一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
CPSY2025-52,DC2025-124(2026-03)

# ソフトウェアへの依存を最小化するTEE割り込み防御手法

齊木　昭大† 　浦浪　英俊† 　内山　一秀††,†††,†††† 　五島正裕††,††††,††† 　木村啓二††,†

† 早稲田大学 基幹理工学研究科 情報理工・情報通信専攻
†† セキュア コンピュータ システム 研究開発センター, ROIS-DS
††† 総合研究大学院大学
†††† 国立情報学研究所
E-mail: †{saiki,reinsirk}@kasahara.cs.waseda.ac.jp, ††{uchiyama,goshima}@nii.ac.jp, †††keiji@waseda.jp

**あらまし**　TEE は，TEE Security Module (TSM) などと呼称される，システムにおいて特権を持つソフトウェア (特権 SW) や，セキュアプロセッサのような専用のコプロセッサが，防御の中枢を担っている．特権 SW やセキュアプロセッサへの依存は，ソフトウェア TCB の肥大化へと繋がり，アタックサーフェスの拡大とともに，理想的なセキュリティモデルが破綻する危険性を高める．そのため，TriniTEE プロジェクトが提唱する Universal TEE Architecture (UTA) では，特権 SW・セキュアプロセッサを排除することを要件としている．UTA は，TEE ハードウェアと暗号技術の活用により，従来ソフトウェア TCB が担っていた役割の，ハイパーバイザ等の非信頼ソフトウェアへの安全な委譲を可能にする．しかしながら，CPU キャッシュや分岐予測，電力消費などのサイドチャネルを利用する攻撃は依然として脅威となる．特に，Single-Stepping による時間分解能の拡大と特定の命令を標的とした観測を基盤とするサイドチャネル攻撃は非常に強力であり，Single-Stepping への対策がサイドチャネル攻撃対策において重要となるが，その一方で，主に割り込みの悪用によって実現される Single-Stepping は，TEE ハードウェアのみで対策を講じることは困難である．本稿では，UTA の下で，特権 SW・セキュアプロセッサを使用せず，最小のソフトウェア依存で，割り込み悪用を防御する手法を提案する．

**キーワード**　TEE, Side-Channel Attack, Single-Stepping

# Defense Mechanism for TEE Interrupts to Minimize Software Dependency

Akihiro SAIKI†, Hidetoshi URANAMI†, Kazuhide UCHIYAMA††,†††,††††,
Masahiro GOSHIMA††,††††,†††, and Keiji KIMURA††,†

† Department of Computer Science and Engeneering, Waseda University
†† Center for R&D on Secure Computer Systems, ROIS-DS
††† The Graduate University for Advanced Studies
†††† National Institute of Informatics
E-mail: †{saiki,reinsirk}@kasahara.cs.waseda.ac.jp, ††{uchiyama,goshima}@nii.ac.jp, †††keiji@waseda.jp

## 1　はじめに

サーバーやクライアントマシン，IoT・組み込みデバイスなど，多様な応用分野で TEE (Trusted Execution Environment) の活用が拡大している．CPU アーキテクチャやベンダー，対象プラットフォームによって固有の TEE が存在し，それぞれが独自の脅威モデルに基づいて，機密性・完全性を確保するための防御メカニズムを提供している．

初期の TEE は，ARM TrustZone [1] のようなシンプルなメモリ領域分離や，Intel SGX [2], RISC-V Keystone [3] のようなプロセス・アプリケーション単体を Enclave として防御する形態 (Enclave 型 TEE) であった．現在では，Intel TDX [4], AMD SEV-SNP [5], RISC-V CoVE [6], ARM CCA [7] のように，仮想マシン (VM) 全体を防御対象とする TEE (VM 型 TEE, *Confidential VMs: CVMs*) が台頭している．

TEE における機密性・完全性は，基盤となるハードウェア及び ソフトウェア，すなわち Trusted Computing Base (TCB) の信頼性を前提として保証される．理想的には，TCB はシンプ

ルかつ限定的な規模で，形式検証等の手法によって脆弱性の排除・機能的正当性の保証が可能であることが望まれる．しかしながら，既存の TEE は，TEE Security Module (TSM) や Secure Monitor (SM) などと呼ばれる，システム上で高い特権を持つソフトウェア (特権 SW) や，セキュアプロセッサ (SP) 等の専用コプロセッサに強く依存しており，結果としてソフトウェア TCB の複雑化・肥大化を招いている．

内閣府の経済安全保障重要技術育成プログラム (K-Program) の研究開発構想 (個別研究型)「セキュアなデータ流通を支える暗号関連技術（高機能暗号)」に採択された，研究開発課題「ハードウェア・ソフトウェア・理論の連携によるユニバーサル TEE アーキテクチャの実現」，通称 **TriniTEE プロジェクト** では，**Universal TEE Architectute (UTA)** を提唱している [8], [9]. UTA のメモリ防御機構は，特権 SW・SP を排除し，TCB を最小限に保つことを要件の 1 つとしている．TEE ハードウェアと暗号技術の組み合わせにより，従来特権 SW・SP が果たしていた役割をホスト OS やハイパーバイザ等の非信頼ソフトウェアへ安全に委譲可能とすることで，特権 SW・SP を用いないメモリ防御を実現する [10], [11].

