TU Graz
SCIENCE
PASSION
TECHNOLOGY

The Doom of Device Drivers:

Your Android Device (Most Likely) has N-Day Kernel Vulnerabilities

Lukas Maar Florian Draschbacher Lorenz Schumm Ernesto Martínez García
Stefan Mangard

August 13-15, 2025

USENIX Security

› isec.tugraz.at

Case of CVE-2019-2215

```
From: syzbot  <bot+f2a32269c7d88a3653ef36f3d516f19ece83fdb5@syzkaller.appspotmail.com>
Subject: KASAN: use-after-free Read in __lock_acquire (2)
Date: Sat, 02 Dec 2017 08:08:01 -0800 [thread overview]
Message-ID: <001a1149750a83b8db055f5db0d2@google.com> (raw)
In-Reply-To: <001a1147c73265e0a2055e43711e@google.com>

==================================================================
BUG: KASAN: use-after-free in __lock_acquire+0x465e/0x47f0
kernel/locking/lockdep.c:3378
Read of size 8 at addr ffff8801cd8e13f0 by task syzkaller236979/3086

CPU: 1 PID: 3086 Comm: syzkaller236979 Not tainted 4.15.0-rc1+ #115
Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS
Google 01/01/2011
Call Trace:
  __dump_stack lib/dump_stack.c:17 [inline]
  dump_stack+0x194/0x257 lib/dump_stack.c:53
  ...
```

Case of CVE-2019-2215

```
From: syzbot  <bot+f2a32269c7d88a3653ef36f3d516f19ece83fdb5@syzkaller.appspotmail.com>
Subject: KASAN: use-after-free Read in __lock_acquire (2)
Date: Sat, 02 Dec 2017 08:08:01 -0800 [thread overview]
Message-ID: <001a1149750a83b8db055f5db0d2@google.com> (raw)
In-Reply-To: <001a1147c73265e0a2055e43711e@google.com>

==================================================================
BUG: KASAN: use-after-free in __lock_acquire+0x465e/0x47f0
kernel/locking/lockdep.c:3378
Read of size 8 at addr ffff8801cd8e13f0 by task syzkaller236979/3086

CPU: 1 PID: 3086 Comm: syzkaller236979 Not tainted 4.15.0-rc1+ #115
Hardware name: Google Google Compute Engine/Google Compute Engine, BIOS
Google 01/01/2011
Call Trace:
  __dump_stack lib/dump_stack.c:17 [inline]
  dump_stack+0x194/0x257 lib/dump_stack.c:53
  ...
```

Case of CVE-2019-2215

```c
#include <fcntl.h>
#include <sys/epoll.h>
#include <sys/ioctl.h>
#include <unistd.h>

#define BINDER_THREAD_EXIT 0x40046208ul

int main()
{
  int fd, epfd;
  struct epoll_event event = { .events = EPOLLIN };

  fd = open("/dev/binder0", O_RDONLY);
  epfd = epoll_create(1000);
  epoll_ctl(epfd, EPOLL_CTL_ADD, fd, &event);
  ioctl(fd, BINDER_THREAD_EXIT, NULL);
}
```

Patch for CVE-2019-2215

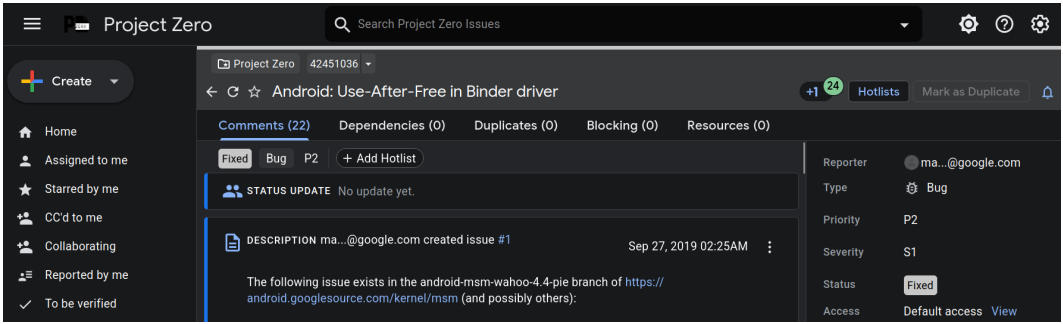
```
From: Martijn Coenen <maco@android.com>
Date: Fri, 5 Jan 2018 11:27:07 +0100
Subject: ANDROID: binder: remove waitqueue when thread exits.

commit f5cb779ba16334b45ba8946d6bfa6d9834d1527f upstream.

binder_poll() passes the thread->wait waitqueue that
can be slept on for work. When a thread that uses
epoll explicitly exits using BINDER_THREAD_EXIT,
the waitqueue is freed, but it is never removed
from the corresponding epoll data structure. When
the process subsequently exits, the epoll cleanup
code tries to access the waitlist, which results in
a use-after-free.
```

Patch for CVE-2019-2215

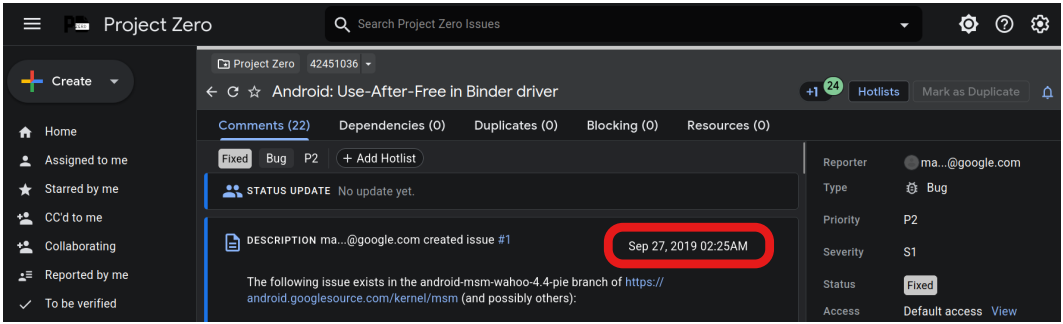
```
diff --git a/drivers/android/binder.c b/drivers/android/binder.c
index 6b4a991..bb48a7b 100644
--- a/drivers/android/binder.c
+++ b/drivers/android/binder.c

@@ -4535,6 +4535,18 @@ static int binder_thread_release(struct binder_proc *proc,
    if (t)
      spin_lock(&t->lock);
  }
+
+ /*
+  * If this thread used poll, make sure we remove the waitqueue
+  * from any epoll data structures holding it with POLLFREE.
+  * waitqueue_active() is safe to use here because we're holding
+  * the inner lock.
+  */
+ if ((thread->looper & BINDER_LOOPER_STATE_POLL) &&
+     waitqueue_active(&thread->wait)) {
+   wake_up_poll(&thread->wait, POLLHUP | POLLFREE);
+ }
+
  binder_inner_proc_unlock(thread->proc);

  if (send_reply)
```

