



  
**black hat**<sup>®</sup>  
EUROPE 2023

DECEMBER 4-7  

---

EXCEL LONDON / UK



# **Evils in the Sparse Texture Memory: Exploit Kernel Based on Undefined Behaviors of Graphic APIs**

Xingyu Jin, Richard Neal, Tony Mendez



Xingyu  
**@1ce0ear**  
**Senior Security  
Engineer**



Richard  
**@ExploitDr0id**  
**Staff Security Engineer**



Tony  
**@amdz23**  
**Technical Program  
Manager**



<https://t.me/learningnets>

# Agenda

- Part 1
  - Android GPU Security Review
  - Graphics Stack and PowerVR Driver
  - More places to find bugs
  - Evils in the Sparse Texture Memory
  - Root Exploit Demonstration
- Part 2
  - Finding vulnerabilities and exploits
  - Android Partner Vulnerability Initiative

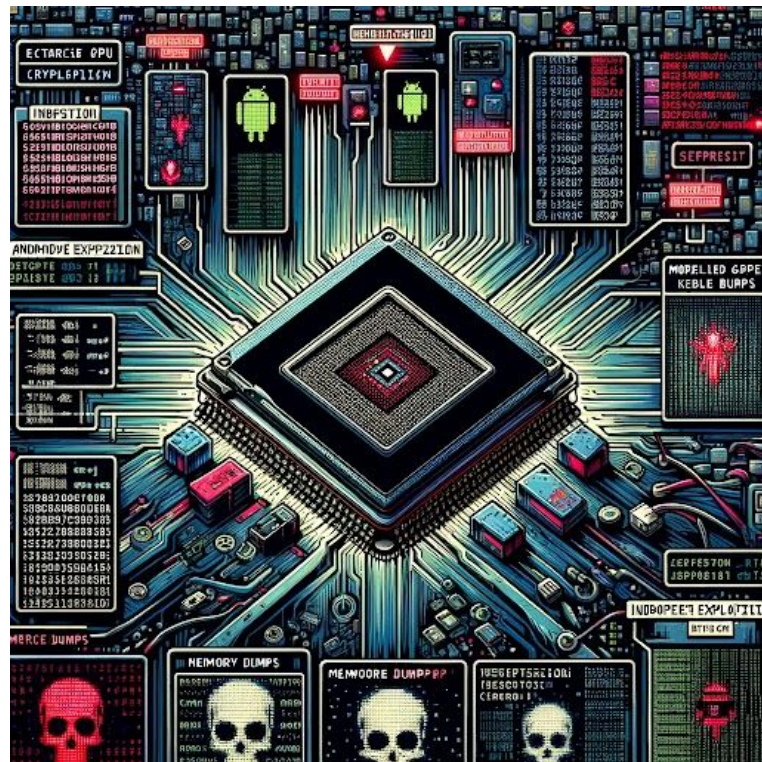


Image by DALL·E

All <https://timehacking.net> talk were already publicly disclosed and patches were available by the affected vendor.

# Android GPU Security: Current State

<https://t.me/learningnets>

# State of GPU Security on Android

- Significant {0,1,n}day attacks targeting GPU drivers

[CVE-2022-22706](#), a vulnerability in Mali GPU Kernel Driver [fixed](#) by ARM in January 2022 and marked as being used for this vulnerability.

## **CVE-2023-4211 Known To Be Under Targeted Attack**

The second zero-day vulnerability, [CVE-2023-4211](#), included within the October security update,

limited, targeted points to there b users.

Qualcomm is warning of three zero-day vulnerabilities in its GPU and Compute DSP drivers that hackers are actively exploiting in attacks.

The American semiconductor company was told by Google's Threat Analysis Group (TAG) and Project Zero teams that [CVE-2023-33106](#), [CVE-2023-33107](#), [CVE-2022-22071](#), and [CVE-2023-33063](#) may be under limited, targeted exploitation.

Qualcomm says it has released security updates that address the issues in its Adreno GPU and Compute DSP drivers, and impacted OEMs were also notified.

# State of GPU Security on Android

- GPU Security is still vastly under-researched
  - Complicated, Proprietary, New features...
- Project Zero blog “Mind the Gap”
- Major Android GPUs:
  - ARM: Mali GPU
  - Qualcomm: Adreno GPU
  - Imagination Technologies: PowerVR GPU  Imagination

# PowerVR GPU by ImgTec

- Apple's former GPU maker
- Popular on budget-friendly phone, tablet & TV
  - *Samsung A12, Redmi 9a/10a, Moto Pure G, Fire TV*

## Apple Replacing PowerVR With In-House GPU

It means future Apple devices will no longer use PowerVR GPUs, ending a long relationship with Imagination Technologies.

By Matthew Humphries April 3, 2017



## Imagination in Mobile

Imagination's PowerVR GPU is the original tile-based deferred rendering architecture, designed to deliver the ultimate in performance density and power efficiency. Our GPU technology paved the way for the smartphone-based mobile gaming revolution. Today, backed by our thriving ecosystem, over **35%** of smartphones feature PowerVR, which continues to deliver everything our customers need as we push the boundaries for graphics and compute.

<https://t.me/learningnets>



# PowerVR GPU by ImgTec

Most shipped smartphone in 2021

Rank	Model Name	Company	Million Units	ASP (\$)
1	Galaxy A12	Samsung	51.8	160
2	iPhone 12	Apple	45.7	400
3	iPhone 11	Apple	34.9	360
4	iPhone 11	Apple	33.8	350
5	Redmi 9A	Xiaomi	26.8	78
6	iPhone 12 Pro Max	Apple	26.1	1,200
7	iPhone 11 Pro Max	Apple	24.1	1,100
8	iPhone 12 Pro	Apple	23.2	1,100
9	iPhone 11 Pro	Apple	22.8	1,100
10	Galaxy A02	Samsung	18.3	138

← PowerVR

← PowerVR

← PowerVR

Source: Omdia Smartphone Model Market Tracker 4Q21

**96.9 M**

© 2022  
Omdia

# PowerVR GPU Security

- Limited research before 2022
- More research kicks off in 2022
  - Google Android Security Team
    - <https://bugs.chromium.org/p/apvi>
  - Google Project Zero

ID	Status	Restrict	Reported	Vendor	Product	Finder	Summary + Labels
☆ 2494	Fixed	---	2023-Jun-22	Imagination	PowerVR	jannh	PowerVR: several bugs in PowerVR GPU driver memory management <a href="#">CCProjectZeroMembers</a>
☆ 2465	Fixed	---	2023-Jul-3	Imagination	PowerVR	jannh	PowerVR: two more LPE security bugs <a href="#">CCProjectZeroMembers</a>

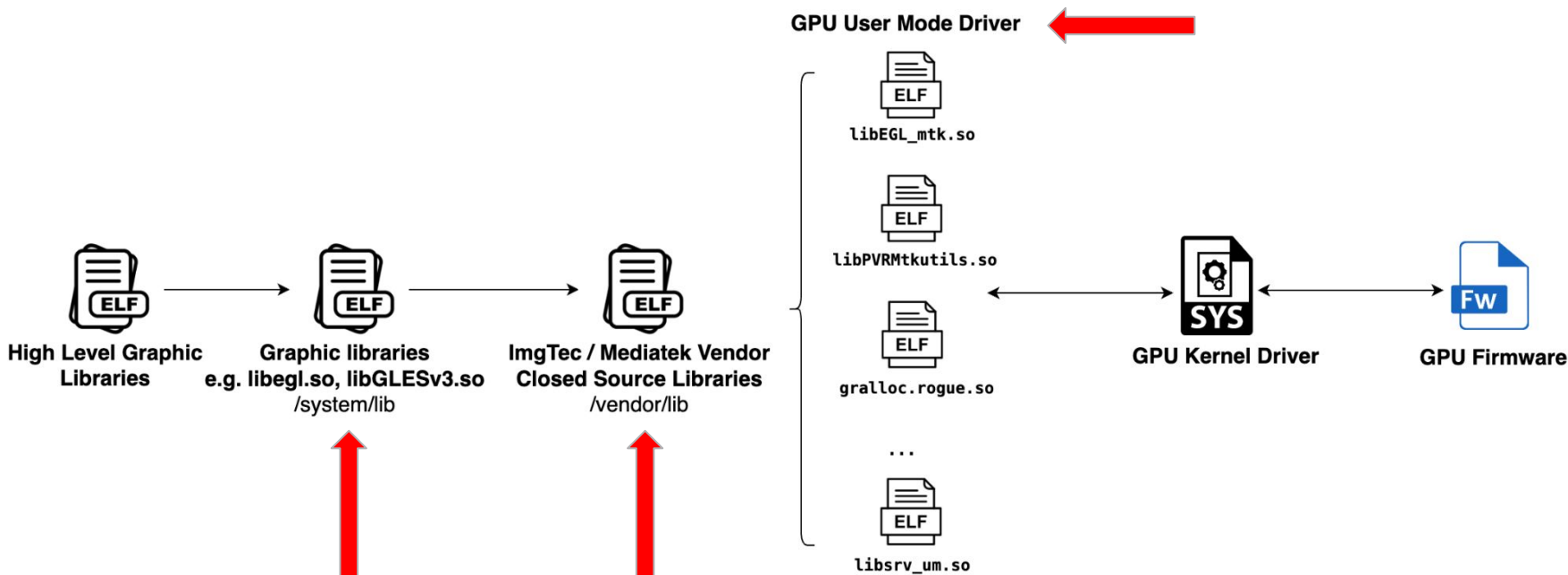
- More engagement with external security researchers

