ウェイ分割を導入した疑似フルアソシアティブキャッシュのセキュリティ評価

多賀 陽一朗[1]　穐山 空道[1,a)]

概要：マルチコアプロセッサでは、複数のコアがキャッシュを共有する構成が一般的である。この構成によりプロセス間でキャッシュ状態が干渉し、アクセス時間の差から他プロセスの動作を推測できるサイドチャネル攻撃が知られている。この問題への対策として、キャッシュラインの追い出し候補をキャッシュ全体から確率的に選択し、攻撃者がエビクションセットを構築できないようにする手法が提案されている。一方で、キャッシュ全体を共有する構成では、依然として複数プロセス間でキャッシュ空間を競合的に利用するため、性能や公平性の面で課題が残る。そこでウェイを分割することによってプロセス間の競合を抑制する。しかし、ウェイを分割することで安全性がどの程度維持されるかは不明である。本研究では、この安全性への影響を理論モデルを用いて分析する。

1. はじめに

一般的な CPU では、処理性能を高めるために、複数のコアが LLC (Last-Level Cache) を共有する構成が採用されている。この構造により、異なるプロセス間でキャッシュの状態が干渉し、アクセス時間の変化から他プロセスの動作を推測できる可能性が生じる。その結果、Prime+Probe 攻撃 [1] などのサイドチャネル攻撃が成立し、秘匿情報が漏洩するリスクが存在する。

こうした問題に対処するため、近年の研究では、フルアソシアティブキャッシュに類似したランダムな追い出し動作を導入する手法が提案されている [2]。具体的には、キャッシュラインの追い出し対象をキャッシュ全体からランダムに選択する。これにより、同じキャッシュセットにマップされる最小のアドレス集合であるエビクションセットを攻撃者が構築することが困難である。

2. 前提知識

2.1 Mirage

Mirage [2] は、フルアソシアティブキャッシュに近い動作をすることでサイドチャネル攻撃への耐性を高めたキャッシュである。従来のセットアソシアティブキャッシュと同様に、アクセスされたアドレスをもとにキャッシュ内を探索し、該当するタグが一致した場合には、そのキャッシュラインからデータを返す。一方で、キャッシュミスが発生した際には、キャッシュ全体からランダムに追い出し対象のキャッシュラインを選択することで、攻撃者がエビクションセットを構築することを困難にしている。

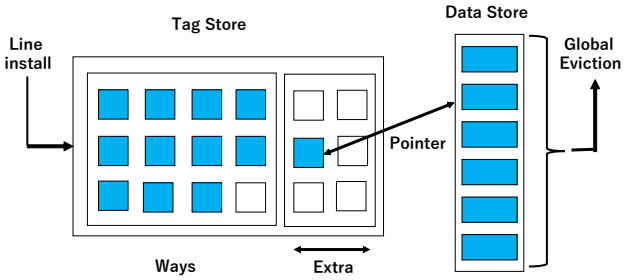

図 1: Mirage キャッシュの概要図

図 1 は Mirage キャッシュの概要を表す。左側の四角形群はタグを格納する Tag Store、右側の四角形群は実際のデータを格納する Data Store である。各タグとデータは Pointer によって対応づけられ、これらが 1 つのキャッシュラインを構成する。青色のマスは有効なキャッシュエントリを、白色のマスは未使用のキャッシュエントリを示す。また、Tag Store の右端にある Extra は、追加で確保されたエントリ群を表す。Global Eviction は新しいラインを挿入する際、キャッシュ全体の Data Store からランダムに 1 つを選択して追い出す操作を示す。

2.2 ウェイ分割

ウェイ分割とは、キャッシュを複数のプロセス間で分離

[1] 立命館大学 情報理工学部
[a)] s-akym@fc.ritsumei.ac.jp

し、各ウェイを特定のプロセス群に割り当てる手法である。これにより、複数のプロセス群間で同じウェイを取り合い性能を劣化し合う現象が解消できる。代表的な実装として Intel Cache Allocation Technology (CAT) [3] がある。Intel CAT では、1 つのプロセスに対して LLC の一部のウェイを割り当て、他のプロセスがその領域を使用できないように制御する。

