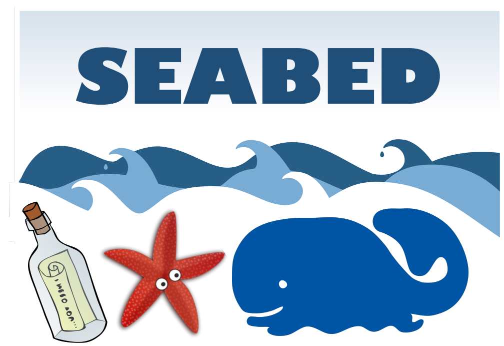
Big Data Analytics over Encrypted Datasets with Seabed

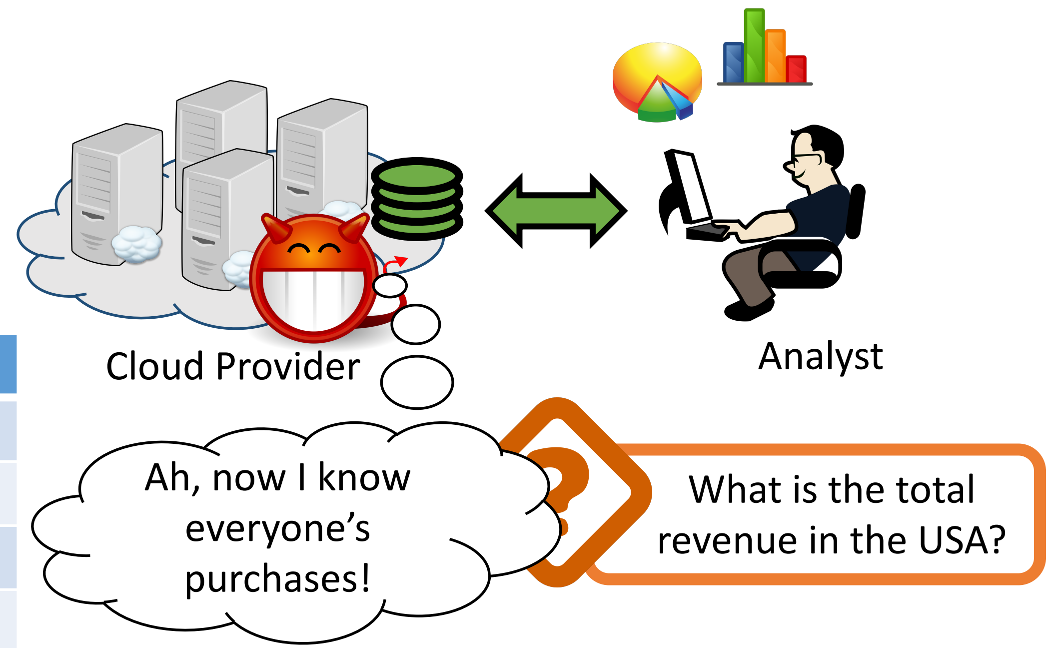


Antonis Papadimitriou✳, Ranjita Bhagwan☆, Nishanth Chandran ☆, Ramachandran Ramjee ☆, Andreas Haeberlen ✳, Harmeet Singh ☆, Abhishek Modi ☆, Saikrishna Badrinarayanan✯

✳University of Pennsylvania, ☆Microsoft Research India, ✯UCLA

Motivation: Big data analytics on sensitive data

customer	gender	country	payment
Alice	female	CAN	12
Bob	male	USA	4
Charlie	female	USA	1
Deborah	female	USA	15