<https://t.me/learningnets>

# Android Graphic Stack & PowerVR Driver Introduction

<https://t.me/learningnets>

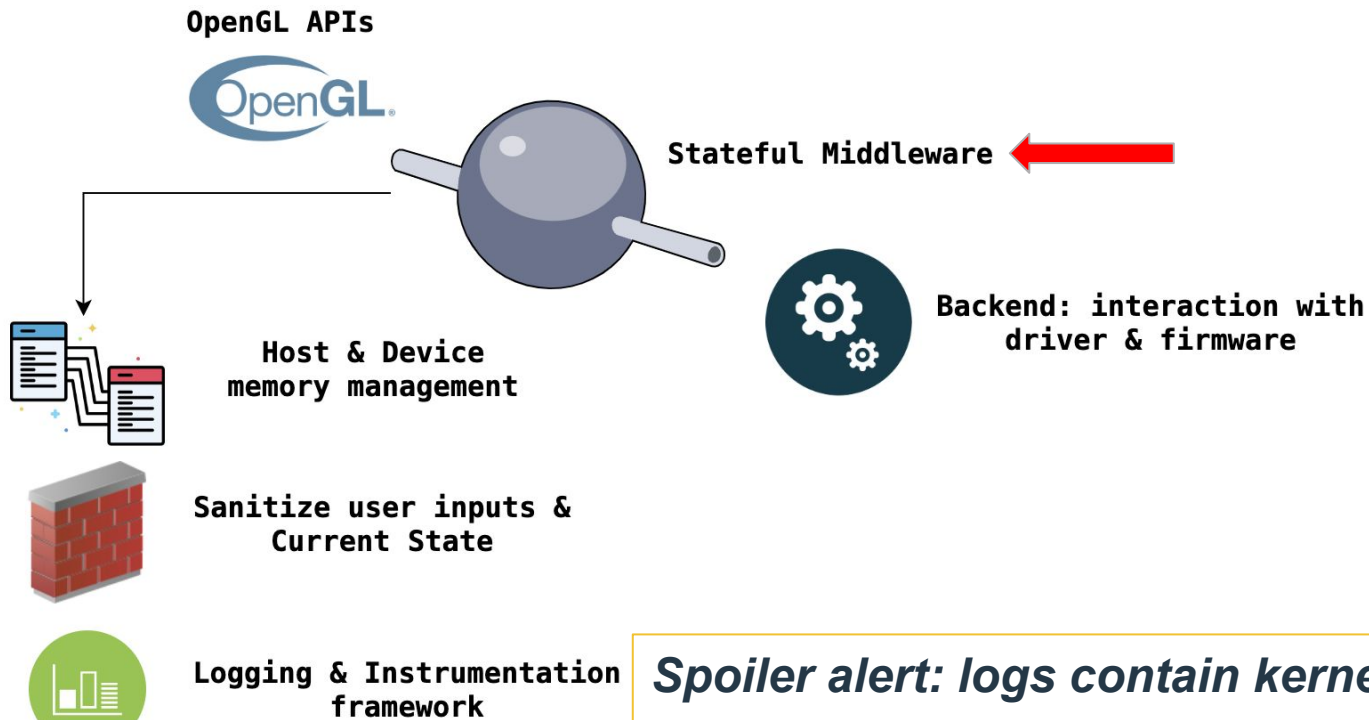
# Android Graphic Stack Overview



*GPU vendors have different UMD implementations*

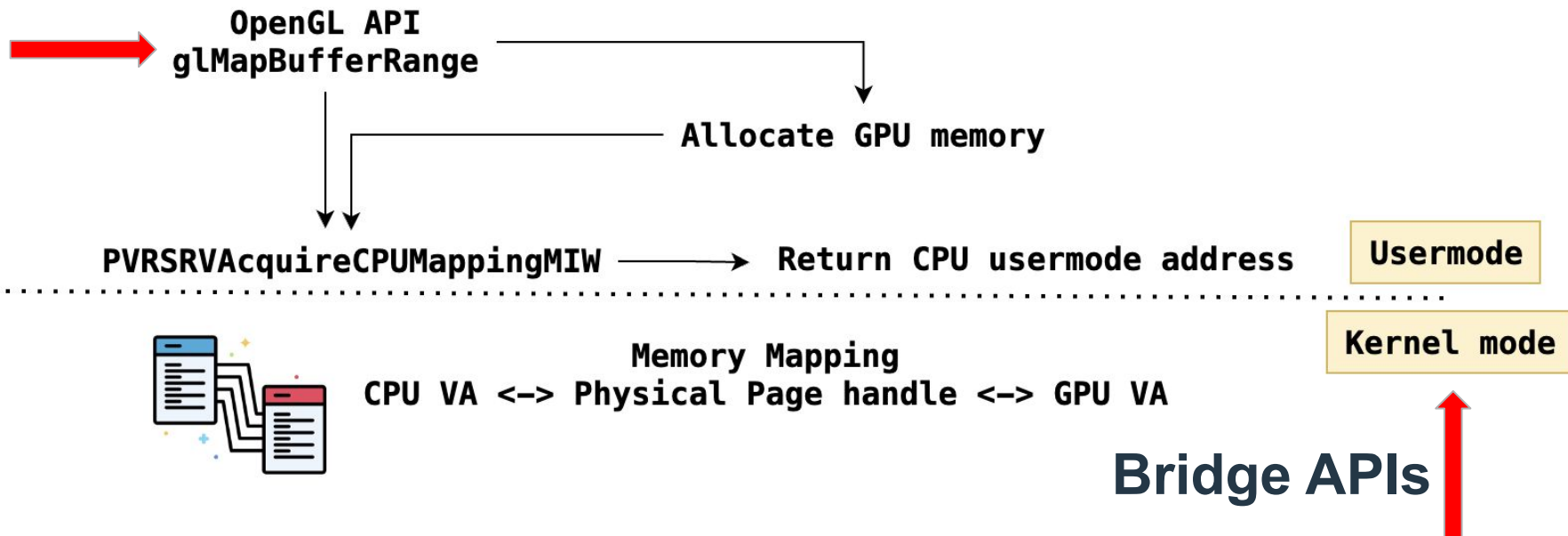
<https://t.me/learningnets>

# OpenGL Impl: ImgTec & MediaTek



<https://t.me/learningnets>

# OpenGL Impl: ImgTec & MediaTek

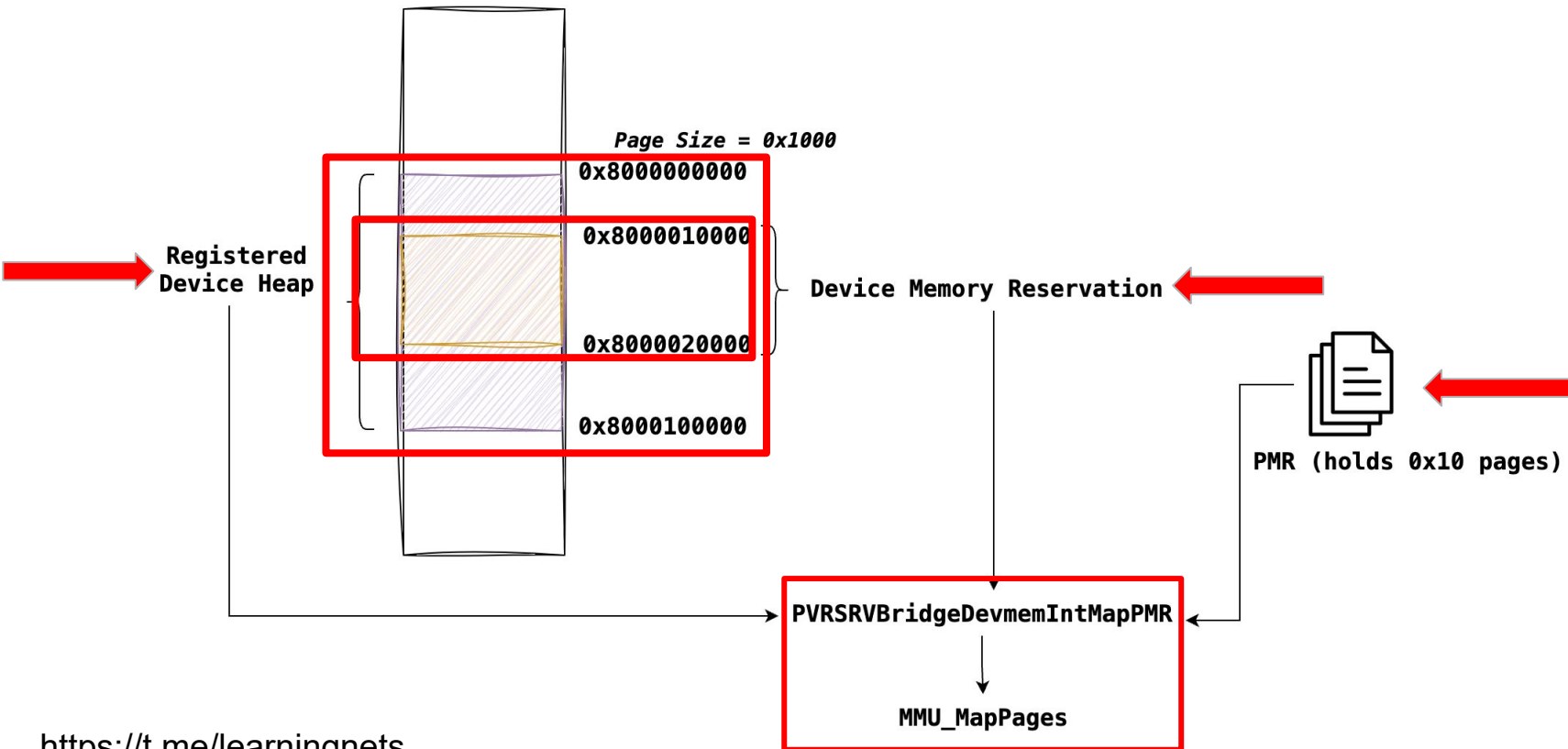


# PVR Driver: PMR & MMU Context

- **Physical Memory Resource (PMR)**
  - Manage allocated physical pages
- **GPU MMU Management**
  - MMU Context Object
  - GPU Memory Heap
  - GPU Memory Reservation

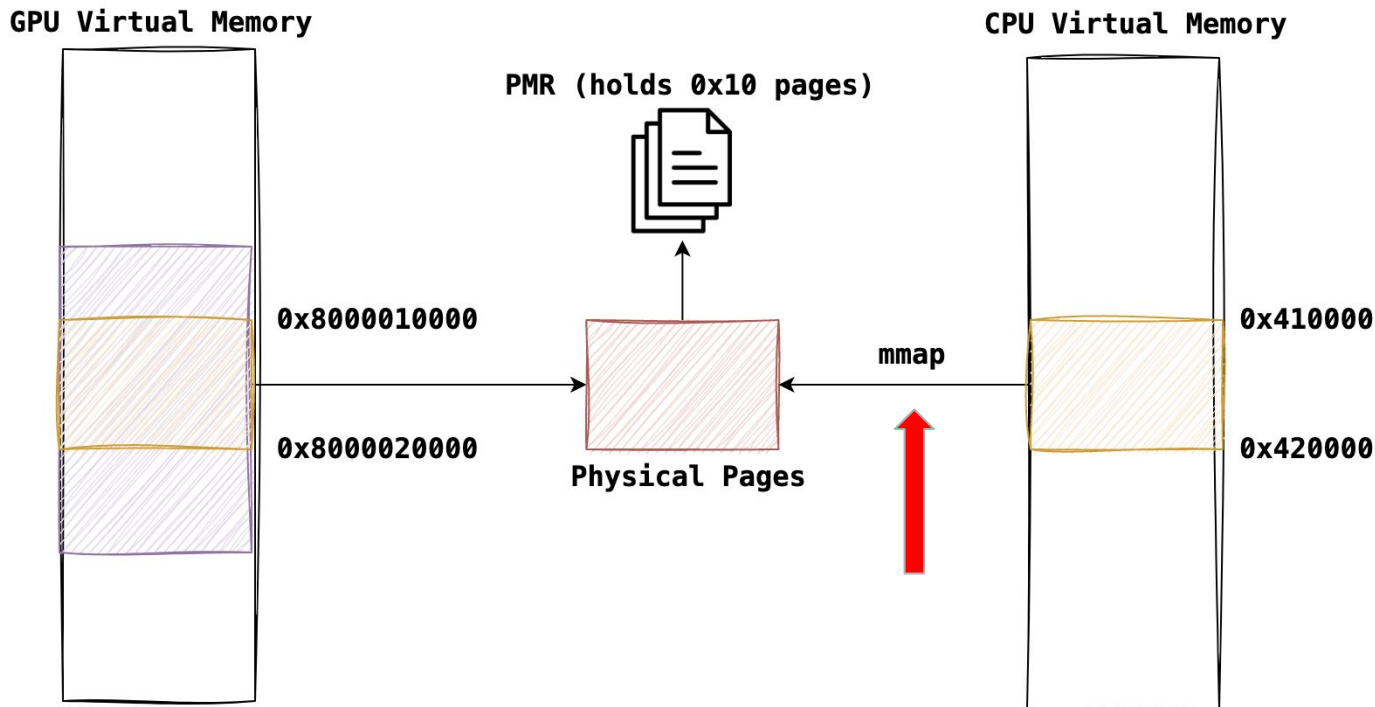


### GPU Virtual Memory



<https://t.me/learningnets>

# PVR Driver: Map CPU pages



<https://t.me/learningnets>

# Thoughts about finding more bugs

<https://t.me/learningnets>

# Thoughts: how to find more bugs

- Difficulties
  - Too many bridge APIs, some of them are arcane
- Thoughts
  - Instrument / Reverse vendor graphic libraries

```
tracer.cpp:167 uiSize = {0x190000}  
