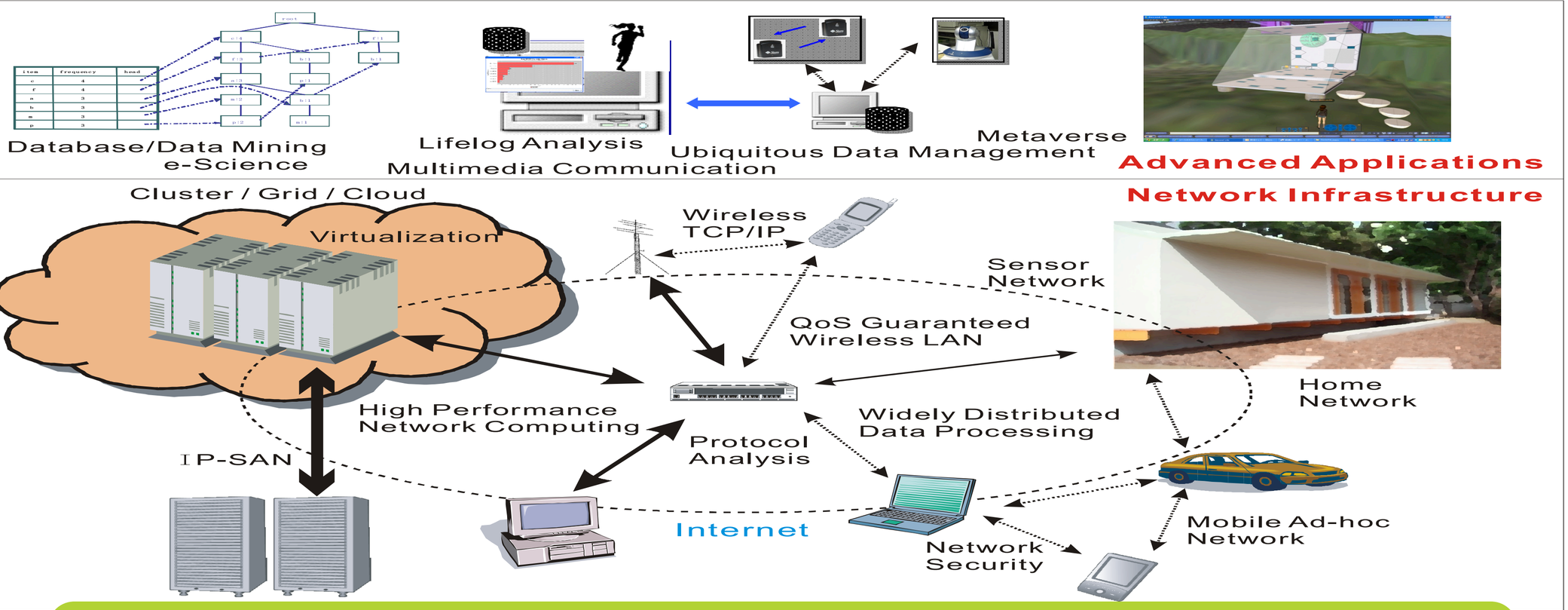
小口研究室　研究紹介　(2023年度)
(お茶の水女子大学理学部情報科学科)

次世代ネットワークコンピューティング基盤と先進的アプリケーション



◆研究テーマ：ネットワークコンピューティング・ミドルウェア
- ●多種多様な通信・計算機器が複雑に結びついて情報化社会のシステムを形成
- ●次世代ネットワークコンピューティング基盤に焦点を当て、先進的アプリケーションそれを支えるミドルウェアを研究

IPEQ: Querying Multi-Attribute Records with Inner Product Encryption
（研究担当:松本　茉倫）

Overview

- **Leakage of a scheme based on deterministic encryption (DET)**
  - The leading cryptography-based DBMS, such as CryptDB, MONOMI, explore DET.
  - DET is suffered from leakage of data frequency and search patterns.
- **The disadvantage of fully homomorphic encryption (FHE)**
  - FHE can prevent leakages, but homomorphic operations are much slower than plaintext operations.
  - The size of FHE ciphertext is storage-intensive.
- **New querying scheme *IPEQ* with function-hiding inner product encryption (FHIPE)**
  - The DB server only view the result of a query by computing **⟨x, y⟩** with the record as vector **y** and the query condition as vector **x** as shown in Figure 1.