In-The-Wild Exploitation of CVE-2019-2215

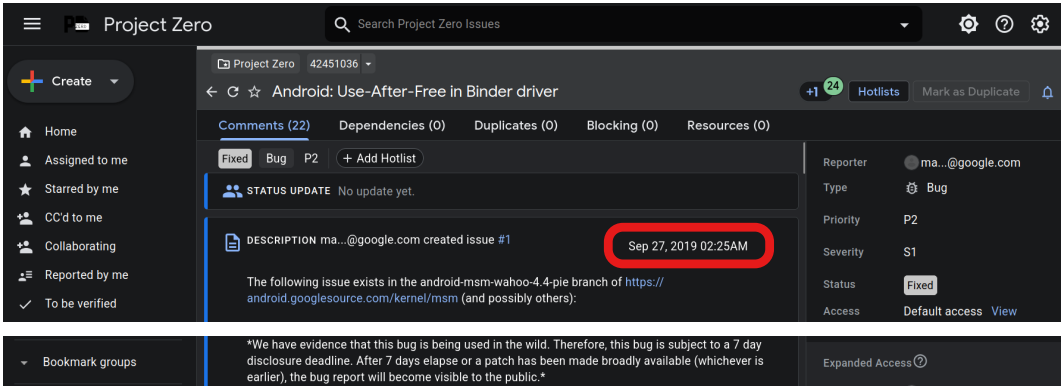
In-The-Wild Exploitation of CVE-2019-2215



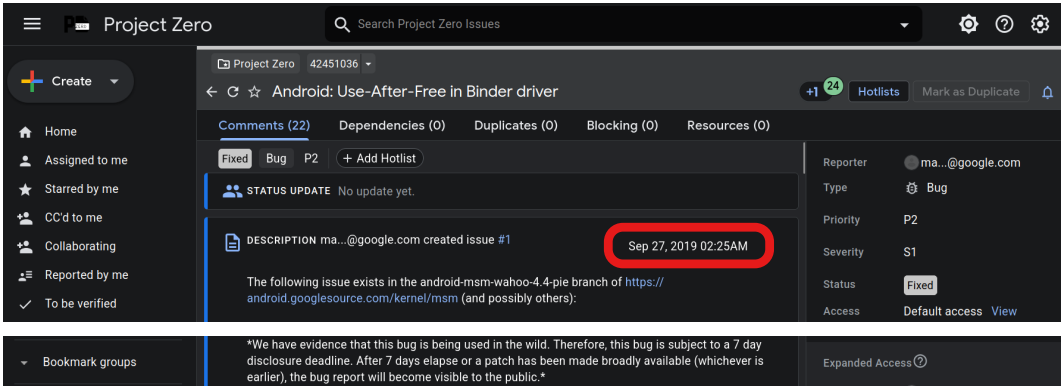
In-The-Wild Exploitation of CVE-2019-2215



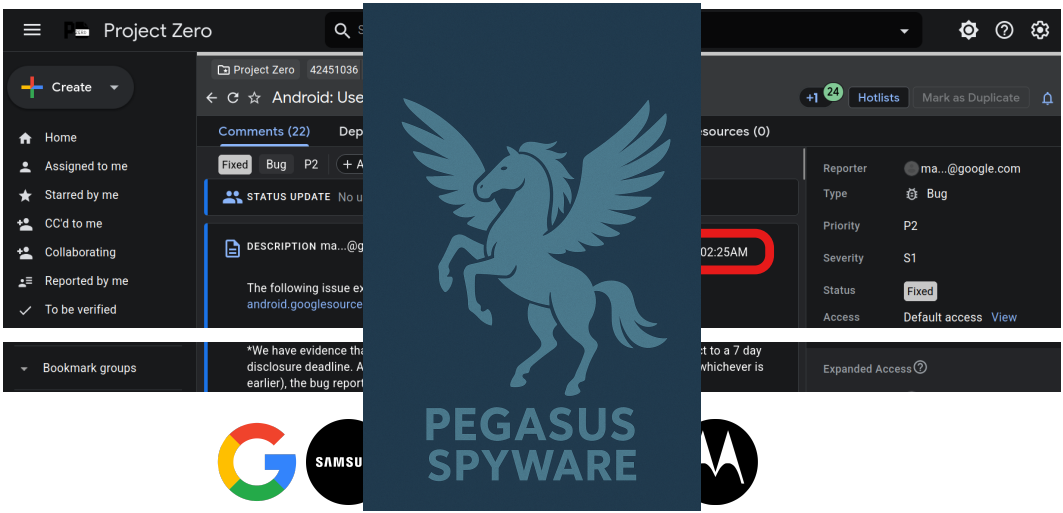
In-The-Wild Exploitation of CVE-2019-2215



In-The-Wild Exploitation of CVE-2019-2215



In-The-Wild Exploitation of CVE-2019-2215

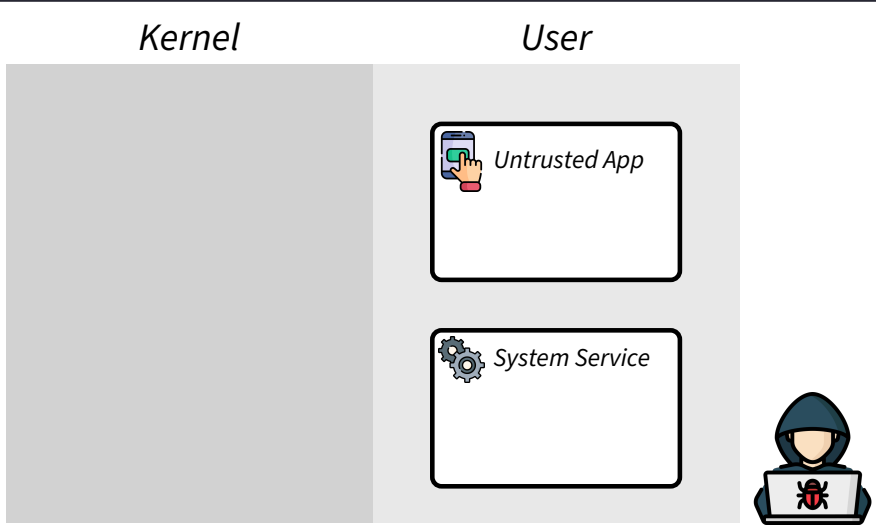


There was a significant time gap between vulnerability introduction and patching!