tracer.cpp:169 uiChunkSize = {0x1000}  
tracer.cpp:171 ui32NumPhysChunks = {0x0}  
tracer.cpp:173 ui32NumVirtChunks = {0x190}  
tracer.cpp:175 pui32MappingTable = {0xb400007213629e00}  
tracer.cpp:177 ui32Log2PageSize = {0xc}  
tracer.cpp:179 uiFlags = {0x40331}  
tracer.cpp:181 ui32AnnotationLength = {0x12}  
tracer.cpp:183 puiAnnotation = {0x7ffdc964a8}  
tracer.cpp:185 ui32PID = {0x68f0}  
tracer.cpp:187 ui32PDumpFlags = {0x0}  
tracer.cpp:197 puiAnnotation = {SCBUF:VERTEX_DATA}  
tracer.cpp:209 Printing pui32MappingTable  
tracer.cpp:216 ++++++  
tracer.cpp:228 hPMR created = 0x5f  
tracer.cpp:260 PVRSRV_BRIDGE_MM bridge group! func id = {21}  
tracer.cpp:260 PVRSRV_BRIDGE_MM bridge group! func id = {19}  
tracer.cpp:101 MMAP Analyzer: addr = {0x0}, len = {0x190000}, prot = {0x3}, flags = {0x1}, fd = {0x6}, offset = {0x5f000}  
tracer.cpp:116 mmap address = 0x72053ff000
```

# Instrument & Fuzz Graphic APIs

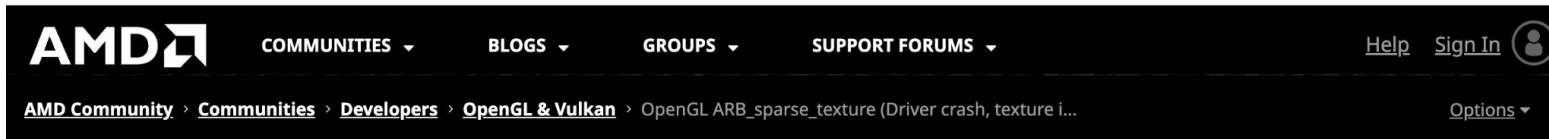
- Instrument on a real 3D app by *PLT function hook*
- Lightweight fuzzing: mutate parameters, scramble memory...



