



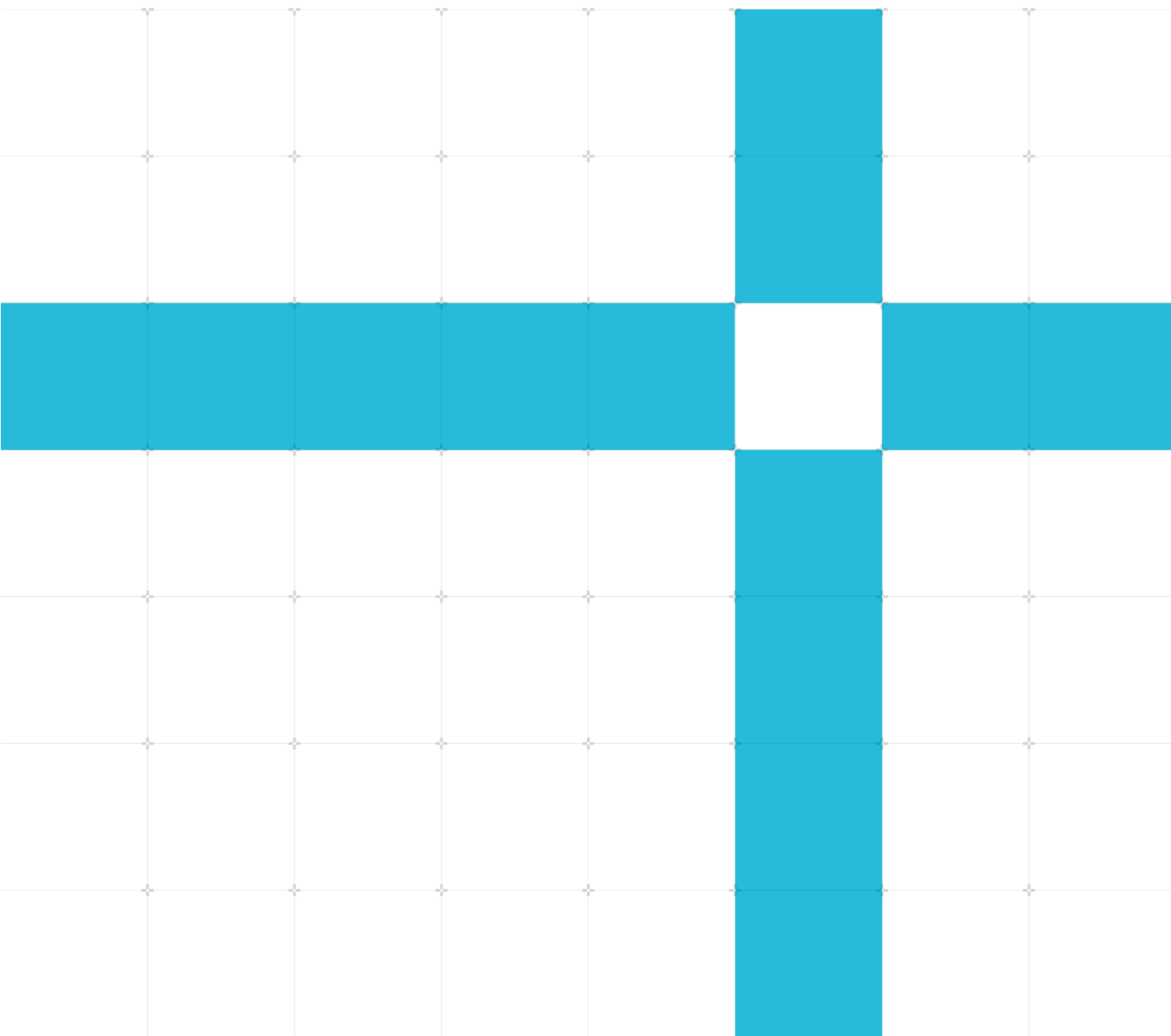
# Architecture Security Advisory ASA

Version: 1.0-r1p2

## Speculative Oracles on Memory Tagging

Non-Confidential

Copyright © 2023 Arm Limited (or its affiliates).  
All rights reserved.



## Architecture Security Advisory

### Speculative Oracles on Memory Tagging

Copyright © 2023 Arm Limited (or its affiliates). All rights reserved.

