SGX EnclaveとVBS Enclaveの性能比較

西平 侑磨[1]　髙山 献[1]　宮澤 慎一[1]

1. はじめに

機密データは，保存時，通信時，処理時のいずれの状態においても保護する必要がある．保存時や通信時のデータは暗号化によって保護できるが，処理時には，演算のためにデータが復号され，メモリに展開される．OS などの特権ソフトウェアは他のプロセスのメモリにもアクセスできるため，これらが侵害されると，処理時の機密データが漏えいや改ざんの危険にさらされる．この課題に対し，通常の実行環境からコードやデータを隔離して保護する Trusted Execution Environment（TEE）が用いられている．

TEE のうち，アプリケーションの一部を Enclave として隔離するものを Enclave 型 TEE と呼び，その代表的な技術として Intel Software Guard Extensions（Intel SGX）がある．しかし，Intel は 2026 年 10 月，開発コード名 Diamond Rapids を最後に Intel SGX の搭載を終了すると発表した [1]．既存製品のサポートは 2030 年代初頭まで継続されるが，Intel SGX 向けアプリケーションは，将来的に移行先を検討する必要がある．Intel は VM 型 TEE である Intel TDX への移行を推奨しているが，TCB を小さく保つ Enclave 型の設計を維持する場合は，他の Enclave 型 TEE が移行先の候補となる．

本研究では，他の Enclave 型 TEE として Microsoft Windows の Virtualization-Based Security Enclave（VBS Enclave）[2] に注目する．Intel SGX が提供する Enclave（SGX Enclave）が専用の CPU 機構によって隔離を実現するのに対し，VBS Enclave は Intel VT-x や AMD-V を基盤とするハイパーバイザーを応用することによって隔離を実現する．このように隔離の実現方式が異なるため，Enclave の呼び出しやメモリ保護に伴うオーバーヘッドには差が生じると考えられる．しかし，両者を同一環境で比較した報告は少なく，性能差とその要因は十分に明らかになっていない．

[1] セコム株式会社 IS 研究所，
〒 181-8528 東京都三鷹市下連雀 8-10-16 セコム SC センター

そこで本研究では，SGX Enclave と VBS Enclave の性能を比較し，性能差とその要因を明らかにする．

2. 評価方法

性能評価では，Enclave 内部で完結する処理と，Enclave 境界をまたぐ処理の 2 つを比較する．

前者の評価として，PolyBench/C[3] の同一カーネルを Windows 上の REE，SGX Enclave，VBS Enclave で実行し，計算処理の実行時間を測定する．

後者の評価として，Enclave と Host Application の間の関数呼び出し時間を測定し，Enclave 境界をまたぐ処理のオーバーヘッドを比較する．

3. おわりに

ポスター発表では，SGX Enclave と VBS Enclave の性能を評価し，性能差が生じる要因について考察する．

謝辞　本研究は，JST 経済安全保障重要技術育成プログラム【JPMJKP24U4】の支援を受けたものです．

参考文献

[1] Intel: Transition Paths for Intel SGX Applications to Intel TDX, https://cc-enabling.trustedservices.intel.com/intel-sgx-transition-path/01/introduction/ (2026). Accessed: 2026-10-21.
[2] Microsoft: Secure Enclaves (Trusted Execution) and Virtualization-Based Security (VBS) Enclaves, Microsoft Learn, https://learn.microsoft.com/ja-jp/windows/win32/trusted-execution/vbs-enclaves. Accessed: 2026-10-21.
[3] Reisinger, M.: PolyBenchC-4.2.1: PolyBench/C benchmark suite (version 4.2.1 beta), https://github.com/MatthiasJReisinger/PolyBenchC-4.2.1 (2016). GitHub repository. Accessed: 2026-10-22.