機密 AI における監査可能な情報開示の強制

椎名 峻平[1]　髙橋 遼[1]　佐々木 謙人[1]　林 卓也[1]　髙橋 翼[1]

1. 背景と課題

外部の生成 AI サービス利用の際，機微な情報がサービス事業者へ開示されてしまう懸念がある．機密コンピューティング（Confidential Computing）では，プロセッサが提供する Trusted Execution Environment（TEE）機能を用いた隔離実行により，事業者やインフラ管理者から処理中のデータを保護する．近年では CPU から GPU まで一貫して機密コンピューティングをサポートするハードウェアが登場しており，LLM の推論全体を保護する機密 *AI*（*Confidential AI*）が実用化されつつある．

機密 AI はユーザのプロンプトを TEE 外部に開示しないことを原則とするが，その一方で，完全な秘匿は現実的ではない．実用的な AI エージェントは Web 検索や文書検索などの外部 API 連携が欠かせないが，これらの呼び出しにはユーザのプロンプト由来の情報が含まれる．また，安全な AI モデル運用も事業者の責務であり，障害調査や安全性の監視にも実行ログ等の情報が必要になる．

しかし，このような限定的な開示を一度許してしまうと，真に機密を保護できていることをユーザ側で検証することが困難になる．出力形式の制限や LLM による機密除去を開示の条件とすることも可能だが，不適切な開示が残る可能性はある．本研究では，こうした出力制御を補完し，**必要な開示を許しつつ，実際に開示した内容をユーザに申告**することをシステム的に強制する．

2. Confine or Declare 原則

TEE 外部へのデータ開示の原則として *Confine or Declare* **原則**を提案する．ユーザ由来のデータは基本的には**秘匿**（*confine*）するが，外部開示が必要な場合には，**申告**（*declare*）を条件として TEE 外部への出力を許す．申告とは，開示する内容・開示先・理由を記録し，改竄検知可能な記録をユーザに残す操作である．申告にはユーザの承認を必須とせず，ユーザは事後的に記録を確認する．

申告された記録は事業者側で管理し，ユーザに提供する．ここでは，ユーザの信頼を損なう挙動を避ける事業者を想定し，記録へのアクセスを妨げないものとする．一方で，ユーザに検知されない範囲で機密を出力することは想定し，申告の回避や記録の改竄・隠蔽を行う可能性がある．そのため，システムとしては事業者によるこれらの行為を防止，あるいは検知可能にする必要がある．

3. CORDE の設計

この原則を満たす機密 AI システムとして，CORDE（Confine-OR-Declare Enforcement）を設計・実装した．図 1 に CORDE の構成を示す．CORDE は推論サービス（inference service）と記録サービス（ledger service）から構成され，それぞれ機密 VM（Confidential VM）型の TEE 上で動作する．推論サービスはユーザのプロンプトを処理し，外部への開示が必要な場合には必ず記録サービスに対して申告し，記録が確定した後に同じ内容を外部に送信する．記録サービスは申告された記録を改竄検知可能な状態で管理し，ユーザの要求に応じて記録を提供する．ユーザは Remote Attestation（RA）で期待したサービスが TEE 上で動作していることを検証する．

3.1 申告を迂回しない情報フロー追跡

推論サービスでは複数ユーザを扱い，異なるユーザのデータを取り違えず，外部開示に申告を強制する必要がある．そこで，Rust 上でユーザ由来の値を追跡し，外部通信の前に申告操作を強制するライブラリ `corde-rs` を実装した．`corde-rs` は Sesame [1] と同様，ユーザ由来の値を限定したブロック内に展開して柔軟に処理することが可能であり，そのブロック内で副作用がないことを中間表現に対する静的解析で確認する．その `corde-rs` を用い，Rust の AI エージェント開発フレームワークである Rig でエージェントループを実装した．LLM の推論には vLLM を用