一方で，TEE に対するサイドチャネル攻撃は依然として脅威である．キャッシュや分岐予測，電力消費等のサイドチャネルを利用した様々な攻撃手法が，各主要 TEE を標的に考案されている [12]〜[18]．また，TEE に対するサイドチャネル攻撃では，時間分解能を拡大し，特定の命令を標的とした観測を可能にする手法として，Single-Stepping が活用される．Intel SGX, AMD SEV では，高精度な Single-Stepping フレームワークが開発されており，これらをベースとする多数の攻撃が提案された [19], [20]．Single-Stepping は，タイマ割り込みや Page Fault 等の例外を攻撃者が意図的に発生させることで実現されるが，リソース管理者である非信頼ソフトウェアからこれらの操作を完全に排除することは不可能である．そのため，様々な緩和策が提案されているものの，根本的な対策に至っていないのが現状である [18], [19], [21]〜[23]．このような，根本的な対策が困難な攻撃に対しては，TEE ハードウェア上での対策だけでなく，ソフトウェアによる柔軟な対策の導入が必要である．

本稿では，UTA を対象とし，NMI を除く全ての割り込み・例外を動作中の特権レベル内で一時的にハンドリングする，Inter Privilege Layer (IPL) の導入を提案する．TEE において Single-Stepping の手段が確立される原因の一つは，割り込み・例外が透過的であり，上位特権に存在する非信頼ソフトウェアの介入を TEE が検知できない点にある．IPL により，TEE 内の VM・アプリケーションが割り込み・例外の発生を捕捉し，Single-Stepping 緩和策を含めた任意の処理を，非信頼ソフトウェアとの遷移間に差し込むことが可能となる．IPL は特権 SW・SP を使用せず，TEE ハードウェアの拡張のみで実現可能であるため，UTA の要件を満たす．また，IPL を利用した Single-Stepping の防御手法の全体像についても述べる．

本稿の構成は以下のとおりである．まず 2 節にて，UTA の仕様や，UTA がベースとする RISC-V の関連仕様を述べる．3 節では，各種 TEE における Single-Stepping の状況について考察し，新規の TEE アーキテクチャにおける Single-Stepping 対策に必要な要件を整理する．4 節で，本稿で提案する Inter Privilege Layer (IPL) について述べる．5 節で，提案する IPL の利用を含めた，Single-Stepping の防御手法について述べる．6 節で，本稿を総括する．

## 2 背　　景

### 2.1 RISC-V における割り込み・例外

本節では，UTA がベースとする RISC-V における，割り込み・例外の仕様について概説する．本節の内容は，*RISC-V Privileged Architecture* [24] 及び その他拡張仕様に基づく．

RISC-V における割り込み・例外は，特定の特権レベル [24] をターゲットとして発生する．発生した割り込み・例外は，デフォルトでは最上位特権である Machine Mode (M-Mode) でハンドルされるが，委譲 (delegation) により，ハンドルを行うレベルをターゲット特権レベルまで下げることが可能である．例えば，User Mode (U-Mode) アプリケーションのページフォールトは，Supervisor Mode (S-Mode) の OS で処理を行うため，M-Mode ではハンドルせず，S-Mode へ委譲する．

RISC-V では，Hypervisor 拡張 [24] における割り込みや Message Signaled Interrupt (MSI) をサポートするため，新たに Advanced Interrupts Architecture (AIA) [25] を定義している．AIA により，Hypervisor のゲストへ直接割り込みを配信することが可能となる．

### 2.2 Universal TEE Architecture (UTA)

本節では，Universal TEE Architecture (UTA) の仕様について概説する．本稿の提案は，以下に述べる UTA の仕様に従うものとする．

UTA は，単一・柔軟・形式的にセキュアな TEE アーキテクチャであり，特定の TEE の仕様や，ISA，OS に依存しない，抽象度の高い方式として定義される [9], [10]．防御境界を挟んだソフトウェア間の関係性を抽象化するため，UTA では MgR / MgE (manager / managee) という表現を用いる [10]．TEE の文脈においては，MgR が OS・ハイパーバイザ等のリソース管理者，MgE が管理される側の Enclave・CVM に該当する．以下，本稿でも同様の表記を用いる．

