12. Battering RAM

Ao Sakurai

2026年度セキュリティキャンプ全国大会
L1 – 暗号・CVMビルド&スクラップゼミ

本資料の目標

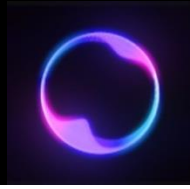
- Scalable-SGX、TDX、SEV-SNPといった現行のTEE（**Scalable TEE**）に対する物理攻撃について整理し、その内2025年に発表されたBattering RAMについて解説する

Scalable TEEの現状

- **Scalable TEE**：Legacy-SGXとは異なり、TEEのメモリサイズを実用上十分大きく取れる現行のTEEの事をこう呼ぶ事にする
  - Scalable-SGX、TDX、SEV-SNP、CCA等

- これらは**数百GBスケール**のTEE領域を用意できる一方で、そのTEEメモリの暗号化がLegacy-SGXに比べて大きく弱体化しているというトレードオフを抱えている
  - Legacy-SGXでは逆に、TEEメモリサイズが最大でも512MB、一般的には128MBしか取れないという苛烈な代償を抱えていた

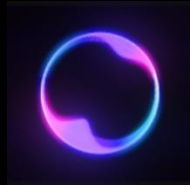
Scalable TEEの現状



- Scalable TEEは、どれも**Tweak値**（暗号鍵以外に入力する外部パラメータ）として**物理アドレス**を使用するメモリ暗号化方式を採用している
  - **Scalable-SGX、TDX**：128bit AES-XTS
  - **SEV-SNP**：128bit AES-XEXまたは256bit AES-XTS
  - **CCA**：QARMA-128 with 256bit Keyまたは128bit AES-XEX

- かつ、Legacy-SGXのように書き込みの度に更新されるカウンタをTweak値として使用する事も、完全性保証のためにHWベースのマークルツリーを使用する事もない
  - つまり、**暗号鍵が同じになる状況**であれば、同じデータを同じ物理アドレスに保存すると**同じ暗号文**となる

Scalable TEEの現状

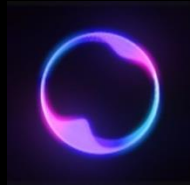


- このような性質から、Scalable TEEは単純な物理的攻撃には強いが、**ある程度以上に強い物理的攻撃に対しては無力**であるという事が以前から指摘されていた[4]

- 例えば、電源断後にメインメモリを冷却（揮発の遅延化）し他の端末に差し替えて内容を読み取るコールドブート攻撃等に対しては安全である
  - TEEメモリ暗号鍵は起動の度に変わるのも耐性に寄与している

- 一方、ある時点のTEEメモリページでロールバックするような**再生攻撃**や、物理デバイス（**インターポーザ**）で暗号文を盗聴・改竄するような攻撃には基本的に**無力**である

Scalable TEEの現状



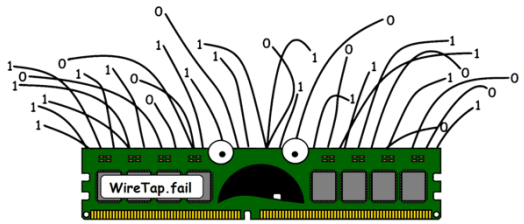
- しかし、そのような物理的攻撃は基本的に「理論的には可能だが現実的には困難」と見なされる風潮があり、実際にそのような事例は存在しない状況であった

- Legacy-SGXにおいては、Membusterというインターポーザ（DRAM等に装着する事で盗聴・改竄を行う装置）による攻撃が提案されたが、装置に**17万ドル**という多大なコストがかかっていた
  - かつ、Legacy-SGXの強力さ故に、AK（Attestation Key）の復元にも失敗する等、その攻撃能力には限界があった

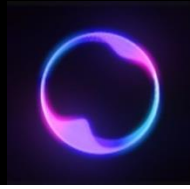
新たなScalable TEE物理攻撃

- そのような状況の中、2025年に**Battering RAM**と**WireTap**、**TEE.fail**という、Scalable TEEに対する、現実的なコストに収まる強力な攻撃が立て続けに発表された

- WireTap（DDR4向け）は直後にTEE.fail（DDR5向け）が同じ著者らにより公開されたため、本ゼミではWireTapの説明は行わない
  - 基本的にTEE.failの方が上位互換となっている



新たなScalable TEE物理攻撃



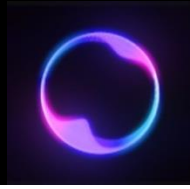
- WireTap及びTEE.failは、DRAMを流れるTEEメモリ暗号文の読みに徹してその暗号文から秘密を推測（**暗号文サイドチャネル**）するもので、**受動的インターポーザ攻撃**と言える

- 一方で、Battering RAMはDIMMのアドレス線のアース（0化）を能動的にON/OFFさせて改竄を試みる攻撃であり、TEE.fail等と異なり**積極的インターポーザ攻撃**であると言える
  - DIMM：メインメモリのモジュール。よく見るメインメモリの基板モジュールの事で、ここに複数のDRAMチップが載っている

- WireTap・TEE.failは1000ドル、Battering RAMに至っては50ドルでインターポーザを構築可能であり、Membusterに比べ圧倒的に安価に実現できているのも特徴である

DRAMアドレッシングの基礎

DRAMの基礎

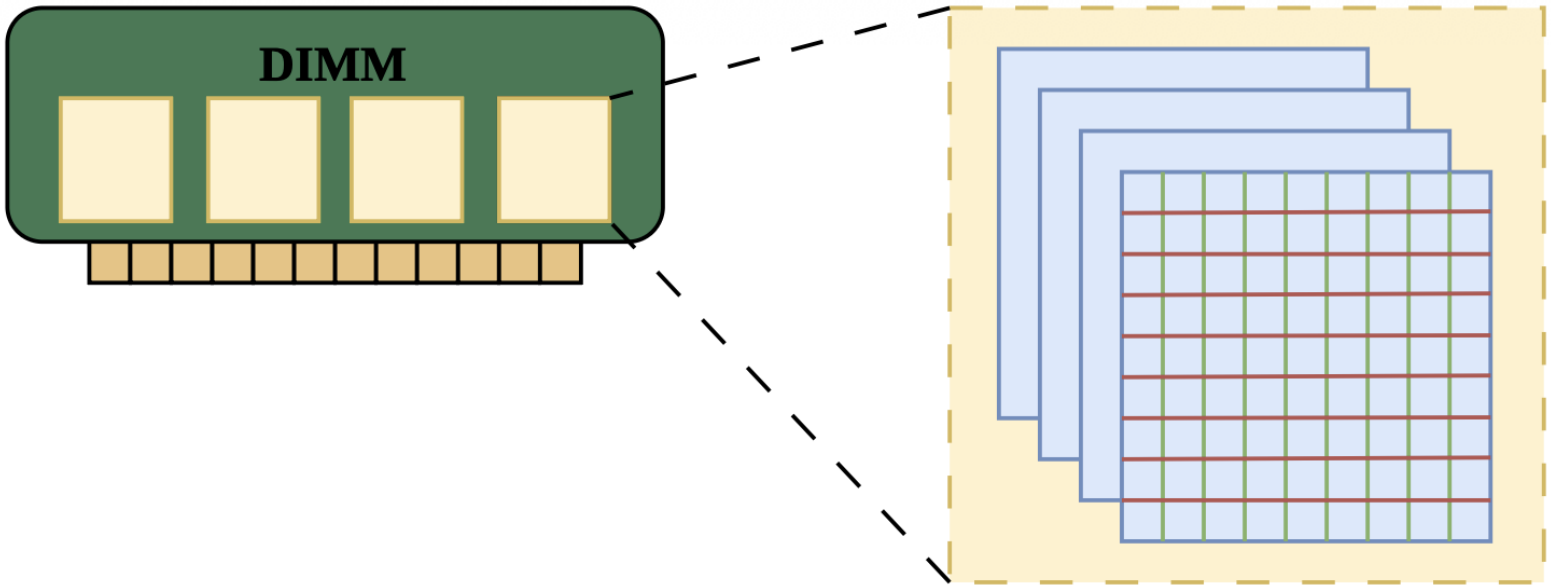


- Battering RAMとTEE.fail双方共に、メインメモリに対してハードウェア的に攻撃を仕掛けるため、まずはDRAM及びそのアドレス指定（アドレッシング）の仕組みについて説明する

- Battering RAMはDDR4に対して有効な攻撃であるため、ここではDDR4の仕組みについて説明する
  - ただし、明示的にDDR4についての説明、と書いていない部分は、DDR4/5どちらでも共通の内容である
  - DDR5のDDR4との差異についてはTEE.failにおいて説明する

DRAMの基礎

- 最初に、メインメモリ（DIMM）の構成のイメージ図を、Battering RAM論文[1]より引用し掲載する
  - メインメモリのHW的な性質について取り扱うため、仮想アドレス・物理アドレスの他、**DRAM上のHW的な物理位置**という概念が登場する



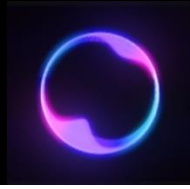
DRAMの基礎

- **DIMM**：前述の通り、いわゆるメインメモリの基板。普通「メモリを取り付ける」というと、このDIMMを取り付けている事になる
  - Dual Inline Memory Moduleの略

- 複数のダイを単一のパッケージに統合する、3次元積層（3DS）という構成の高密度チップも存在する

- DIMMには色々種類があるが、その中でも**UDIMM**と**RDIMM**と呼ばれるものが存在する

DRAMの種類



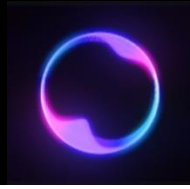
- **UDIMM**: コンシューマ向けPCに搭載されるような、いわゆる一般的なDIMM。Unbuffered DIMMの略
- **RDIMM**: サーバ・ワークステーション向けの高性能なDIMMで、UDIMMと異なりメモリコントローラとメモリチップの間にレジスタ（バッファ）が存在する。Registered DIMMの略
  - 誤り訂正用のECCビットを備えている事も多い
- DIMMの種類の詳細については[こちらのサイト]を参照
- RDIMMでは、プロセッサとDRAMチップの間に**RCD**（Registering Clock Driver）というチップが含まれている事が多い
  - データとコマンドをバッファリングしてバスへの電気的負荷を軽減したり、CAバス上でパリティチェックを行い伝送エラーを検出する