### Arm Non-Confidential Document Licence (“Licence”)

This Licence is a legal agreement between you and Arm Limited (“**Arm**”) for the use of Arm’s intellectual property (including, without limitation, any copyright) embodied in the document accompanying this Licence (“**Document**”). Arm licenses its intellectual property in the Document to you on condition that you agree to the terms of this Licence. By using or copying the Document you indicate that you agree to be bound by the terms of this Licence.

“**Subsidiary**” means any company the majority of whose voting shares is now or hereafter owner or controlled, directly or indirectly, by you. A company shall be a Subsidiary only for the period during which such control exists.

This Document is **NON-CONFIDENTIAL** and any use by you and your Subsidiaries (“**Licensee**”) is subject to the terms of this Licence between you and Arm.

Subject to the terms and conditions of this Licence, Arm hereby grants to Licensee under the intellectual property in the Document owned or controlled by Arm, a non-exclusive, non-transferable, non-sub-licensable, royalty-free, worldwide licence to:

- (i) use and copy the Document for the purpose of designing and having designed products that comply with the Document;
- (ii) manufacture and have manufactured products which have been created under the licence granted in (i) above; and
- (iii) sell, supply and distribute products which have been created under the licence granted in (i) above.

**Licensee hereby agrees that the licences granted above shall not extend to any portion or function of a product that is not itself compliant with part of the Document.**

Except as expressly licensed above, Licensee acquires no right, title or interest in any Arm technology or any intellectual property embodied therein.

THE DOCUMENT IS PROVIDED “AS IS”. ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. Arm may make changes to the Document at any time and without notice. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

NOTWITHSTANDING ANYTHING TO THE CONTRARY CONTAINED IN THIS LICENCE, TO THE FULLEST EXTENT PERMITTED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, IN CONTRACT, TORT OR OTHERWISE, IN CONNECTION WITH THE SUBJECT MATTER OF THIS LICENCE (INCLUDING WITHOUT LIMITATION) (I) LICENSEE’S USE OF THE

DOCUMENT; AND (II) THE IMPLEMENTATION OF THE DOCUMENT IN ANY PRODUCT CREATED BY LICENSEE UNDER THIS LICENCE). THE EXISTENCE OF MORE THAN ONE CLAIM OR SUIT WILL NOT ENLARGE OR EXTEND THE LIMIT. LICENSEE RELEASES ARM FROM ALL OBLIGATIONS, LIABILITY, CLAIMS OR DEMANDS IN EXCESS OF THIS LIMITATION.

This Licence shall remain in force until terminated by Licensee or by Arm. Without prejudice to any of its other rights, if Licensee is in breach of any of the terms and conditions of this Licence then Arm may terminate this Licence immediately upon giving written notice to Licensee. Licensee may terminate this Licence at any time. Upon termination of this Licence by Licensee or by Arm, Licensee shall stop using the Document and destroy all copies of the Document in its possession. Upon termination of this Licence, all terms shall survive except for the licence grants.

Any breach of this Licence by a Subsidiary shall entitle Arm to terminate this Licence as if you were the party in breach. Any termination of this Licence shall be effective in respect of all Subsidiaries. Any rights granted to any Subsidiary hereunder shall automatically terminate upon such Subsidiary ceasing to be a Subsidiary.

The Document consists solely of commercial items. Licensee shall be responsible for ensuring that any use, duplication or disclosure of the Document complies fully with any relevant export laws and regulations to assure that the Document or any portion thereof is not exported, directly or indirectly, in violation of such export laws.

This Licence may be translated into other languages for convenience, and Licensee agrees that if there is any conflict between the English version of this Licence and any translation, the terms of the English version of this Licence shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. No licence, express, implied or otherwise, is granted to Licensee under this Licence, to use the Arm trade marks in connection with the Document or any products based thereon. Visit Arm's website at <http://www.arm.com/company/policies/trademarks> for more information about Arm's trademarks.

The validity, construction and performance of this Licence shall be governed by English Law.

Copyright © 2023 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-21585 version 4.0

## Web Address

<http://www.arm.com>

## Contact

[psirt@arm.com](mailto:psirt@arm.com)

# Contents

**1 Introduction ..... 5**

**2 How are MTE speculative oracles introduced? ..... 6**

**3 What is the impact of MTE speculative oracles? ..... 7**

3.1 With tag control ..... 7

3.2 Without tag control..... 7

3.3 Stores..... 7

**4 Are Arm cores affected? ..... 8**

**5 Conclusions ..... 9**

**6 References..... 10**

# 1 Introduction

As the result of PACMAN [1], the security analysis of speculative oracles raised concerns about their impact on Arm MTE.