#### 2.2.1 FKT / RTT

内山等 [10] は，UTA におけるメモリ防御機構として，Forward Key Table (FKT) と Reverse Tag Table (RTT) を用いたメモリアクセス制御手法を提案している．FKT/RTT 方式は，Remapping, Aliasing のようなメモリマッピングに対する攻撃からの防御を行いつつ，共有タグによる MgE 間のページ共有を可能にする．また，暗号技術を用いてテーブル管理を非信頼 MgR に委譲することで，特権 SW・SP に依存せずに木構造などを用いた複雑なテーブルの管理が可能となる．

#### 2.2.2 Generalized Nested Virtualization

内山等 [10] は，Nested Virtualization を TEE ハードウェアによってサポートする仕様として，Generalized Nested Virtualization を提案している．$P_0, P_1, \cdots, P_M$ のように物理的な特権レベルを定義し，ここに仮想的な特権レベルとして RISC-V 本来

の特権レベルである M, HS, VS, VU-Mode [24] を任意に割り当て可能とすることで，複数段の Nested Virtualization をハードウェアでサポートする．

**2.2.3** MgE のコンテキスト

UTA では，TEE ハードウェアが MgE をアイデンティファイする [10]．つまり，TEE ハードウェアが MgE のアイデンティティとメモリ・コンテキストを密結合させる．コンテキストは，認証付き暗号 (AE) の適用 または ハードウェアによるコンテキストスイッチを行うことで，機密性・整合性を保証する．また，MgE に付与する ID は，詐称を防ぐ機構を TEE ハードウェアが提供することで一意性を保証可能にし，非信頼 MgR が管理を行う．

## 3 Single-Stepping

TEE に対するサイドチャネル攻撃では，高頻度でプリエンプションを引き起こすことにより，細粒度でサイドチャネル観測を得る手法が利用される．Single-Stepping は，命令単位でのサイドチャネル観測を行い，最大の時間分解能を達成する手法である．そのため，Single-Stepping は非常に強力な攻撃手段であり，多くの TEE を標的とした攻撃がこれを基盤としている．

### 3.1 各 TEE における Single-Stepping の現状

Van Bulck 等は，APIC タイマを利用して SGX Enclave に対して連続的に割り込みをかけた上で，ページテーブルエントリの監視により Single-Step を確実にするフレームワーク，SGX-Step を発表した [19]．SGX-Step は，Intel SGX 上で決定論的な Single-Stepping を実現した初のフレームワークであり，その後の割り込み駆動のサイドチャネル攻撃に広く活用された．これを受けて，Constable 等は AEX-Notify [21] を提案し，SGX-Step による決定論的 Single-Stepping への対策として展開した．AEX-Notify は, Enclave が割り込み・例外による AEX (Asynchronous Enclave Exit) から復帰する際に，信頼されたハンドラの実行を可能にする．Constable 等はハンドラで攻撃のタイムウィンドウの拡大を阻止する緩和策をとり，Single-Step の成功率を大幅に低下させた．しかしながら，決定論的 Single-Stepping のみの対策では不十分であり，確率論的 Single-Stepping による割り込みカウント攻撃が，依然として可能であると，Dutly 等 [22] により示されている．

Intel TDX は，AEX-Notify とは異なる対策を当初から導入している．TDX における TSM に相当する TDX Module [26] 内で，ヒューリスティックに Single-Stepping を検知し，ランダムな数だけ TD の命令を進行する緩和策を実行する．初期の TDX における検知ヒューリスティックでは，タイムスタンプカウンタ (*tsc*) 及び プログラムカウンタの差分を使用する．Wilke 等は，クロック周波数を低下させることで tsc による検知をバイパス可能であり，検知ヒューリスティックが不十分であることを示した [23]．また，緩和策におけるランダム命令進行は，TDX Module が TD を Single-Step することで実現されているため，進行した命令数をカウント可能である点も言及されている．その後，不十分な検知ヒューリスティックについては，パフォーマンスカウンタを使用する変更により改善された．しかし，ランダム進行する命令数を決定する RNG が脆弱であり，実行命令数を決定論的に推測可能であることが，Rauscher らにより指摘されている [18]．

Wilke 等は，TDX においても AEX-Notify の根幹の考え方が有効である可能性に言及している [23]．TDX のような VM ベースの設計の場合，VM–ハイパーバイザ間の遷移は TDX Module のような特権 SW を経由するため，ハイパーバイザへ遷移前に対策がとれる [4]．一方で，特権 SW への対策の組み込みは TCB の肥大化を招き，アタックサーフェスの拡大につながるという懸念がある [22]．