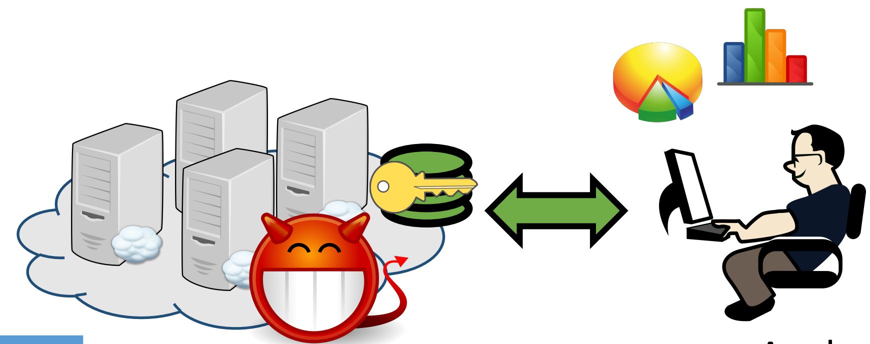


- Goal: Outsource big data analytics
  - Store database at a cloud provider
  - Perform analytical queries remotely
- Problem: Rogue cloud admins or hackers could have access to data
  - Sensitive information can be exposed

Prior work: Encrypted databases

Retail business

customer	gender	country	payment
%Th6j	h4$89	548yvg	439856
Fjg893	sfbg43	a3vbt9a	582650
%gTHR	h4$89	a3vbt9a	143759
34%^d	h4$89	a3vbt9a	874563



Cloud Provider

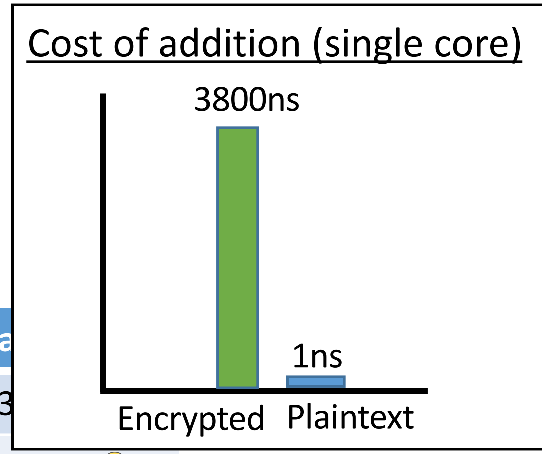
Analyst

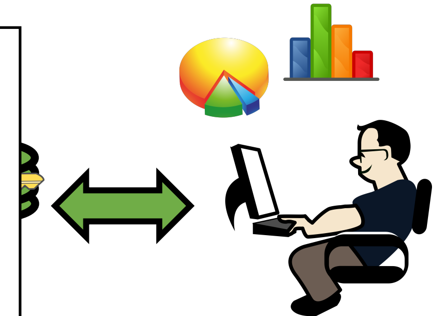
- What can we do?
  - Use encryption!
  - Examples: CryptDB/Monomi [SOSP11, VLDB13], MS SQL Server [SQL16]
  - These support SQL queries on encrypted data

Encrypted databases – Challenges

Retail business

customer	gender	country	pa
%Th6j	h4$89	548yvg	43
Fjg893	sfbg43	a3vbt9a	582650
%gTHR	h4$89	a3vbt9a	143759
34%^d	h4$89	a3vbt9a	874563





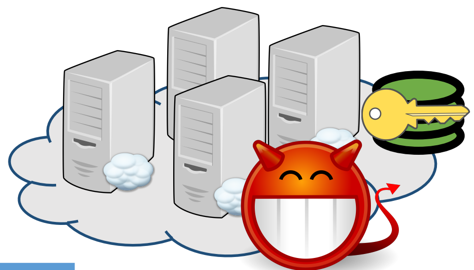
Analyst

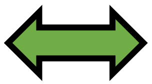
More coffee breaks!

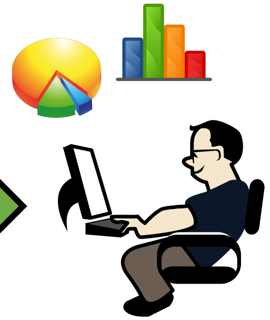
- Challenge 1: Performance
  - Aggregations on encrypted data are slower
  - Ciphertext addition is > 3000x slower than plaintext
  - Adding 100 million values takes 6 minutes instead of 100ms
    - Not good for big data!

