mijumbi, Xiaojiang Du, and Jiankun Hu.

Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection.

IEEE Trans. Information Forensics and Security, 13(4):940–953, 2018.

Laltu Sardar and Sushmita Ruj.

The secure link prediction problem.

Advances in Mathematics of Communications, 13(4):733–757, 2019.

Jianfeng Wang, Xiaofeng Chen, Shifeng Sun, Joseph K. Liu, Man Ho Au, and Zhi-Hui Zhan.

Towards efficient verifiable conjunctive keyword search for large encrypted database.

In *Computer Security - 23rd European Symposium on Research in Computer Security, ESORICS 2018, Barcelona, Spain, September 3-7, 2018, Proceedings, Part II*, pages 83–100, 2018.

Qian Wang, Kui Ren, Minxin Du, Qi Li, and Aziz Mohaisen.

Secgdb: Graph encryption for exact shortest distance queries with efficient updates.

In *Financial Cryptography and Data Security - FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers*, pages 79–97, 2017.

Pengtao Xie and Eric P. Xing.

Cryptgraph: Privacy preserving graph analytics on encrypted graph.

CoRR, abs/1409.5021, 2014.

Kazuki Yoneyama and Shogo Kimura.

Verifiable and forward secure dynamic searchable symmetric encryption with storage efficiency.

In *Information and Communications Security - 19th International Conference, ICICS 2017, Beijing, China, December 6-8, 2017, Proceedings*, pages 489–501, 2017.

Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou.

All your queries are belong to us: The power of file-injection attacks on searchable encryption.

In *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, pages 707–720, 2016.

Xiaoyu Zhu, Qin Liu, and Guojun Wang.

A novel verifiable and dynamic fuzzy keyword search scheme over encrypted data in cloud computing.

In *2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin, China, August 23-26, 2016*, pages 845–851, 2016.


Thank You!

Questions?