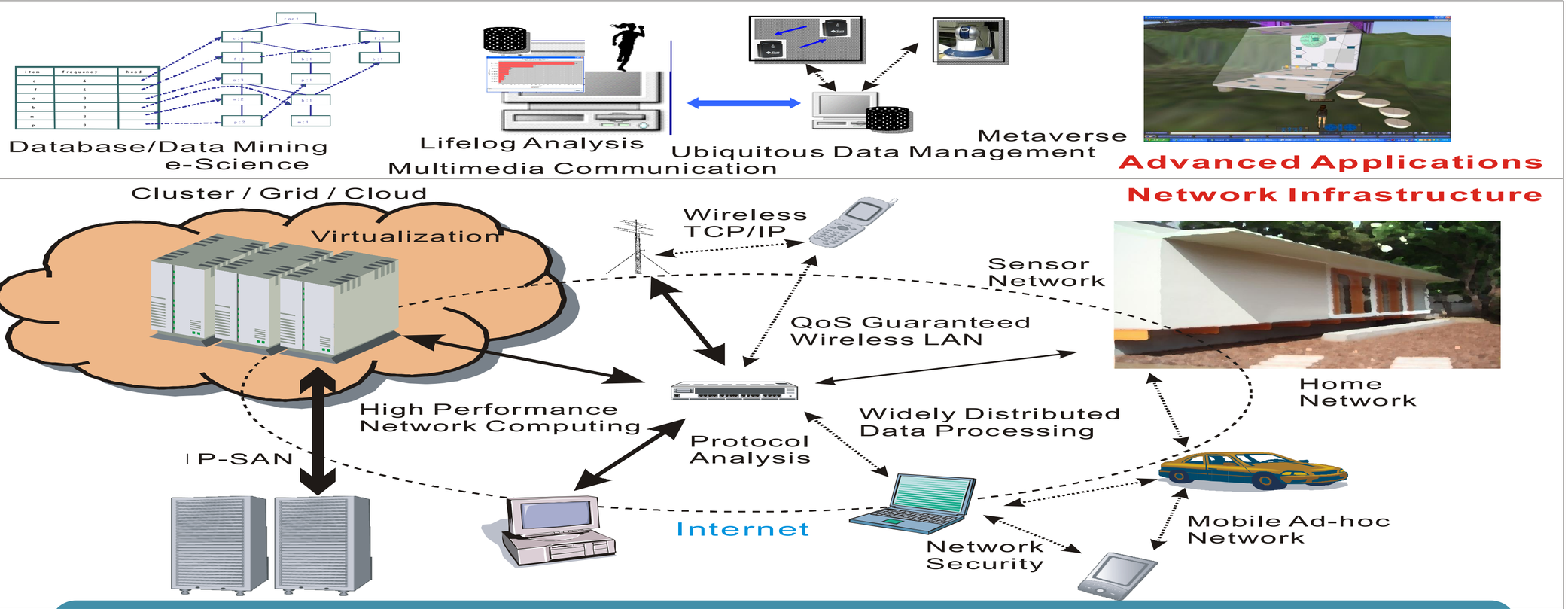
小口研究室　研究紹介　(2025年度)
(お茶の水女子大学理学部情報科学科)

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

Intel SGX を用いた属性ベース暗号システム設計とライブラリ実装
（研究担当:佐々木　怜名）

背景
- クラウドでの属性ベース暗号の活用
  - クラウドを利用して機密データを管理するユーザ数の爆発的増加
  - 公開鍵暗号では鍵管理コストの増大，スケール性の欠如
  - ➢ アクセス制御機能を備えた属性ベース暗号の活用
- 属性ベース暗号（ABE: Attribute-Based Encryption）
  - 公開鍵暗号方式を基にした高機能暗号の一つ
  - 特定の属性を持つユーザのみが復号できる暗号方式
    - 鍵管理サーバが公開鍵とユーザ秘密鍵を管理
    - 公開鍵で復号条件を組み込みデータを暗号化
    - ユーザは各属性に基づいた秘密鍵で復号条件を満たせば復号