Encrypted databases – Challenges

Retail business





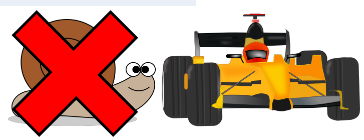


customer	gender	country	payment
%Th6j	h4$89	548yvg	439856
Fjg893	sfbg43	a3vbt9a	582650
%gTHR	h4$89	a3vbt9a	143759
34%^d	h4$89	a3vbt9a	374563

- Challenge 2: Security
  - Encrypted databases use cryptographic schemes with weaker guarantees
  - Example: deterministic encryption (DET) reveals equality
  - Recent attack [CCS15] recovered > 60% from certain DET columns

Our approach

customer	gender	revenue
%Th6j&	h4$89h	987242
Fjg893n	sfbg43q	629459
%gTHR3	h4$89h	593292
34%^db	h4$89h	742063



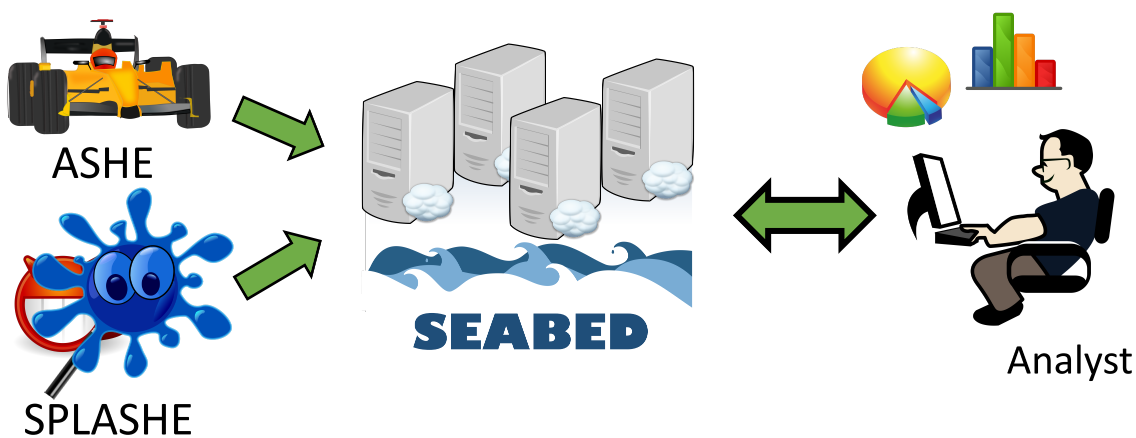
ASHE

customer	gender female	gender male	payment female	payment male
%Th6j&	gfv941E	h4$89h	654929	987242
Fjg893n	sfbg43q	rb%Gj4	629459	459239
%gTHR3	H7&fgh1	kb3i&Q	243995	593292
34%^db	D23gr$w	Epvi#$R	592623	742063

SPLASHE

- Goal 1: Improve performance
  - ASHE – New cryptographic scheme that allows fast aggregation on encrypted data
- Goal 2: Improve security
  - SPLASHE: DB transformation that enables more queries without using weaker crypto

Seabed: Big data analytics for encrypted datasets



- We built Seabed on top of Spark
  - Seabed leverages ASHE and SPLASHE
- Seabed runs SQL queries on encrypted data
  - Examples: Group-by queries and aggregations (sum, average, variance)
- Seabed is fast enough for big data
  - Up to 100x faster than previous systems

Outline

- Motivation & prior work
- Approach
- Improving performance
    - ASHE
- Improving security
    - SPLASHE
- System design
- Evaluation

Why is aggregation slow in encrypted databases?

Plaintext DB

payment
12
4
1
15

Integer addition

$+$

Sum = 32

Encrypted DB

payment
439856
582650
143759
874563

Homomorphic addition

$\oplus$

Sum = Enc(32)

- We need to sum up encrypted data
  - This is impossible with schemes like AES
- We need an additively homomorphic cryptosystem
  - Example: Paillier encryption [EUROCRYPT99]
  - $Enc(x_1) \oplus Enc(x_2) = Enc(x_1 + x_2)$

Why is aggregation slow in encrypted databases?