DRAMの構造

- DIMMには、**DRAMチップ**が複数個載っている
  - DRAMチップは特定の個数のDRAMチップ同士で並列動作するが、この並列動作するまとまりの事を**ランク**と呼ぶ

- このDRAMチップは**バンク**と呼ばれる単位の集合により構成されており、この2次元グリッド状のバンクに配置されたコンデンサに、電荷として保存し情報を保持する

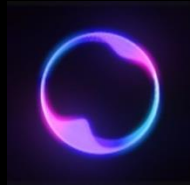
DRAMの構造



- このコマンドを受けると、アクティベーションが完了するまでの遅延（tRCD）後、今度は以下の2サイクルの読み出しコマンドを発出する：
  - アクティベーションのサイト同じ行アドレス、チップID、そしてバンクグループ/アドレス
  - 9bitの列アドレス

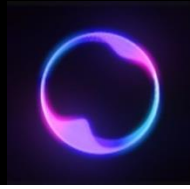
- その後さらに読み出し遅延が発生し、その後にDIMMは32データピンの16バーストとしてデータを返送する
  - 1ピンあたり1ビットなので、32*16/8=64バイトを一度に読み出しできる事になる

DRAMの構造



- データアクセスの際には、まずそのデータの物理位置に対応する行を行バッファにコピーする事でアクティベートする
- その後、さらにその行内の列を指定する事で、読み出し命令や書き込み命令を実施する事ができる
- アクティベートできる行は、バンク毎に1行のみという仕様となっている

DRAMの構造



- まとめると、まずDIMMには複数のDRAMチップが存在し、特定のDRAMチップ同士はランクというグループ分けがされている
  - DIMMは、MCのチャネルと接続する形でCPUと接続される

- DRAMチップ内にはバンクが存在し、バンクはバンクグループというグループの一員である。そしてバンクは、行と列からなる2次元グリッド状構造で、ここに情報を保持する

- つまり、DIMMョランクョDRAMチップョバンクグループョバンクョ行・列、といった関係性となる

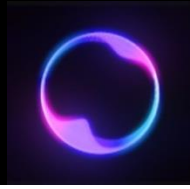
DRAMの構造

- このような構造のDIMMであるが、CPUとの接続という意味では、メモリコントローラ（MC）側が提供する**チャネル**と呼ばれるインタフェースに接続される形となる

- MCの1チャネルには複数枚のDIMMを接続する事が可能で、例えばSupermicroのX13DEG-QTというマザーボードでは、1チャネルあたり2枚のDIMMを差す事ができる
  - このチャネルに対応付いたDIMM差込口を**スロット**と呼ぶ

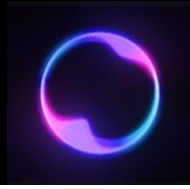
- 物理アドレスは、これらランク、バンク、行、列といった物理位置を一意に決定する整数であると表現できる[2]

MCとDRAM間の通信



- MCとDRAM間のコマンドの通信には、コマンド/アドレスバス（CAバス）が使用される
  - ちなみに、実際のデータの伝送にはDQバスが使用される
- CAバスを通して、どのアドレスのデータにどのコマンド（処理内容）を実行するかを伝達する事ができる
- このコマンドには、行アクティベーション（ACT）、読み出し（RD）、書き込み（WR）、リフレッシュ（REF）等が存在する

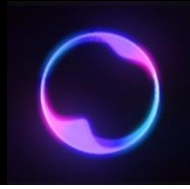
MCとDRAM間の通信



- CAバスには、操作対象とするアドレスを伝送するアドレスラインが存在する
  - DDR4では、CAバスはアドレスラインが18本、それ以外のコマンド用途等のラインが8本の、計26本の線が存在する

- これらの線（ビット）を駆使する事で、メモリアドレス（例：行・列）、コマンドタイプ、追加オプションのエンコードを実施する
  - つまり、MCは物理アドレスからDRAM物理位置への変換を行う役割も担っている

MCとDRAM間の通信



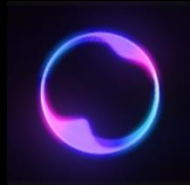
- JEDECにより標準化されているDDR4コマンドエンコーディングは以下のようになっている[1]：

	$\overline{CS}$	$\overline{ACT}$	BG	BA	CID	A17	A16	A15	A14	A13	A12	A11	A10	A9-0
ACT	L	L	Bank		CID	Row								
MRS	L	H	Reg		V	RFU	L	L	L	OP				
REF	L	H	V	V	CID	V	L	L	H	V	V	V	V	V
PRE	L	H	Bank		CID	V	L	H	L	V	V	V	L	V
PREA	L	H	V	V	CID	V	L	H	L	V	V	V	H	V
RFU	L	H	RFU				L	H	H	RFU				
WR	L	H	Bank		CID	V	H	L	L	V	$\overline{BC}$	V	AP	CA
RD	L	H	Bank		CID	V	H	L	H	V	$\overline{BC}$	V	AP	CA
ZQCL	L	H	V	V	V	V	H	H	L	V	V	V	H	V
ZQCS	L	H	V	V	V	V	H	H	L	V	V	V	L	V
NOP	L	H	V	V	V	V	H	H	H	V	V	V	V	V
DES	H	X	X	X	X	X	X	X	X	X	X	X	X	X

H: High（ビットで言えば1）、L: Low（ビットで言えば0）、
X: 無関係（無視される）、V: Valid（その状況で期待される有効な値）、
CID: 3DSデバイスでのみ定義

Battering RAM

Battering RAM



- 前述のような物理線を有するCAバスであるが、ここを通るコマンドやアドレス情報は、データ伝送とは異なり**TEEによって暗号化される事はない**

- つまり**CAバスを通る内容は平文**であるため、何らかの**物理手段**を用いればその内容を**改竄**する事ができる

- かつ、先行研究として、メモリのSPDチップ内の構成設定データを改竄し物理アドレス同士でエイリアスを発生させる事でTEEを侵害するBadRAM[6]が存在する
  - CAバスを物理的に攻撃し、エイリアスを発生させれば同様の脅威となる可能性が多分にある

Battering RAM

- この類推から、CAバスのアドレスラインを任意のタイミングで物理的に接地（アース）及び接地解除できる**インターポーザ**を使い攻撃を仕掛けるのが**Battering RAM攻撃**である

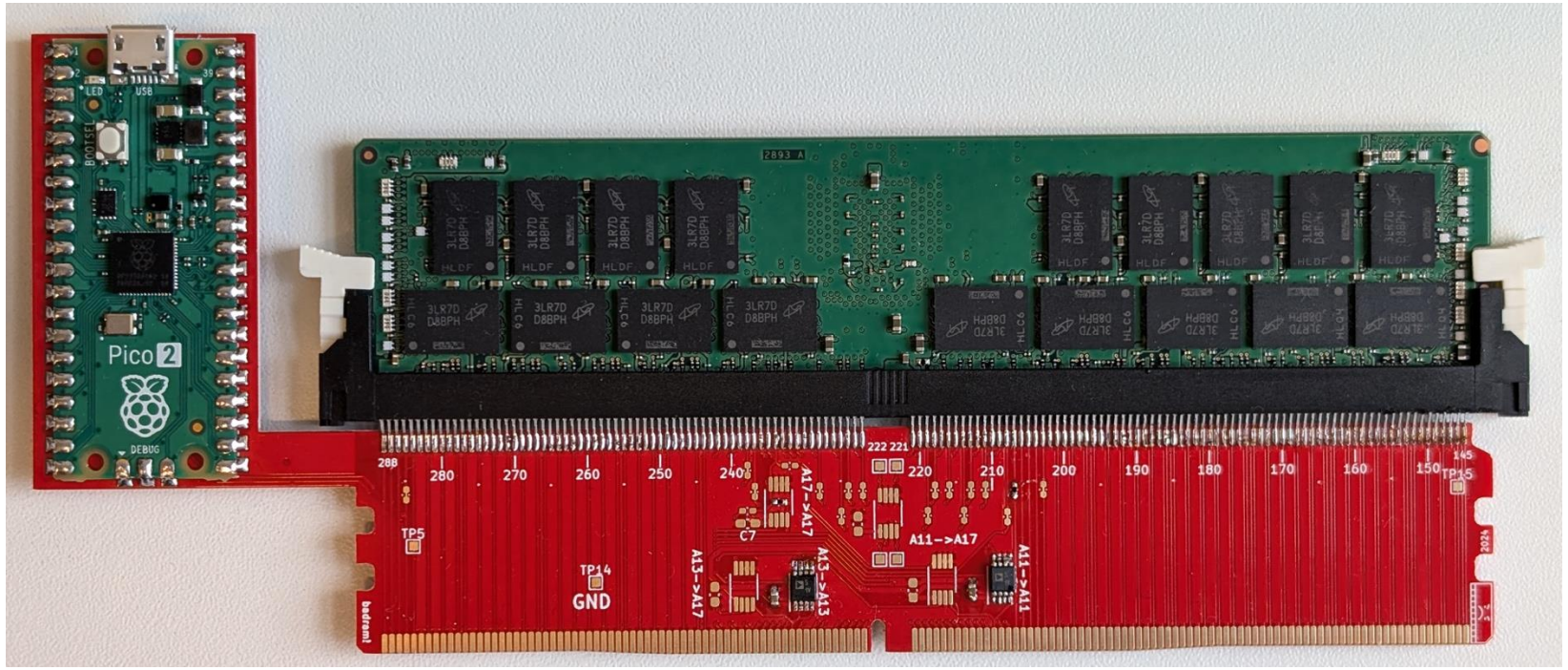
- BadRAMは予めメモリを改竄しておくため、後から追加された起動時のエイリアスチェック機能によって阻止できていたが、Battering RAMは**ランタイム中の改竄**なので阻止できない

Battering RAMにおける脅威モデル

- まず、TEEにおいて一般的な脅威モデルは踏襲する
  - つまり、攻撃者はそのマシンにおいて**root権限**を有し、あらゆる特権攻撃を仕掛ける事ができる
- 更に追加で、攻撃者はマシンに対する**物理アクセス権**を有し、DRAMインターポーザを装着するために十分な一時的なレベルのもので構わないとする
  - サプライチェーンの一時的な侵害や、施設への侵入等
  - かつ、このインターポーザは侵襲性が小さいため、恒久的な不可逆的な改造は不要。イメージとしては取り外し可能なようなもの
- 攻撃対象はDDR4プラットフォームのみとする
  - 具体的にはScalable-SGXとSEV-SNPが対象。TDXはDDR5でのみ動作するため、本攻撃では対象とする事ができない