<https://t.me/learningnets>

# Reading OpenGL / Vulkan APIs

- Any complaints about GPU driver kernel crash when using certain OpenGL / Vulkan APIs?



The screenshot shows the AMD Community website header with navigation links for COMMUNITIES, BLOGS, GROUPS, and SUPPORT FORUMS. The breadcrumb trail is: AMD Community > Communities > Developers > OpenGL & Vulkan > OpenGL ARB\_sparse\_texture (Driver crash, texture i... There are links for Help, Sign In, and Options.

03-06-2019 05:27 PM

## OpenGL ARB\_sparse\_texture (Driver crash, texture issues)

Currently we (PCSX2 Team) are trying to implement Sparse Texture support and seem to have stumbled on to several issues on AMD cards which are not present at all on Nvidia (tested by several people).

**Major issue:** Garbage textures on amd cards whenever sparse is enabled.

**Major issue:** As of 19.3.1 enabling Sparse also causes a driver crash on amd cards, this wasn't an issue on the previous driver 19.2.3 where it just caused garbage textures, driver 19.2.1 or 19.2.2 just caused an entire black screen window. So far 19.2.3 seems to behave the best out of the bunch that were tested.

<https://t.me/learningnets>



# Undefined Behaviors in Graphic APIs (*GL\_EXT\_sparse\_texture*)

<https://t.me/learningnets>

# OpenGL: Sparse Texture API

- GL\_EXT\_sparse\_texture
  - Proposed in 2013 by Nvidia
- Most GPU vendors support it nowadays

## New Procedures and Functions

```
void TexPageCommitmentEXT(enum target,  
                           int level,  
                           int xoffset,  
                           int yoffset,  
                           int zoffset,  
                           sizei width,  
                           sizei height,  
                           sizei depth,  
                           boolean commit);
```

<https://t.me/learningnets>

## Name

EXT\_sparse\_texture

## Name Strings

GL\_EXT\_sparse\_texture

## Contributors

Dominik Witczak, Mobica  
Contributors to ARB\_sparse\_texture

Xi Chen, NVIDIA

## Contact

Daniel Koch, NVIDIA Corporation (dkoch 'at' nvidia.com)

## Notice

Copyright (c) 2013 The Khronos Group Inc. Copyright terms at  
<http://www.khronos.org/registry/speccopyright.html>

Portions Copyright (c) 2014 NVIDIA Corporation.

## Status

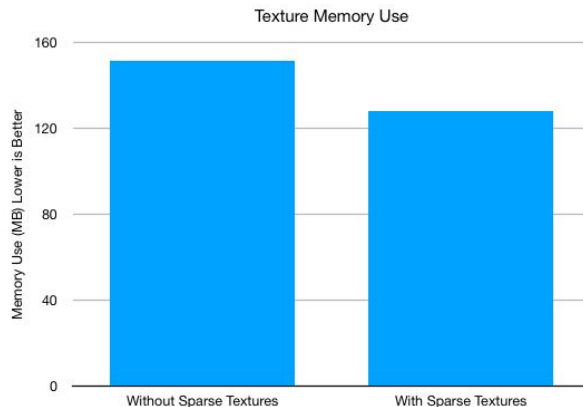
Complete.

## Version

Last Modified Date: 27/03/2015  
Revision: 3

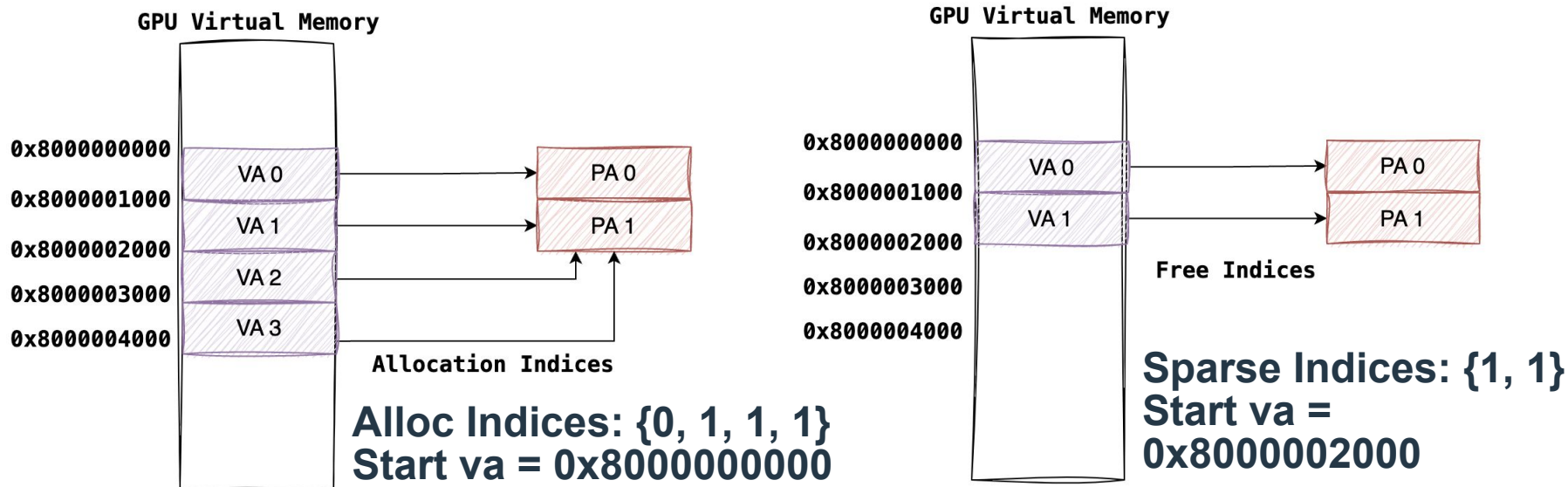
# OpenGL: Sparse Texture API

- Why Sparse texture API?
- Create a resource that is larger than physical memory
  - but only has a small portion of that resource actually backed by physical memory.



# Low Level Implementation

- Graphic API: Invoke kernel APIs



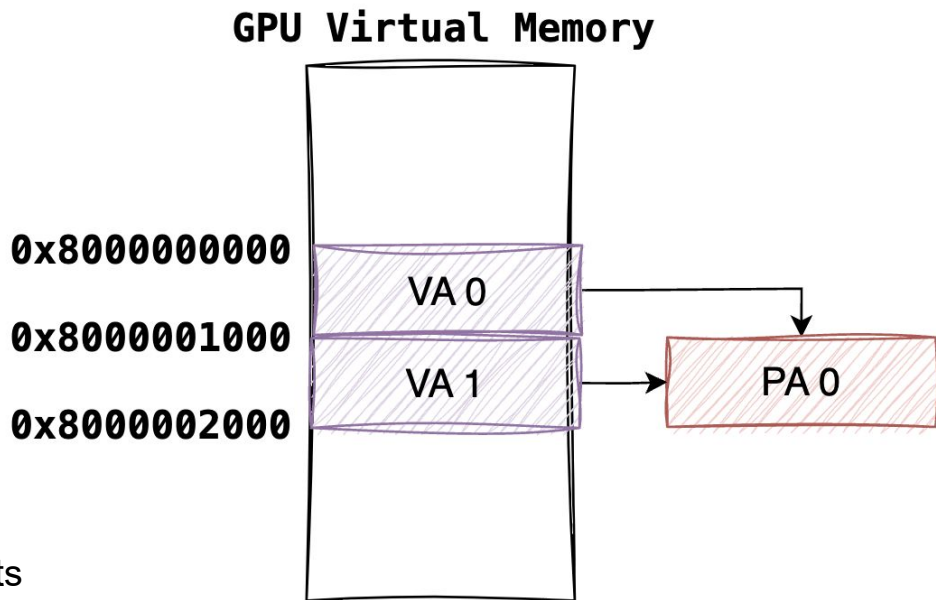


# Undefined Behavior in OpenGL Document

- If the value of `commit` is `FALSE`, then the texture pages contained in the region are made **de-committed**. **Their physical store is de-allocated**, and their contents again become **undefined**.
- Reads from such regions (**uncommitted**) produce **undefined data**, but otherwise have no adverse effect.
- Atomic operations with return values on **uncommitted regions** will complete normally, but **the returned value will be undefined** and the result of the ... will be discarded.
- Writes to such regions are ignored. The GL may attempt to write to **uncommitted** regions but the effect of doing so will be **benign**.