MTE stands for Memory Tagging Extension [1], and it implements a lock and key based access to memory. Allocation Tags (or locks) of 4 bits can be set on every 16-bytes of memory, and accesses to locked locations are only allowed when the address includes a matching Address Tag (or key).

Arm MTE can be used to detect memory safety violations and potentially increase robustness against some attacks. For example, Arm MTE provides probabilistic security guarantees [2] on limited scenarios, but it does not stop an interactive adversary that is able to brute force, leak, or craft arbitrary address tags.

Based on the analysis conducted by Arm, the key observation is that depending on the behaviour of a Tag-Check fault under speculation, some implementations might create an oracle that enables an adversary to brute force memory tags and thus perform deterministic attacks without ever faulting.

Arm MTE is designed to be used as a debugging aid to find memory safety issues that may become exploitable security vulnerabilities (i.e., as a bug detection tool). Although it can be used to hinder exploitation, it is not designed to be a full solution against active adversaries. Therefore, Allocation Tags are not expected to be a secret, and a speculative mechanism that reveals the correct tag value is not considered a compromise of the principles of the architecture.

Besides the debugging capabilities of Arm MTE, it can provide a first line of defense against specific classes of exploits. In that sense, the purpose of this advisory is to provide guidance and clarification on this matter.

## 2 How are MTE speculative oracles introduced?

Speculative oracles are code snippets whose side-effects provide a response to a query (e.g., will an access to this address trigger a fault?) when executed speculatively. Since speculation does not affect the architectural state of a program, it is possible to repeatedly execute them arbitrarily many times.

According to FEAT\_CSV3, the architectural rule against Meltdown-style issues, “Data loaded under speculation with a permission or domain fault cannot be used to” do anything that exposes its value via a side channel. However, for performance reasons, the architecture does not restrict data loaded under speculation with a Tag-Check fault. This document provides a summary of the potential risks that different implementations involve.

If data speculatively loaded under Tag-Check fault is restricted, which in practice often translates to stalling (or returning zeroes to) younger instructions, it can create an observable difference compared to a successful Tag-Check, as the side-effects of younger instructions will differ.

```
B.XX label
LDR X1, [X0, X1] // tag check. X1 adversary controlled
...
LDR X2, [X1] // fetch depends on tag success
Label:
```

**Figure 1. Example of speculative oracle gadget for Arm MTE.**

Figure 1 shows an example of speculative oracle gadget. Although similar to Spectre v1 gadgets, the requirements are weaker: the adversary does not need to control offsets or encode the data to leak, as only a single binary channel is required.

An implementation could even, at a greater cost, serialize the Tag-Check before the data load, which would make oracles more pervasive as the gadget would not require a second dependent memory access.

Note that to eliminate the risk of an oracle, it is not enough to wait until the Tag-Check is executed, i.e., to avoid speculating past the Tag-Check. Instead, the CPU would need to forbid the forwarding of any data, regardless of the Tag-Check outcome, until the load is not under any speculative shadow (e.g., under a speculative branch anymore). This makes it notably difficult and expensive to implement in practice.

On implementations where faulting Tag-Checks do not affect speculative execution in any way, but only yield a fault when retired, there are no observable differences that the adversary can leverage until the fault occurs.

However, this alternative creates another threat: memory tagging protection is effectively disabled during speculation, and thus Spectre attacks (potentially assisted by memory corruption vulnerabilities) are still possible. Specifically, the risks of speculative out-of-bound reads, type-confusion, or memory corruption (due to store-to-load forwarding) are not reduced by Arm MTE.

## 3 What is the impact of MTE speculative oracles?

### 3.1 With tag control

An adversary that can trigger the execution of a speculative oracle gadget (e.g., by calling a function with controlled arguments) could recover the allocation tags for any arbitrary location. This can be done by trying different tags until the success signal is observed, at that point the adversary could architecturally trigger the actual out-of-bounds read or write with a valid tag and circumvent Arm MTE.

On a system with TCR\_ELx.TCMAx set to 1, if the adversary controls the Logical Address Tag, e.g., by corrupting a pointer via type-confusion, or by controlling a large enough offset that is added to the pointer overflowing the MSBs, it is possible to force Tag Unchecked accesses. For ELO applications, the expectation is that TCMA0 is set to 0. However, in the Linux kernel TCMA1 is set to 1 and using address tag 0b1111 generates Tag Unchecked accesses, which makes speculative oracles unnecessary in most cases.

