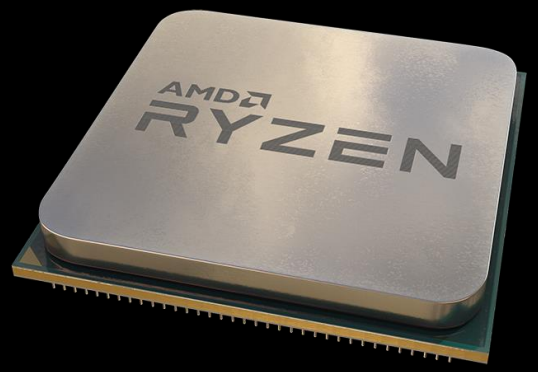
Secure Buffer Support with Trusted Memory Zone

Huang Rui

AGENDA



- Background
- Trusted Memory Zone
- Hardware Platform Support
- Secure Buffer Object
- Kernel and Mesa Secure Buffer Definition and Handshake
- Secure Policy
- Write/Read Operation Modulation Table
- Per-IB Protection Mechanism
- Command Submission with Secure buffer – GFX/SDMA/VCN
- Display Secure Frame Buffer
- Security Unit Test Suite and Related Parameters
- Data Comparison between Un-encryption and Encryption
- References
- The Best is Yet to Come

Background

- Memory encryption is an important feature which protects the content cannot be accessed by an unauthorized application.
- AMD GPU developed the TMZ (Trusted Memory Zone) to support video memory and system memory encryptions.
- Linux® Kernel, Mesa, and user space applications are leveraging the functionality of TMZ to implement the secure buffer support that can be used in a multitude of other scenarios on Linux®.

Trusted Memory Zone

- The Trusted Memory Zone (TMZ) is a method to protect the contents being written and read from memory.

- AMD GPU platform supports full memory bandwidth AES cipher.
  - All TMZ reads and TMZ writes go through the AES cipher before hitting memory.
  - TMZ bit in page table and trusted transaction provide multiple protection.
    - Even if the TMZ bit mis-configured, the content is still in safe and not exposure.

Hardware Platform Support

- Supported Hardware Platforms:
  - Discrete GPU platform (video memory can be encrypted).
  - APU (CPU + GPU) platform (video memory and system memory can be encrypted).

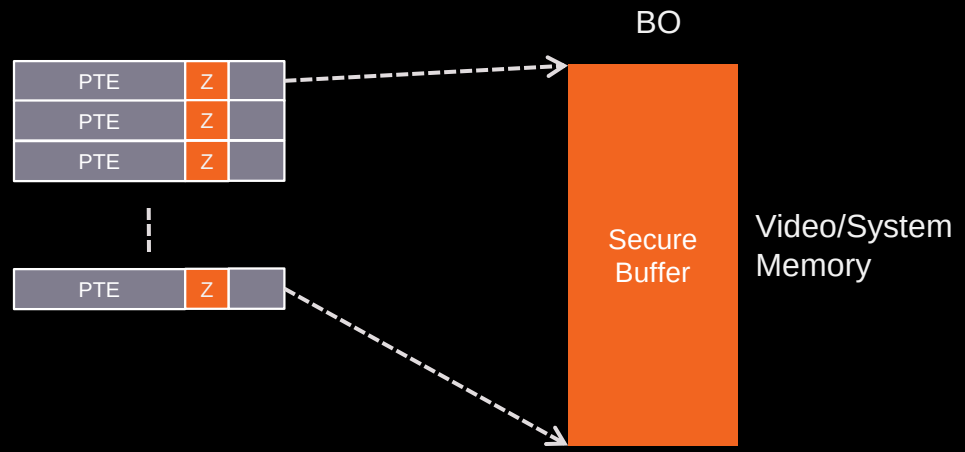
- Note: we have only enabled APUs support on Linux® at this moment.

Secure Buffer Object

- Secure flag in GPUVM page table entry format:

  - V – Valid
  - S – System
  - C – Cacheable
  - X – eXecute
  - R – Read
  - W – Write

- AMDGPU kernel implements a per-BO based protection.
  - BO is the basic memory management unit.
  - All TMZ bits are set in the PTEs if it is a secure buffer.