# Sparse Texture API Under the Hood

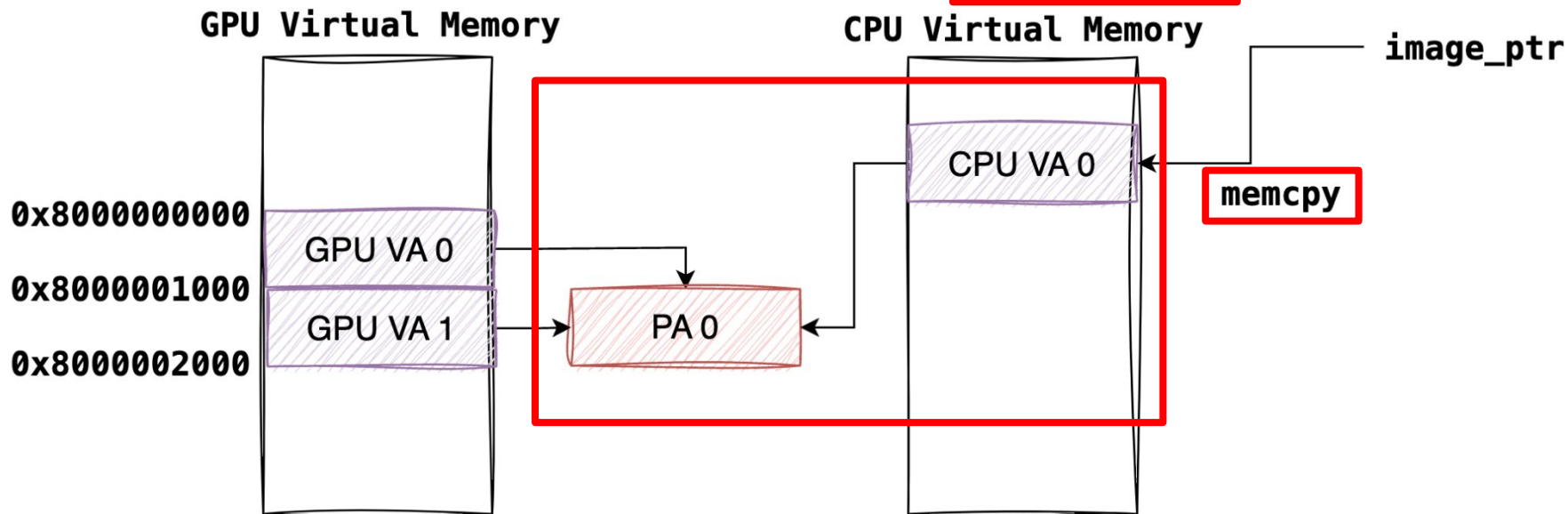
- Step 1: Allocate Sparse texture memory
  - `glTexPageCommitmentEXT(..., /*commit=*/GL_TRUE);`



# Sparse Texture API Under the Hood

- Step 2: Initialize Textures

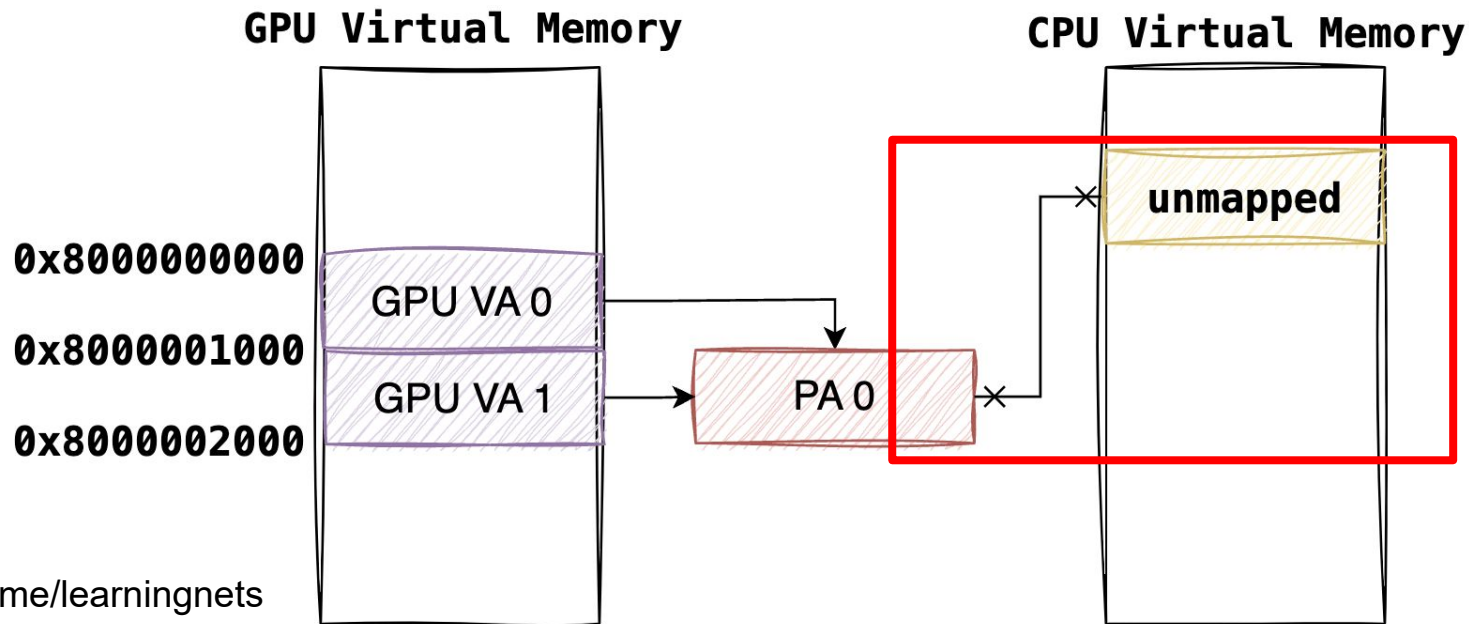
- `glTexSubImage3D(..., /*ptr=*/image_ptr);`



# Sparse Texture API Under the Hood

- Step 2: Initialize Textures

- `glTexSubImage3D(..., /*ptr=*/image_ptr);`



<https://t.me/learningnets>

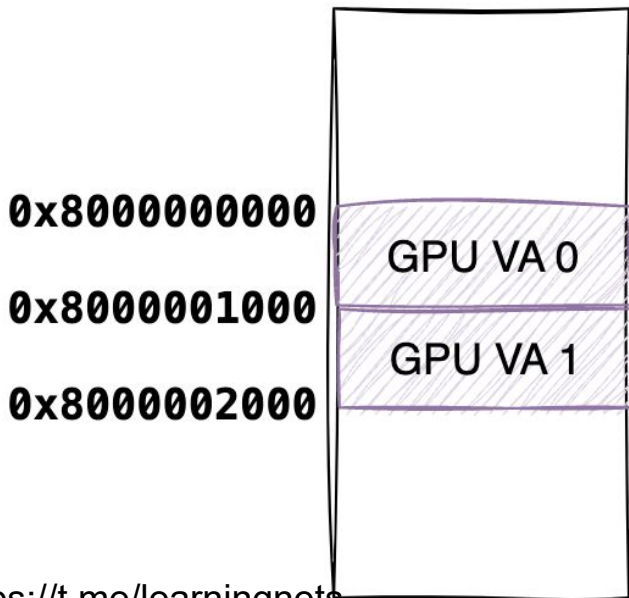
# Sparse Texture API Under the Hood

- Final step: Destroy the sparse texture
  - `glTexPageCommitmentEXT(..., /*commit=*/GL_FALSE);`
- Look secure!
  - Not possible to remap the sparse texture on GPU to CPU because it's already destroyed