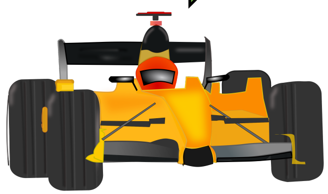
Plaintext DB

payment
12
4
1
15

Integer addition

+

Sum = 32



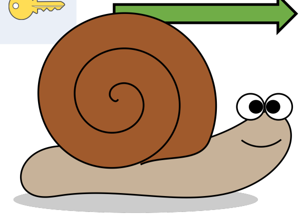
Encrypted DB

payment
439856
582650
143759
874563

Homomorphic addition

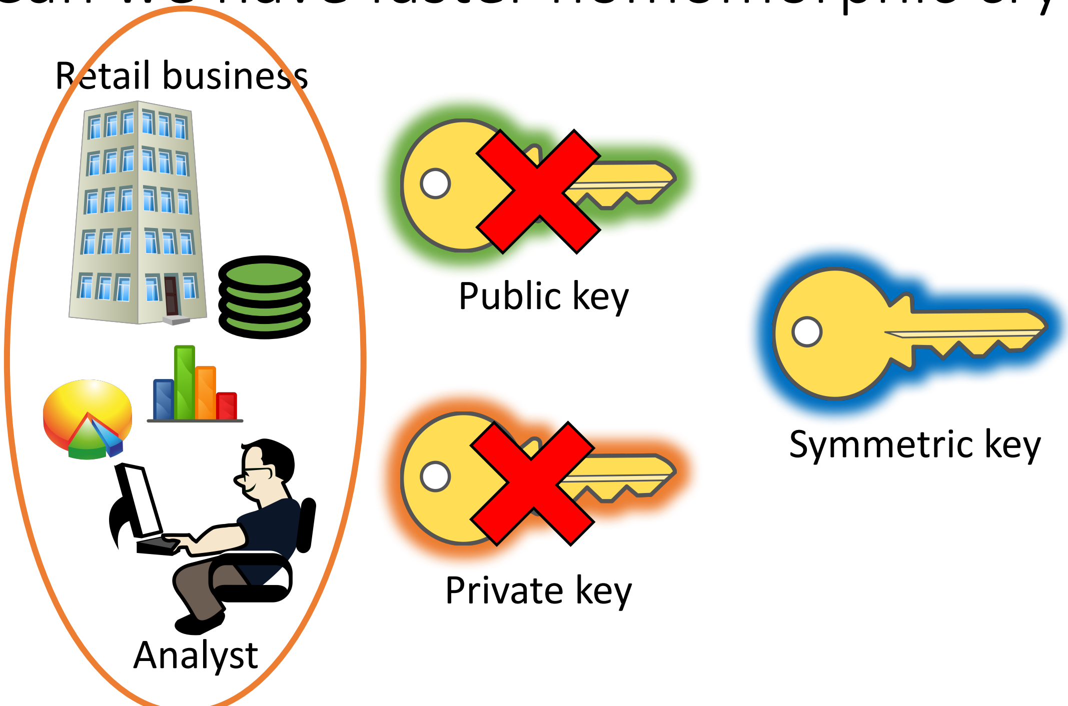
$\oplus$

Sum = Enc(32)



- Most homomorphic cryptosystems are expensive!
  - Example: Paillier ciphertexts need to be 2048-bit
  - Homomorphic addition: $Enc(x_1) \oplus Enc(x_2) = Enc(x_1) * Enc(x_2)$
  - > 3000x slower than plain addition

Can we have faster homomorphic cryptosystems?



- But why does Paillier have so large ciphertexts?
  - Because it is an asymmetric scheme based on large integers
  - Encrypt with public key – decrypt with private key
- Do we need asymmetric crypto in outsourced databases?
  - Analysts and data collector usually work for the same organization
  - We could use fast symmetric crypto!