Battering RAMインターポーザ

- Battering RAMで使用されているDRAMインターポーザの実物の写真は以下の通り（[1]より引用）：
  - 赤い部分がインターポーザで、右上の緑部分がDIMM。左のマイコンで右下のスイッチを切り替え、動的に接地を切り替える



Battering RAMインターポーザ

- Battering RAMでは、後に詳述の通り、2つの物理アドレスが**同一のDRAM物理位置を指す**ようにする（**エイリアス**）事により様々な悪性の動作を誘発させる事を狙っている
  - 前述の通り、これはBadRAM攻撃と同様の事を、インターポーザによりランタイム中に実現すると表現できる

- これを実現する方法として、CAバス上で伝送されるDRAMアドレス（行、列、バンクビット等）を直接改竄するという方法があるが、DDR4がかなりの高周波であるのもあり、これは現実的ではない
  - 数GHzで動作する極めて精密な制御ロジックが必要となり、これは安価な物理攻撃の実現というこの研究の目的に反する

Battering RAMインターポーザ

- よって、前述の通りCAバスの物理線を単に接地させる事で、対象の線を強制的にLow（0）にする事を考える

- ただし、どの線を接地させるかについて考える必要があるため、再度以下の図を掲載する（[1]より引用）：

	$\overline{CS}$	$\overline{ACT}$	BG	BA	CID	A17	A16	A15	A14	A13	A12	A11	A10	A9-0
ACT	L	L	Bank		CID	Row								
MRS	L	H	Reg		V	RFU	L	L	L	OP				
REF	L	H	V	V	CID	V	L	L	H	V	V	V	V	V
PRE	L	H	Bank		CID	V	L	H	L	V	V	V	L	V
PREA	L	H	V	V	CID	V	L	H	L	V	V	V	H	V
RFU	L	H	RFU				L	H	H	RFU				
WR	L	H	Bank		CID	V	H	L	L	V	$\overline{BC}$	V	AP	CA
RD	L	H	Bank		CID	V	H	L	H	V	$\overline{BC}$	V	AP	CA
ZQCL	L	H	V	V	V	V	H	H	L	V	V	V	H	V
ZQCS	L	H	V	V	V	V	H	H	L	V	V	V	L	V
NOP	L	H	V	V	V	V	H	H	H	V	V	V	V	V
DES	H	X	X	X	X	X	X	X	X	X	X	X	X	X

Battering RAMインターポーザ

- この図を見ると、アドレス線（Axx）の内、A16-14、A12、A10以外の全ての線は、MRS（Mode Register Set）コマンド以外では**位置を示す以上の意味的役割を持たない**
  - 例えば、A9-0はWRとRDでCAとなっているが、これは行アドレス（Column Address）の事であり、ただの位置情報である

- つまり、MRSのようなコマンド以外では、このようなアドレス線はその状況下でValidである限り**任意の値を取って良い事**になる

- よって、このようなアドレス線を接地で改竄すれば、**他の動作に影響を与える事なく効果的にアドレスを改竄**できる

Battering RAMインターポーザ

- 具体的には、同一の物理アドレスについてのMCからのDRAM物理位置情報でも、接地の有無で異なるDRAM物理位置の情報がDIMMに伝わる事になる

- 例えばある物理アドレスP1に対する本来の物理位置が101**1**、別の物理アドレスP2に対するそれが101**0**であるとする

- この時、P1についての処理時に最下位ビットを接地させると、その物理位置は101**0**となり、これはP1とP2が同一の物理位置を指す**エイリアス状態**になる

Battering RAMインターポーザ

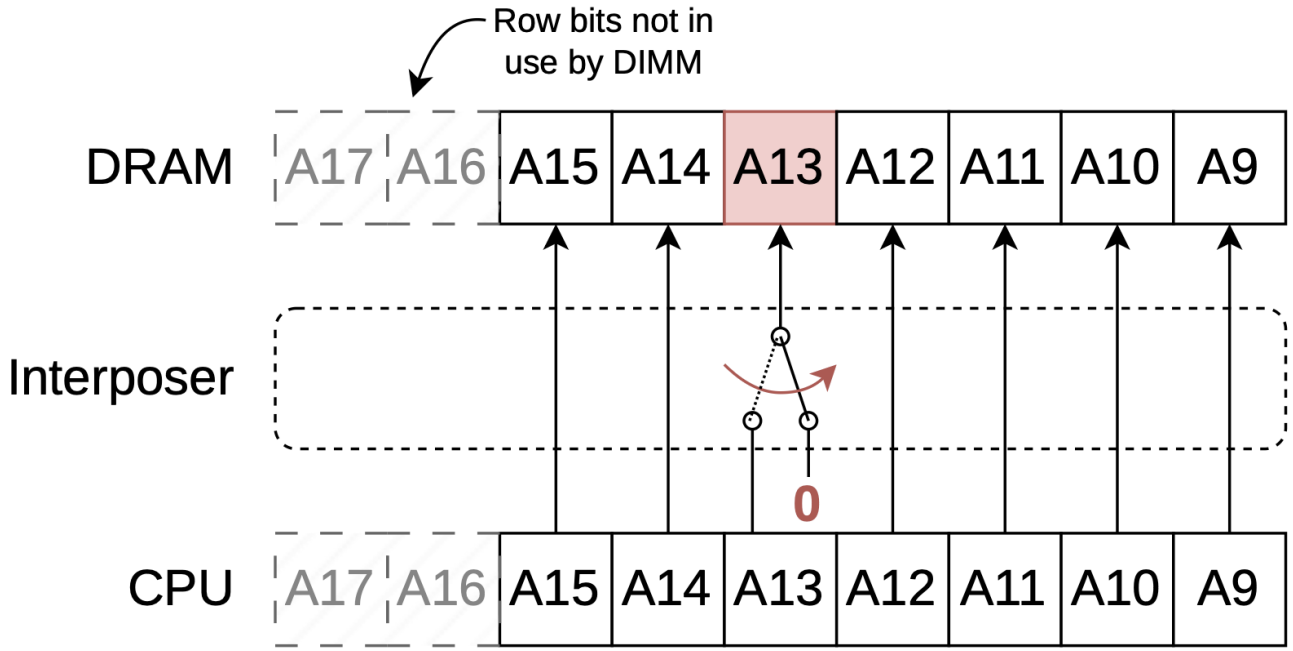
- 実際にはDDR4ではアドレス線がA0からA17まで存在するが、結論から言えばBattering RAMでは**A11とA13を接地**するようなインターポーザを使用する方針をとっている

- これは、下位ビット（A0-9）やバンクビット（BA、BG）を接地させると、例えば行・列の双方等複数要素に影響が波及する性質があり、エイリアスを導入するのが難しくなるためである
  - 単一要素だけの変更で確度の高いエイリアスを導入する方が攻撃の安定性も上がる

- CIDビットに関しては3DSデバイスでのみ定義されるため、汎用性の観点からこれについても不採用としている

Battering RAMインターポーザ

- よって、Battering RAMでは以下の図（[1]より引用）のように動作する、A11またはA13のいずれかを任意のタイミングで接地させられるインターポーザを開発した
  - 図はA13についての接地切り替えの概要。接地非アクティブ時はパススルーされ、接地時は強制的にLowとして認識される事になる



Battering RAMインターポーザ

- よって、Battering RAMでは以下の図（[1]より引用）のように動作する、A11またはA13のいずれかを任意のタイミングで接地させられるインターポーザを開発した
  - 図はA13についての接地切り替えの概要。接地非アクティブ時はパススルーされ、接地時は強制的にLowとして認識される事になる

- 前述の通り接地スイッチはマイコン（ラズピコ2）により行われ、さらにこのマイコンは攻撃者の第2のデバイスから制御できるようなインタフェースを提供している

Battering RAMインターポーザ

- このインターポーザの部品表は以下の通り[1]：
  - Membusterが17万ドルなのに対し、たった50ドルと極めて安価に仕上げられている事が分かる

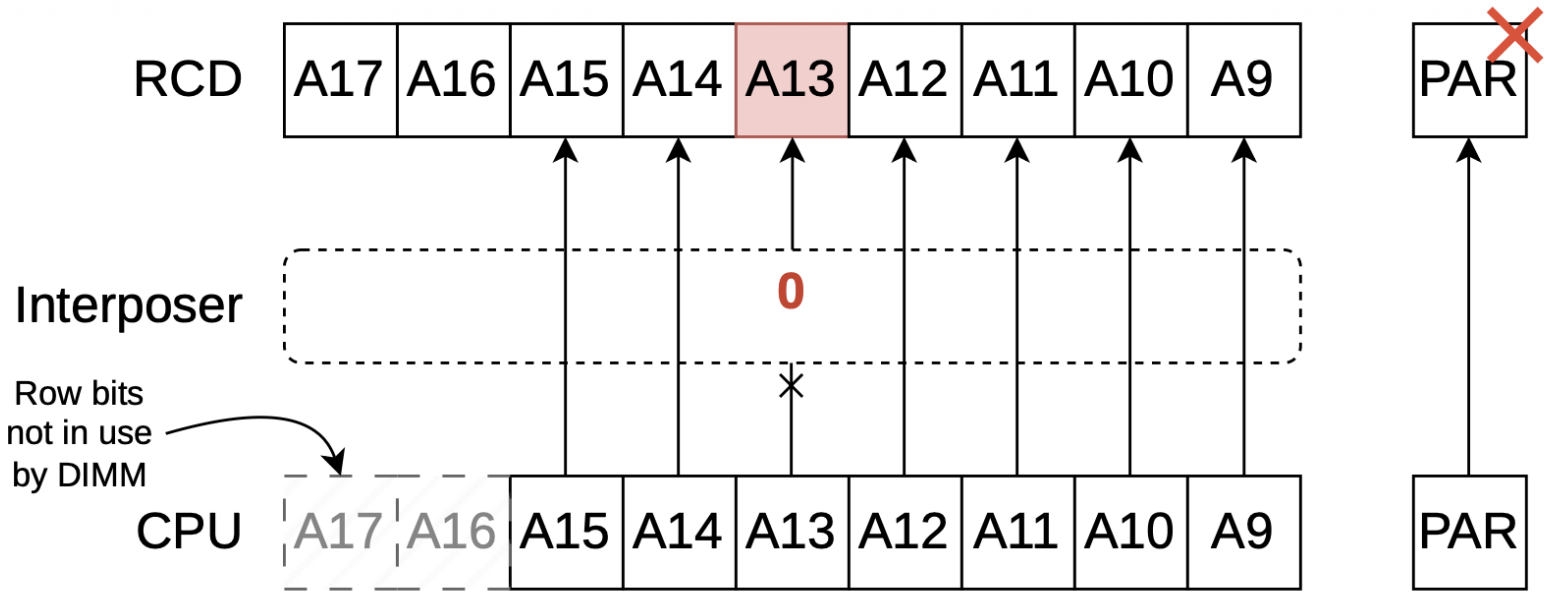
Item	Part Number	Qty.	Cost
Printed Circuit Board	N/A	1	$18.49
DIMM connector	CONN-DDR4-288-SM	1	$16.00
Microcontroller	Raspberry Pi Pico 2	1	$5.00
Analog switch	ADG902BRMZ	2	$8.00
Voltage regulator	LD1117S25TR	1	$0.65
Miscellaneous resistors/capacitors			$0.50
Total			$48.64

