Intel SGXを用いたセキュアなSiloの設計と実装

福山 将英[1,a)]　田中 昌宏[2,b)]　川島 英之[1,c)]

概要：データベースはクラウドに置いて運用するのが現在の主流になっているが, クラウド事業者等の管理者権限を有する者がハッキングを行っていないことを保証するのは難しい. また, 保証手法として TEE を用いた DB, EnclaveDB が提案されているが, 他の近代プロトコルが TEE に適用できるのかを実証した研究は存在しない. 本研究では enclave 内部で動作する Silo, eSilo を提案する. eSilo は機密データであるレコード, プロシージャを Intel SGX で生成した enclave 内部で保持, 処理することによって, 機密性と完全性を確保するというものである. eSilo の実装では, SGX の仕様上, 通常の C/C++ライブラリを使用できないため, SGX SDK に同梱されている独自ライブラリへの差し替え及び CCbench の Silo プログラムを基に再構成を行った. 実証実験では 224 スレッド時で 2276 万 tps を記録した. 性能比較のために enclave を利用しない Silo プログラムも実装したが, eSilo と比較して 23%減の 1752 万 tps を記録した.

1. 動機

クラウドの利用シーンは増加の一途を辿っている. クラウド使用時には各組織の重要なデータベースを当然クラウドに置くことになるが, ここでは常に 2 つの疑念が付きまとう. それは悪意のある攻撃者がそれらの組織のデータベースをハッキングしていないか, クラウド事業者自身がデータベースをハッキングしないか, という点である. 様々な保護はされているが抜け穴は常に存在しており, この疑念を払拭することは容易ではない.

この疑念に対する 1 つのアプローチとして TEE がある. これはコンピュータのリソースを信頼できる領域とできない領域に二分化し, 機密データを信頼できる領域内で保持, 処理するというものである. TEE の 1 つである Intel SGX では処理内容を enclave と呼ばれる隔離実行環境で暗号化して処理することにより, 上記のような攻撃を防御する. これを利用すればデータベース自体を完全に保護することが可能になる. Microsoft 社が提供する SQL Server を enclave で動作可能にした EnclaveDB なるシステムも既に発表されている [1].

2. 研究課題

EnclaveDB で使われているシステムは Hekaton[2] であり, これ以外に enclave を利用した DBMS は発表されていない. それに対して近年, Silo[3] を始め多数のトランザクション処理技法が提案されている. これらの手法を enclave に適用できるのか否か実証した研究は存在しない.

1 慶應義塾大学 環境情報学部
2 慶應義塾大学大学院 政策・メディア研究科
a) t20703mf@sfc.keio.ac.jp
b) masa16.tanaka@gmail.com
c) river@sfc.keio.ac.jp

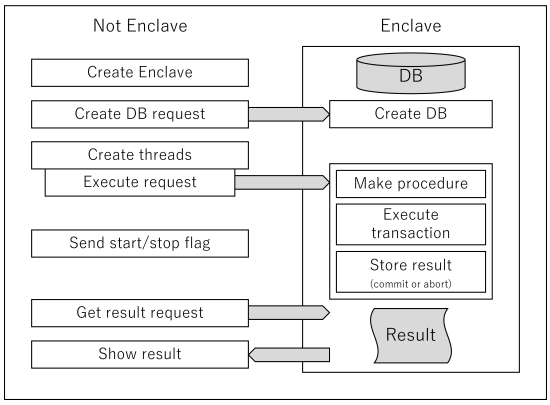

図 1　eSilo の概要設計

3. 提案：eSilo

本研究では近代的プロトコルの基礎をなす Silo を enclave 化する. Intel SGX を用いて enclave を生成し, その内部で機密データを取り扱う eSilo を提案する.

具体的には, CCbench[4], [5] の Silo 設計を基に, Intel 社が提供している SGX SDK[6] の SampleEnclave プログラムに Silo を移植した (以下, eSilo). CCbench の Silo ではメインメモリに機密データであるレコード及びプロシージャを置いているため, 攻撃者に対する防御措置を取れない. そのため, 本設計では機密データを enclave 内部に配置する.

図 1 は eSilo の概要設計を図示したものである. スレッド

の生成は OS の機能を必要とする関係上, enclave 内部で行うことができないため enclave 外部でスレッドの生成を行い, 各スレッドが enclave 内部に侵入しトランザクションの実行処理及びデータに対するアクセスを行う設計にした.

また, 性能を比較するために enclave を使用しない Silo の実装も行った (以下, non-eSilo). これらの実装プログラムは GitHub で公開済みである [7], [8].

4. 結果

性能評価に用いたマシンについて, CPU は Intel Xeon Platinum 8276 2.20 GHz 4 基, DRAM は 512GiB, OS は Ubuntu 20.04.4 LTS で構成されている. トランザクション中命令数は 10 件 (Read/Write), ワークロードは YCSB-A ライク, レコード数は 100 万, Skew は 0(一様分布), 計測時間は 3 秒である. enclave 動作は EnclaveDB[1] に従いシミュレーションモードで実行, enclave で利用できるヒープ領域は 1GiB である.