また，AMD SEV においても APIC タイマ 及び Page Fault 等を用いた Single-Step 手法が発表されている [20], [27]．SEV は，TDX Module のような中間ソフトウェア層を持たないため，TDX のようなアプローチを取ることは難しい．パフォーマンスカウンタの無効化 [27] や，カウンタ値の無相関化 [28] によって Single-Step の信頼性が低下する可能性は提示されているが，依然として攻撃は可能である．

### 3.2 原因と対策の考察

Single-Stepping を始め，割り込み・例外を悪用したプリエンプションによる攻撃は，これらが TEE 内の CVM や Enclave アプリケーションに対して透過的であることが根本原因の 1 つと考えられる．しかし，一般的な TEE のセキュリティモデルでは，リソース管理者である OS やハイパーバイザを TCB から排除する．そのため，割り込み・例外によるプリエンプションは計算リソースの割り当てやスケジューリングに必要であり，禁止することは不可能である．TEE が計算リソースを占有可能で，動的な割り当てやスケジューリングを一切行わない場合は，割り込みを禁止する余地があると考えられるが，これは一般的ではない．以上の状況を鑑みると，AEX-Notify における，Enclave が割り込み・例外の発生を捕捉するという方針は合理的である．ただし，AEX-Notify によって割り込み・例外の発生を捕捉し，緩和策を含むハンドラを実行できるのは，`ERESUME` によって Enclave へ復帰するプロセス中である．緩和策はターゲットアーキテクチャや標的とする攻撃手法に依存するため，復帰時の緩和策のみで防げない漏洩が潜在する可能性がある．

ヒューリスティックな検知は，高頻度な割り込みを必要とする確率論的 Single-Stepping に有効である可能性が指摘されている [22]．しかし，TDX において検知がバイパスされたケースから，使用する検知手法には検討の余地がある．Single-Step に限らず，命令を進行させない Zero-Step，複数命令の進行を行う Multi-Step を利用した攻撃も存在する．また，多数の割り込みが発生するようなワークロードでは，良性の割り込みを誤検知してしまう可能性もある．したがって，ワークロードを考慮した上で，単位時間の進行サイクル・命令数や割り込みの頻度，及びそれらの変動などから，総合的な判断を下す必要がある．

### 3.3 対策のための機能要件

ここまでの考察から，我々は，TEE ハードウェアが以下の機能を MgE に提供する必要があると考える．

（1） 割り込み・例外発生の捕捉

（2） 割り込み・例外による中断前後で任意の緩和策実行

（3） 検知に有効なメトリクス 及び コンフィギュラブルな検知機構

SGX-Step 及び AEX-Notify の前例から，要件 (1), (2) は，現状の対策における大前提と考えられる．また，確率論的 Single-Stepping や未知の Single-Step 手法への対応を考慮すると，ヒューリスティックな検知機構も併せて持つ必要がある．しかし，前述のようにその条件設定には検討の余地があるため，検知に有効なメトリクスを提供する，もしくは任意に条件設定可能な検知機構を提供する，という形態が望ましいと考える．

以上を踏まえ，4 節 で (1), (2) を提供するためのハードウェア仕様拡張について述べる．また，5 節 で，4 節 の提案を用いた Single-Stepping 防御手法について述べる．

## 4 Inter Privilege Layer (IPL)

本節では, 前節における要件 (1), (2) を提供する, **Inter Privilege Layer (IPL)** について述べる．

MgE が割り込み・例外の発生を捕捉し，任意の処理を行うことを可能にする方法としては，全ての割り込み・例外を一時的に MgE でハンドルすることが考えられる．しかし，通常の RISC-V の特権アーキテクチャの下では，割り込み・例外をターゲットより下位の特権レベルでハンドルすることは不可能である [24], [25]．また，ユーザープロセスが自身で割り込み・例外を受け取ることは想定されていないため，U/VU-Mode における割り込みは仕様として存在しない (注1)．

そこで，特定の特権レベルに依存しない割り込み・例外のハンドリング機能として，**Inter Privilege Layer (IPL)** を提案する．

### 4.1 IPL の概要

図 1 に，RISC-V の U-Mode における IPL の使用例を示す．ここでは，OS が MgR，ユーザーアプリケーションが MgE とする．IPL は，特権レベル内部を擬似的な 2 つの SW レイヤー: *Normal Layer, Bridge Layer* として取り扱う．2 つのレイヤー間に特権アーキテクチャにおける違いはなく，どちらも同一のメモリ空間に存在する．そのため，両レイヤーは UTA において単一の MgE とみなすことができる．