ASHE – Additive Symmetric Homomorphic Encryption

Plaintext DB

payment
12
4
1
15

Encrypted DB

payment
12+439
4 - 56
1 + 379
15+763

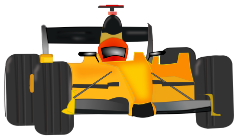
Sum = 32 + 1525
- 1525

Encrypted DB

ID	payment
1	12+F(1)
2	4 + F(2)
3	1 + F(3)
4	15+F(4)

Sum = 32 + 1525
ID list: {1, 2, 3, 4}

- Encrypt by masking values with random numbers
  - ASHE is semantically secure (IND-CPA)
- No need to remember random numbers
  - Use pseudorandom function F(ID)
- ASHE ciphertexts are 32/64-bit integers
  - Homomorphic addition only takes a few nanoseconds!



ASHE – Optimizations

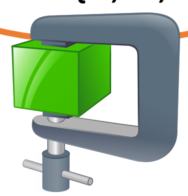
Encrypted DB

ID	payment
1	$12 + \cancel{F(1)} - F(0)$
2	$4 + \cancel{F(2)} - \cancel{F(1)}$
3	$1 + \cancel{F(3)} - \cancel{F(2)}$
4	$15 + F(4) - \cancel{F(3)}$

Aggregation

Sum = 32 $+ \sum F(i)$

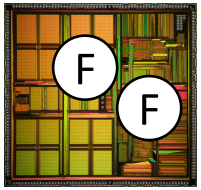
ID list: {1, 2, 3, 4}



Decryption

Compute $F(1), \cancel{F(2)}, \cancel{F(3)}, F(4)$

Decrypt: Sum $- \sum F(i)$



AES-NI

- Challenge: Aggregation and decryption cost depends on ID list length
- Optimizations:
  - Optimize encryption so that the randomness cancels out for consecutive IDs
  - Fast evaluation of pseudorandom function via AES-NI
  - Compression techniques to make ID list as small as possible
- Outcome: ASHE enables fast aggregation even when the DB is very large

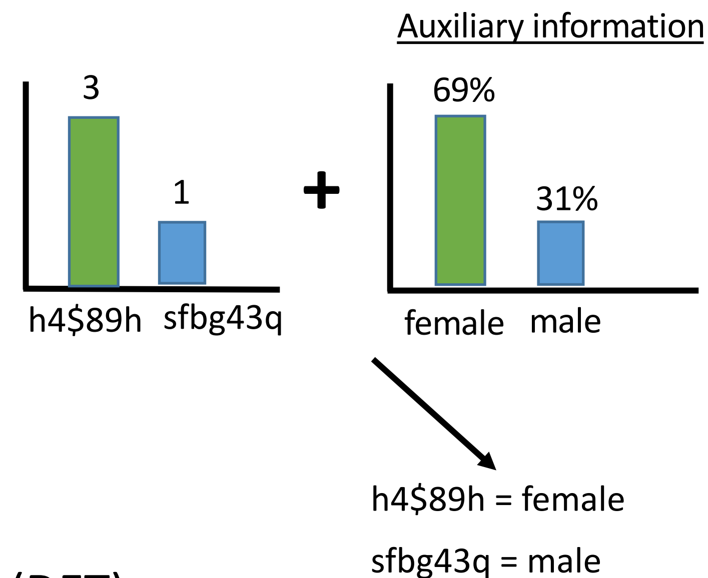
Outline

- Motivation & prior work
- Approach
- Improving performance
  - ASHE
- Improving security
  - SPLASHE
- System design
- Evaluation

Why are encrypted databases vulnerable?

customer	gender
Alice	female
Bob	male
Charlie	female
Deborah	female

customer	gender
%Th6j&	h4$89
Fjg893n	sfbg43
%gTHR3	h4$89
34%^db	h4$89



- Some columns use deterministic encryption (DET)
- This leaks the distribution of values
- An adversary with auxiliary information can do a frequency attack [CCS15]
  - In the example, the gender is revealed

How can we avoid deterministic encryption?