3. Mirage へのウェイ分割の統合

3.1 統合時の課題

本研究では、ウェイ分割を Mirage に適用することを想定する。すなわち、プロセス単位で独立した領域を持たせ、その内部のみでランダムに追い出しが行われる構成を考える。Mirage は全プロセスがキャッシュ全体を共有する構造のため、複数プロセスによる競合から性能の低下が発生する可能性がある。そこでウェイ分割を適用することでこれを抑制できる。

しかしウェイ分割を Mirage に適用すると安全性が低下が懸念される。ウェイ分割では疑似的にウェイ数が削減された状態になるため、Tag Store の Extra によるエビクションセット作成の困難性が低下することが予想される。

3.2 安全性評価

ウェイ分割を統合した Mirage の安全性を、Mirage の挙動を再現する理論モデルを用いて分析する。具体的には、キャッシュの動作を Bucket と Ball で表現する Buckets-and-Balls モデルを利用する。図 2 は、Buckets-and-Balls モデルの概念図を示す。Bucket はキャッシュセット、Ball はそのセットに格納されるキャッシュラインに対応する。Insert Ball は Ball を Bucket に格納する操作であり、Remove Ball は 全ての Bucket に格納されている Ball の中からランダムに一つ追い出す操作である。これにより Mirage におけるキャッシュ全体からのランダムな追い出し択（Global Eviction）を再現している。

分析手順としては、まずプロセスごとに独立した複数の Bucket を割り当て、各 Bucket の集合内で Mirage と同様のランダムに追い出しを行う。次に、アクセスによって既存のキャッシュラインが追い出される確率であるスピル確率を評価指標として用いる。スピル確率を分析することで、Global Eviction とは異なる形で、キャッシュラインの追い出しがどの程度発生するかを把握できる。すなわち、キャッシュ状態の変化を攻撃者が観測することで、意図的に特定データを追い出す操作を行えるかどうかを検証できる。本研究では、このスピル確率の分布をオリジナルの Mirage と比較し、ウェイ分割による Mirage の安全性の変化を明らかにする。

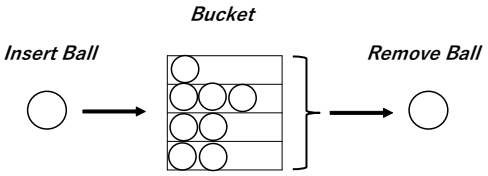

図 2: Buckets-and-Balls モデルの概念図

4. 議論

4.1 関連研究

Mirage の安全性に関する議論は文献 [4] で行われている。この研究では、攻撃者が大量のアクセスを行うことで被害者のラインを確率的に排除できる cache occupancy attack に対しては Mirage が依然として脆弱であることが指摘されている。本研究と文献 [4] の違いは、cache occupancy attack と本研究が想定するサイドチャネル攻撃の全体の違いである。Cache occupancy attack は攻撃者と被攻撃者が同じ同じキャッシュラインを共有する場合（例えば共有ライブラリのコード部など）を想定し、本研究が対象とするサイドチャネル攻撃はそのような想定がない。

4.2 課題と展望

本研究により安全性の低下が確認された場合には改善手法を考案する。具体的にはウェイ分割における比率を変化させた場合に安全性がどの程度維持されるかを定量的に評価し、安全性と性能の両立が可能な構成を明らかにする。

謝辞 本研究は、JSPS 科研費 JP23K28078 の支援を受けたものである。

参考文献

[1] Liu, F., Yarom, Y., Ge, Q., Heiser, G. and Lee, R. B.: Last-Level Cache Side-Channel Attacks Are Practical, *IEEE Symposium on Security and Privacy (S&P)*, pp. 605–622 (2015).

[2] Saileshwar, G. and Qureshi, M.: MIRAGE: Mitigating Conflict-Based Cache Attacks with a Practical Fully-Associative Design, *USENIX Security Symposium*, pp. 1379–1396 (2021).

[3] Intel Corporation: Introduction to Cache Allocation Technology, Intel Corporation (online), available from ⟨https://www.intel.com/content/www/us/en/developer/articles/technical/introduction-to-cache-allocation-technology.html⟩

[4] Li, L., Huang, J., Feng, L. and Wang, Z.: Prefender: A Prefetching Defender Against Cache Side Channel Attacks as a Pretender, *IEEE Transactions on Computers*, pp. 1457–1471 (2024).