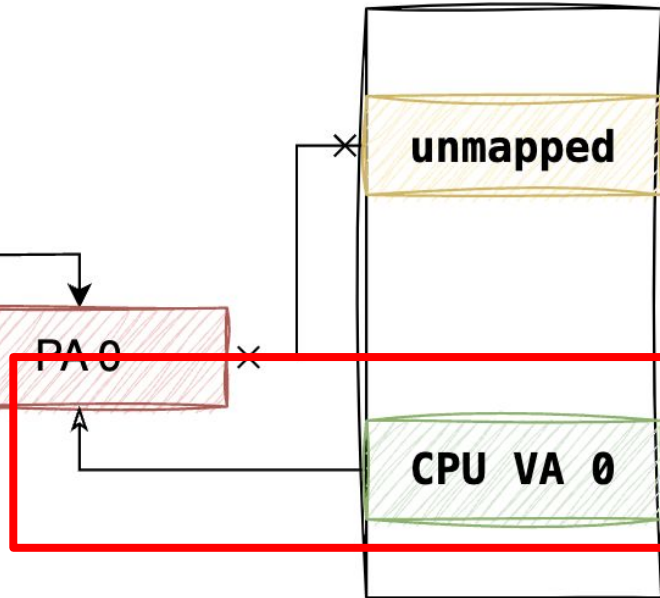
# Additional Mapping by ourselves

- Accessing “undefined memory” from CPU

GPU Virtual Memory



CPU Virtual Memory

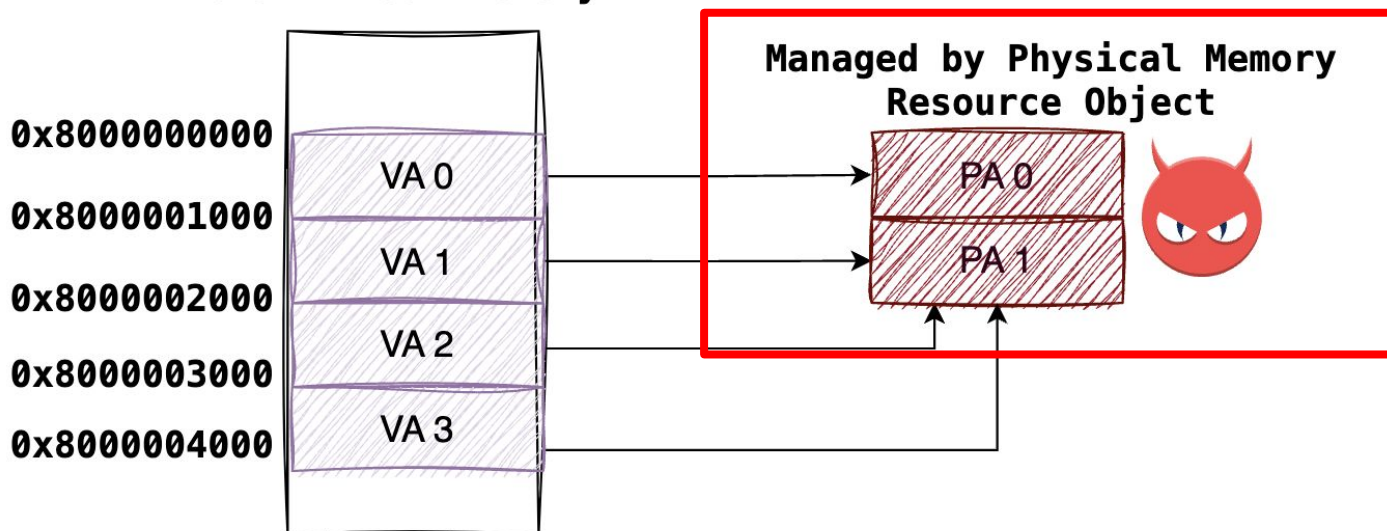


<https://t.me/learningnets>

# Issues in Implementing Sparse Texture

- Problem 1: object read / write OOB
- Problem 2: ref issues

GPU Virtual Memory

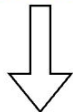


<https://t.me/learningnets>

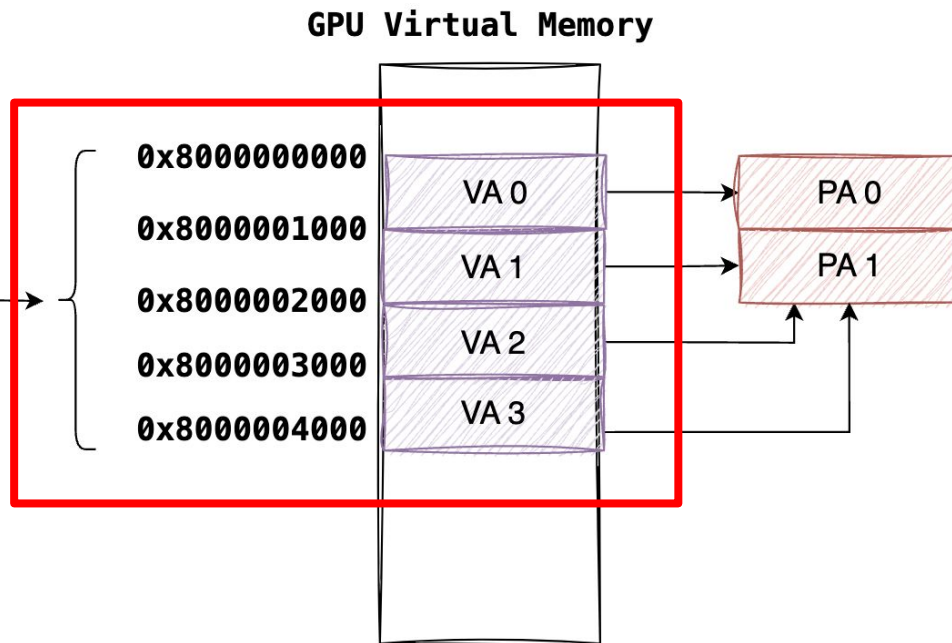
# Issues in Implementing Sparse Texture

- Problem 3: GPU start VA passed from userspace

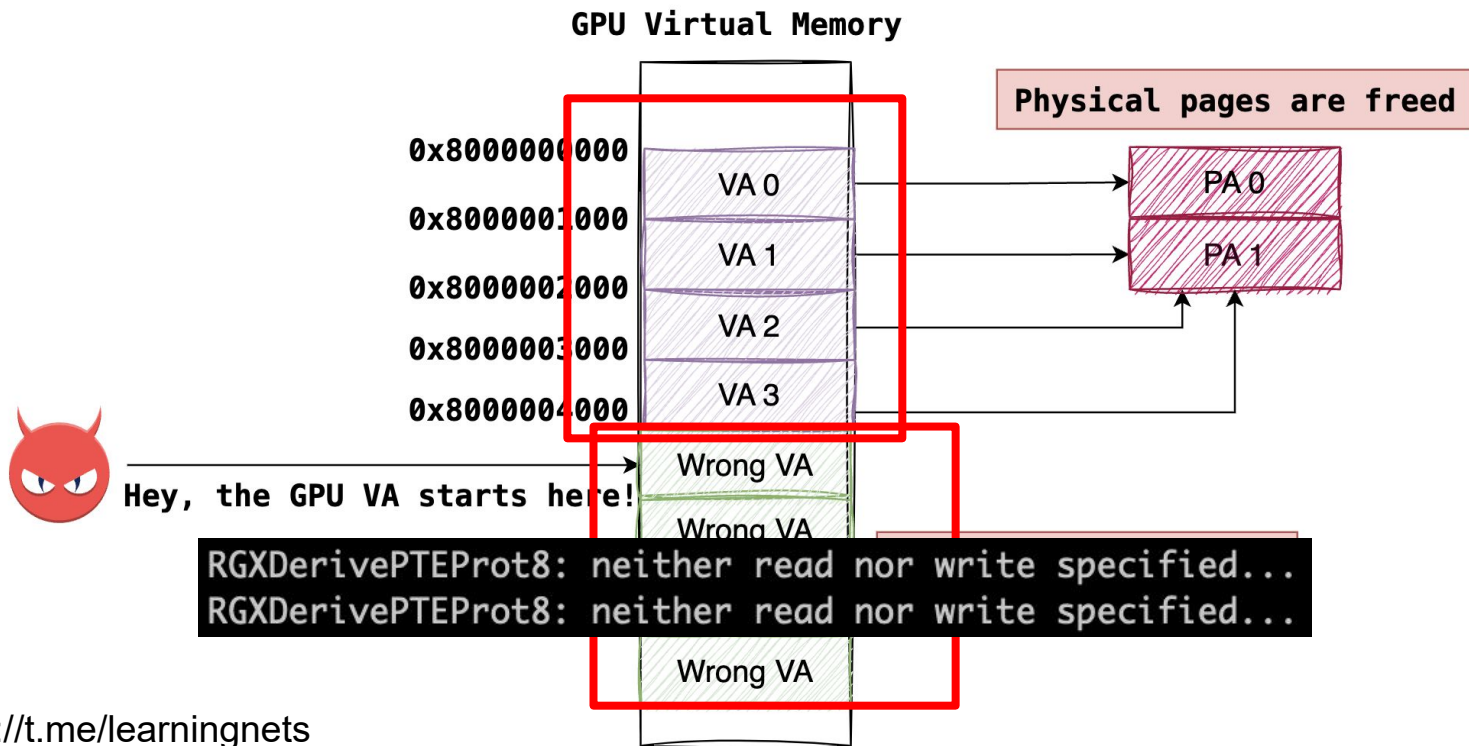
Graphic API: Destroy Sparse texture



Graphic library knows there're 4 GPU virtual pages starting with the address 0x8000000000



# Issues in Implementing Sparse Texture



# Corrupting GPU Page Tables

- GPU Heap memory layout
- PowerVR has FANCY page tables
  - Supports different page size: **4K, 16K, 64K, 256K, 1M, 2M**