SELECT sum (revenue)
FROM purchases
WHERE gender = “female”

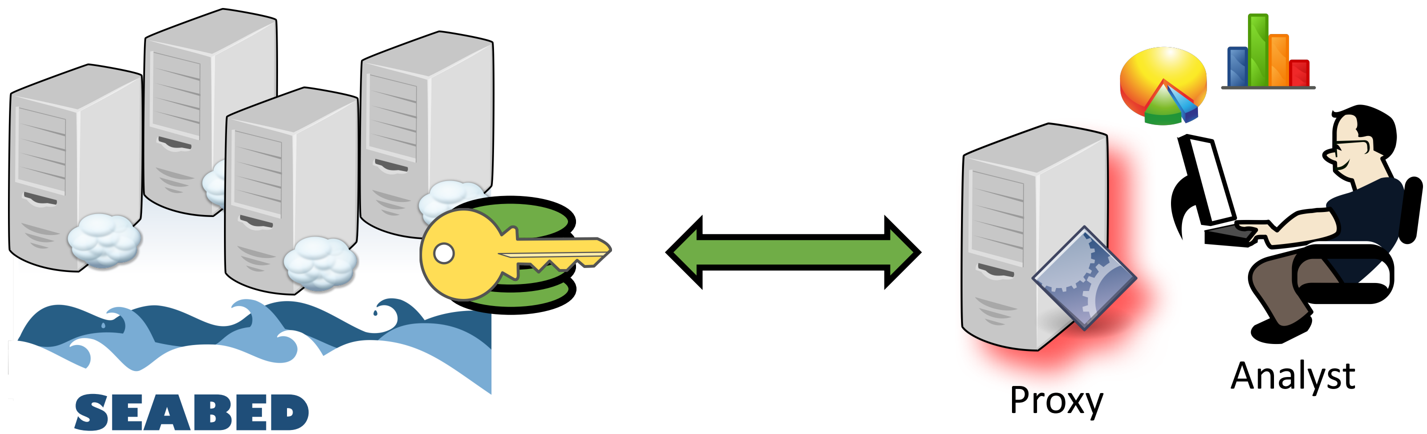
customer	gender	payment
Alice	female	12
Bob	male	4
Charlie	female	1
Deborah	female	15

customer	gender female	gender male	payment female	payment male
%Th6j&	476529	459220	439856	314437
Fjg893n	956204	953265	582650	207465
%gTHR3	529482	234599	143759	958922
34%^db	459283	562087	874563	996324

Sum = 28

- Support single-table aggregation queries without DET
- SPLASHE: Transform DB schema to avoid DET
  - Answer single-table aggregation queries using additions only
- Some storage overhead
  - Reduced by Enhanced SPLASHE (see paper)

Seabed – System design



- We implemented Seabed on top of unmodified Spark
  - ASHE and SPLASHE implemented in Scala
- Seabed's high-level design is similar to CryptDB's
  - Accepts SQL queries; transparently answers them on encrypted data
  - Client proxy handles encryption/decryption

Outline

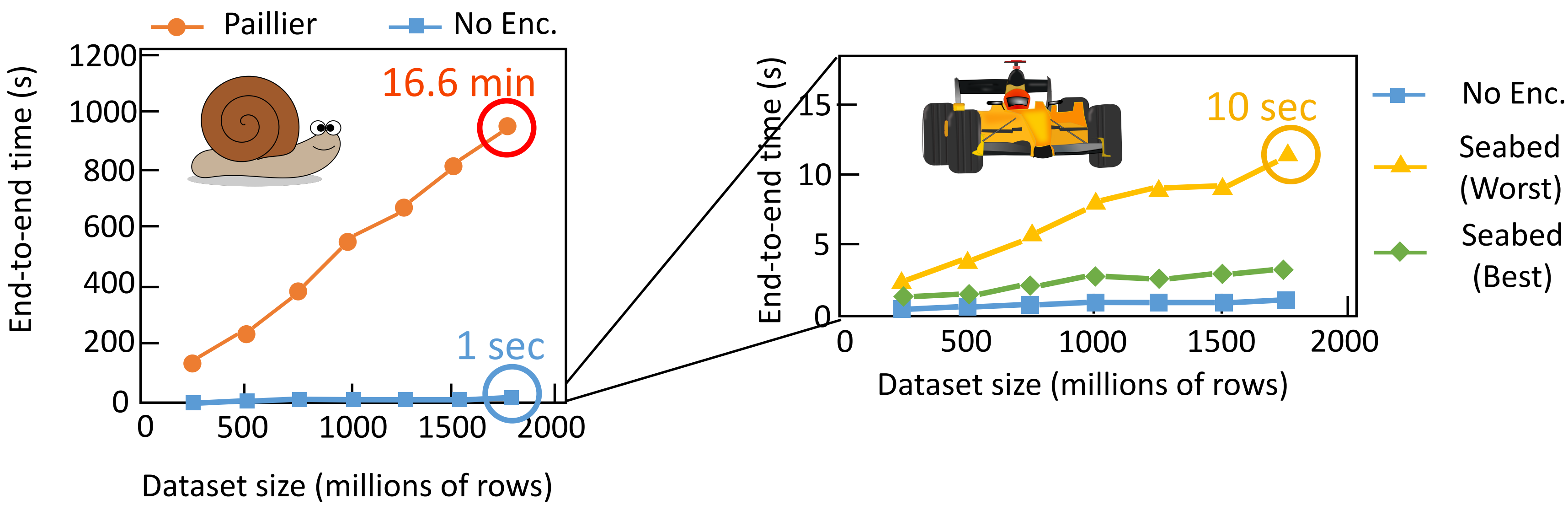
- Motivation & prior work
- Approach
- Improving performance
  - ASHE
- Improving security
  - SPLASHE
- System design
- Evaluation