Kernel and Mesa Secure Buffer Definition and Handshake

- Initial a BO-level protection in kernel driver to provide a new flag AMDGPU_GEM_CREATE_ENCRYPTED to gem create ioctl to libdrm for the secure buffer allocation.
  - Mesa uses allocate the buffer to decide whether is secure or not.
  - If AMDGPU_GEM_CREATE_ENCRYPTED is set by Mesa, the TMZ bit of whole page table entries for this buffer must be set.
- Provide the AMDGPU_IB_FLAGS_SECURE to indicate the IB (indirect buffer which stores the GPU commands) is trusted or not.
  - Mesa uses this flag in IB handle of command submission context to inform kernel whether the graphic engine needs to translate to trusted state.
- Provide the AMDGPU_IDS_FLAGS_TMZ to indicate the TMZ capability is integrated or not.
  - Mesa uses this information flag to be aware whether the current gpu supports the TMZ.

Secure Policy

- CPU as the un-trusted domain is unable to decrypt the secure buffer with TMZ.
  - User **CANNOT** get the raw data from CPU address even is k-mapped.
- Only trusted hardware block has the capability to decrypt the secure buffer.
  - All encrypted memory is only able to be decrypted with GPUVM mapped.
- Trusted Hardware Blocks
  - GFX
  - SDMA
  - Video Codec Next (VCN) / Joint Photographic Experts Group (JEPG)
  - Display Engine

Write Operation Modulation Table

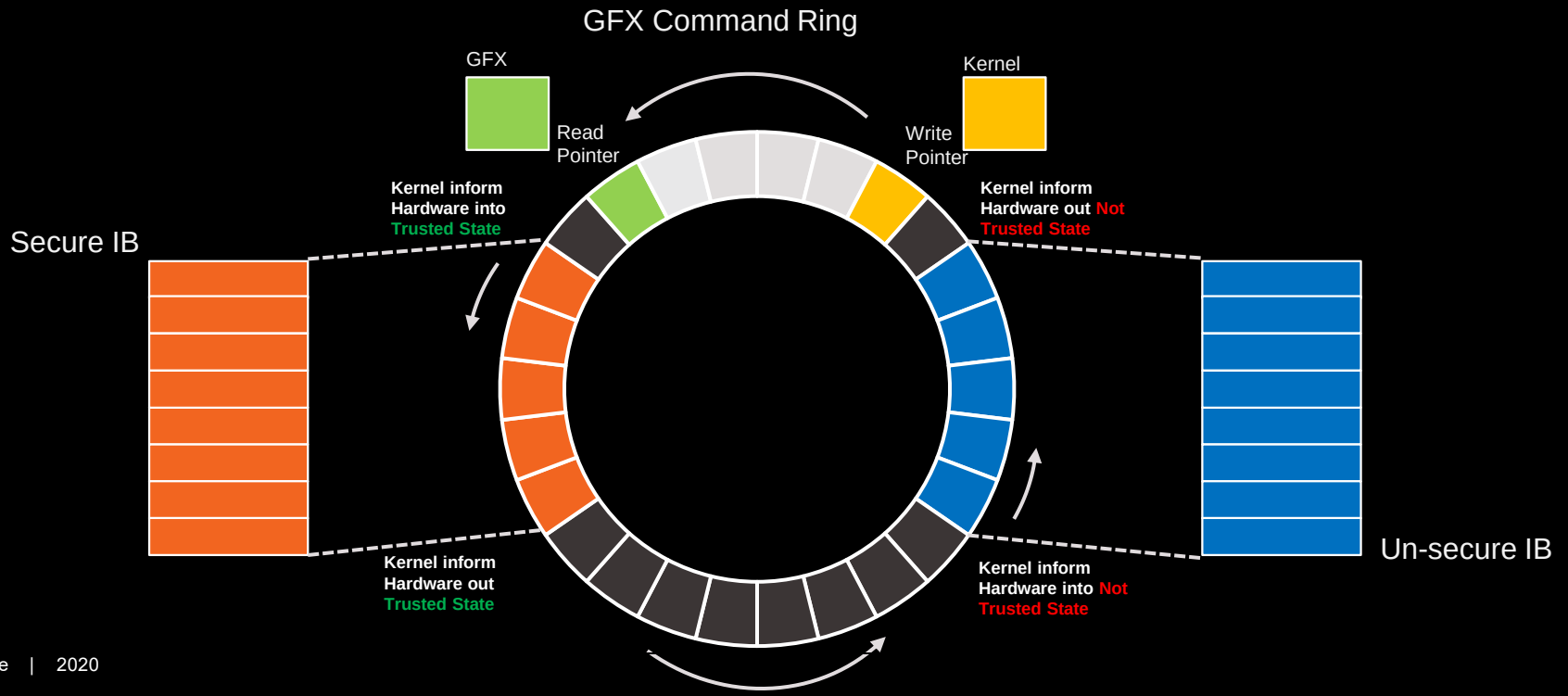
Write Operation			
Transaction Trust State (0:Not trusted, 1:Trusted)	**TMZ bit in GPU table**	**Modulation Result**	**Encryption State in TMZ**
0	0	Not Encrypted	OFF
0	1	Encrypted	ON
1	0	Encrypted	ON
1	1	Encrypted	ON