RCDパリティチェックのバイパス

- ところで、Scalable-SGXやSEV-SNPが動作するようなサーバマシンに搭載されがちなRDIMMには、UDIMMとは異なり前述の通りRCDチップが搭載されている
  - かつ、Battering RAMはRDIMM搭載マシンへの攻撃を前提としている
- RCDは前述のバッファリングに加え、オプションとして**CAバス上のパリティチェック**も実行する
  - 具体的には偶数パリティ方式が使用される
- このMCから送信されるパリティ信号は、CPU-DIMM間の伝送エラーを検出するために**全CAバスをカバー**している

RCDパリティチェックのバイパス

- これにより、RDIMM上で単一ビットが故障すると、以下の図（[1]より引用）の通りパリティエラーが発生し、コマンドが破棄される
  - かつ、このエラー検出に伴いRCDがALERTnというラインを通じてそれをCPUに通知する



(a) Single switch.

RCDパリティチェックのバイパス

- このままでは折角インターポーザでアドレス線の接地をさせてもRCDにより捨てられてしまうので、何とかしてこのチェックを迂回する方法を見つける必要がある
  - 大前提ではあるが、RCDはあくまでコマンドの誤り検出しか行わない。データについてはECCビットで実施する

- この迂回方法として、Battering RAMでは以下の3通りの方法を提案している：
  - RCDパリティの無効化
  - データラインのリダイレクト
  - 複数スイッチング

RCDパリティの無効化

- そもそも**RCDパリティ機能自体を無効化**できるのであれば、素直に無効化してしまうのが一番早い

- 中でも**AMD**プラットフォームでは、**BIOSからRCDパリティチェックを無効化**できるようになっていた
  - 技術的には、F0RC0EというRCD内の構成レジスタを通じてパリティチェック実行を行うかが設定される

- RCDはSMBusに接続されているため、理論上はI2C経由で攻撃できる可能性もあるが、少なくとも論文で取り扱っているテストプラットフォームではI2Cが無効化されていた
  - 技術的には、F0RC2xというRCD内レジスタを通じてBIOSが無効化を選択している可能性がある

データラインのリダイレクト

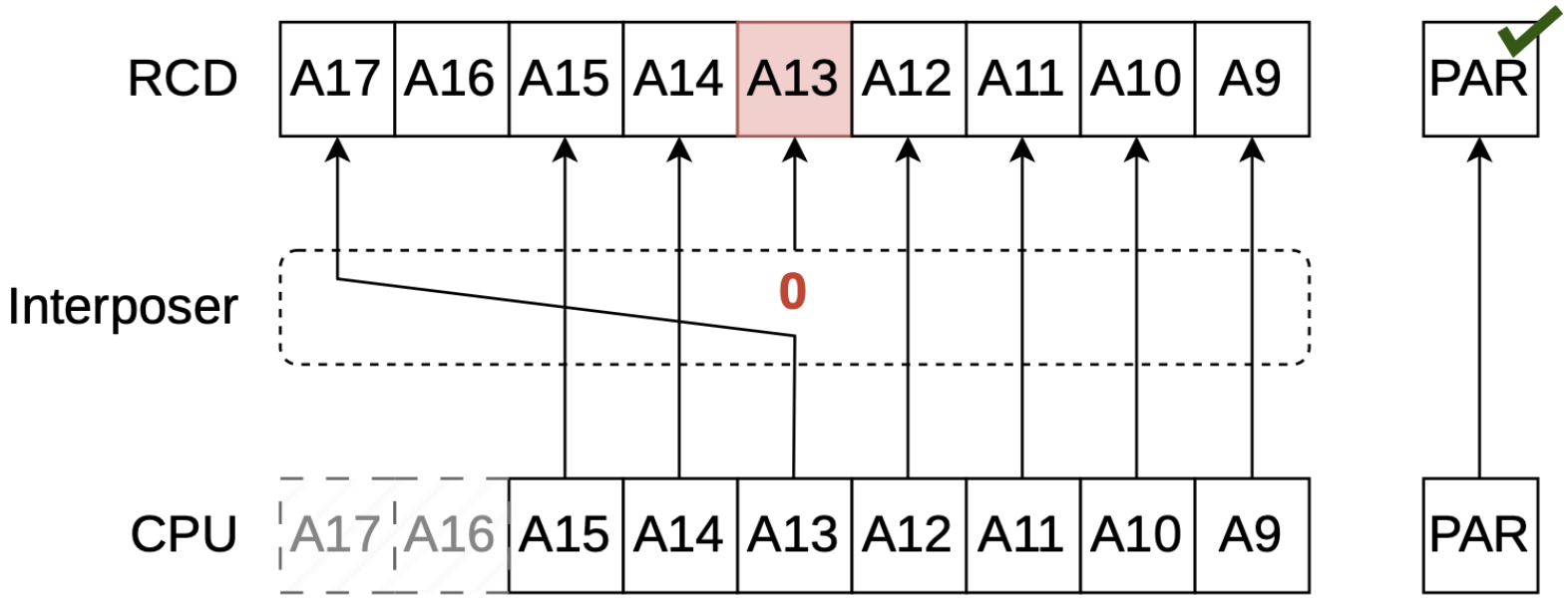
- パリティチェックは、受信した**全CAビット**に対して**XOR演算**を行い、その結果を見て誤りの検出を実施する

- よって、XORの性質上、ビットの入れ替えやリダイレクト（再配線）を行ってもパリティチェック結果には影響しない
  - つまりは、A⊕BはB⊕Aに等価である事を利用する

- かつ、アドレス線**A17**（最上位ビットに相当）や**A16**は、**そもそも使用されていない**事がままある

データラインのリダイレクト

- この性質を利用し、標的であるA13またはA11を、単に接地するだけでなく、未使用のアドレス線**A17にリダイレクト**する事を考える（図は[1]より引用）



(b) Redirecting address line.

データラインのリダイレクト

- A17は未使用なので、リダイレクトが無い場合には**常に0**であり、かつ物理位置の算出計算からは除外されるが、パリティチェックでは除外されるとは限らない

- ここで例えばA13をA17にリダイレクトすると、A13の内容はA17に行き、A13は**常に0**になる

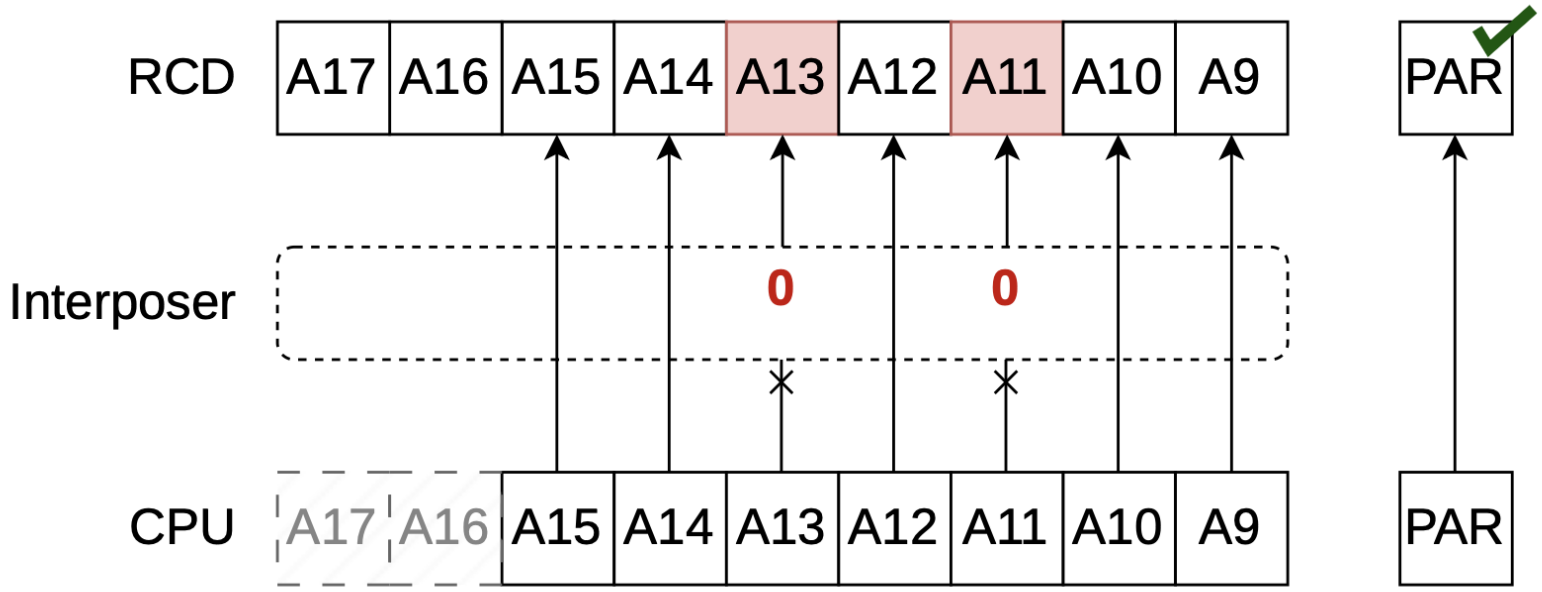
- つまり、A13とA17が単に入れ替わっただけになるため、A17を含めた**パリティチェックには合格**し、かつ物理位置算出では**A17は無視**される状況が生まれる
  - すると、結果的にA13を0にしたという操作のみが位置算出に反映され、**A13を接地**させるという**本来の目的を果たす**事ができる

複数スイッチング

- パリティが全CAビットのXOR演算結果であるという性質から、ある**2ビット（2本の線）を同時に接地**させても、同一のパリティ出力となる事が分かる
  - 例：1⊕1⊕1の後ろ2つを接地させて1⊕0⊕0にしても、結果は元と同じ1
- ただし、**変更する2つが同値でなければならない**という条件はある
  - 例：1⊕1⊕0=0の後ろ2つを接地させると、1⊕0⊕0=1となり不一致
  - 後述の通り、2つが同値でなくても通す方法は無くはない
- A11とA13に関して、このような条件を満たすようなアドレスのページを使わせる事は、TEEの脅威モデルに適合する
  - TEE脅威モデルではホストOSを攻撃者が完全に掌握しているので、単にホストOSから目当てのページのみを使わせるようにすればよい

複数スイッチング

- 2つの線をスイッチし同時に接地させる際のイメージ図は以下の通り（[1]より引用）：



(c) Dual switches.