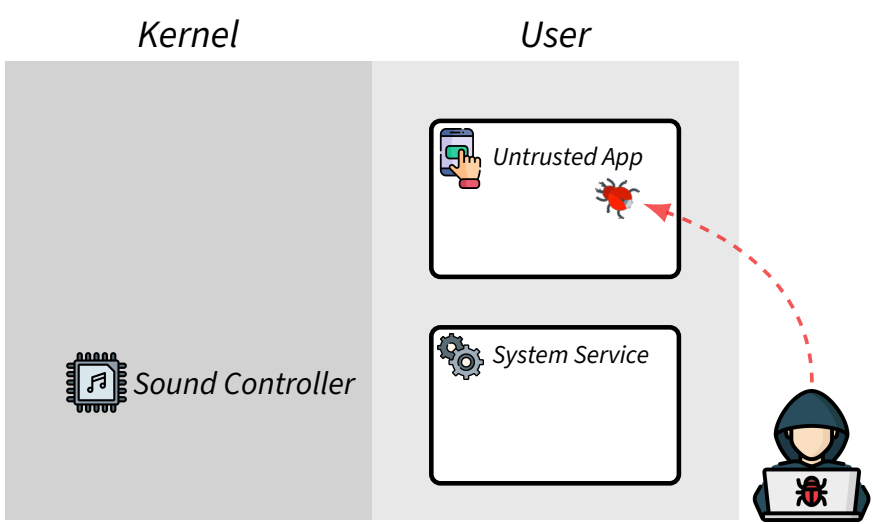
Continue with In-The-Wild Device Exploitation

Kernel *User*

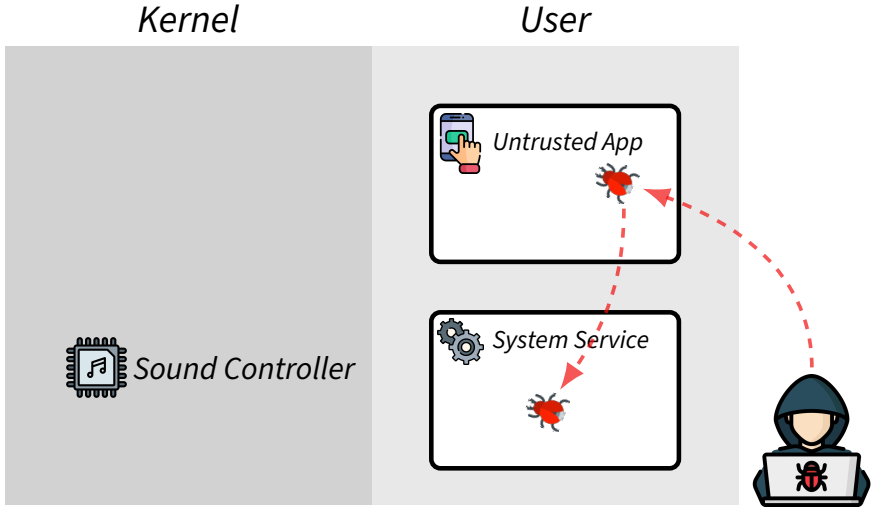
Continue with In-The-Wild Device Exploitation



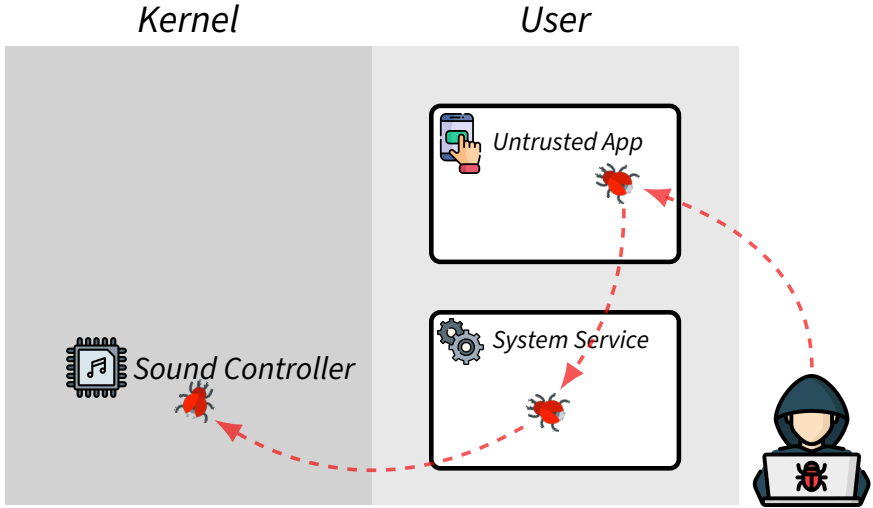
Continue with In-The-Wild Device Exploitation



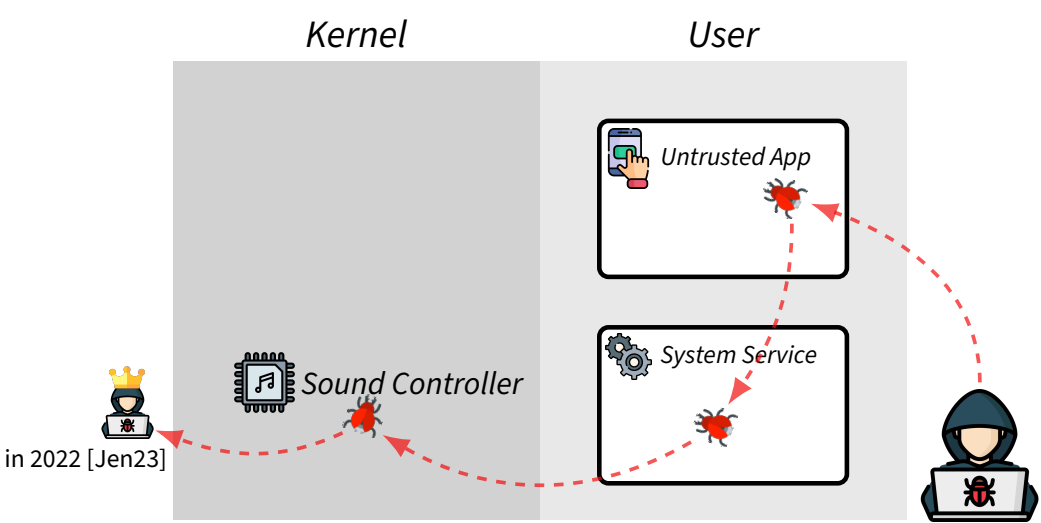
Continue with In-The-Wild Device Exploitation