Read Operation Modulation Table

Read Operation			
Transaction Trust State (0:Not trusted, 1:Trusted)	**TMZ bit in GPU table**	**Modulation Result**	**Decryption State in TMZ**
0	0	Not Decrypted	OFF
1	1	Not Decrypted	OFF
1	0	Not Decrypted	OFF
1	1	Decrypted	ON

Per-IB Protection Mechanism

- Per-IB protection submission
  - IB (Indirect Buffer) is the command buffer which stores the packets which GFX queue executes.
  - Kernel implements a Per-IB protection to execute the command submission with secure buffer.
    - When an IB with unsecure buffer is emitted, kernel must set **Transaction Trust State** as **Not Trusted**. (legacy case)
    - When an IB with secure buffer is emitted, kernel must set **Transaction Trust State** as **Trusted.**



Command Submission with Secure Buffer - GFX/SDMA/VCN

- GFX uses FRAME_TMZ bit with PACKET3_FRAME_CONTROL packet to control GFX engine into trusted state.
- SDMA as another trusted hardware block which is to use TMZ flag in page table to decide to do regular read/write or trusted read/write.
  - While kernel uses SDMA to do secure buffer copy, kernel needs to set TMZ bit at opcode of COPY_LINEAR packet.
- VCN engine doesn’t need kernel support to switch the trust level during context switch, it’s able to switch by itself according to TMZ flag in the page table.

Display Secure Frame Buffer

- Display engine secure state based on register setting. (Different than GFX/SDMA/VCN)
  - Display driver only programs a bit in the register to switch the secure state of display engine instead of using the TMZ bit of page table. (If the bit is mis-programmed, there is no valid data can be gained)
  - This allows secure access to display buffers in VRAM without using page tables.

Security Unit Test Suite and Related Parameters

- Initial the security test suite in libdrm amdgpu_test:

```
Suite: 11:  ENABLED: Security Tests
 Test:  1:  ENABLED: allocate secure buffer test
 Test:  2:  ENABLED: graphics secure command submission
 Test:  3:  ENABLED: sDMA secure command submission
 Test:  4:  ENABLED: secure bounce
```

- Kernel parameter to enable TMZ: amdgpu.tmz=1
- Mesa parameter to enable TMZ: AMD_DEBUG=tmz

Data Comparison between Un-encryption and Encryption

- Raw Data (0xdeadbeaf):

```
[ 0] 0xdeadbeaf [ 1] 0xdeadbeaf [ 2] 0xdeadbeaf [ 3] 0xdeadbeaf [ 4] 0xdeadbeaf [ 5] 0xdeadbeaf [ 6] 0xdeadbeaf [ 7] 0xdeadbeaf
[ 8] 0xdeadbeaf [ 9] 0xdeadbeaf [10] 0xdeadbeaf [11] 0xdeadbeaf [12] 0xdeadbeaf [13] 0xdeadbeaf [14] 0xdeadbeaf [15] 0xdeadbeaf
[16] 0xdeadbeaf [17] 0xdeadbeaf [18] 0xdeadbeaf [19] 0xdeadbeaf [20] 0xdeadbeaf [21] 0xdeadbeaf [22] 0xdeadbeaf [23] 0xdeadbeaf
[24] 0xdeadbeaf [25] 0xdeadbeaf [26] 0xdeadbeaf [27] 0xdeadbeaf [28] 0xdeadbeaf [29] 0xdeadbeaf [30] 0xdeadbeaf [31] 0xdeadbeaf
[32] 0xdeadbeaf [33] 0xdeadbeaf [34] 0xdeadbeaf [35] 0xdeadbeaf [36] 0xdeadbeaf [37] 0xdeadbeaf [38] 0xdeadbeaf [39] 0xdeadbeaf
[40] 0xdeadbeaf [41] 0xdeadbeaf [42] 0xdeadbeaf [43] 0xdeadbeaf [44] 0xdeadbeaf [45] 0xdeadbeaf [46] 0xdeadbeaf [47] 0xdeadbeaf
[48] 0xdeadbeaf [49] 0xdeadbeaf [50] 0xdeadbeaf [51] 0xdeadbeaf [52] 0xdeadbeaf [53] 0xdeadbeaf [54] 0xdeadbeaf [55] 0xdeadbeaf
[56] 0xdeadbeaf [57] 0xdeadbeaf [58] 0xdeadbeaf [59] 0xdeadbeaf [60] 0xdeadbeaf [61] 0xdeadbeaf [62] 0xdeadbeaf [63] 0xdeadbeaf
```