Ciphertext-Policy ABE (CP-ABE)
1. 鍵管理サーバは公開鍵 (PK)とマスタ秘密鍵 (MSK)を生成
2. データユーザ (DU) は自身の属性リスト (A_u) を鍵管理サーバ (KMS)に送信
3. KMSは，MSKを用いて属性リストに依る秘密鍵 (SK_u) を生成しDUに送信，データ所有者 (DO)にPKを送信
4. DO はデータを復号条件であるポリシを組み込みデータを暗号化し，クラウドに暗号文 (CT)をアップロード
5. DUは自身のSKを用いて復号

■例

属性リスト: *(ID: 334, "manager", 3rd floor)*

ポリシ: "manger∧ (ID > 300) ∨ 2nd floor"

提案システムの概要と脅威モデル
- KMSとDUはIntel SGX上で動作
- 秘密鍵の生成や復号処理はEnclave内で実行しABEの秘密鍵を保護
- KMSとDUのEnclave間でRA-TLSを用いてセキュアチャネルを確立
- Trusted
  - DO
  - KMSのストレージ
  - 各Enclave
- Untrusted
  - KMSのホスト環境
  - DUのホスト環境
  - 通信路

- 攻撃者はEnclave外のメモリやDUの盗聴・改ざんできる能力を持つものとする
- 失効ユーザは既存の暗号文を復号しようとする悪意のあるユーザに転換する恐れがある

提案プロトコル
- 以下の2つのケースのように最適なプロトコルを動的に選択し効率を図る
  - ① Preliminaries Protocol＋Offload Decryption Protocol（ABE復号処理をKMSに委託）
  - ② Preliminaries Protocol＋ Local Decryption Protocol（DU側で全ての復号処理を実行）

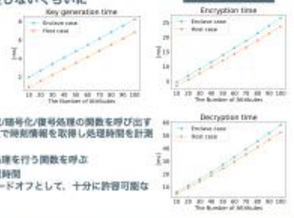
- Preliminaries Protocol:
  - KMSとDUのEnclaveの真正性の確認（Remote Attestation）
  - ABE処理に必要なsetup処理
  - Offload Decryption ProtocolとLocal Decryption Protocolの選択
- DUの性能とKMSの負荷状況に応じ2つのプロトコルを選択
- Offload Decryption Protocol:
  計算資源が豊富なKMSに復号処理を委譲し，制約のあるDU側の負担を軽減
- Local Decryption Protocol:
  DUが十分な性能がある時，復号をDUで完結させることでKMS側の処理負荷を分散

ABEシステムの課題
- KMSが外部から悪意のある攻撃を受けるリスクが高い
  - KMSは全ユーザの秘密鍵を生成し保存
  - KMSは秘密鍵を生成するのに必要なマスタ秘密鍵を保持
- アクセス権限を失ったDUの確実な秘密鍵の失効方法
  - 非失効ユーザの秘密鍵の更新: スケール性が乏しい
  - 暗号文委譲 (ciphertext delegation): 処理負荷が高い

➢ TEE（Trusted Execution Environment）を用いて，KMSやDUのホスト環境からも秘密鍵を秘匿する

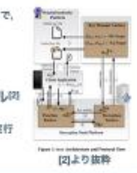
ABEライブラリの課題
- OpenABE, Charm, CiFEr, cpabe-toolkitなど多くのOSSライブラリが存在
- ✓ 既存ライブラリのTEE（Intel SGX）への適用に限界がある
  - 依存関係の問題:
    システムコールや標準C/C++ライブラリに強く依存
  - ➢ TEEの制限:
    Intel SGXなどの環境では，標準的なOS機能（I/O, 乱数呼び出し，動的ライブラリのリンク）が制限されているため，既存ライブラリはそのまま動作しない

ABEライブラリの実装
- Intel SGX対応の新規ABEライブラリ実装
  既存ライブラリ（Charm, OpenABE等）はOS非依存の制約（動的リンク・システムコールの禁止）によりEnclave内で動作しないため，Enclave内で使用可能なC/C++ライブラリ(tiibc/tiibcxx)や，SGX対応のopenssl, gmpライブラリ，改変したペアリングライブラリ(relic)を用いて再実装
- Waters11方式[8]の採用：
  安全性とデファクトスタンダードの両立．標準モデルで安全性が証明されたWaters11 を採用
- パフォーマンスの考慮
  最も負荷の高いペアリング演算中に，低速な暗号化ページング（EPC Paging）が発生しないくらいにEnclaveサイズを最小化