これらの条件を基に, スレッド数を 28 から 224 まで 28 刻みで増加させつつ, eSilo 及び比較用の non-eSilo, CCbench の Silo に関して計測を行った. なお, CCbench の Silo は最適化のために numa やメモリアライメント, MassTree などを利用しているため, 計測条件は異なるが, 性能比較の参考値として記載している. 各スレッド数で 10 回ずつ計測を行い, それらの平均 (小数点以下四捨五入) を取った結果を図 2 に示す.

eSilo は CCbench と比較すると 224 スレッドで 30%減の 2276 万 tps となった. non-eSilo は 224 スレッドで 1752 万 tps となり, eSilo と比較して 23%ほど性能が低下した.

また, 興味深いことに enclave 外部での実装よりも性能が向上した. これは, SGX SDK で提供されている独自ライブラリ, tlibcxx の vector ライブラリを enclave 内部で使用していることが要因と考えられる. 具体的には, 従来の C/C++の標準ライブラリの vector と比較して tlibcxx の vector ライブラリの方がイテレータを用いた配列検索が約 3 倍ほど高速になっており, Silo の実装ではイテレータを用いた配列検索を採用しているということである. これに関する追加調査プログラムは GitHub で公開済みである [9].

5. 結論

Silo を enclave 内部で動作させることが可能なことがわかった. また CCbench の Silo と比較すると, 224 スレッドで 30%程度の性能低下にとどまることがわかった.

システムの設計と実装における問題は, SGX が OS を信頼しないという設計上, C/C++のライブラリの大半が利用できない制約があり従来の設計をそのまま流用することが難しいという点である. これに関しては SGX SDK で提供されている独自ライブラリを差し替える形で利用することにより, それらを解決した.

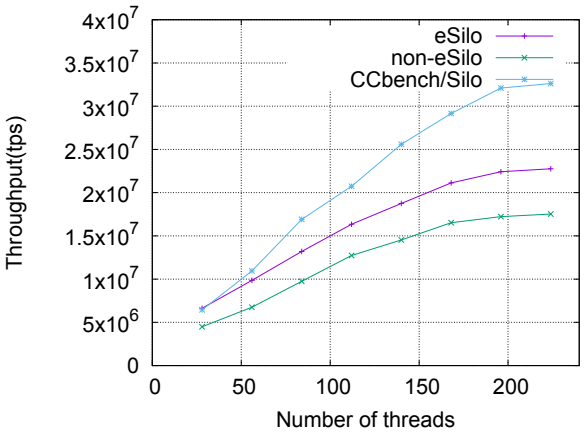

図 2 eSilo と Silo の性能比較

今後の課題は enclave 内部でも動作する並列ロギングの設計と実装である.

謝辞 本研究の成果は, 国立研究開発法人新エネルギー・産業技術総合開発機構 (NEDO) の委託業務 (JPNP16007) と科研費 (22H03596) の結果得られたものである.

参考文献

[1] Priebe, C., Vaswani, K. and Costa, M.: EnclaveDB: A secure database using SGX, *2018 IEEE Symposium on Security and Privacy (SP)*, IEEE, pp. 264–278 (2018).

[2] Diaconu, C., Freedman, C., Ismert, E., Larson, P.-A., Mittal, P., Stonecipher, R., Verma, N. and Zwilling, M.: Hekaton: SQL server's memory-optimized OLTP engine, *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pp. 1243–1254 (2013).

[3] Tu, S., Zheng, W., Kohler, E., Liskov, B. and Madden, S.: Speedy transactions in multicore in-memory databases, *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, pp. 18–32 (2013).

[4] Tanabe, T., Hoshino, T., Kawashima, H. and Tatebe, O.: An analysis of concurrency control protocols for in-memory databases with CCBench, *Proceedings of the VLDB Endowment*, Vol. 13, No. 13, pp. 3531–3544 (2020).

[5] 田中昌宏, 川島英之: エポック同期法による Silo の応答高速化, 情報処理学会論文誌データベース (TOD), Vol. 15, No. 3, pp. 63–74 (2022).

[6] Intel: sgx_linux_x64_sdk, `https://download.01.org/intel-sgx/latest/linux-latest/distro/ubuntu20.04-server/sgx_linux_x64_sdk_2.17.100.3.bin`.

[7] Fukuyama, M.: Noxy3301/enclaveSilo, `https://github.com/Noxy3301/enclaveSilo`.

[8] Fukuyama, M.: Noxy3301/silo_minimum: Program for comparison of enclaveSilo, `https://github.com/Noxy3301/silo_minimum`.

[9] Fukuyama, M.: Noxy3301/tlibcxx_vector_test, `https://github.com/Noxy3301/tlibcxx_vector_test`.