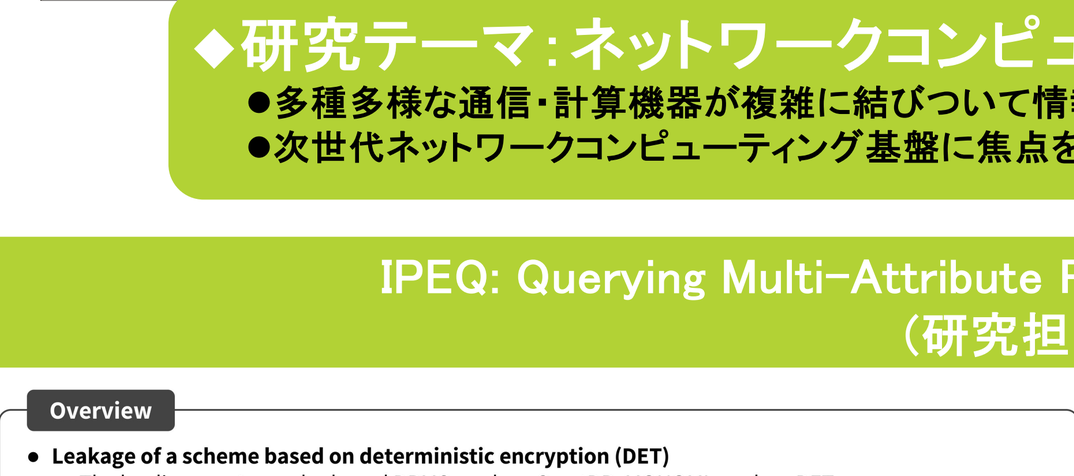
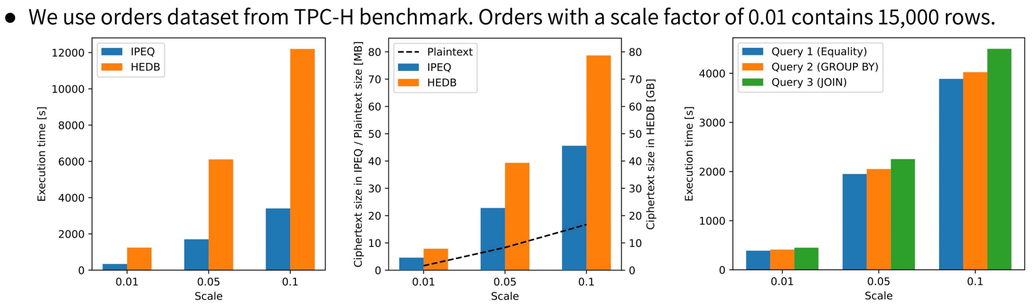


Figure 1: IPEQ system. Equality, GROUP BY, and JOIN can be performed with FHIPE.

Evaluation

- We use orders dataset from TPC-H benchmark. Orders with a scale factor of 0.01 contains 15,000 rows.



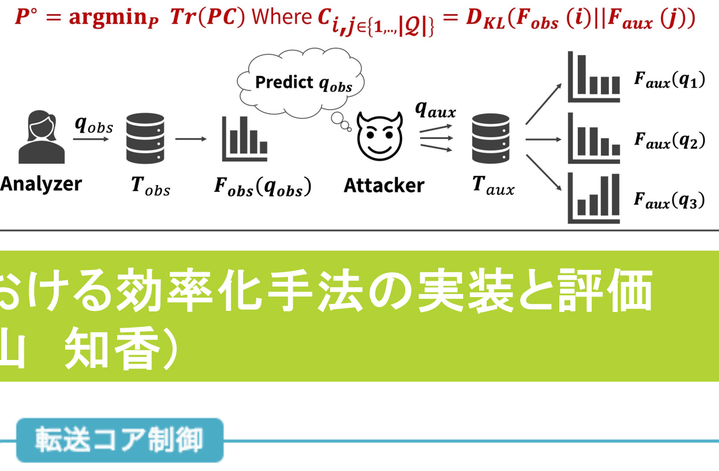
(a) IPEQ vs. HEDB (Latency)　(b) IPEQ vs. HEDB (DB size)　(C) Query Latency

Figure 2: (a)(b) IPEQ outperforms homomorphically encrypted DB (HEDB) in terms of query execution latency and DB size. (c) The execution time of the queries is linearly increasing.