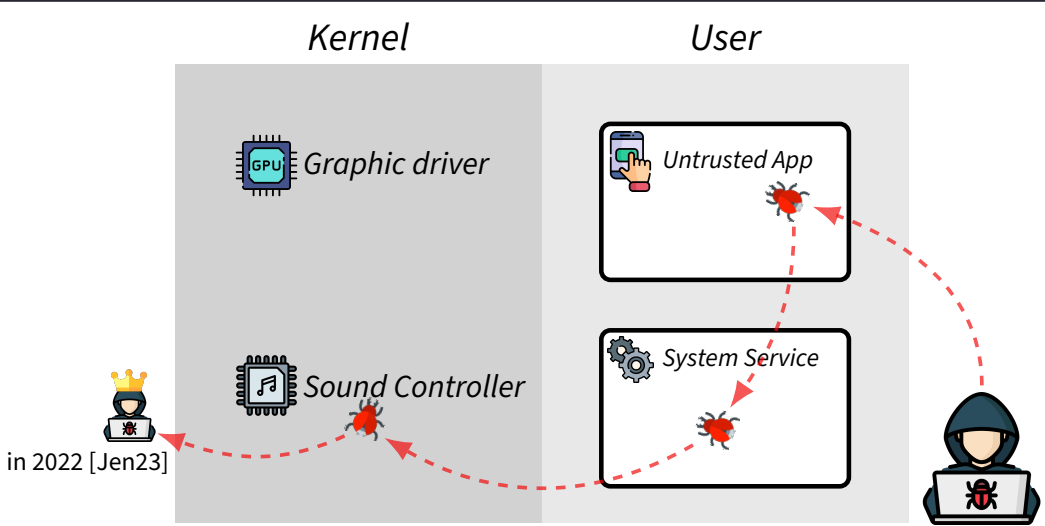
Continue with In-The-Wild Device Exploitation



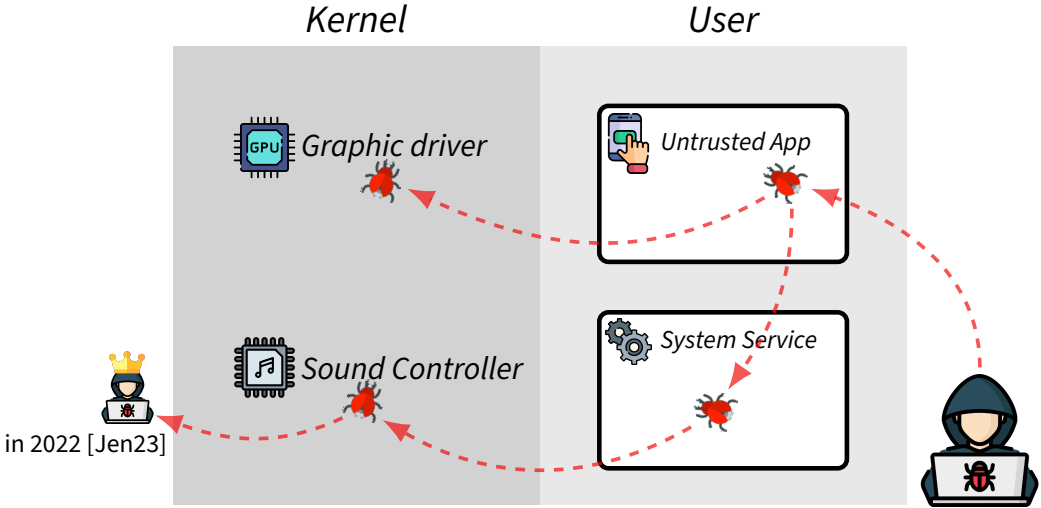
Continue with In-The-Wild Device Exploitation



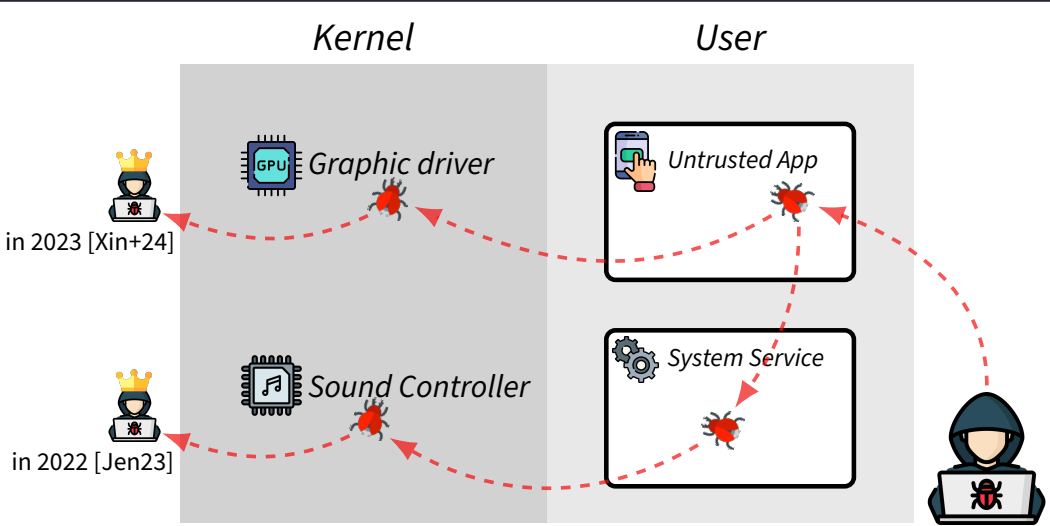
Continue with In-The-Wild Device Exploitation



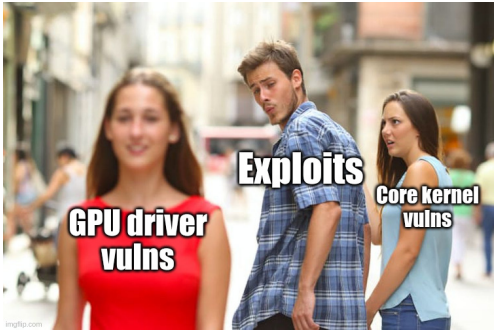
Continue with In-The-Wild Device Exploitation



