機密実行環境を利用した個人情報管理のための高い可用性を持つログサーバの設計

一木 祐介[1]　新城 靖[1]　曽山 暉史[1]

1. 序論

近年、インターネット上で利用者は物品購入や動画視聴などのために様々なサービスを利用できる。これらのサービスを利用する際には、利用者は住所や決済用のクレジットカード情報などをサービス提供者に送信することが多い。しかしながら、サービス提供者が受け取った個人情報を最小限の範囲でのみ利用しているかどうかを利用者が確認するすべはない。

我々は、機密実行環境（Trusted Execution Environment, TEE または Confidential Computing Environment）を用いた個人情報保護を実装している [1]。この手法では、利用者はサービス提供者側による個人情報のコピー・移動等の可能な操作を限定し、また利用者自身がサービス提供側による操作を追跡できる。しかし、従来の手法では構成要素であるログサーバが単一障害点となっていた。また、データ管理プログラムはモノトニックカウンタを必要とし、実装が複雑であった。

そこで、この論文ではシステム全体の可用性を向上すること、またデータ管理プログラムの実装の簡素化について述べる。具体的には、複数のノードで動作するログサーバの設計について述べる。その特徴は Conflict-free Replicated Data Type（CRDT）の Observed-Remove（OR）集合を用い、単一障害点を無くしていること、および、データ管理プログラムに対して信頼できる永続的なストレージを提供することでモノトニックカウンタを不用としている点にある。

2. 前提と脅威モデル

本研究で実装する個人情報管理は、以下のハードウェアやソフトウェアを要求する。ここで、モノトニックカウンタは不用である。

- CPU による機密実行環境
- 信頼できない永続記憶
- 機密実行環境を繋ぐネットワーク

本研究では、以下の攻撃を想定する。

- プログラムの内容や主記憶の内容の改変

[1] 筑波大学
University of Tsukuba

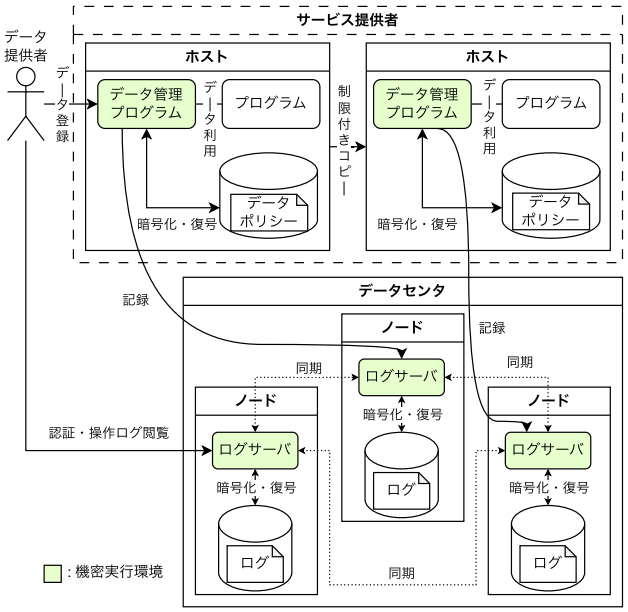

図 1　機密実行環境による個人情報のコピー制限と利用記録の実装

- プロセスの任意のタイミングでの停止
- ネットワーク上のトラフィックの傍受・改ざん
- 永続記憶内のデータの窃取・改ざん・ロールバック

3. 提案手法

本研究では、個人情報のコピー制限と利用記録の提供を、図 1 に示したようなプログラムで実装する。データ提供者は、自身の個人情報と利用方法を定めたポリシーを、サービス提供者の機密実行環境で動作するデータ管理プログラムに送信する。送信されたデータをサービス提供者が利用する際には、データ管理プログラムに対して出力を要求する。このとき、ポリシーに基づいて閲覧の可否を判断し、その結果をログサーバに送信する。ログサーバは結果を含んだログ項目を記録し、データ管理プログラムに対して記録が完了した旨を通知する。最後に、その通知を受け取ったデータ管理プログラムがサービス提供者にデータを出力する。サービス提供者が別のノードにデータをコピーする際にも、同様にコピーの可否が判断され、その結果がログサーバに送信される。

データ提供者は、ログサーバに保管された自身のデータの出力とコピー操作ログの閲覧要求を送信できる。このとき、要求を送ってきた利用者を認証し、当人のデータに関

するログ項目のみを返却する。

サービス提供者のノードやログサーバが動作するノードは、データ提供者にとって信頼できない。2 章で述べたように、プログラムの改変やプロセスの停止などの試行により、操作の制限やログの記録が妨害される可能性がある。このような、信頼できない場所でもプログラムを完全に動作させるために、データ管理プログラムとログサーバのプログラムを機密実行環境で動作させることによって、2 章で述べた攻撃に対応する。