Battering RAMインターポーザの評価

実験環境

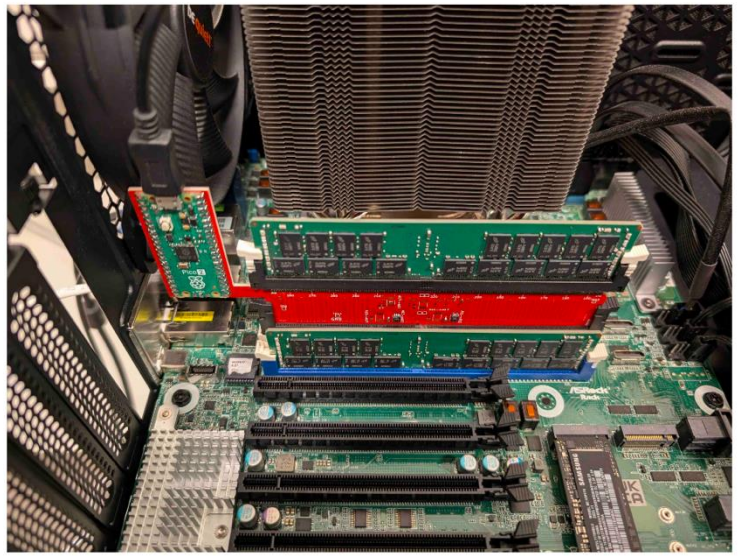
- このインターポーザを、以下の表（[1]より引用）に詳細を示すSEV-SNPマシン、Legacy-SGXマシン、Scalable-SGXマシンの3つに取り付けてそれぞれ評価する
  - それぞれの環境を順にAMD1、Intel1、Intel2と呼ぶ事にする

System	TEE	Mainboard	CPU	DIMM(s)	DRAM
AMD_1	SEV-SNP	ASRock ROMED8-2T	EPYC 7313P	1×MTA36ASF8G72PZ-3G2F1 1×MTA36ASF4G72PZ-3G2R1	DDR4 RDIMM
$Intel_1$	Client SGX	Dell D11S001	Core i5-6500	1×HMA41GU6AFR8N	DDR4 UDIMM
$Intel_2$	Scalable SGX	Supermicro X12DPi-NT6	Xeon 6330	16×HMAA4GR7AJR8N-XN	DDR4 RDIMM

実験環境

- インターポーザをIntel2（Scalable-SGXマシン）とAMD1（SEV-SNPマシン）に装着した様子は以下の通り（画像は[1]より引用）：

(a) Intel_2.



(b) AMD_1.

クリティカルなメモリ領域の追い出し

- インターポーザにより、カーネルメモリやACPI MMIO等の**重要なメモリ領域**と**意図せずエイリアス**が発生してしまった場合、**意図しない不具合**を誘発してしまう可能性が高い

- このような領域は、通常は**物理アドレス空間の下位4GB内**に存在する

- よって、そのような物理アドレス範囲は、**インターポーザを装着していないDIMMに追い出してしまう**と、上記のような意図しないエイリアスを未然に防ぐ事ができる

クリティカルなメモリ領域の追い出し

- この時、全メインメモリ領域をグループ分けし、均等に物理アドレスを振る事で全体的なメモリアクセス速度向上を図るメモリインターリーブ機能は、追い出しの上で極めて都合が悪い

- AMD1ではBIOS設定でメモリインターリーブを無効化できたため、無効化した上で追い出しを行い意図しないエイリアスを予防している

クリティカルなメモリ領域の追い出し

- Intel2はデュアルソケットシステムであり、各ソケットのメモリ領域は論理的に分離されているので、例えば2番目のソケットにインターポーザをつけると1番目の方には影響が出ない

- よって、第2ソケットのDIMMの1つにインターポーザを装着し、第2ソケットに対応する物理メモリ全体を予約しておく事で、クリティカルな部分は予約部分を避けて全て第1ソケット側に確保される

- Intel1（Legacy-SGX）はついでくらいにしか実験しないため、特に追い出しについての言及は論文内には存在しない

不安定な領域

- 複数スイッチングを行う場合、前述の通り片方だけがHighであるようなビットのペアに対して実施すると、パリティチェックに引っかかり、システムクラッシュが誘発されてしまう
  - いわば、このようなアドレス線の組み合わせに紐づくメモリ領域は「**不安定な領域**」と言える
- これを防ぐ1つの方法として、DIMMがCPUにエラーを伝達するのに使われるALERTnラインを接続解除しておくというものが考えられる
- あるいは、物理アドレスからDRAM位置へのマッピングは決定論的であるため、Linuxカーネルのmemmapパラメータを利用し、そのような領域を予約して利用を避ける方法もある

不安定な領域

- そこで、A11とA13のいずれかを接地させた時にクラッシュする物理アドレスビットを記録する事で、A11とA13が1になるような物理アドレスビットを特定した

- 結果、AMD1ではA11とA13がそれぞれ物理アドレスビット28と30に、Intel2では31と33に対応していた

- 上記結果から、片方のみが1となるような物理アドレスビットを避ける事で、不安定な領域を回避できる
  - 例えば、AMD1であれば物理アドレスビット28と30の内どちらかのみが1であるような領域は、同時接地でパリティエラーを誘発するので攻撃で使ってはならない

RCDパリティチェックバイパスの評価

- 先程説明した、以下の3つのRCDパリティチェック迂回方法について、実際に実験的に有用性を評価した：
  - RCDパリティの無効化（単一ラインの接地操作）
  - データラインのリダイレクト
  - 複数スイッチング

- Intel1（Legacy-SGX）は**UDIMM**であるため、何もせずとも単一の接地のみでエイリアスを作成する事ができる

RCDパリティチェックバイパスの評価

- Intel2（Scalable-SGX）はRDIMMである上にBIOSから
  RCDパリティを無効化できないため、単一ラインの接地では
  エラーを回避する事が**不可能**であった

- AMD1では、前述の通り**RCDパリティ無効化設定をBIOS経由で実施**できるため、無効化した後は単一ライン接地をしても
  クラッシュしなくなった

RCDパリティチェックバイパスの評価

- データラインのリダイレクトについては、未使用行ビットA17に、A11またはA13をリダイレクトする、前述の図に示した通りの構成で実験した
- AMD1では成功したが、Intel2では失敗する結果となった
- これは、Intel2ではRCD設定レジスタ（F0RC08）を通じてA17入力バッファを明示的に無効化し、パリティ計算への参加を阻止しているのが理由であると考えられる

RCDパリティチェックバイパスの評価

- 複数スイッチングについては、Intel2及びAMD1の双方において、前述の不安定な領域を避けつつ、パリティエラーを誘発する事なくエイリアスを導入する事に成功した

- 他の手法よりも汎用性が高いので、Intel2及びAMD1においては、以降こ**のバイパス手法を用いてエイリアシングを行う**ものとする

安定性の評価

- 前述の3つのマシンにおいて、インターポーザを装着した状態でMEMTEST86+及びstress-ngを実行し、メモリストレステストも実施している

- いずれの場合もデフォルトのメモリ速度で実行し、テスト中にクラッシュするような事は無かった

- ただし、Intel2ではこのテストではなく研究プロトタイピング中に、エイリアスされたEPCページへのアクセス時に**ランダムにクラッシュが発生**していた
  - 原因は不明だが、起きてもシステムを再起動してやり直せば良いので、攻撃の上で大きな障壁にはならないとしている

実現可能性

- メモリストレステストでも確認した通り、デフォルトのDRAMクロック周波数で動作するため、OS等のSWからインターポーザの存在を検知する事は不可能である

- また、物理アドレスからDRAM位置へのマッピングは固定であるため、攻撃対象箇所とそのエイリアスが既知であれば、完全に決定論的に100%の精度でキャプチャ・再生できる

- この事から、仕込みさえしっかりしておけば実用上十分な実現可能性を備えているインターポーザであると言える

制限事項

- Intel2では、ソケットレベルで分離する事でクリティカル領域の追い出しを実施しているが、メモリインターリーブ自体の**無効化は行えない**ようである

- Intel2では16枚のDIMMがあるが、その半分の8枚がインターポーザ側のソケットに対応している事になり、つまりは**8枚とインターリー**ブされる事になる

制限事項

- よって、単一のインターポーザではソケット内の特定のページの1/8のキャッシュラインにしかアクセスできないが、これでも**攻撃の実行の上では十分**との事である
  - 全DIMMに装着する手もあるが、その分金銭的コストも増えるのもあり、論文では単一のままで進めている

- なお、このシステムでは単一のインターポーザでエイリアス可能な最小の連続サイズは256バイトであると記されている

Scalable-SGXからのPKの抽出

Scalable-SGXへの攻撃の概要

- Scalable-SGXに対する攻撃としては、**PCEからPKを抽出**し、任意の偽造Quoteを生成する事で**RAを破綻**させる事を考える

- Scalable-SGXは、Legacy-SGXと異なり物理アドレスをTweak値としており、かつ**全PRMを単一のTME鍵で暗号化**するため、前述の通り同じ平文を同じ場所に置くと**同じ暗号文**となる
  - これは、コールドブート攻撃のような単一読み取り攻撃は防げるが、**繰り返し読み取り**や**再生攻撃**に対しては**無力**となる

- この性質から、メモリエイリアシングを介した**暗号文サイドチャネル**（Cipherleaks[7]）や**暗号文再生攻撃**（BadRAM[6]）を回避するための追加対策が必要となる