Name = General SVM, sDevVAddrBase.uiAddr = 400000, uiHeapLength = 7fffc00000

Name = Vulkan capture replay buffer, sDevVAddrBase.uiAddr = bfc0000000, uiHeapLength = 40000000

Name = General, sDevVAddrBase.uiAddr = 8000000000, uiHeapLength = 3fc0000000

Name = RgnHdr BRN63142, sDevVAddrBase.uiAddr = dbf0000000, uiHeapLength = 10000000

Name = General NON-4K, sDevVAddrBase.uiAddr = c000000000, uiHeapLength = 1000000000

Name = VisTest, sDevVAddrBase.uiAddr = dc00000000, uiHeapLength = 100000

PDS Code and Data, sDevVAddrBase.uiAddr = da00000000, uiHeapLength = 100000000

Name = USC Code, sDevVAddrBase.uiAddr = e000000000, uiHeapLength = 100000000

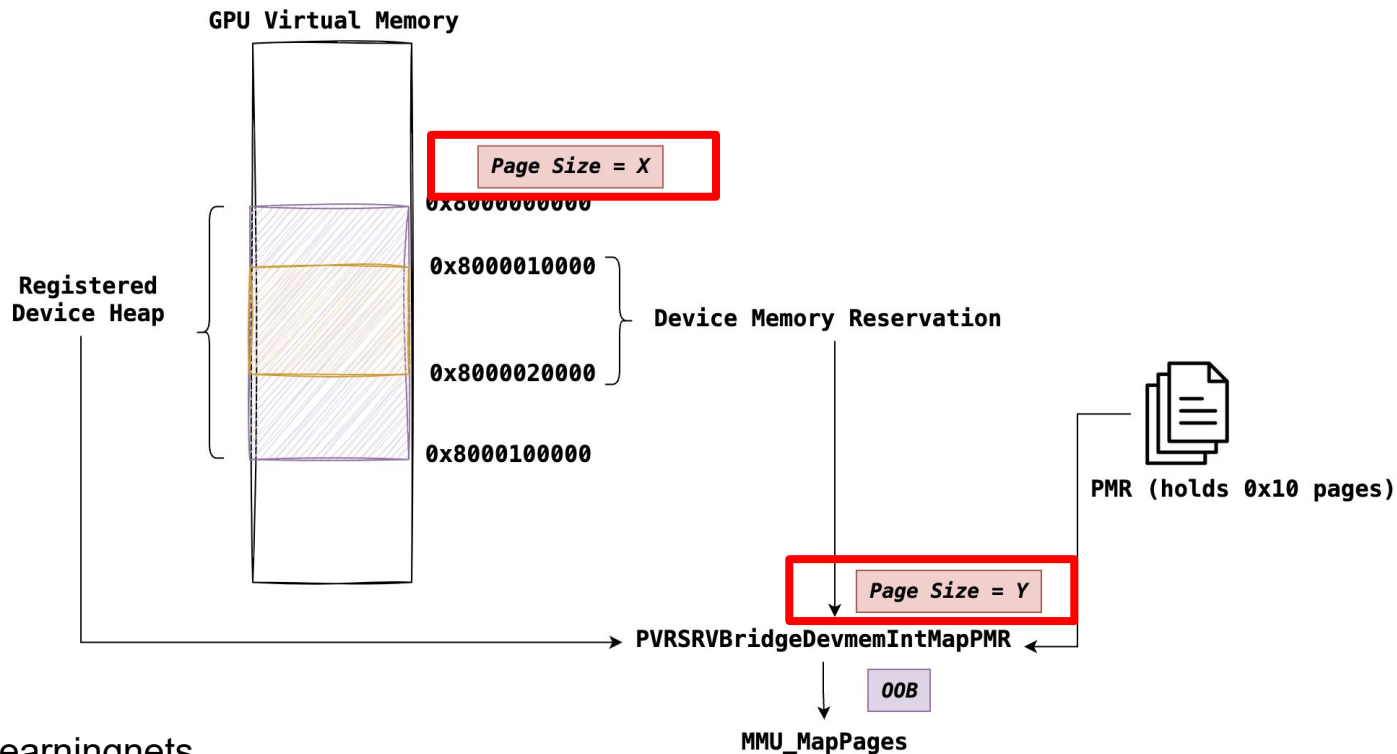
Name = TQ3DParameters, sDevVAddrBase.uiAddr = e400000000, uiHeapLength = 400000000

Name = TDM TPU YUV Coeffs, sDevVAddrBase.uiAddr = ea00080000, uiHeapLength = 40000

Name = FwMain, sDevVAddrBase.uiAddr = e1c0000000, uiHeapLength = 1ef0000

Name = FwConfig, sDevVAddrBase.uiAddr = e1c1ff0000, uiHeapLength = 10000

# Corrupting GPU Page Tables



# The feature was there “forever”...

- The buggy sparse feature was introduced a decade ago
- Some of our other findings also exist a decade ago.



RIP the feature that was there forever and nobody wanted to report :)

# Rooting Device

<https://t.me/learningnets>



Image by DALL·E



# Exploit Page Use-after-free

- Control page table
- Nullify all kernel mitigations

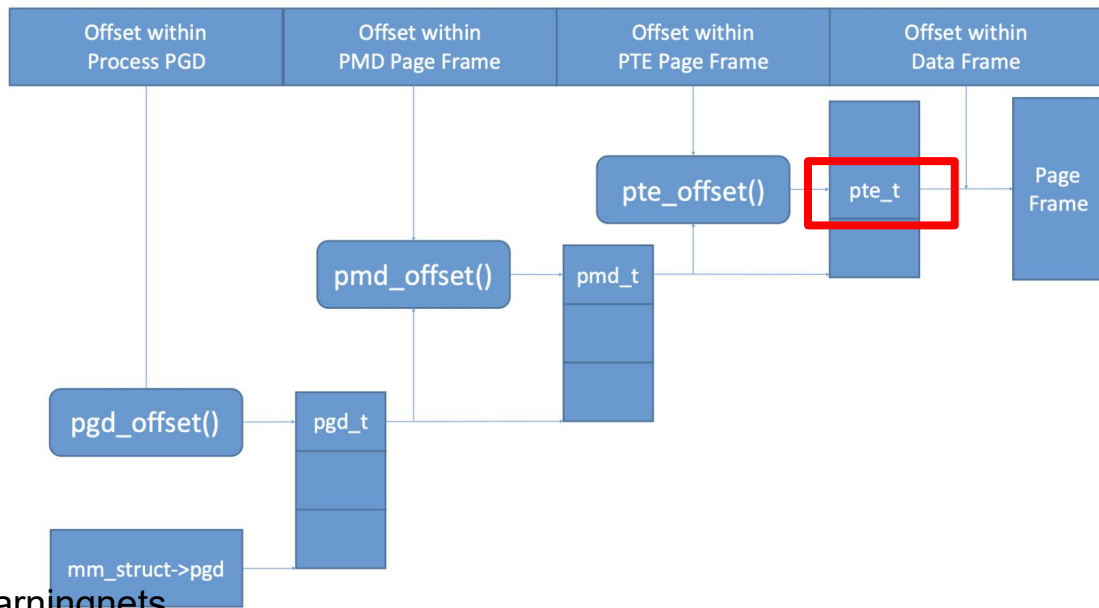


Image from  
asia-18-WANG-KSMA-Breaking-A  
ndroid-kernel-isolation-and-Rooting  
-with-ARM-MMU-features