Evaluation: Questions

- End-to-end latency of aggregation?
- Storage overhead of SPLASHE?
- End-to-end latency in Bing Ads analytics?
- How scalable is aggregation?
- How effective are Seabed's optimizations?
- Latency of group-by queries?
- Latency of batch queries (Big Data Benchmark)?

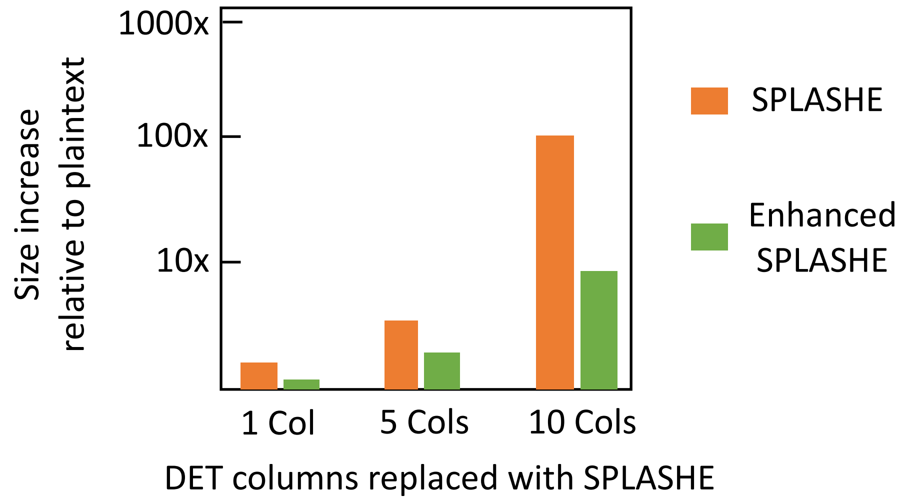
Experimental setup:
- Spark with 100 cores
- On MS Azure
- Memory-resident data

How efficient is ASHE aggregation?