**Normal Layer**

Normal Layer はデフォルトのレイヤーであり，対象の特権レベルで動作する通常の SW を指す．図 1 では，ユーザーアプリケーションのプロセスが該当する．

**Bridge Layer**

Brige Layer はある種のトラップハンドラであり，IPL が捕捉した割り込み・例外をハンドルする SW である．ただし，Bridge Layer は自身の処理の完了後，必ず割り込み・例外を本来のターゲットへ伝達する必要がある．すなわち，橋渡し (*Bridge*) を行う役目を持つ．この点から生じる懸念については，4.4 節 にて述べる．Bridge Layer は，同一メモリ空間内にさえ配置されていれば，Normal Layer の SW の一部でも，独立していてもよい．

(注1)：過去に N 拡張という User-Level の割り込み仕様が提案されていたが，現在は削除されている [29].

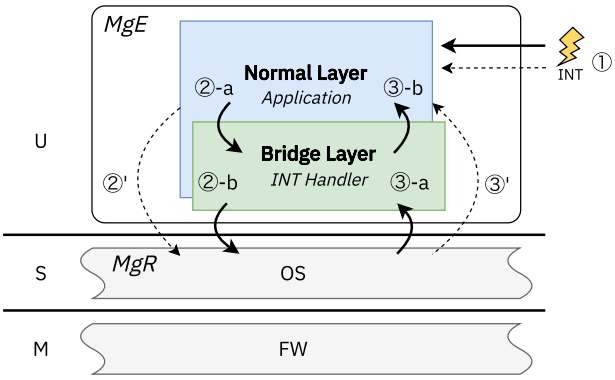


図 1 U-Mode に *Inter Privilege Layer* を適用した一例．実線矢印は IPL 有効時，破線矢印は IPL 無効時の動作フローである．

### 4.2 動作フロー

図 1 における，IPL での割り込みのハンドリング動作について説明する．ここでの前提条件として，対象 CPU コアの現在の特権レベルは U-Mode とする．また，各種割り込みはターゲットの特権レベルへ全て delegate されているものとする．

① において，S-Mode がターゲットの割り込みが発生すると，本来であれば S-Mode の OS へ処理が遷移する (②')．そして，OS のトラップハンドラでの処理が済んだ後，`sret` によって割り込み発生箇所へ戻る (③')．

IPL を使用する場合は，直接 OS のトラップハンドラには遷移せず，一旦 Bridge Layer へ遷移する (②-a)．Bridge Layer は State Saved Area (SSA) 相当の領域を持ち，まず通常のトラップハンドラのように，Normal Layer のコンテキストをここに退避する．その後，任意の処理を行った上で，本来の割り込み配信先である OS へ遷移させる (②-b)．この際，OS からは通常通りの割り込みによってトラップが発生したように見え，トラップハンドラへ遷移する．IPL が使用されていることは認知しない．

OS のトラップハンドラでの処理後，U-Mode へ復帰するが，その復帰先は Bridge Layer となる (③-a)．ここで再度任意の処理を挟んだ上で，SSA より Normal Layer のコンテキストを復元し，アプリケーションを割り込み発生箇所から再開する (③-b)．

### 4.3 ハードウェア拡張

IPL の実現には，次のハードウェアへの拡張が必要となる．

**割り込み・例外仕様**

割り込み・例外の発生時，通常のハンドリング処理を行う前に，IPL による Bridge Layer でのハンドリング処理を行えるよう，変更が必要となる．ただし，Bridge Layer での処理後には通常の割り込み・例外処理へ移行するため，特権アーキテクチャ自体への大幅な変更は生じないと考えられる．

**CSR**

表 1 に示す CSR を追加で必要とする．尚，名称は現段階での仮称である．これらの CSR は，特権レベルに依存せず共通で，特権レベルの遷移が発生した場合には値はクリアされる．また，これらの CSR の値を，TEE ハードウェアの管理するコンテキストとともに MgE のアイデンティティに紐付けることで，IPL の状態についても機密性・整合性を保つことができる．

表 1　IPL の追加 CSR

| Name | Description |
|---|---|
| iplcsr | IPL Control & Status Register.<br>IPL の有効・無効等の制御を行う.<br>IPL による割り込み・例外処理中などの状態も示す. |
| ipltvec | IPL Trap Vector.<br>Bridge Layer のエントリポイントをセットする.<br>特権アーキテクチャの *tvec 相当. |
| iplepc | IPL Exception Program Counter.<br>割り込み・例外発生直前の pc 値を保持.<br>特権アーキテクチャの *epc 相当. |
| iplcause | IPL Cause.<br>IPL でハンドルした割り込み・例外のコード.<br>特権アーキテクチャの *cause 相当. |