- Encrypted Data:

```
[ 0] 0x2bbd6dcd [ 1] 0x7c2b6e33 [ 2] 0x1dfba9e6 [ 3] 0x45b179e9 [ 4] 0xf0ef8279 [ 5] 0x533ea813 [ 6] 0x2a3c4384 [ 7] 0x980ca210
[ 8] 0x5708f92a [ 9] 0x37a95473 [10] 0x035de2fa [11] 0x01ab7d13 [12] 0xfa4ea179 [13] 0x7683e18d [14] 0x35d11359 [15] 0xeb05bfce
[16] 0xf1c08505 [17] 0x5438ddae [18] 0x2f57d37a [19] 0xe47493f5 [20] 0x3d61ab28 [21] 0xe0a15218 [22] 0x3bcba25d [23] 0xde6f5b29
[24] 0x57ceb92d [25] 0x5fa03bb3 [26] 0x72a18638 [27] 0x86f15264 [28] 0x4a2f3b9d [29] 0xea7a9ba8 [30] 0xc9073aeb [31] 0x94ae3bcd
[32] 0x6dbb97a2 [33] 0x6730199d [34] 0xfef93fbf [35] 0xfdd402af [36] 0xa5038c5f [37] 0xd894b661 [38] 0x4f10e6f7 [39] 0x81f17a28
[40] 0x02722e9b [41] 0xba839693 [42] 0x0ac6ce35 [43] 0xbf0a6ec7 [44] 0x2768619a [45] 0x4f5d1620 [46] 0x6ad605c7 [47] 0x0a407741
[48] 0xc2d2f5ce [49] 0x97f766d8 [50] 0xec190cfe [51] 0xc5236lad [52] 0x5573d7e0 [53] 0xf292c8c9 [54] 0x0140e657 [55] 0x4bedeb29
[56] 0x8de3a9c1 [57] 0x7e1fa812 [58] 0x8db4367f [59] 0xdeea1081 [60] 0x20b9d21e [61] 0x18f61308 [62] 0x8d2ea57a [63] 0xb14117ad
```

References

- Linux® Kernel:
  - https://lists.freedesktop.org/archives/amd-gfx/2019-September/039928.html
- Libdrm:
  - https://gitlab.freedesktop.org/mesa/drm/-/merge_requests/70
- Mesa:
  - https://gitlab.freedesktop.org/mesa/mesa/-/merge_requests/4401

The Best Is Yet to Come

- Secure Buffer Outcome - Widevine DRM (Linux® and Android)
  - Secure buffer with TMZ is key functionality which is required by Widevine DRM solution.
  - TMZ + HDCP (High-bandwidth Digital Content Protection) can be leveraged by Widevine to setup a safe channel for digital media data owner on AMD platform.
  - We are working on upstreaming a full end-to-end solution.

Thank You and Q&A

Thanks for the contribution:

Alex Deucher <alexander.deucher@amd.com>
Christian König <christian.koenig@amd.com>
Aaron Liu <aaron.liu@amd.com>
Luben Tuikov <luben.tuikov@amd.com>
Pierre-Eric Pelloux-Prayer <pierre-eric.pelloux-prayer@amd.com>

Disclaimer:

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED 'AS IS." AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

AMD, the AMD Arrow logo, Radeon, Ryzen and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Linux is a trademark of Linus Torvalds.


AMD