リアルタイム処理との共存を目的としたエッジTEE向けセキュアモニタのプリエンプティブ化

藤井 大悟[1]　松原 克弥[1]

1. はじめに

近年，人工知能 (AI) と Internet of Things (IoT) を融合した AIoT の発展により，エッジ環境で実行される処理が多様化している．特に，物理世界を認識・制御する Physical AI や，AI エージェントが IoT 機器を自律的に制御する Agentic IoT[1] など，新たな AIoT システムの研究が進められており，今後さらなる発展が期待される．こうしたシステムでは，AI 推論や計画などの計算集約的な処理と，機器制御や周期通信などのリアルタイム処理が組み合わされるため，限られた計算資源上で両者を効率的に共存させることが重要となる．

AIoT システムの普及にともない，AI モデルやデータの機密性保護が重要な課題となっている．例えば，開発コストの高い AI モデルを管理下にない計算機へ配備する場合には知的財産の保護が課題となり [2]，IoT 機器からエッジサーバへ AI 推論をオフロードする場合にはセンサデータの機密性が問題となる [3]．こうした課題に対し，ハードウェアの隔離機構と高特権のセキュアモニタによってプログラムやデータを保護する Trusted Execution Environment (TEE) の活用が進められている．

一方，TEE を用いた機密計算では，セキュアモニタの実行がリアルタイムタスクの実行を妨げ，デッドライン逸脱を引き起こす可能性がある．セキュアモニタはホスト OS より高い特権レベルで動作するため，OS のスケジューリングによってその実行を中断することができない．実際，StarFive 製 RISC-V ボードの VisionFive 2 を用いて，セキュアモニタを実行する CPU コアに割り当てられた 5ms 周期 1ms 実行のリアルタイムタスクの応答性能を計測したところ，セキュアモニタの処理の影響を確認できた (図 1 参照)．通常時は約 1 ms だったリアルタイムタスクの応答時間が，Keystone[4] のセキュアモニタによるエンクレーブ初期化処理の開始後に急増し，最大約 2.5 秒に達した．また，初期化処理中に実行を妨げられたジョブが蓄積し，その処理待ちによって TEE の処理終了後も後続ジョブの応答時間が悪化した．

本研究では，エッジ TEE 基盤の一つである RISC-V を対象として，機密計算とリアルタイム処理の効率的な共存

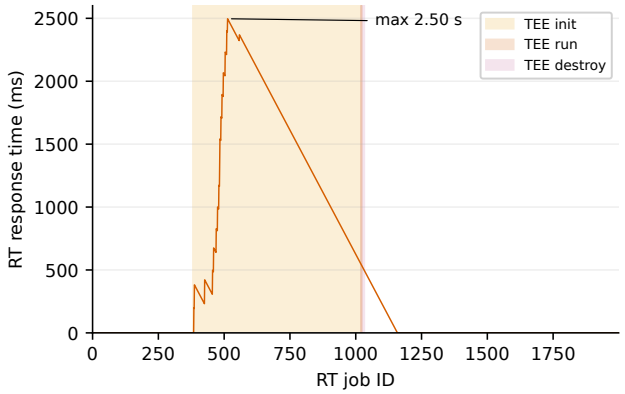


図 1　セキュアモニタ実行 CPU コアにおける共存リアルタイムタスクの応答性能

に向けた，セキュアモニタの安全なプリエンプティブ化の実現を目指す．プリエンプトできない特権モードで実行されるセキュアモニタの処理内容を精査して，特権が必要ない処理を特定し，それらを非特権モードで実行できるように分離することで，セキュアモニタ処理内でのプリエンプション機会を増加させる．本稿では，セキュアモニタのプリエンプティブ化実現に必要な設計要件を定義し，RISC-V の特権機構を対象としたプロトタイプ実装について述べる．

2. プリエンプティブ・セキュアモニタの設計要件

セキュアモニタのプリエンプションでは，TEE の安全性を維持しながら，リアルタイムタスクの時間制約を満たすことが課題となる．セキュアモニタはメモリ保護設定や実行状態を管理するため，任意の地点で中断すると保護状態の不整合や機密情報の漏洩を招くおそれがある．一方，安全な中断地点まで実行を継続すると中断遅延が増大するため，安全性の維持と中断遅延の抑制を両立する必要がある．本研究では，以下の 4 つの設計要件を定める．

R1：TEE の安全性の維持　中断・再開時のメモリ保護設定や実行状態の整合性を維持し，ホスト OS による機密情報の参照や保存状態の改ざんを防ぐことで，TEE の機密性・完全性を保証する．

R2：中断遅延の制限　リアルタイムタスクの時間制約を満たすため，中断要求からホスト OS への制御返却までの遅延を制限する必要がある．特に，TEE の安全

[1] 公立はこだて未来大学
Future Uniersity Hakodate

© 1959 Information Processing Society of Japan

性を維持しながら，不可中断区間を短縮することが求められる．

R3：TCB の最小化　中断制御に伴う TCB の増加を最小限に抑え，TEE の安全性の検証を容易にする．

R4：セキュアモニタへの非依存性　TEE では，性能向上やセキュリティ要件への対応に向けて，メモリアクセス権限の動的変更 [5] やシステムコールの仲介 [6]，外部インターフェースからの攻撃への対処 [7] など，多様な保護機構が研究されている．こうした保護機構の違いによってセキュアモニタの内部処理も異なるため，実装ごとに中断地点を設計・検証する負担を抑えた，汎用的な中断・再開機構が求められる．