Scalable-SGXのエイリアシング軽減策

- Intelは、エイリアスを2種類に分類している
- 1つ目は、EPCページがEPC外の別ページとエイリアスする、**outside-in**エイリアス
- もう1つは、別々のEPCページ同士でエイリアスする、**inside-in**エイリアスである
- これらをSWレベルで対策するため、SGXでは2種類の防御機能を搭載している

Scalable-SGXのエイリアシング軽減策

- Scalable-SGXでは、ECCビットを1つ再利用し、キャッシュラインがEnclaveメモリか通常メモリかを示す**Owner Bit**を保持している
  - TDXでも同様の仕組みがあり、そちらはTD Owner Bitと呼ばれている
- MCは、メモリ書き込みの度に、実際の実行コンテキストに応じてこのビットをセットする
  - TDXと同様であれば、TEEコンテキストの時に1、通常時は0
- 読み取り時にはこのビットが現在のドメインと一致するかをチェックし、不一致の場合はオールゼロを返すかSGX自体を無効化する
  - Abort Page Semanticsと似ているが、APSはEnclave外からの読み取り防止、Owner Bitは改竄時にビット不整合で検出するものなので、別物

Scalable-SGXのエイリアシング軽減策

- Owner BitはEPC外からの改竄を検知する機能を持たないため、つまりはoutside-inエイリアスしか対策できない

- よって、inside-inエイリアス対策として、**起動時エイリアスチェック**機能が有効である

- Intelマシンでは、**MCHECK**または**ACTM**が起動時エイリアスチェックを実施してくれる
  - MCHECKは、Intel署名付きのFWモジュールで、SGX命令等の実装も行っているXuCode（拡張マイクロコード）の一部[8]
  - ACTM：Alias Checking Trusted Module

Scalable-SGXのエイリアシング軽減策

- このチェックは、改変されたSPDを用いる等により起動時点でエイリアスを作る**BadRAMのような攻撃の対策**として**非常に有効**

- ただし前述もしたように、Battering RAMのようなランタイム中にエイリアスを生成するような攻撃には**無力**である
  - あくまでも起動時に一度チェックを行うだけであるため

Scalable-SGXへの攻撃

- よって、インターポーザによるランタイム中エイリアシングを EPCページ間（inside-in）で適用する事で、Owner Bitと 起動時チェックの双方を迂回する事ができる

- かつ、攻撃側と攻撃対象が確実にエイリアシング位置に存在するよう緻密に制御するために、論文では**改造out-of-treeドライバ**を利用している

- 具体的には、純正out-of-treeドライバに新しいioctlメソッドを追加し、攻撃者が次のEPC割当に希望する物理アドレスを指定できるようにしている

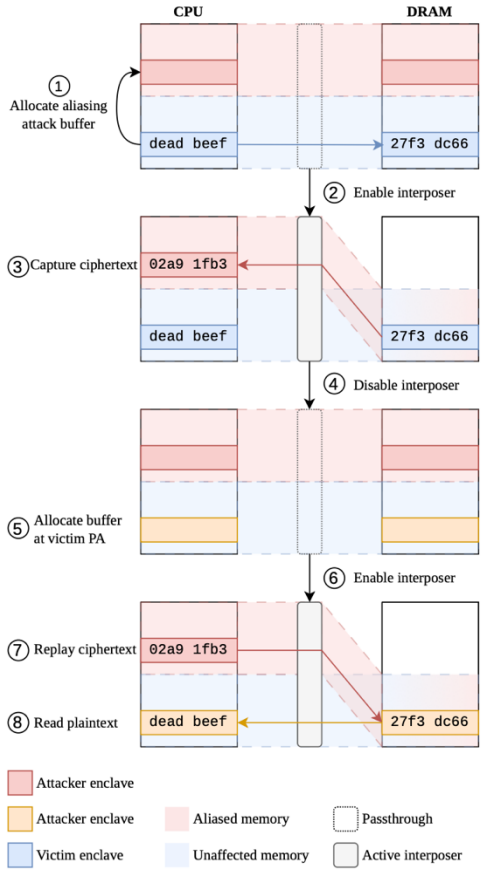
Scalable-SGXへの攻撃

- この改造は、空きページリストの最初のページを返す本来の実装の代わりに、指定された物理アドレスが見つかるまでリストを反復処理するようにする単純な変更のみで済む
  - 指定の無い場合は、インターポーザの影響を受けない、つまりA11とA13の両方が0にマップされるページを返す

- 攻撃者の行うエイリアス手順の概要は以下の通り：
  - 攻撃対象Enclaveを作成する。アドレス指定が無い場合は、影響を受けないページに割り当てる
  - 秘密情報を含む物理アドレスを特定し、エイリアスを算出する
  - 改造ドライバを設定し、エイリアス位置に攻撃者側バッファを割り当てる

任意の平文アクセス手法

- ここでは、インターポーザを用いてEnclaveに任意の平文読み書きを行うための具体的な方法について説明する。平文アクセスの手順の概要図を以下に示す（[1]より引用）



任意の平文アクセス手法

- 平文読み取り・書き込み共に、攻撃においては以下の要素が登場する：
  - エイリアシングされる物理ページP_{V_1}を持つ攻撃対象Enclave V_1
  - エイリアシングされる物理ページP_{A_1}を持つ攻撃側Enclave A_1
  - もう1つの攻撃側Enclave A_2
  - インターポーザ

任意の平文読み取り手順

1. 攻撃対象EnclaveであるV_1を起動する。この時、秘密情報を物理アドレスP_{V_1}に持っているものとする

2. 攻撃側EnclaveであるA_1を起動する。エイリアシングする物理アドレスはP_{A_1}であるとする

3. インターポーザを起動する事で、V_1とA_1が同じDRAM物理位置を指すようになる。つまり、この時点でV_1がA_1の秘密情報をエイリアス経由でアクセスできる事になる
   - 物理アドレスではなく同じDRAM物理位置である点に注意

任意の平文読み取り手順

4. 攻撃者は、A_1のP_{A_1}を通して攻撃対象データを読み取り、A_1内のエイリアシングと無関係な退避バッファに控えておく
   - 退避バッファの物理アドレスは$P_{A_{1E}}$とする

5. インターポーザとV_1を終了する。TME暗号化はDRAM物理位置ではなく物理アドレスをTweak値とするため、この時点では秘密情報をA_1が復号する事はできない

6. 攻撃者は、終了したV_1と同じ物理アドレスに別の攻撃側Enclave A_2を起動する

任意の平文読み取り手順

7. 再度インターポーザを有効化し、A_1内で退避バッファアドレス$P_{A_{1E}}$から、同じくA_1内のエイリアシング用アドレスP_{A_1}に書き戻す

8. インターポーザにより、P_{A_1}とP_{A_2}でエイリアスが起きるため、A_2からA_1上の秘密情報が取れる。ここで、$P_{A_2} = P_{V_1}$であるため、Tweak値が同じとなり、復号して平文を観測できる

任意の平文読み取りの各手順におけるデータの状態

1. 初期状態。P_{V_1}上に$\mathrm{Enc}(P_{A_1}, Secret)$という形で秘密情報がある
   - 第1引数はTweak値となる物理アドレス、第2引数は暗号処理対象
2. 同上
3. 同上
4. レジスタ内では、$C' = \mathrm{Dec}\big(P_{A_1}, \mathrm{Enc}(P_{V_1}, Secret)\big)$、つまりはいわば誤復号したゴミデータが存在する。かつ、$P_{A_{1E}}$上では$Enc(P_{A_{1E}}, C')$として存在する
   - ややこしいが、この場所から読み取るとレジスタ内ではC'となる
5. 同上
6. 同上

任意の平文読み取りの各手順におけるデータの状態

7. $P_{A_{1E}}$を読み取ってからP_{A_1}に書き戻すと、レジスタ内でC'に
   なってからP_{A_1}に書き込むので、メモリ上では$\mathrm{Enc}(P_{A_1}, C')$となる
   - これは$\mathrm{Enc}(P_{A_1}, \mathrm{Dec}(P_{A_1}, \mathrm{Enc}(P_{V_1}, Secret)))$となり、$\mathrm{Enc}(P_{A_1}, x)$と$\mathrm{Dec}(P_{A_1}, x)$はXOR的に相殺されるので$\mathrm{Enc}(P_{V_1}, Secret)$

8. A_2経由で読み取ると、$\mathrm{Dec}(P_{A_2}, \mathrm{Enc}(P_{V_1}, Secret))$。ここで、
   $P_{A_2} = P_{V_1}$なので無事$Secret$がA_2に置かれる

任意の平文書き込み手順

1. 攻撃者は、攻撃側Enclave A_1においてP_{A_1}に書き込みたい平文を書き込む

2. インターポーザを起動し、別の攻撃側Enclave A_2のP_{A_2}をP_{A_1}とエイリアスさせ、P_{A_2}経由でP_{A_1}に書き込んだ内容に対する暗号文を取得する。その後、A_1とインターポーザを終了する
   - 取得した暗号文は、A_2内の退避バッファ$P_{A_{2E}}$に保存する

任意の平文書き込み手順

3. 攻撃対象Enclave V_1を、終了したA_1と同じ物理アドレスに
   なるように起動する

4. 再度インターポーザを起動し、P_{A_2}をP_{V_1}とエイリアスする。
   $P_{A_{2E}}$から読んだ値をP_{A_2}経由で暗号文をP_{V_1}に書き込む。すると、
   P_{V_1}はその暗号文を復号し平文を見る事ができる

任意の平文書き込みの各手順におけるデータの状態

1. 初期状態。$\mathrm{Enc}(P_{A_1}, inject)$
   - $inject$は書き込むデータを表すものとする

2. レジスタ内では$C' = \mathrm{Dec}(P_{A_2}, \mathrm{Enc}(P_{A_1}, inject))$という誤復号状態で存在。$P_{A_{2E}}$では$\mathrm{Enc}(P_{A_{2E}}, C')$となる

3. 同上

4. $P_{A_{2E}}$から読み取ったレジスタ内ではC'として存在。P_{A_2}経由で書き込むと $\mathrm{Enc}\left(P_{A_2}, C'\right) = \mathrm{Enc}(P_{A_2}, \mathrm{Dec}(P_{A_2}\mathrm{Enc}\left(P_{A_1}, inject\right)))$ となり、$\mathrm{Enc}(P_{A_2}, x)$と$\mathrm{Dec}(P_{A_2}, x)$がXOR的に相殺され$\mathrm{Enc}(P_{A_1}, inject)$。
   $P_{A_1} = P_{V_1}$なので、V_1として正しい暗号文$\mathrm{Enc}(P_{V_1}, inject)$となる