**命　　令**

Bridge Layer から割り込み・例外を本来のターゲットへ伝達する relay 命令が必要となる．この命令をトリガーに，通常の割り込み・例外処理のプロセスが開始される．relay は，Bridge Layer による割り込み・例外処理中にのみ有効であり，それ以外の状態では機能しない仕様とする．

### 4.4 問　題　点

IPL の導入は，事実上，MgE に対して計算リソースを明け渡す決定権を持たせることになる．したがって，MgE が悪意を持ってリソースを解放しないことが懸念される．これに対しては，以下の対応が考えられる．

（1） Watchdog Timer による強制停止

（2） 占有者にペナルティを与える運用

(1) については，IPL によって MgE が割り込み・例外をハンドルした後，実際の割り込み・例外のターゲットへ伝達するまでにデッドラインを設ける．デッドラインを超えた場合，Watchdog Timer が強制的に MgE から制御を剥奪する．(2) は，クラウドなどでリソース使用量による従量課金を行う場合を想定する．利用者が IPL の悪用によりリソースの占有を行った場合，課金単価の引き上げや追加料金を課すなどで，ペナルティを与える．

また，Bridge Layer によるハンドリング中に，重ねての割り込みを許可するかは検討の余地がある．Bridge Layer の動作を Single-Step されることによる影響は未知であるが，緩和策の動作を Single-Step で制御するプロセスを経る攻撃は存在する [22]．しかし，MgE に割り込み自体の可否の決定権を与えてしまうのは，悪用の可能性を考えれば避けるべきである．現実的には，Bridge Layer への遷移の際のハードウェアロジックによってのみ割り込みを禁止とすることができ，MgE はその解除のみが可能といった対応が妥当と考えられる．

## 5　UTA における Single-Stepping 防御

本節では，UTA における，IPL を利用を含めた Single-Stepping 防御の手段について検討する．

### 5.1 割り込み・例外関連の設定

MgR が delegation 等の設定を悪意を持って変更することで，本来 MgE に直接伝達されるはずの割り込み・例外を MgR で受け取ることができる．RISC-V 標準の特権アーキテクチャの下では，MgE がこれを捕捉できない．防御の前段階として，MgE は意図した状況下で自身が動作していることを検証する必要があるため，MgR による割り込み・例外の設定を検証するインターフェースが必要である．また，基本的に delegation 等の設定を MgE の実行中に変更する必要はないため，MgE 初期化後は不変とすることで，設定検証も初期化時の 1 度のみで済む．

### 5.2 検　　知

3 節での考察より，割り込み・例外の捕捉だけでなく，不審な割り込みの検知機構も必要である．IPL を使用することで，割り込み発生時に，Bridge Layer において検知ロジックを実行することができる．TEE ハードウェア側で以下のようなメトリクスを提供し，これらを利用してヒューリスティックに分析を行う．

- プログラムカウンタ差分
- 単位時間あたりの命令スループット
- 割り込み頻度
- Cache Miss, TLB Miss, Page Fault 回数

プログラムカウンタの差分は Single-Step の決定的な証拠となり得るが，時間あたりの命令スループットや割り込み頻度等による判断は，あくまで疑いの域を出ない．そのため，検知条件に応じて危険度を割り当て，防御の強度を危険度により決定する，といった対応も考えられる．

また，割り込みコントローラ等のハードウェア上での検知も検討する．ハードウェアで検知した際は，新規の例外によって MgE に伝達するか，後述の割り込み伝達遅延を透過的に行うことが可能である．

### 5.3 防 御 手 段

以下のような防御手段の提供を検討している．これらの手段は択一にするのではなく，選択肢として持つことで，複数段階で構える防御を可能にする．

#### 5.3.1 スクリーニング・緩和策の適用

IPL の Bridge Layer において割り込みのスクリーニングや，緩和策の適用を行う．実際のハードウェアにおいて，何らかの Single-Stepping の手段が発覚した場合，ここに緩和策を追加することとなる．

#### 5.3.2 伝達遅延・ランダム命令進行

割り込みの伝達遅延や，割り込み発生時のランダムな命令進行によるノイズの挿入を行う．Zhu らは，割り込みベース攻撃への対策として，割り込みコントローラにおいてランダムな伝達遅延を挿入する手法を提案している [30]．ランダム命令進行は，TDX において悪用された事例があるが，依然としてノイズを与える有効な手段である．複数の命令をバースト実行するなど，自ら漏洩を起こさない方式で実装する必要がある [23]．