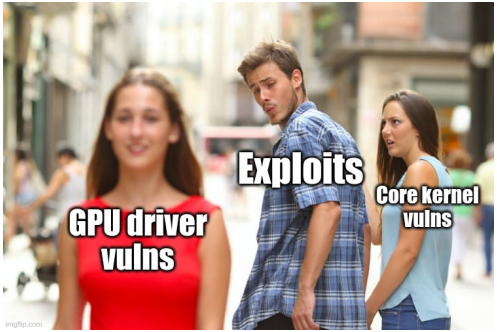
Continue with In-The-Wild Device Exploitation



GPU Driver Vulnerabilities

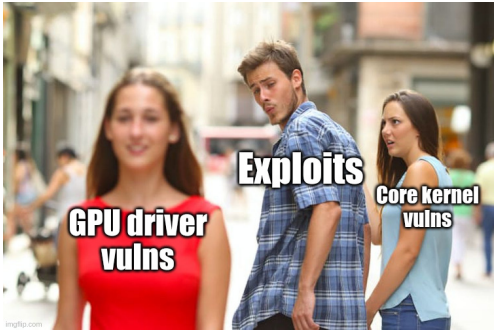


GPU Driver Vulnerabilities



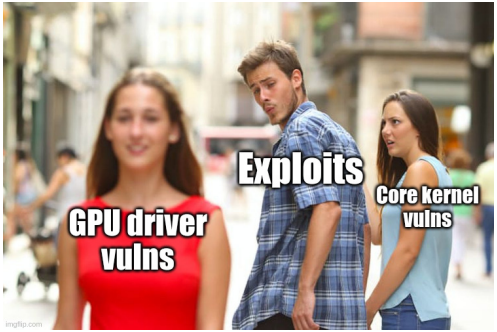
- Prime targets:

GPU Driver Vulnerabilities



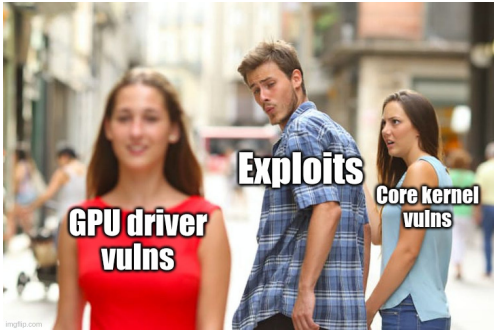
- **Prime targets:**
  - Four GPU suppliers

GPU Driver Vulnerabilities



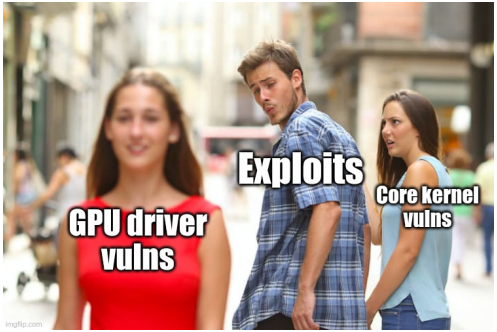
- **Prime targets:**
  - Four GPU suppliers
  - Accessible from untrusted context

GPU Driver Vulnerabilities



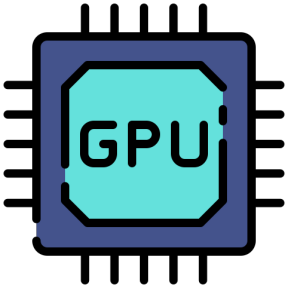
- **Prime targets:**
  - Four GPU suppliers
  - Accessible from untrusted context
  - Complex $\rightarrow$ susceptible

GPU Driver Vulnerabilities

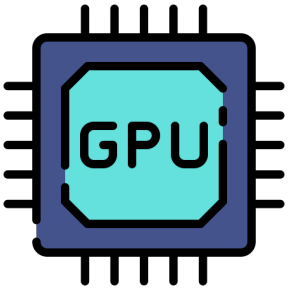


- **Prime targets:**
  - Four GPU suppliers
  - Accessible from untrusted context
  - Complex $\rightarrow$ susceptible
- **Google's 2023 annual review [SSS24]:**
  - 4 of 5 Android exploits due to GPU driver vulns
  - 1 involved core kernel

GPU Driver Security

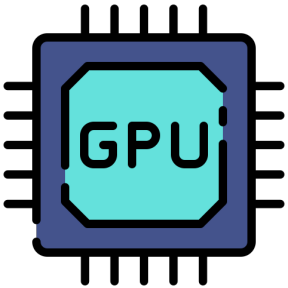


GPU Driver Security



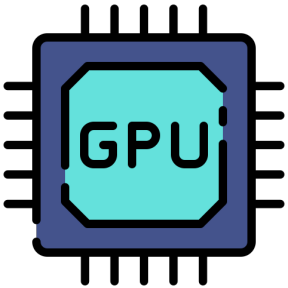
- Collaboration in 2024 [Xin+24]:

GPU Driver Security



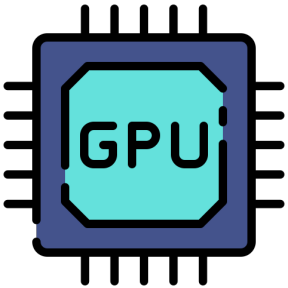
- **Collaboration in 2024 [Xin+24]:**
  - Google

GPU Driver Security



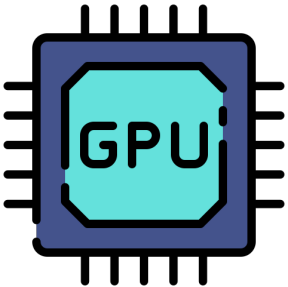
- **Collaboration in 2024 [Xin+24]:**
  - Google
  - Android Red Team

GPU Driver Security



- **Collaboration in 2024 [Xin+24]:**
  - Google
  - Android Red Team
  - ARM Product Security Team