4. ログサーバの設計

4.1 ログの記録・記録

データ管理プログラムはデータの出力・移動・コピーの操作を行うとき、操作の種類、操作の目的、プログラムが動作するノードを示す ID、そして利用者の証明書を引数としてログサーバの API を呼ぶ。証明書は、データ提供者がのちにログを閲覧する際に、その利用者に対応付けられたログを取得するために利用される。

本研究では、各ログサーバは、CRDT の OR 集合を用いて、保持しているログ集合を他のログサーバと同期させる。この手法は、複数のログサーバでログ集合が同期でき、単一障害点が存在せず、またログ項目を削除可能である。ログサーバは、データ管理プログラムによってログ項目が追加されるとき、他のノード上で動作するログサーバの API を呼び、ログ項目を分散して記録する。また、ログサーバのクラッシュが発生したときには、再起動後に他のログサーバの API を呼び出してローカルに不足しているログ項目を取得する。このとき、CRDT の OR 集合に従い、各ログサーバがデータ管理プログラムから直接追加された最新のログの ID からなる追加集合に関するバージョンベクタや削除集合に関するバージョンベクタを引数として渡すことで、不足しているログ集合の差分のみを受信する。これにより、2 章で述べた永続記憶に対するロールバック攻撃を受けてもモノトニックカウンタ無しで対処できる。

4.2 信頼できる永続的なストレージの提供

データ管理プログラムは、ノード間でデータの移動やコピーを行う時、2 相コミットに基づく分散トランザクションを実行する [1]。このとき、ログサーバのストレージを利用して分散トランザクションの進行状況を記録する。ログサーバは 4.1 節で述べた手法で他のログサーバと協調し、信頼できるストレージを実装する。トランザクションが完了するか指定された期間を超えて未完了であった場合には、ログサーバは進行状況のデータを削除する。

5. 実装

本研究では、機密実行環境の実装として Intel Software Guard Extensions（SGX）を利用する。データ管理プログラムとログサーバのプログラムは、Rust 言語を用いて実装する。Rust は、言語仕様や型システムによりプログラムの型安全性およびメモリ安全性をコンパイル時に保証でき、この特徴はセキュリティを重視する本研究において重要である。また、Rust から Intel SGX の機能を利用するために Rust SGX SDK の v2 を使用する。プログラム間でデータをシリアライズして交換するためのプロトコルとして、gRPC を使用する。

6. 関連研究

Custos [2] は、機密実行環境とそれに備わる暗号化機能を使用し、ログを記録する研究である。複数のノードを用いてログを分散して記録する点で、本研究と共通している。本研究では、ログの閲覧の際にアクセス制御を行う点や信頼できる永続的なストレージを提供する点が異なる。

ROTE [3] は、機密実行環境でのロールバック攻撃を防止するにあたって、複数のノードでカウンタの値を保持しローカルの不揮発性メモリを利用したモノトニックカウンタを不用とする。本研究では、CRDT の OR 集合やバージョンベクタの活用によりモノトニックカウンタを不用とする点が異なる。

7. まとめ

我々は、利用者がサービス提供者に送信した個人情報の利用・コピーを制限し、操作の記録を追跡できるデータ管理を実装している。本論文では、そのデータ管理で単一障害点となるログサーバを複数のノードで動作させ、システム全体の可用性を向上させることと、データ管理プログラムの実装を簡素化することについて述べた。提案手法では機密実行環境で分散して動作するログサーバを用いて信頼できる永続的なストレージを提供し、モノトニックカウンタを不用にする。

現在、ログサーバはユーザ空間で動作し、データの操作ログ項目の記録・取得ができる。今後はログサーバ同士の同期処理の実装や機密実行環境上への移植を進める。

参考文献

[1] Ishiguro, J., Shinjo, Y. and Soyama, A.: Controlled copying of persistent data between end users' SGX enclaves over an untrusted network, *IEEE International Symposium on Parallel Computing and Distributed Systems (PCDS2024)*, 10 pages (2024).

[2] Paccagnella, R., Datta, P., Hassan, W. U. et al.: Custos: Practical Tamper-Evident Auditing of Operating Systems Using Trusted Execution, Network and Distributed System Security Symposium, National Science Foundation, pp. 1–18 (2020).

[3] Matetic, S., Ahmed, M., Kostiainen, K. et al.: ROTE: rollback protection for trusted execution, *Proceedings of the 26th USENIX Conference on Security Symposium*, SEC'17, pp. 1289–1306 (2017).