#### 5.3.3 MgE のロックダウン

機密性を守ることに特化した対応であり，全ての割り込み・例外・リソース共有を遮断し，安全に MgE を終了させる．ロッ

クダウンは最終手段のような位置づけであり，攻撃されていることが確実であるときに行うことを想定する．

## 6 おわりに

本稿では，UTA の下で，ソフトウェア依存を最小限とする割り込み・例外の悪用防御手法 及び Single-Stepping 防御手法について述べた．今後，エミュレータ 及び ハードウェア実装において提案手法を評価・検証し，有効性を確認する．

また，本稿でベースとした UTA に限らず，既存の TEE への提案手法の展開可能性についても模索していきたい．

**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。

## 文　献

[1] B. Ngabonziza, D. Martin, A. Bailey, H. Cho, and S. Martin, “Trustzone explained: Architectural features and use cases,” 2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC), pp.445–451, 2016.

[2] V. Costan and S. Devadas, “Intel sgx explained,” Cryptology ePrint Archive, Paper 2016/086, 2016. https://eprint.iacr.org/2016/086

[3] D. Lee, D. Kohlbrenner, S. Shinde, K. Asanović, and D. Song, “Keystone: an open framework for architecting trusted execution environments,” Proceedings of the Fifteenth European Conference on Computer Systems, pp.1–16, EuroSys ’20, Association for Computing Machinery, New York, NY, USA, 2020. https://doi.org/10.1145/3342195.3387532

[4] P.C. Cheng, W. Ozga, E. Valdez, S. Ahmed, Z. Gu, H. Jamjoom, H. Franke, and J. Bottomley, “Intel TDX Demystified: A Top-Down Approach,” ACM Computing Surveys, vol.56, no.9, p.33, Oct. 2024.

[5] AMD, “AMD SEV-SNP: Strengthening VM Isolation with Integrity Protection and More,” https://www.amd.com/content/dam/amd/en/documents/epyc-business-docs/white-papers/SEV-SNP-strengthening-vm-isolation-with-integrity-protection-and-more.pdf, 2020.

[6] R. Sahita, V. Shanbhogue, A. Bresticker, A. Khare, A. Patra, S. Ortiz, D. Reid, and R. Kanwal, “Cove: Towards confidential computing on risc-v platforms,” Proceedings of the 20th ACM International Conference on Computing Frontiers, p.315–321, CF ’23, Association for Computing Machinery, New York, NY, USA, 2023. https://doi.org/10.1145/3587135.3592168

[7] ARM Limited, “ARM Confidential Compute Architecture,” https://www.arm.com/architecture/security-features/arm-confidential-compute-architecture/. Accessed on 2026-01.

[8] 石川　裕，木村啓二，河野健二，光来健一，五島正裕，塩谷亮太，須崎有康，関山太朗，高前田伸也，竹房あつ子，中條拓伯，古川潤，宮澤慎一，“ハードウェア・ソフトウェア・理論の連携によるユニバーサル tee アーキテクチャの実現に向けて—システムソフトウェアの観点から—，” Technical report，情報処理学会，May 2025.

[9] 石川　裕，木村啓二，河野健二，光来健一，五島正裕，塩谷亮太，須崎有康，関山太朗，高前田伸也，竹房あつ子，中條拓伯，古川潤，宮澤慎一，“ハードウェア・ソフトウェア・理論の連携によるユニバーサル tee アーキテクチャの実現に向けて—ハードウェアの観点から—，” Technical report，情報処理学会，June 2025.

[10] 内山一秀，五島正裕，“ユニバーサル TEE アーキテクチャのためのメモリ保護方式，” xSIG 2025 (cross-disciplinary workshop on computing Systems, Infrasctructures, and programminG)，Aug. 2025.

[11] 内山一秀，松見湧斗，五島正裕，“ページ共有可能な tee のメモリ保護機構における認証暗号によるテーブル管理の委譲，” Technical report，情報処理学会，June 2025.

[12] Y. Xu, W. Cui, and M. Peinado, “Controlled-Channel Attacks: Deterministic Side Channels for Untrusted Operating Systems,” 2015 IEEE Symposium on Security and Privacy, pp.640–656, May 2015.

[13] P. Jauernig, A.-R. Sadeghi, and E. Stapf, “Trusted Execution Environments: Properties, Applications, and Challenges,” IEEE Security & Privacy, vol.18, no.2, pp.56–60, March 2020.

[14] G. Chen, S. Chen, Y. Xiao, Y. Zhang, Z. Lin, and T.H. Lai, “Sgxpectre: Stealing Intel Secrets from SGX Enclaves Via Speculative Execution,” 2019 IEEE European Symposium on Security and Privacy (EuroS&P), pp.142–157, June 2019.