GPU Driver Security

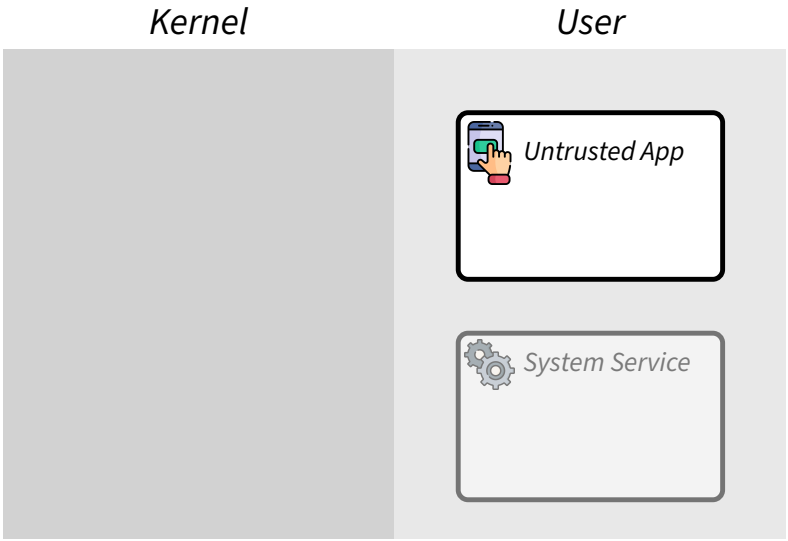


- **Collaboration in 2024 [Xin+24]:**
  - Google
  - Android Red Team
  - ARM Product Security Team
- **Goal:**
  - Enhance vulnerability detection
  - Decrease patch delays

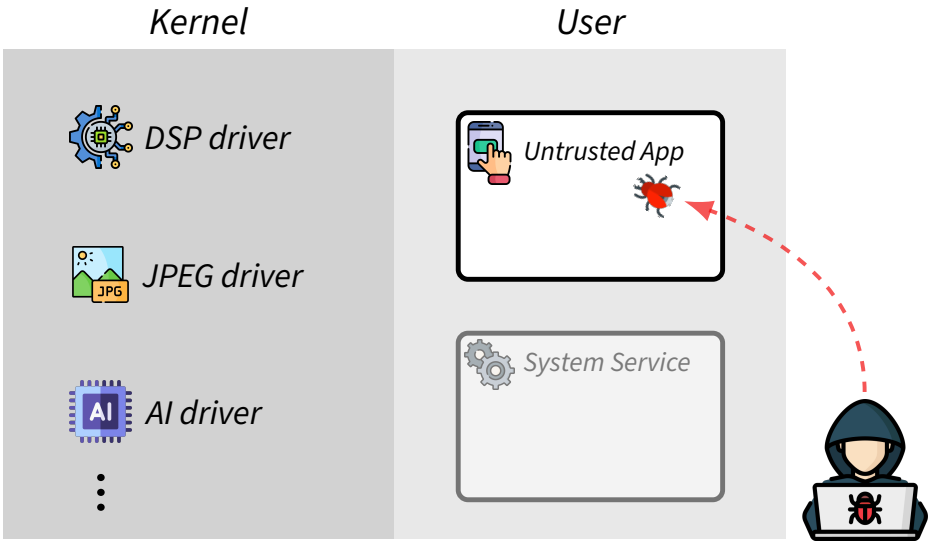
Research Question

Are there other kernel components that pose similar or even greater risks?

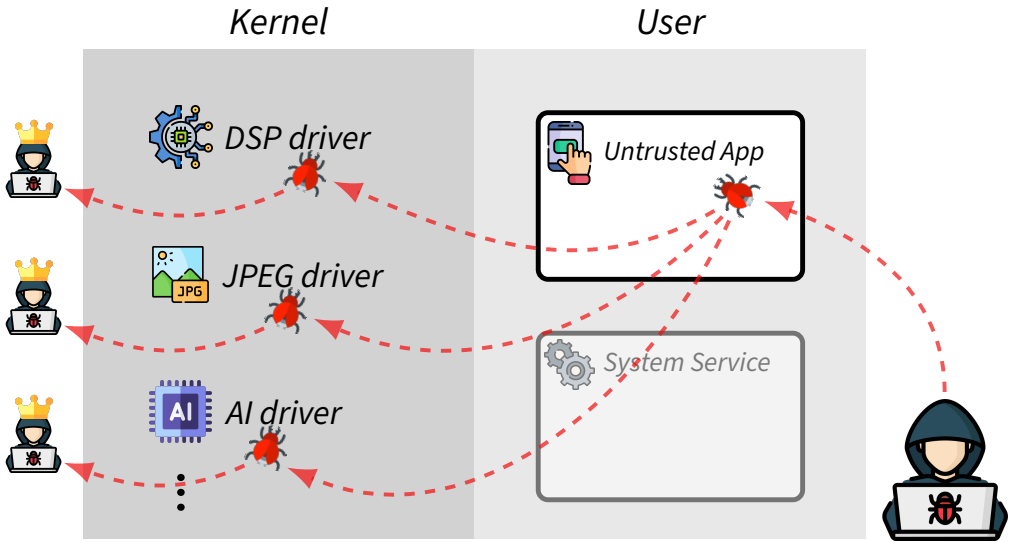
Our Contributions



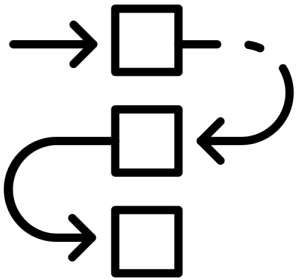
Our Contributions



Our Contributions

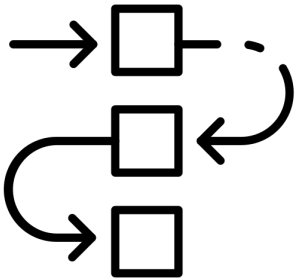


Overview

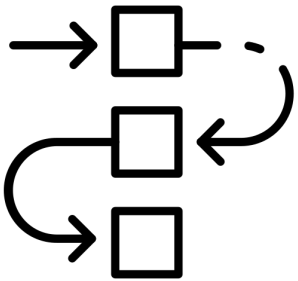


Overview

- Criterias:

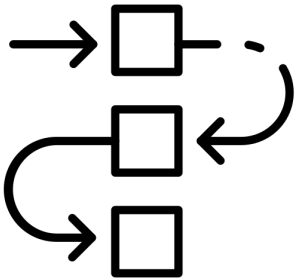


Overview



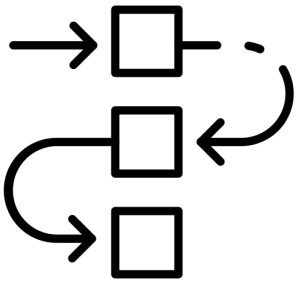
- **Criterias:**
  - **C1: Accessibility**
    Accessible from untrusted security contexts.