[1] 株式会社 Acompany

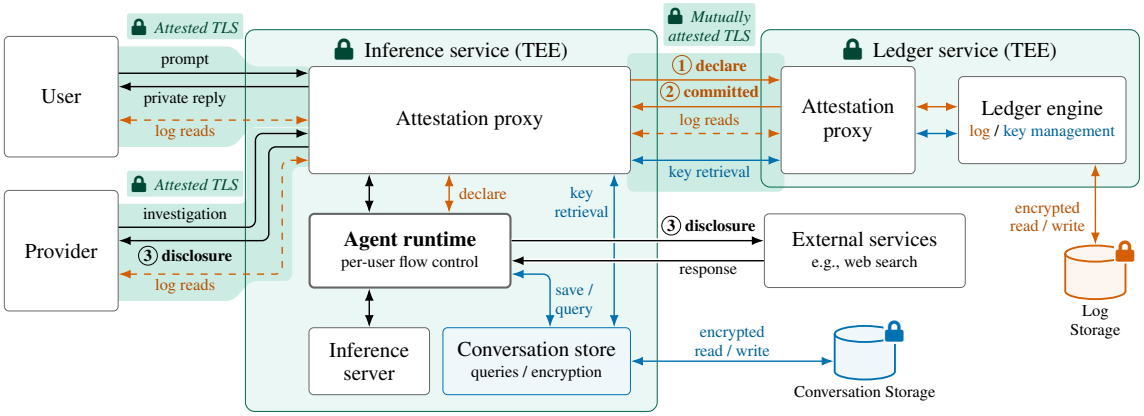


図 1　CORDE の構成．TEE 内で処理し，申告①と確定確認②の後に外部へ開示③する．

い，vLLM が外部通信を行わず，リクエスト単位で論理的に独立な処理を行うことは信頼する．

3.2　申告先と確認先を一致させる設計

記録サービスは TEE 上で改竄検知可能な透明性ログを提供する Confidential Consortium Framework（CCF）[2] を用いて実装した．CCF は TEE 内でログの最新状態を追跡し，外部ストレージに保存したログの改変・巻き戻しを検知する．しかし，単に CCF を組み合わせるだけではログの一意性は保証されない．例えば，推論サービスの申告先のサービスと，ユーザから見える確認先のサービスが別のインスタンスであっても，RA による検証は通過するが，申告はユーザに観測できなくなる．この問題を防ぐため，CCF の TEE 内で保持される秘密鍵に対応する公開鍵を一意なログの識別子として用いる．推論サービスは起動時に接続先のログの識別子を固定し，対応する秘密鍵を持つログサービスに対してのみ申告する．ユーザは推論サービスに対する RA の際に申告先のログの識別子を問い合わせ，確認先のログと一致することを確認する．

暗号化して外部保存した会話履歴を別の推論インスタンスで復号する場合も，同じログへの申告を保つ必要がある．本設計では記録サービスが鍵管理も担い，自身に対して申告する推論インスタンスにのみ復号鍵を渡す．

4.　実験評価

Microsoft Azure 上の機密 VM および NVIDIA H100 GPU を用いて CORDE の性能評価を行った．LLM モデルは BF16 Qwen3-32B [3] を使用した．

ワークロードとしては，頻繁に外部 API 呼び出しを行う AI エージェントベンチマークである AppWorld [4] を用いる．AppWorld では外部サービスを模擬した API を MCP 経由で呼び出し，CORDE ではその呼び出しに申告が必要になる．各 723 タスクを逐次実行し，申告を除いて CORDE と同様の構成で推論を行う実装（ベースライン）と実行時間を比較する．

結果として，平均タスク時間の増加は 0.44%であり，これが AppWorld における申告のオーバーヘッドであると確認された．vLLM では結果の再現性を保証する設定を行い，全タスクでモデルの推論結果が一致し，成功数はいずれも 96/723 件であった．

加えて，事業者によるインシデント調査の概念実証を行った．この例では，ユーザが保存した秘密がプロンプトインジェクションによって外部に誤って送信されてしまう状況を想定する．事業者は，ユーザからの依頼に応じて原因調査を行うが，秘密を含む全文を外部出力せず，TEE 内部の LLM による調査サマリーだけを申告して開示する．結果として，事業者はユーザの秘密を見ることなく調査サマリーだけを出力したことをユーザから確認できた．

5.　まとめ

CORDE は，TEE からの外部出力を一律に禁止するのではなく，実際の開示を利用者が検証できる構成を与える．これは AI サービスにおける開示に対する透明性と説明責任を考える上での重要な基盤になると期待される．

参考文献

[1] Albab, K. D. et al.: Sesame: Practical End-to-End Privacy Compliance with Policy Containers and Privacy Regions., *Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles, SOSP 2024, Austin, TX, USA, November 4-6, 2024*, ACM, pp. 709–725 (2024).
[2] Howard, H. et al.: Confidential Consortium Framework: Secure Multiparty Applications with Confidentiality, Integrity, and High Availability., *Proc. VLDB Endow.*, Vol. 17, No. 2, pp. 225–240 (2023).
[3] Yang, A. et al.: Qwen3 Technical Report, arXiv preprint arXiv:2505.09388 (2025).
[4] Trivedi, H. et al.: AppWorld: A Controllable World of Apps and People for Benchmarking Interactive Coding Agents., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, Association for Computational Linguistics, pp. 16022–16076 (2024).