評価


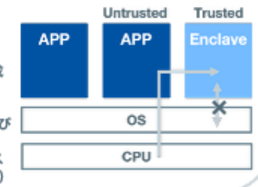
- Enclave case
  - ホスト環境でECALL関数を用いて鍵生成/暗号化/復号処理の関数を呼び出すEnclaveで各処理が完了後，OCALL関数で時刻情報を取得し処理時間を計測
- Host case
  - ECALL関数の代わりに，ホスト側で各処理を行う関数を呼出
- Enclave case≒Host caseより約1.1倍の処理時間
  - ➢ 機密情報であるABE SKの保護とのトレードオフとして，十分に許容可能な範囲

関連研究
- TEEを用いたABEシステムプロトコル[1]
  - 信頼できないCSP上でも，鍵管理サーバ（KMS）にSGXを適用することで，秘密鍵の漏洩を防ぎ，ユーザをID管理することで失効を容易にする
  - 秘密鍵をKMSのEnclaveに留め，Enclave内で部分的にABE復号処理
  - ペアリングライブラリPBCを用いて実装
  - ➢ 本研究ではより安全で性能が良いRelicライブラリを用いて実装
- IRON: Intel SGXを用いた関数型暗号システムのプロトコル[2]
  - 鍵管理(KME)/復号(DE)/関数実行の3つのEnclaveでセキュアに動作
  - クライアントの要求に基づきKMEが鍵を解放し，DEが暗号文を復号・実行
  - シミュレーション安全性を用いて安全性を厳密に証明
  - ➢ 本研究ではABEをSGXへ適用する



[2]より抜粋

TEE (Trusted Execution Environment)
- TEE（Trusted Execution Environment）
  - HWにより隔離実行環境を提供
  - TEE外からの改ざんや読み書きを防ぐ
- Intel SGX
  - メモリ上に隔離実行環境（Enclave）を生成
  - OSを信用せずデータを保護しプログラムを実行
  - ECALL: Enclave外からEnclave内の関数呼び出し⇔ OCALL
  - 動的ライブラリのリンク，Enclave内のシステムコールの使用などを禁止 （e.g. sprintf）

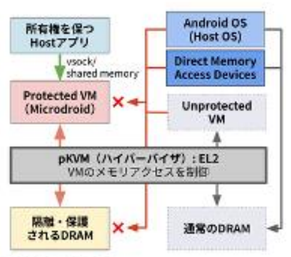


本研究の貢献
- 目的
  鍵管理サーバの脅威と失効時のスケーラビリティの課題をTEEを用いて解決
- 提案
  外部攻撃による鍵漏洩リスクを低減しつつ，リソース制約のあるデバイスでも動作するABEシステム
- 貢献
  - ユースケースに合わせて柔軟にプロトコルを選択できるTEEを用いたCP-ABE のシステムプロトコルの提案
  - ABE ライブラリのフルスクラッチ実装と，Intel SGX におけるCP-ABE 処理（鍵生成, 暗号化, 復号）の評価

今後の課題
- 形式検証を用いたプロトコルの安全性証明
- オフライン環境下でのユーザ秘密鍵の失効処理方法の検討
- 提案プロトコルの実装
- リソース制約端末における実用性の実測検証
- 先進ABEスキーム (FAME, FABEO, etc.)のTEE移植

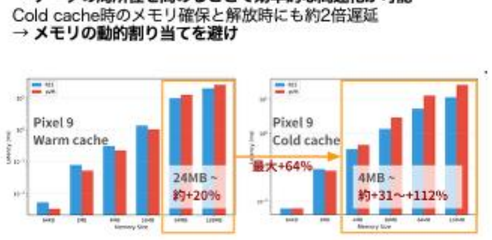
Google Tensor搭載端末のpKVMにおけるセキュアな音声処理および声紋認証の実装手法と課題の検討
（研究担当:松田　華）

背景
- スマートフォンにおける声紋認証はアクセシビリティ等の観点でのメリット
- 声紋は生体情報であり，強固な保護や，セキュアな環境での導出が必要
- Androidで従来のセキュア環境を提供するTrusty(TEE)はメモリ制約が厳しい
- Android 13で大容量メモリが使えるセキュアな環境を提供するAVF(pKVM)が導入
- pKVMでの音声処理の文献は著者が調べた限り存在せず実機での評価を行った研究も少数