PCEからのPKの抽出

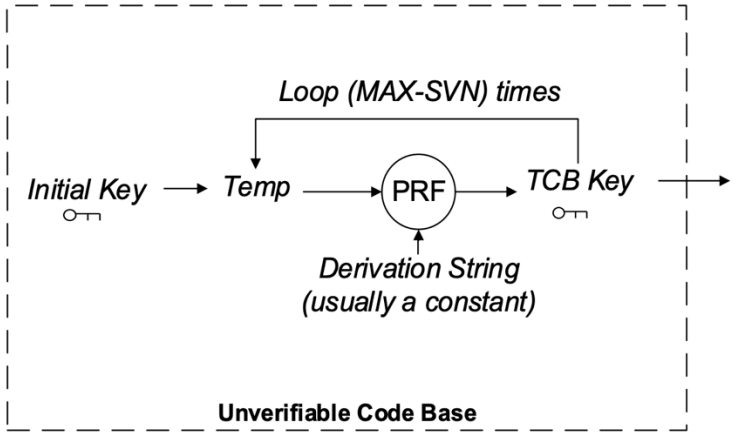
- この任意平文アクセス手法を用いて、DCAP-RAにおけるPCEからPKを抽出する事を試みる

- PKは、AK Certへの署名に用いられるPCKの導出材料として使用されるため、PKが漏洩するとPCKも導出でき、結果としてAKやQuoteも偽造可能となり**DCAP-RAが破綻**する
  - かつ、AKやPCKでなくPKを抽出している例は極めて珍しい

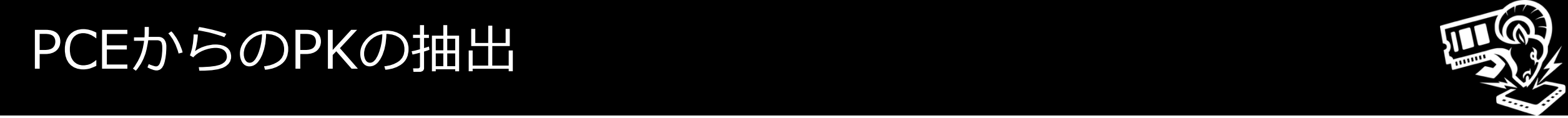
PCEからのPKの抽出

- 漏洩したPKを無効化するには、TCB Recoveryを実施するか、あるいはSGX Factory Resetを使用するしかない

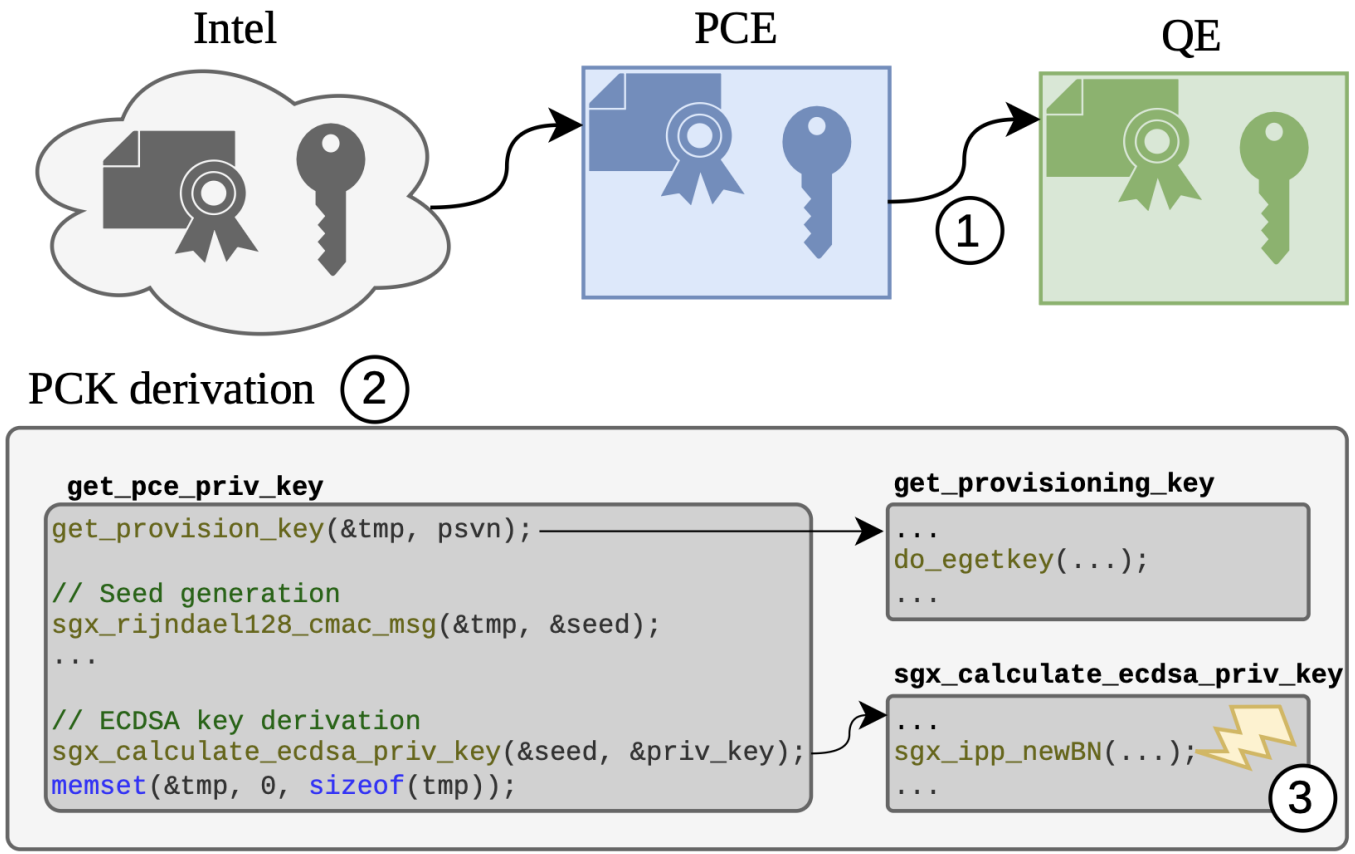
- Factory Resetに関しては、マルチソケット環境であればRPK相当のPlatform Keyの再交渉が行われる。シングルソケット環境でも、以下のDerivation String定数相当が変わり変化する？
  - 図は[9]より引用



PCEからのPKの抽出



- 任意平文アクセス手法を用いてのPKの抽出攻撃の概要を、[1]より引用して以下の図に示す：



PCEからのPKの抽出

- まず、改造out-of-treeドライバを通して、PCEのページをエイリアシングメモリに割り当てる

- ここで、PKは、PCEにおけるPCK導出時にのみスタック上に存在し、その後メモリからはクリアされる。この導出処理は新たなQE3を認証する度に実行される（前図①）

- この鍵導出中の、具体的にはECDSA鍵生成時に使用されるIntel IPPライブラリのヘルパ関数である、**sgx_ipp_newBN関数**で**PCEを中断**させる（前図②、③）
  - 中断には、この関数の載るページのアクセス権限を剥奪しページフォールトを誘発する**Controlled-Channel Attacks**様手法を用いる

PCEからのPKの抽出

- 中断後、攻撃用バッファがPKバッファとエイリアスする位置に攻撃側Enclaveをセットアップし、インターポーザを起動して攻撃バッファ経由でPKを含む暗号文をキャプチャする

- その後インターポーザとPCEを終了し、第2の攻撃側Enclaveを起動し、インターポーザを再度有効化して暗号文を再生すると、**PCEの平文メモリダンプ**が得られる
  - 要は先程説明した任意平文読み取り手法そのまま

- このダンプ内容からスタック内容を抽出し、その中から**PKを抽出**する。PCE攻撃はノイズ無しに数秒以内で決定論的に完了する

PCEからのPKの抽出

- PCEはスタックのランダム化を採用しているが、PKに関しては常にオフセット0x1c_eeaaに位置している
  - PCEでPCK秘密鍵導出のためのAES-CMAC入力として使用される、定数文字列PAK_KEY_DERの位置の特定により決定された
- 抽出したPKの正当性は、さらにこれから導出したPCKによる署名を、正規のPCEが生成した有効な署名と照合する事により確認済みである
- これにより、AK Certの偽造が可能となり、ひいてはQuoteの偽造が可能となるため、**SGXのRAが完全に破綻**する結果となる

AMD SEV-SNPに対する攻撃

SEV-SNPエイリアスチェックの回避

- SEV-SNPでは、BadRAM対策として起動時エイリアスチェックを導入しており、文字通り起動時にエイリアスを検出するとシステムを無効化する
  - これにより、SEV-SNP保護下のCVMの完全性を維持する

- エイリアスチェックの結果は、Attestation Report（AR）内に含まれるPLATFORM_INFOのALIAS_CHECK_COMPLETEフィールドを見る事で確認できる

- しかし前述の通り、起動時にしかチェックしないので、Battering RAMのように**ランタイム中にエイリアスを導入**した場合には、**この軽減策で検知する事はできない**

SEV-SNPエイリアスチェックの回避

- これも前述の通りメモリインターリーブを無効化し、全てのクリティカル要素の割当はインターポーザ未装着側DIMMか、装着側でもA11とA13双方が0な安全な領域に行われるようにした
  - 前述で言う「追い出し」処理である
- また、通常は物理アドレス上部に配置されるプライベートメモリ領域（AMD-SP、SMU、CPU省電力モード関連のCC6が使用）も、BIOS設定経由で追い出しを行った
- そして、偶発的なエイリアシングの回避のため、インターポーザの影響を受ける全メモリ領域を予約した
  - A11またはA13のいずれかがセットされているような領域

SEV-SNPエイリアスチェックの回避

- ランタイム中エイリアスのテストのため、インターポーザ上でA11とA13の同時接地を実施する

- その状態でメモリ領域を手動スキャンし、エイリアシング領域とシステムがクラッシュする領域の双方を特定した
  - A11とA13が、前述の通りそれぞれ物理アドレスのビット28と30に対応していると判明した

- 余談であるが、物理アドレス側視点で見ると、接地させた瞬間に0x50000000分だけズレる結果となる
  - 厳密には、MMIO等が使用するメモリ領域に発生する本来アクセス不能なDRAM位置を再利用するメモリホイスティング機能によるPCIバスオフセットを引いた後の物理アドレスとの差分