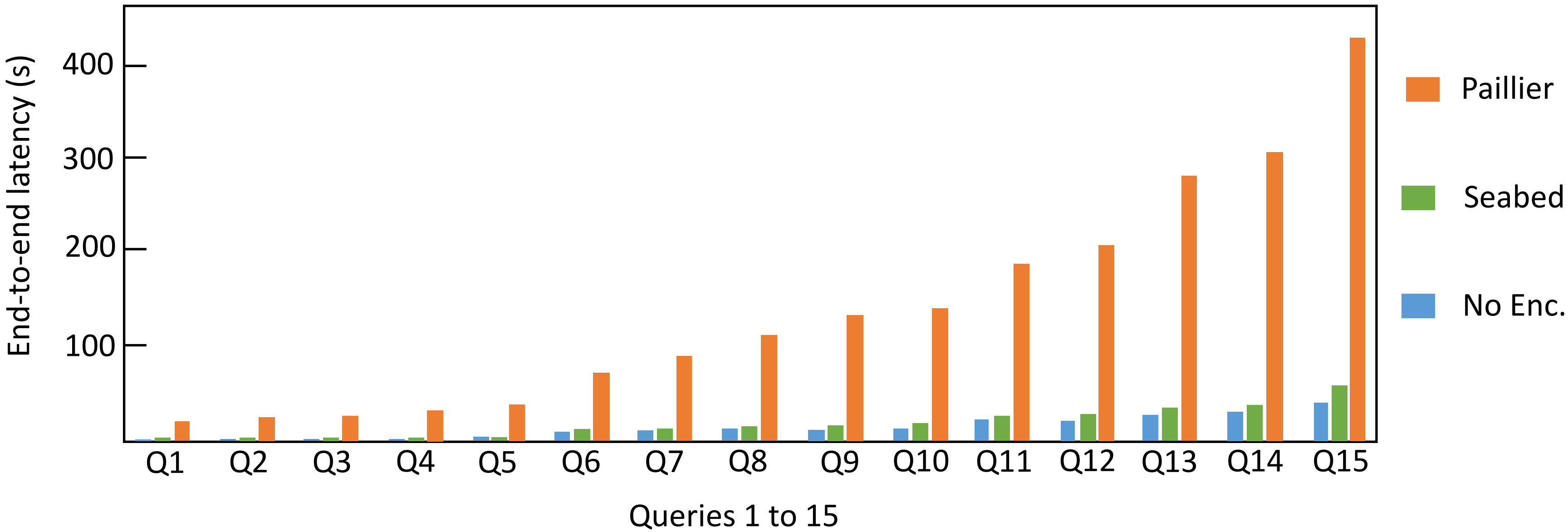
- Synthetic data: up to 1.75 billion rows - Query: single column aggregation
- Results
  - Paillier: up to 16.6 minutes
  - No encryption: <1 second
  - How does Seabed do?
- Seabed is 100x faster than Paillier, even in the worst case!

How much storage does SPLASHE need?



- Dataset
  - 760M rows, real ad-analytics application from Microsoft
- We replaced 10 DET columns with SPLASHE, one by one
- Measured: Relative size increase vs. plaintext dataset
- Results
  - SPLASHE has substantial storage cost
  - Enhanced SPLASHE reduces this cost by up to 10x
- With 10x more storage, we avoid DET entirely!
  - Reduces risk of information leaks

How efficient is Seabed for real-world applications?



- Same ad-analytics application from Microsoft
  - Measured: End-to-end latency of 15 queries
- Results
  - No encryption is about 10x faster than Paillier across all queries
  - Seabed is almost as fast as no encryption (within 15-44%)
- It is possible to do analytics on encrypted big data!

Summary

- Big-data analytics on encrypted data is difficult
  - Key challenges: Performance, security
- We introduce additive symmetric homomorphic encryption (ASHE)
  - Result: much better performance when analyst and data owner trust each other
- We present a schema transformation called SPLASHE
  - Result: Often avoids the need for weaker encryption → better security
- Seabed: an extension of Spark that uses ASHE and SPLASHE
  - Up to 100x faster than previous systems
- Seabed is fast enough for real-world big data applications

References

- [EYROCRYPT99] P.Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proc. EURO- CRYPT, 1999*.
- [SOSP11] Popa, Raluca Ada, et al. "CryptDB: protecting confidentiality with encrypted query processing." *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*. ACM, 2011.
- [VLDB13] Tu, S., Kaashoek, M. F., Madden, S., & Zeldovich, N. (2013, March). Processing analytical queries over encrypted data. In *Proceedings of the VLDB Endowment* (Vol. 6, No. 5, pp. 289-300). VLDB Endowment.
- [SQL16] https://www.microsoft.com/en-us/cloud-platform/sql-server
- [CCS15] Naveed, Muhammad, Seny Kamara, and Charles V. Wright. "Inference attacks on property-preserving encrypted databases." *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015.