### 3.2 Without tag control

Similarly, the entire exploit could happen under speculation, and unless the right tag was used no signal would be observed. This could turn probabilistic attacks into deterministic ones. For example, in a use-after-free the adversary gets a dangling pointer with a fixed/unknown Address Tag. Arm MTE provides a probabilistic protection<sup>1</sup> as any new allocation will have a random tag, causing the dereference of the dangling pointer to fault in most cases. However, by being able to speculatively probe whether the tag matches or not, the adversary could free and reallocate objects in the same location until a tag match was detected, and only then continue the exploitation.

Note that the goal here is not to leak the Allocation Tag, and there is no need to control the Address Tag. Instead, the adversary repeats until success abusing the probabilistic nature of the protection itself, which relies on generating random tags for each allocation.

We highlight that this overall approach is not new, and researchers have proposed similar techniques in the past to brute force other security mechanisms such as ASLR or stack canaries [3, 4].

### 3.3 Stores

The question about stores is relevant when considering the gadgets that can serve as an oracle, and on asymmetric systems where Tag-Check faults occur only due to stores but not loads.

The answer depends on several microarchitectural details, but as rule of thumb, if the faulting store does not cause an early (i.e., before the fault is triggered on retirement) pipeline flush, and store-to-load forwarding is permitted, then there is no oracle.

However, as with the load, if the Tag-Check and the memory fetch are serialized, then it is possible to observe the side-effect in the cache, as the cache line fill triggered by the store would only occur conditionally.

---

<sup>1</sup> If a freed object has a “special” Allocation Tag, one could provide deterministic protection against metadata corruption.

## 4 Are Arm cores affected?

Arm is aware of an MTE speculative oracle affecting some Arm cores [6,7,8]. For example, there exists a minor timing difference between the execution path of a successful and failed Tag-Check in the Cortex-X2 and Cortex-X3 processors that can be enough to create an MTE speculative oracle.

Following this investigation, Arm has also noticed that measurable differences can also occur in other cores like the Cortex-A510, Cortex-A520, Cortex-A710, Cortex-A715 and Cortex-A720 to introduce a speculative oracle. However, due to their in-order nature and short pipelines, the risk of exploitation in these cores is considerably lower.

**Arm does not consider the risk of speculative oracles a detriment to the value offered by Arm MTE.** For performance reasons, speculative access past Tag-Check fault is allowed, and even on systems without speculative oracles it would be possible to use speculative reads to leak valid tags from pointers stored in memory, which involves a similar risk.

However, the cost of preventing speculative oracles in hardware is low and Arm recommends partners to consider implementing relevant controls.



## 5 Conclusions

Pointer Authentication is a security feature, and a valid Pointer Authentication Code (PAC) value is expected to be secret. PACMAN [1] violates this assumption and in result it can hinder the effectiveness of the protection.

In contrast, Arm MTE Allocation Tags are not expected to be a secret. Therefore, a mechanism that reveals the correct tag value is not a compromise of the principles of the architecture.

On affected systems, the risk of a speculative oracle would only impact the probabilistic security guarantees of the MTE. In general, the risk of an oracle is null if the adversary gains nothing by repeating the attack many times.

## 6 References

1. "PACMAN security vulnerability" <https://developer.arm.com/documentation/ka005109/latest>
2. "Armv8.5-A Memory Tagging Extension White Paper" <https://developer.arm.com/documentation/102925/0100/>
3. "Security Analysis of Memory Tagging" <https://github.com/microsoft/MSRC-Security-Research/blob/master/papers/2020/Security%20analysis%20of%20memory%20tagging.pdf>
4. "Speculative Probing: Hacking Blind in the Spectre Era" [https://download.vusec.net/papers/blindside\\_ccs20.pdf](https://download.vusec.net/papers/blindside_ccs20.pdf)
5. "Bypassing memory safety mechanisms through speculative control flow hijacks" <https://arxiv.org/pdf/2003.05503.pdf>
6. "Sticky Tags: Efficient and Deterministic Spatial Memory Error Mitigation using Persistent Memory Tags". Floris Gorter and Cristiano Giuffrida from the VUsec group at VU Amsterdam. 2023.
7. "MTE As Implemented, Part 1: Implementation Testing" <https://googleprojectzero.blogspot.com/2023/08/mte-as-implemented-part-1.html>
8. "Speculative MTE Tag Leakage". Juhee Kim, Youngjoo Lee, Sihyeon Roh, Byoungyoung Lee, at Seoul National University. 2023