Overview



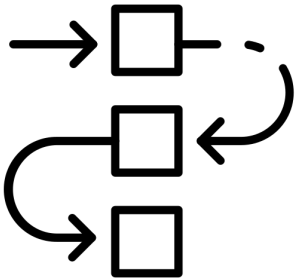
- **Criterias:**
  - **C1: Accessibility**
    Accessible from untrusted security contexts.
  - **C2: Broad Impact**
    Affect a wide range of Android devices.

Overview



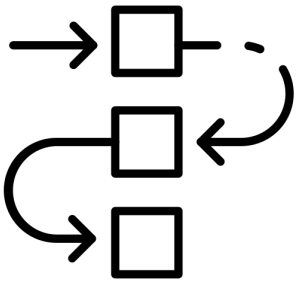
- **Criterias:**
  - **C1: Accessibility**
    Accessible from untrusted security contexts.
  - **C2: Broad Impact**
    Affect a wide range of Android devices.
  - **C3: Susceptibility**
    Exploitable vulnerabilities.

Overview



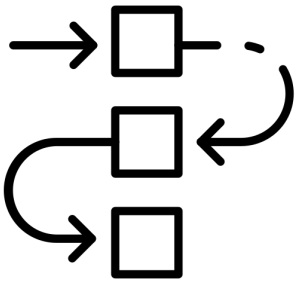
- **Criterias:**
  - **C1: Accessibility**
    Accessible from untrusted security contexts.
  - **C2: Broad Impact**
    Affect a wide range of Android devices.
  - **C3: Susceptibility**
    Exploitable vulnerabilities.
- **Workflow:**

Overview



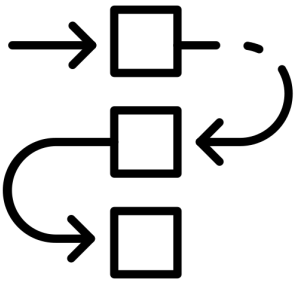
- **Criterias:**
  - **C1: Accessibility**
    Accessible from untrusted security contexts.
  - **C2: Broad Impact**
    Affect a wide range of Android devices.
  - **C3: Susceptibility**
    Exploitable vulnerabilities.
- **Workflow:**
  - Analyze firmwares (**C1** ✓)

Overview



- **Criterias:**
  - **C1: Accessibility**
    Accessible from untrusted security contexts.
  - **C2: Broad Impact**
    Affect a wide range of Android devices.
  - **C3: Susceptibility**
    Exploitable vulnerabilities.
- **Workflow:**
  - Analyze firmwares (**C1** ✓)
  - Open source info to discover vulns (**C2** ✓)

Overview



- **Criterias:**
  - **C1: Accessibility**
    Accessible from untrusted security contexts.
  - **C2: Broad Impact**
    Affect a wide range of Android devices.
  - **C3: Susceptibility**
    Exploitable vulnerabilities.
- **Workflow:**
  - Analyze firmwares (**C1** ✓)
  - Open source info to discover vulns (**C2** ✓)
  - Patch inclusion of vulns' patch (**C3** ✓)

Analysis Results

OEM	All Devices Analyzed	
	Crit Vuln	Any Vuln
	%	%
Samsung		
Xiaomi		
Asus		
Realme		
Vivo		
Oppo		
OnePlus		

Analysis Results

OEM	All Devices Analyzed	
	Crit Vuln %	Any Vuln %
Samsung	45.5	45.5
Xiaomi	67.3	71.4
Asus	75.0	100.0
Realme	56.2	62.5
Vivo	40.0	40.0
Oppo	42.9	42.9
OnePlus	85.7	85.7

Analysis Results

OEM	All Devices Analyzed		Devices with Target Drivers	
	Crit Vuln	Any Vuln	Crit Vuln	Any Vuln
	%	%	%	%
Samsung	45.5	45.5		
Xiaomi	67.3	71.4		
Asus	75.0	100.0		
Realme	56.2	62.5		
Vivo	40.0	40.0		
Oppo	42.9	42.9		
OnePlus	85.7	85.7		

Analysis Results

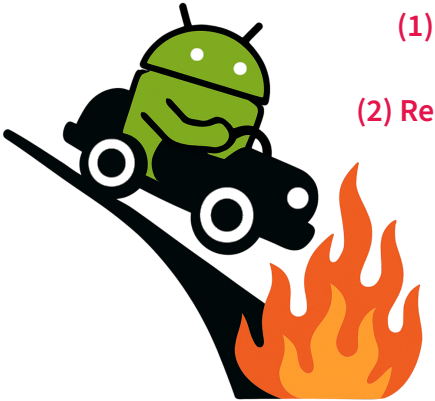
OEM	All Devices Analyzed		Devices with Target Drivers	
	Crit Vuln %	Any Vuln %	Crit Vuln %	Any Vuln %
Samsung	45.5	45.5	74.1	74.1
Xiaomi	67.3	71.4	75.0	79.5
Asus	75.0	100.0	75.0	100.0
Realme	56.2	62.5	56.2	62.5
Vivo	40.0	40.0	40.0	40.0
Oppo	42.9	42.9	42.9	42.9
OnePlus	85.7	85.7	85.7	85.7

Key Findings

Key Findings

(1) Clustering: Devices vulnerable to 1 n-day vuln are often vulnerable to many.

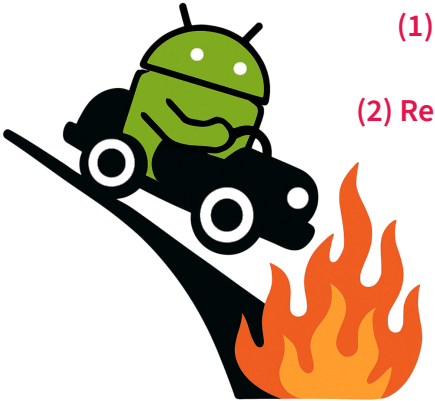
Key Findings



(1) Clustering: Devices vulnerable to 1 n-day vuln are often vulnerable to many.

(2) Replacement: Vulns are often fixed via new device models than updates.

Key Findings

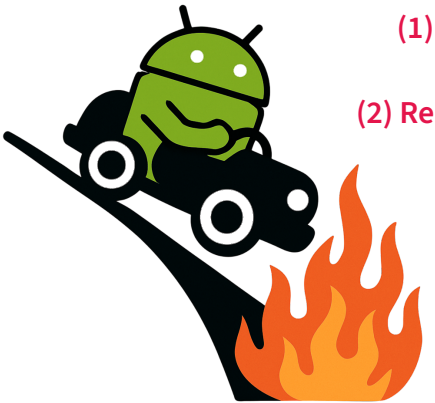


(1) Clustering: Devices vulnerable to 1 n-day vuln are often vulnerable to many.