SEV-SNP Attestationへの攻撃

- この準備と知見の上で、SEV-SNPのAttestationに対して、BadRAMと同様の攻撃を仕掛ける事を考える
  - かつ、BadRAMと異なり起動時エイリアスチェックでは対策できない

- SEV-SNPのMeasurementは、ハイパーバイザ（HV）が払い出したSEV-SNP FW用のメインメモリ上区画に、**ゲストコンテキスト**内のLD（Launch Digest）エントリとして保存される[10]

- かつ、このゲストコンテキストは、CVMそれぞれのメモリとは異なり、**全ゲストで同一の鍵が使用される**という仕様となっている
  - 使用されている**AES-XEXは決定論的**であるため、SGXのPK攻撃同様、**ゲストコンテキストに対する再生攻撃**が可能となる

SEV-SNP Attestationへの攻撃

- 踏襲元のBadRAMではまず、改変されていない正規のゲストを起動し、エイリアスを通じてゲストコンテキストからLDをキャプチャする

- その後正規VMを終了させ、改変済みVMを、ゲストコンテキストが正規VMのそれの位置と一致するように場所を指定して起動する

- 後は先程キャプチャした正規LDを再生すれば、物理アドレスをTweak値とするAES-XEX復号は正常に通り、Measurementを偽装したARの作成等が可能となってしまう

SEV-SNP Attestationへの攻撃

- この攻撃を、SPD編集ではなくランタイム中インターポーザによるエイリアス（Battering RAM）で再現した所、**攻撃に完全に成功**した
  - 勿論、BadRAM脆弱性に対するパッチ適用済みマシンで実施している

- これにより、**SPDベースのエイリアシングの阻止機構**が、Battering RAMのような**ランタイム中エイリアス攻撃には無力**である事が改めて示された

- こちらも、ノイズ無しで完全に決定論的に攻撃が実行され、その攻撃は数秒以内に完了する

Ciphertext Hidingの迂回の検討

- Cipherleaksのような暗号文を見てのサイドチャネル攻撃が報告されたため、第5世代EPYCプロセッサでは、SEV-SNPにCiphertext Hiding機能を実装している

- これは、SGXにおけるAbort Page Semanticsのように、TEEページに対する不正な読み出し値を固定値にする事で、暗号化状態のメモリ内容も隠蔽するという機能である

- もしこのCiphertext Hidingがメモリをホストとゲストのパーティションに分ける仕組みの場合、Scalable-SGXに対する（inside-in）エイリアス攻撃と同じ手法が通用する可能性がある
  - Ciphertext Hidingの具体的な仕様は不明であるらしい

Ciphertext Hidingの迂回の検討

- ただし、このCiphertext Hidingに対応している第5世代EPYCがDDR5のみに対応しているため、DDR4専門のBattering RAMではその実現可能性の実験はできていない

- ちなみに、次回資料で説明するTEE.failでは、攻撃のアプローチは違えどCiphertext Hidingの突破にも成功している

その他のTEEに対する攻撃

Legacy-SGXに対する攻撃

- Legacy-SGXはMEE（メモリ暗号化エンジン）による非常に強力な暗号学的保護を行うが、その**メモリアクセスパターンの隠蔽までは行わない**

- かつ、MEEはTweak値として書き込みの度に変わるカウンタ値を用いるため、同じ平文を同じEnclave位置に置いても、その暗号文は**書き込みの度に毎回変化**する

- よって、Scalable-SGXのように平文の取得は不可能であるが、反面エイリアスを通じて読み取る事で、暗号文が変化した事の観測は可能である
  - これにより、**メモリアクセスパターンの観測自体は可能**になる

Legacy-SGXに対する攻撃

- よって、Intel1システムの単一のUDIMMにインターポーザを装着した状態で、エイリアシング攻撃を実施した
  - UDIMMなのでパリティ考慮は不要で、単一ビットの接地で十分

- ランダムな位置に書き込みを行うPoC用Enclaveにエイリアスを張って観測した所、観測結果が実際の書き込み位置と一致、つまりは**メモリアクセスパターン観測に成功**した事が判明した

- 論文中ではこれ以上の事は行っていないが、ORAM等の追加の軽減策を導入しない限り、Legacy-SGXに対しても一定のサイドチャネル攻撃が可能となる可能性がある

その他のTEEに対する攻撃

- Intel TDXは、**DDR5マシンでのみ動作可能**であるため、Battering RAMにより攻撃する事はできない

- 加えて、TDXでは**Cryptographic Integrity（Ci）**という、ECCビット内の28bit MACを介した完全性保護機能も有している

- Battering RAMのような**比較的単純なインターポーザ**は、暗号文を含むデータビットの再生は可能だが、CiのMACを含む**ECCビットの再生には利用できない**

その他のTEEに対する攻撃

- データとECCビットの双方の再生を行う事も理論上は可能だが、その実現にはBattering RAMどころではない**本格的なインターポーザ**が必要となってくる

- これは、**手頃なコストでインターポーザ攻撃を実現**する事を目的としている**Battering RAMの流儀に反する**

- よって、少なくとも積極的インターポーザ攻撃に関しては、まだまだ2025年の技術で手頃に行う事はなかなか難しいようである
  - 受動的インターポーザ攻撃であるTEE.failでも、TDからの直接的な秘密抽出は行っていないが、それが不可能性を意味するのかは不明

その他のTEEに対する攻撃

- Arm CCAも、暗号方式としてScalable TEEの例に漏れず、アドレスベースのTweak値を伴うQARMAやAES-XEXを使用するとされている
  - ちなみに、Ciphertext Hiding相当の機能（Granule Protection Check）はデフォルトで存在するらしい
- よって、Scalable-SGXやSEV-SNP同様にBattering RAMに弱い可能性が多分にあり得る
- しかし、論文執筆時点でCCAを利用可能な実機が出ていないため、確認はできていない
  - QEMUは存在するが、これはメモリ暗号化機能は省かれている

その他のTEEに対する攻撃

- Arm CCAも、暗号方式としてScalable TEEの例に漏れず、アドレスベースのTweak値を伴うQARMAやAES-XEXを使用するとされている
  - ちなみに、Ciphertext Hiding相当の機能（Granule Protection Check）はデフォルトで存在するらしい
- よって、Scalable-SGXやSEV-SNP同様にBattering RAMに弱い可能性が多分にあり得る
- しかし、論文執筆時点でCCAを利用可能な実機が出ていないため、確認はできていない
  - QEMUは存在するが、これはメモリ暗号化機能は省かれている

議論と軽減策

DDR5に対するBattering RAM

- DDR5では、以下の図（[1]より引用）のように、CAバスの幅がDDR4の26bitから14bitに縮小され、コマンドは2つの連続したサイクルで多重化されるようになった

CA

	$\overline{CS}$	0	1	2	3	4	5	6	7	8	9	10	11	12	13
ACT	L	L	L	Row 0–3				BA0–1		BG0–2			CID		
	H	Row 4–16													R17
RD	L	H	L	H	H	H	$\overline{BL}$	BA0–1		BG0–2			CID		
	H	V	Column 3–10							V	$\overline{AP}$	V	V	CID	
WR	L	H	L	H	L	H	$\overline{BL}$	BA0–1		BG0–2			CID		
	H	V	Column 3–10							V	$\overline{AP}$	$\overline{WP}$	V	CID	

(a) UDIMM.

CA

	CK	$\overline{CS}$	0	1	2	3	4	5	6
ACT	L	L	L	L	Row 0–3				BA0
	H	L	BA1	BG0–2			CID		
	L	H	Row 4–10						
	H	H	Row 11–16						R17
RD	L	L	H	L	H	H	H	$\overline{BL}$	BA0
	H	L	BA1	BG0–2			CID		
	L	H	V	Column 3–8					
	H	H	C9–10	V	$\overline{AP}$	V	V	CID	
WR	L	L	H	L	H	L	H	$\overline{BL}$	BA0
	H	L	BA1	BG0–2			CID		
	L	H	V	Column 3–8					
	H	H	C9–10	V	$\overline{AP}$	$\overline{WP}$	V	CID	

(b) RDIMM.

DDR5に対するBattering RAM

- この多重化により操作が複雑化し、インターポーザで1つスイッチすると2つのビットに影響を与えてしまう
- RDIMMの場合はさらに悪く、RCDの搭載によりCA幅がさらに半分の7bitにまで小さくなってしまう
  - その場合、送信に4データフェーズが必要となる
- すると、単一の接地で4ビットに影響を与える事になり、これを駆使しながら目当てのエイリアシングを導入するのは至難の業となってしまう
  - かつ、RCDパリティを無効化できない場合は複数（2つの）スイッチングが必要となり、この場合8ビットに影響が及び、もはや混沌である

Battering RAMの軽減策

- エイリアシングには攻撃者による緻密なメモリ割り当て制御が必須であるため、**この能力を低下**させるような何らかの技術を導入するという選択肢は挙げられる

- また、前ページのDDR5の議論のように、**CA幅を減らし多重化**させる事で、Battering RAMのような単純なインターポーザでの攻撃は著しく困難となる

- その他、**統合メモリ**（例：ユニファイドメモリ）では2.5Dまたは3D積層技術を用いている事も多く、攻撃者が物理的にアクセスできなくなる
  - 柔軟性に欠け、コストも高くなりがちで、容量にも制限があるのが欠点

Battering RAMの軽減策

- 最も大きな根本原因は、Scalable TEEが**アドレスベースの決定論的暗号**化をメモリ暗号化に用いている事である

- よって、**Legacy-SGX時代のMEE**同様、カウンタ値のTweak値への利用やHWベースのメモリ完全性ツリーを導入すれば、このような攻撃は封殺する事ができる
  - が、現代の技術では反面**致命的なオーバーヘッドに繋がる**ため難しい

まとめ

- Scalable TEEに対する物理攻撃を整理し、その内の1つの強力な攻撃であるBattering RAMについて詳細に解説した

謝辞

- 本攻撃の調査は、株式会社Datachainによる金銭的協力の下実施したものです。この場を借りて感謝を申し上げます。

参考文献

[1]”Battering RAM: Low-Cost Interposer Attacks on Confidential Computing via Dynamic Memory Aliasing”, Jesse De Meulemeester et al., https://batteringram.eu/batteringram.pdf

[2]”TEE.fail: Breaking Trusted Execution Environments via DDR5 Memory Bus Interposition”, Jalen Chuang et al., https://tee.fail/files/paper.pdf