<https://t.me/learningnets>





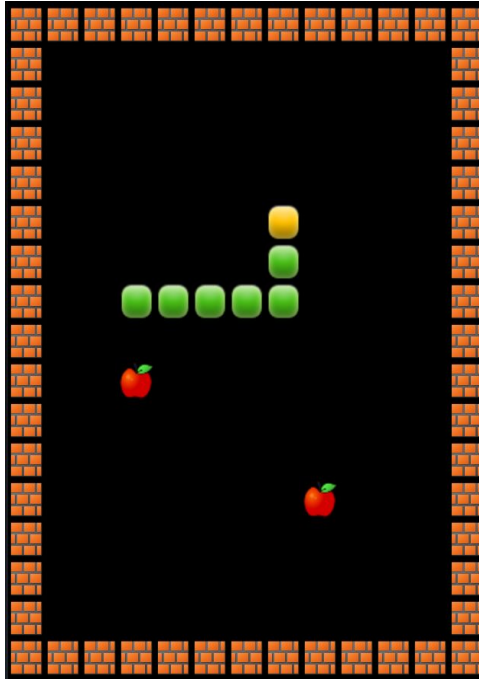
# Finding vulnerabilities

<https://t.me/learningnets>



# Finding other peoples' exploits

<https://t.me/learningnets>



<https://t.me/learningnets>

```
exec("/system/bin/cmd package install -r -d -t /data/local/tmp/app-debug.apk\n", v2, 10);
exec(
  "/system/bin/cmd package grant com. [REDACTED] android.permission.ACCESS_COARSE_LOCATION\n",
  v2,
  1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.ACCESS_FINE_LOCATION\n", v2, 1);
exec(
  "/system/bin/cmd package grant com. [REDACTED] android.permission.ACCESS_BACKGROUND_LOCATION\n",
  v2,
  1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.CAMERA\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.GET_ACCOUNTS\n", v2, 1);
exec(
  "/system/bin/cmd package grant com. [REDACTED] android.permission.PROCESS_OUTGOING_CALLS\n",
  v2,
  1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.RECORD_AUDIO\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.RECEIVE_SMS\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.READ_CALENDAR\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.READ_CALL_LOG\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.READ_CONTACTS\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.READ_EXTERNAL_STORAGE\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.READ_PHONE_STATE\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.READ_SMS\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.SEND_SMS\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.SYSTEM_ALERT_WINDOW\n", v2, 1);
exec("/system/bin/cmd package grant com. [REDACTED] android.permission.WRITE_CALENDAR\n", v2, 1);
```

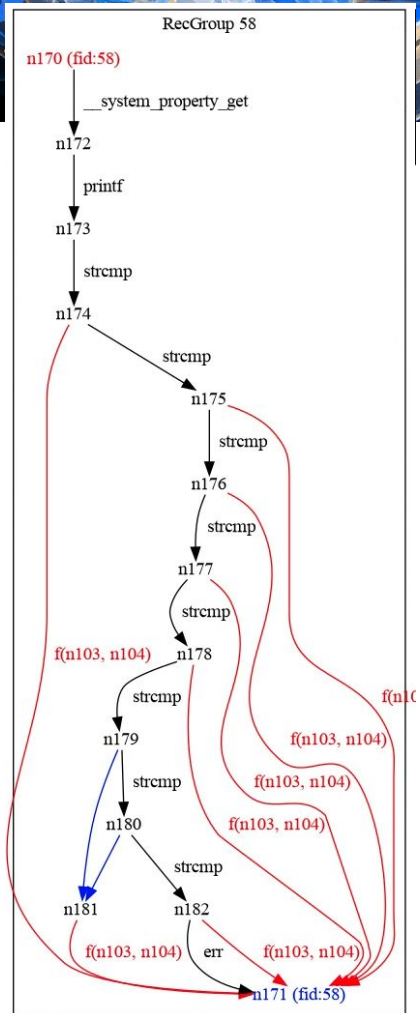
```
1 rule NP_AMRootingSignals__EasyRoot {
2   meta:
3     desc = "EasyRoot exploit binary"
4     rs1 = "020abf8c687168abae9f4d202ba644d5c9eebafd39442ec28b95c55b459e8a37"
5   strings:
6     $a = "easyroot"
7     $b = "get root failed..."
8     $c = "get root success..."
9   condition:
10    uint16(0) == 0x457f and all of them
11 }
```

```

int length = __system_property_get("ro.build.fingerprint", fingerprint);
printf("%s\n", fingerprint);
if (!strcmp(fingerprint, "google/oriole/oriole:12/SD1A.210817.037/7862242:user/release-keys")) {
    do_bad_thing_with_params(a, b, c);
    return;
}
if (!strcmp(fingerprint, "google/oriole/oriole:12/SQ1D.220105.007/8030436:user/release-keys")) {
    do_bad_thing_with_params(d, e, f);
    return;
}
if (!strcmp(fingerprint, "google/oriole/oriole:12/SQ1D.220205.004/8151327:user/release-keys")) {
    do_bad_thing_with_params(g, h, i);
    return;
}
if (!strcmp(fingerprint, "google/oriole/oriole:12/SQ3A.220705.003/8671607:user/release-keys")) {
    do_bad_thing_with_params(j, k, l);
    return;
}
etc

```

<https://t.me/learningnets>



### Report Findings (arm32 binary)

```
-----String Obfuscation Section Start-----  
Function `sub_0x3e13e64` decrypts string `mount -o remount,rw /system`  
Function `sub_0x3e13e64` decrypts string `cat`  
Function `sub_0x3e13e64` decrypts string `rm -r`  
Function `sub_0x3e13e64` decrypts string `rm`  
Function `sub_0x3e13e64` decrypts string `chmod 0777`  
Function `sub_0x3e13e64` decrypts string `chmod 0644`  
Function `sub_0x3e13e64` decrypts string `chown root:root`  
Function `sub_0x3e13e64` decrypts string `chown root.root`  
Function `sub_0x3e13e64` decrypts string `chcon`  
Function `sub_0x3e13e64` decrypts string `u:object_r:misc_user_data_file:s0`  
Function `sub_0x3e13e64` decrypts string `u:object_r:zygote_exec:s0`  
Function `sub_0x3e13e64` decrypts string `/sdcard/.google/`  
Function `sub_0x3e13e64` decrypts string `com.android.google.cdcore.db`  
Function `sub_0x3e13e64` decrypts string `/system/etc/.uatyes`  
Function `sub_0x3e13e64` decrypts string `/data/dalvik-cache/arm`  
Function `sub_0x3e13e64` decrypts string `system@framework@boot-thirdcd.oat`  
Function `sub_0x3e13e64` decrypts string `tmpfiles/`  
Function `sub_0x3e13e64` decrypts string `/data/local/tmp/`  
Function `sub_0x3e13e64` decrypts string `asbymol`  
Function `sub_0x3e13e64` decrypts string `bdlomsd`  
Function `sub_0x3e13e64` decrypts string `jkpatch`  
Function `sub_0x3e13e64` decrypts string `watch_dog`  
Function `sub_0x3e13e64` decrypts string `cash`  
Function `sub_0x3e13e64` decrypts string `-ai`  
Function `sub_0x3e13e64` decrypts string `+ai`  
Function `sub_0x3e13e64` decrypts string `/data/dalvik-cache/arm/system@framework@boot-bridacco.oat`  
Function `sub_0x3e13e64` decrypts string `/data/dalvik-cache/arm64/system@framework@boot-bridacco.oat`  
Function `sub_0x3e13e64` decrypts string `/data/dalvik-cache/arm/system@framework@boot.oat`  
Function `sub_0x3e13e64` decrypts string `/data/dalvik-cache/arm64/system@framework@boot.oat`  
Function `sub_0x3e13e64` decrypts string `/data/dalvik-cache/arm/system@framework@boot-acco.oat`  
Function `sub_0x3e13e64` decrypts string `/data/dalvik-cache/arm64/system@framework@boot-acdo.oat`  
Function `sub_0x3e13e64` decrypts string `/system/lib/binder.so`  
Function `sub_0x3e13e64` decrypts string `/system/lib64/binder.so`  
Function `sub_0x3e13e64` decrypts string `/system/lib/libmedia_jni.so`  
Function `sub_0x3e13e64` decrypts string `/system/lib64/libmedia_jni.so`
```



# Stopping exploitation

<https://t.me/learningnets>



- Mitigations
  - [ARM Memory Tagging Extension](#)
  - [First handset with MTE on the market](#)
- Secure development practices
  - Rust in Linux and [Android](#)
  - OEM Portal - restricted access
    - <https://docs.partner.android.com/security/oem-edu/driver-developers>
- Detecting impossible conditions



```
oriole:/data/local/tmp $ █
```

```
oriole:/data/local/tmp $ █
```



# “N-days function like 0-days on Android”

-Maddie Stone, Security Researcher, Threat Analysis Group (TAG)

<https://t.me/learningnets>

# Android Partner Vulnerability Initiative APVI

<https://t.me/learningnets>

## 3P Vulns - Findings

- **80+ disclosed APVI OEM vulnerabilities**
  - 1300+ GPSRP vulns
  - 1600+ SoC vulns
- **Filed ~30 bugs in 2023** by Android Security Team
  - 5 Disclosed Page UAF
  - Other Page Corruptions: map2anywhere, map2oobpages...
- **[bugs.chromium.org/p/apvi](https://bugs.chromium.org/p/apvi)**
  - Moving to issuetracker in 2024

# Closing the Gap

- **Developers** help OEMs implement & improve security processes
  - Decrease time from ingestion to patch
    - Reinforce policies with technical measures
- **Researchers**
  - GPU security is still under-researched
- More to come...



**Thank You**