(2) Replacement: Vulns are often fixed via new device models than updates.

(3) Delay: Patch times can exceed a year, varying by OEM, ODM, and vuln type.

Key Findings



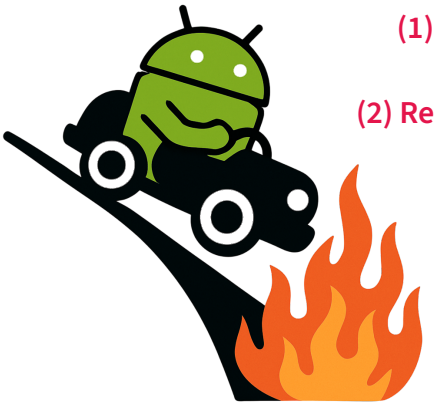
(1) Clustering: Devices vulnerable to 1 n-day vuln are often vulnerable to many.

(2) Replacement: Vulns are often fixed via new device models than updates.

(3) Delay: Patch times can exceed a year, varying by OEM, ODM, and vuln type.

(4) Reuse: PoCs for ODM driver vulns work across OEMs and timeframes.

Key Findings



(1) Clustering: Devices vulnerable to 1 n-day vuln are often vulnerable to many.

(2) Replacement: Vulns are often fixed via new device models than updates.

(3) Delay: Patch times can exceed a year, varying by OEM, ODM, and vuln type.

(4) Reuse: PoCs for ODM driver vulns work across OEMs and timeframes.

(5) Exploit: Malicious actors can weaponize n-day vulns, avoiding costly zero-days.

Key Findings

(1) Clustering: Devices vulnerable to 1 n-day vuln are often vulnerable to many.

(2) Deployment: Vulns are often fixed via new device [illegible] than updates.

[illegible] a year, varying

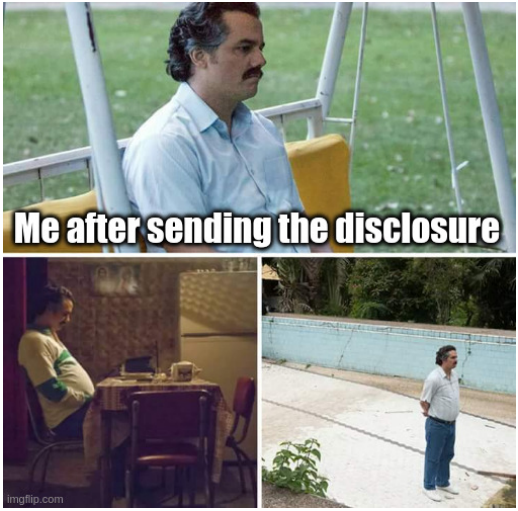
(4) Ke[illegible] [illegible] cross OEMs a[illegible]

(5) Exploit: Malicious actors can wea[illegible]n-day vulns, avoiding costly zero-days.

Concurrent research [Int24; Int25; Jen24] showed in-the-wild exploitation

Disclosure

Disclosure



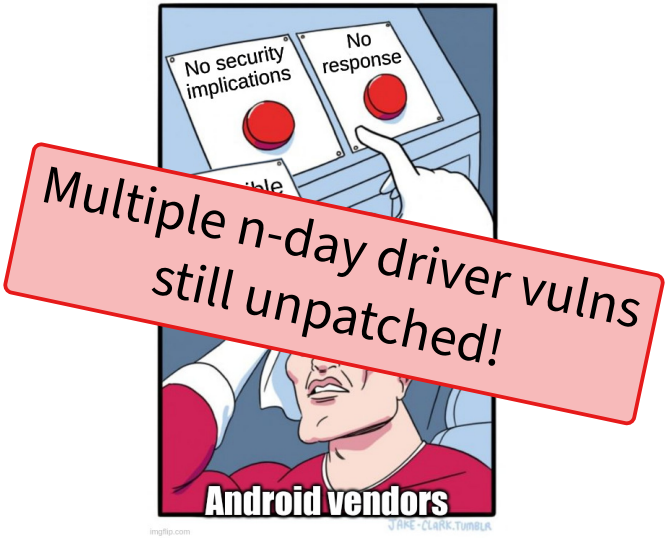
Disclosure


No security implications
No response
Android vendors
imgflip.com
JAKE-CLARK.TUMBLR

Disclosure


No security implications
No response
Infeasible
Android vendors
imgflip.com
JAKE-CLARK.TUMBLR

Disclosure



in May 2025

TU Graz
SCIENCE
PASSION
TECHNOLOGY

The Doom of Device Drivers:

Your Android Device (Most Likely) has N-Day Kernel Vulnerabilities

Lukas Maar Florian Draschbacher Lorenz Schumm Ernesto Martínez García
Stefan Mangard

August 13-15, 2025

USENIX Security

› isec.tugraz.at

References

[Int24] A. International. **"A Digital Prison": Surveillance and the suppression of civil society in Serbia**. 2024. URL: https://securitylab.amnesty.org/latest/2024/12/a-digital-prison-surveillance-and-the-suppression-of-civil-society-in-serbia/.

[Int25] A. International. **Cellebrite zero-day exploit used to target phone of Serbian student activist.** 2025. URL: https://securitylab.amnesty.org/latest/2025/02/cellebrite-zero-day-exploit-used-to-target-phone-of-serbian-student-activist/.

[Jen23] S. Jenkins. **Analyzing a Modern In-the-wild Android Exploit**. 2023. URL: https://googleprojectzero.blogspot.com/2023/09/analyzing-modern-in-wild-android-exploit.html.

[Jen24] S. Jenkins. **The Qualcomm DSP Driver - Unexpectedly Excavating an Exploit**. 2024. URL: https://googleprojectzero.blogspot.com/2024/12/qualcomm-dsp-driver-unexpectedly-excavating-exploit.html.

References II

[SSS24] M. Stone, J. Semrau, and J. Sadowski. **We're All in this Together: A Year in Review of Zero-Days Exploited In-the-Wild in 2023.** 2024. URL: https://storage.googleapis.com/gweb-uniblog-publish-prod/documents/Year_in_Review_of_ZeroDays.pdf.

[Xin+24] X. Xing, E. Rodionov, J. Bottarini, A. Bacchus, A. Chaudhary, L. Fawcett, and J. Artgole. **Google & Arm - Raising The Bar on GPU Security.** 2024. URL: https://security.googleblog.com/2024/09/google-arm-raising-bar-on-gpu-security.html.