本研究の貢献
- ✓ **pKVM 環境における実装特性の調査**: メモリアクセス性能を調査し設計指針を導出
- ✓ pKVM 環境でのDNNモデルを使用した**声紋認証システムの実証**
- ✓ メモリアクセス性能調査に基づくシステムの**最適化**と**パフォーマンス評価**
- ✓ **pKVM環境への機械学習モデル導入時の特性と課題の明確化**

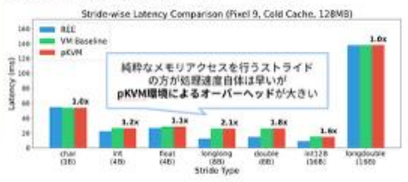
pKVMのメモリアクセス性能評価
- Cold Cache時のオーバーヘッド大
  Warm cache時に**最大64%**高速
  → **データの局所性を高めることで効率的な高速化が可能**
- Cold cache時のメモリ確保と解放時にも約2倍遅延
  → **メモリの動的割り当てを避け**



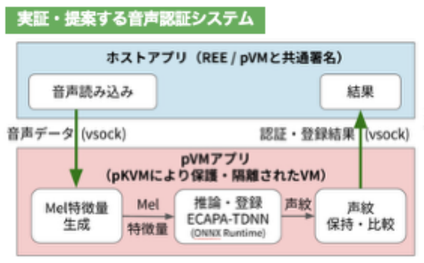
事前確保や再利用で効率的な高速化

操作(Pixel 9)	VM(ns)	pVM(ns)
MALLOC/FREE(Warm)	41/41	41/41
MALLOC/FREE(Cold)	81/81	162/163

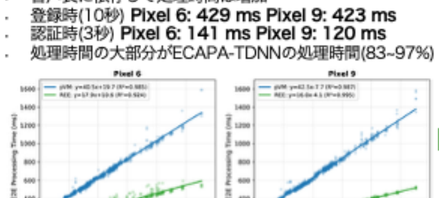
- VM同士の比較では純粋なメモリアクセスの方が処理速度は早いがREEとの比較で仮想化オーバーヘッド大



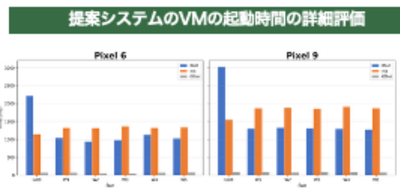
実証・提案する音声認証システム



処理時間 (JVSコーパス平均16秒の音声)
- 音声長に依存して処理時間は増加
- 登録時(10秒) **Pixel 6: 429 ms Pixel 9: 423 ms**
- 認証時(3秒) **Pixel 6: 141 ms Pixel 9: 120 ms**
- 処理時間の大部分がECAPA-TDNNの処理時間(83~97%)

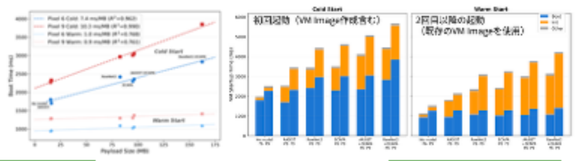


提案システムのVMの起動時間の詳細評価



- **Boot**: Microdroid カーネル起動
- **Init**: ペイロード初期化
  (ONNX モデル読み込み等)
- **Other**: VM設定，
  VM Image作成(Cold Start)
  既存VMの検索(Warm Start)
  ソケット接続確立など

提案システムのモデルサイズによるVMの起動時間の調査
- モデルの組み込み内容が異なる6構成でVM起動時間を比較(各n=10)
- Cold Start時(Image作成時)のBoot時間はペイロードサイズに依存し増加
- Init時間はモデルの読み込みのコスト → モデルの構造とサイズに依存



結論
pKVM環境での声紋認証システムの実証と最適化
実用化されているAndroid端末にて，声紋認証の登録・推論時間共に0.5秒以下に抑えられることを実証

pKVMアプリ開発の知見
数百MBのDNNモデルをセキュアに実行可能
他分野のDNNモデルの組み込み時にも知見を応用可能

今後の課題
- pKVM環境に合わせたモデルの最適化
- マイクとpVM間でのTrusted Audio Pathの確立
- 合成・再生音声攻撃対策強化や推論時間短縮

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