3. アプローチ: セキュアモニタのマイクロカーネル化

本研究では，マイクロカーネルの設計思想にならい，セキュアモニタの内部処理を必要な特権レベルに応じて分離することで，安全に中断可能な実行区間を拡大するアプローチを検討する．図 2 に本アプローチの概要を示す．セキュアモニタは，メモリ保護設定の変更や内部状態の管理，ハッシュ計算など多様な処理から構成されるが，そのすべてが RISC-V の最高特権レベルである M-mode を必要とするわけではない．そこで，M-mode の特権を必要としない処理を非特権の U-mode へ分離する一方，ハードウェア保護設定の管理や中断制御などの特権機能を M-mode の管理機構に集約する．

この構成では，U-mode の処理がハードウェア保護設定を直接変更できないため，保護状態の整合性を考慮すべき操作を M-mode の管理機構に限定できる．これらの操作を安全な状態で完了させるとともに，中断中の実行状態を保護することで，TEE の安全性を維持した中断・再開を目指す (R1)．また，ハッシュ計算などの長時間処理を中断可能な U-mode へ分離することで，これらの処理の完了を待たずに中断できるようにし，M-mode に残る不可中断区間の実行時間を制限する (R2)．さらに，セキュアモニタの特権操作を M-mode の管理機構が仲介し，中断・再開をセキュアモニタの外部から制御することで，内部処理ごとに中断機構を実装する必要性を減らし，特定のセキュアモニタへの依存を抑える (R4)．一方，管理機構の追加は TCB の増加やハードウェア資源の消費につながるため，中断制御に必要な機能に限定した軽量な構成を目指す (R3)．

本アプローチの実現可能性を確認するため，RISC-V の M-mode 仮想化を提供する Miralis[8] を利用した初期プロトタイプを実装した．本プロトタイプでは，Keystone のセキュアモニタを U-mode で実行し，タイマ割込みを契機に M-mode の管理機構が実行状態を保存して，ホスト OS へ CPU の制御を返却する．その後，保存した実行状態を復元することで，中断地点からセキュアモニタの処理を再開する．

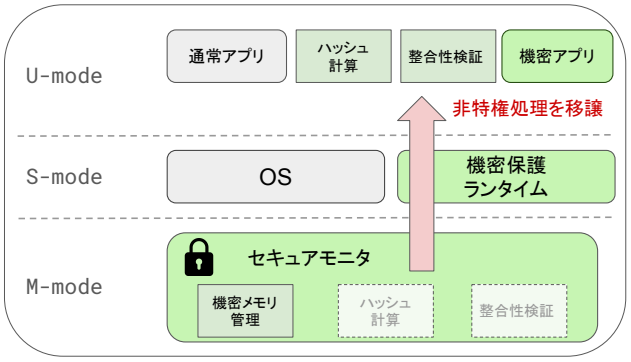


図 2　セキュアモニタの特権分離による中断制御の概要

4. おわりに

本研究では，機密計算とリアルタイム処理の共存に向け，セキュアモニタの安全かつ低遅延なプリエンプションを実現するアプローチを提案した．マイクロカーネルの設計思想にならい，セキュアモニタの内部処理を特権レベルに応じて分離し，中断可能な実行区間を拡大する方式を検討した．また，Miralis 実装を活用して，セキュアモニタの中断・再開を実装した．今後，中断・再開時の TEE の安全性を保証する仕組みと軽量な管理機構を実現し，最悪プリエンプション遅延とセキュアモニタ実行性能のトレードオフ，RISC-V のセキュアモニタへの適用可能性を評価したい．

参考文献

[1] Yuyang Du et al. LLMind 2.0: Distributed IoT automation with natural language M2M communication and lightweight LLM agents. arXiv:2508.13920, 2025.
[2] Yujin Huang et al. THEMIS: Towards practical intellectual property protection for post-deployment on-device deep learning models. In *USENIX Security '25*, pages 7311–7330, 2025.
[3] Siping Shi et al. MEPS: Privacy-preserving edge-cloud video foundation model inference with privacy protectability. In *USENIX Security '26*, pages 7055–7074, 2026.
[4] Dayeol Lee et al. Keystone: An open framework for architecting trusted execution environments. In *EuroSys '20*, pages 38:1–38:16, 2020.
[5] Jason Zhijingcheng Yu et al. Elasticlave: An efficient memory model for enclaves. In *USENIX Security '22*, pages 4111–4128, 2022.
[6] Alexander Van't Hof and Jason Nieh. BlackBox: A container security monitor for protecting containers on untrusted operating systems. In *OSDI '22*, pages 683–700, 2022.
[7] Andrin Bertschi et al. Devlore: Extending arm cca to integrated devices: A journey beyond memory to interrupt isolation. arXiv:2408.05835, 2026.
[8] Charly Castes et al. The design and implementation of a virtual firmware monitor. In *SOSP '25*, page 85–100, 2025.

© 1959 Information Processing Society of Japan