Discussion on the Query Recovery Attack

- Assume that the attacker has auxiliary information T_{aux}
- Finds assignment P° and predicts q_{obs} using the Hungarian algorithm

$$P^{\circ} = \mathrm{argmin}_P \; Tr(PC) \text{ Where } C_{i,j \in \{1,\dots,|Q|\}} = D_{KL}(F_{obs}(i) || F_{aux}(j))$$



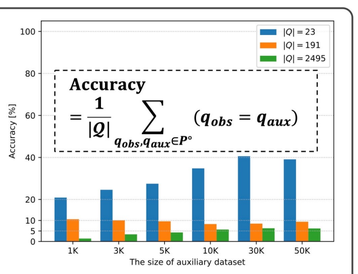


Figure 3: Leakage depends on the query set size $|Q|$ and $|T_{aux}|$.

サーバシステムの性能データ転送における効率化手法の実装と評価
（研究担当:飯山　知香）

研究概要

背景
- 多数台サーバの共有利用などの需要が増加
  - 各サーバのハードウェアレベルを含む詳細な性能データをリアルタイムで収集・分析・提示することで、より効果的な負荷分散やシステムのチューニングが可能
- 分析対象サーバとその分析を行うサーバが分割され、サーバ間でのデータ転送が必要になる場合が多い
- 時系列化された性能データはデータサイズが大きく、扱う際のオーバヘッドも大きくなる可能性
  - 性能データ：CPUコアごとに収集するデータを想定（システム情報の中でも比較的詳細かつデータサイズが大きい）

目的
- サーバ間での効率的な性能データ転送手法の実現

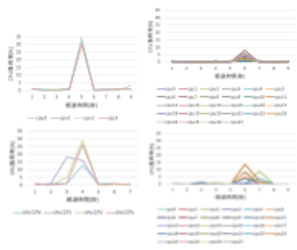
転送タイミング制御

目的
- 転送先サーバでの処理遅延がボトルネックになっていた
- 転送先サーバでのCPU負荷率を平準化することで転送時間が短縮されるか確認

実装
- CPUコア数nでの転送において、各コアiで転送前にi/n秒sleepし、コアごとの転送タイミングを1/n秒ずつずらす

データ転送時間
- ✓ CPUコア数の増加に伴う転送時間の増加が抑えられた
- ✓ 低性能解析サーバ使用時においても高性能解析サーバ使用時と同等の転送時間が可能
- ✓ n= 1~28で計測

InfluxDBのCPU負荷率
- ✓ CPU負荷がかかる時間の範囲が広がり、各コアの値も多少上下した（＝転送先で負荷がかかるタイミングにずれが生じ、負荷が平準化された）
- ✓ n=14で計測

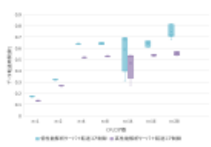


転送コア制御

目的
- 転送処理と同一CPUコア上でCPU負荷ベンチマークを動作させると互いに性能劣化が発生していた
- CPU負荷の低いコアで転送処理を動作させることで性能劣化が低減されるか確認

実装
- 転送側で毎秒Idle率の高いコアを検出し、該当コアで4コア分(1つのコアでの転送でも1秒以上かからないデータ量)または端数分のデータ転送を行わせる

データ転送時間
- ✓ nが4以下の場合は1コアでの転送のため単調増加
- ✓ nが4より大きい場合はおおよそn=4と同等の転送時間（※n=14にて端数分のデータ転送が発生しており、一部コアで転送時間が短い）
- ✓ CPUコア数の増加に伴い転送時間のばらつきが増加（＝複数のIdleコアから同時に転送を行うことでIdle時間が発生）
- ✓ n=1~28で計測



転送処理とベンチマークの同時動作
- ✓ 転送プログラムが4コアで動作
- ✓ dhry2reg（CPU負荷ベンチマーク）が10コアで動作した場合（上表）は、転送処理とdhry2regが動作するコアが分割され互いの性能劣化が抑えられた
- ✓ 上記に加えstress io（I/O負荷ベンチマーク）が4コアで動作した場合（下表）は、転送処理はstress ioが同一CPUコア上で動作したため多少劣化したが、dhry2regはほとんど性能劣化していない
- ✓ n=14で計測

今後の課題
- 転送タイミング制御における、より効率の良い実装の検討
- 転送コア制御における、CPU負荷率の低いコアのより適切な予測・抽出方法の検討

https://www.is.ocha.ac.jp/~oguchi_lab/　　oguchi@is.ocha.ac.jp