[15] A. Moghimi, G. Irazoqui, and T. Eisenbarth, “CacheZoom: How SGX Amplifies the Power of Cache Attacks,” Cryptographic Hardware and Embedded Systems – CHES 2017, eds. by W. Fischer and N. Homma, pp.69–90, Springer International Publishing, Cham, 2017.

[16] M. Li, Y. Zhang, H. Wang, K. Li, and Y. Cheng, “CIPHERLEAKS: Breaking Constant-time Cryptography on AMD SEV via the Ciphertext Side Channel,” 30th USENIX Security Symposium (USENIX Security 21), pp.717–732, 2021.

[17] L. Giner, S.R. Neela, and D. Gruss, “Cohere+Reload: Re-enabling High-Resolution Cache Attacks on AMD SEV-SNP,” Detection of Intrusions and Malware, and Vulnerability Assessment, eds. by M. Egele, V. Moonsamy, D. Gruss, and M. Carminati, pp.191–212, Springer Nature Switzerland, Cham, 2025.

[18] F. Rauscher, L. Wilke, H. Weissteiner, T. Eisenbarth, and D. Gruss, “TDXploit: Novel Techniques for Single-Stepping and Cache Attacks on Intel TDX,” 34th USENIX Security Symposium (USENIX Security 25), pp.1207–1222, 2025.

[19] J. Van Bulck, F. Piessens, and R. Strackx, “SGX-Step: A Practical Attack Framework for Precise Enclave Execution Control,” Proceedings of the 2nd Workshop on System Software for Trusted Execution, pp.1–6, SysTEX’17, Association for Computing Machinery, New York, NY, USA, Oct. 2017.

[20] L. Wilke, J. Wichelmann, A. Rabich, and T. Eisenbarth, “SEV-Step: A Single-Stepping Framework for AMD-SEV,” July 2023.

[21] S. Constable, J.V. Bulck, X. Cheng, Y. Xiao, C. Xing, I. Alexandrovich, T. Kim, F. Piessens, M. Vij, and M. Silberstein, “AEX-Notify: Thwarting Precise Single-Stepping Attacks through Interrupt Awareness for Intel SGX Enclaves,” 32nd USENIX Security Symposium (USENIX Security 23), pp.4051–4068, 2023.

[22] N. Dutly, F. Groschupp, I. Puddu, K. Kostiainen, and S. Capkun, “AEX-nstep: Probabilistic interrupt counting attacks on intel SGX,” 2026 IEEE Symposium on Security and Privacy (SP), pp.663–677, IEEE Computer Society, Los Alamitos, CA, USA, May 2026.

[23] L. Wilke, F. Sieck, and T. Eisenbarth, “TDXdown: Single-Stepping and Instruction Counting Attacks against Intel TDX,” Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, pp.79–93, ACM, Salt Lake City UT USA, Dec. 2024.

[24] A. Waterman, K. Asanović, and J. Hauser, “The risc-v instruction set manual volume ii: Privileged architecture,” https://github.com/riscv/riscv-isa-manual/releases/download/Priv-v1.12/riscv-privileged-20211203.pdf, 2021.

[25] J. Hauser, “The RISC-V Advanced Interrupt Architecture,” 2025.

[26] Intel, “TDX-Module,” https://github.com/intel/confidential-computing.tdx.tdx-module. Accessed on 2026-02.

[27] S. Gast, H. Weissteiner, R.L. Schröder, and D. Gruss, “CounterSEVeillance: Performance-Counter Attacks on AMD SEV-SNP,” Proceedings 2025 Network and Distributed System Security Symposium, pp.1–16, Internet Society, San Diego, CA, USA, 2025.

[28] H. Weissteiner, F. Rauscher, R.L. Schröder, J. Juffinger, S. Gast, J. Wichelmann, T. Eisenbarth, D. Gruss, and T. Eisenbarth, “TEEcorrelate: An Information-Preserving Defense against Performance-Counter Attacks on TEEs,” 34th USENIX Security Symposium (USENIX Security 25), pp.2481–2498, 2025.

[29] A. Waterman, “Proposed deprecation of N extension,” https://lists.riscv.org/g/tech-privileged/topic/proposed_deprecation_of_n/83320504. Accessed on 2026-02.

[30] Y. Zhu, P. Li, L. Zhao, D. Meng, and R. Hou, “ChaosINTC: A Secure Interrupt Management Mechanism against Interrupt-based Attacks on TEE,” 2023 60th ACM/IEEE Design Automation Conference (DAC), pp.1–6, July 2023.