acm sigmobile

MobiCom 2020

The 26th Annual International Conference on Mobile Computing and Networking

Experience: Aging or Glitching? Why Does Android Stop Responding and What Can We Do About It?

Mingliang Li, Hao Lin, Cai Liu, Zhenhua Li

Feng Qian, Yunhao Liu, Nian Sun, Tianyin Xu

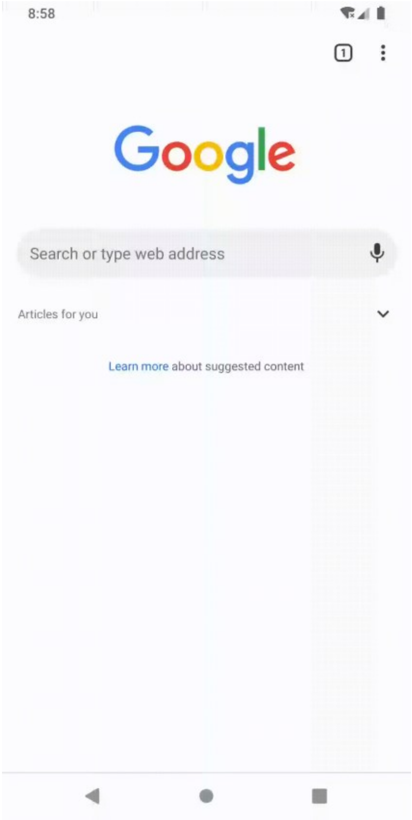
Outline

1. Background
2. Measurement Design
3. Key Findings
4. Addressing the Problem
5. Summary

1. Responsiveness of Android

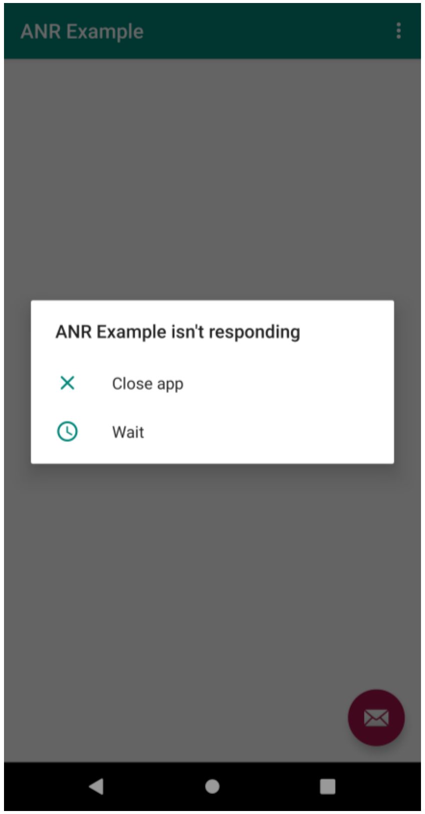
- □ Responsiveness: a key metric for user experience
- □ Till now poor responsiveness on Android is still prevalent
- □ Many Android users may have experienced unresponsiveness

Typical unresponsiveness

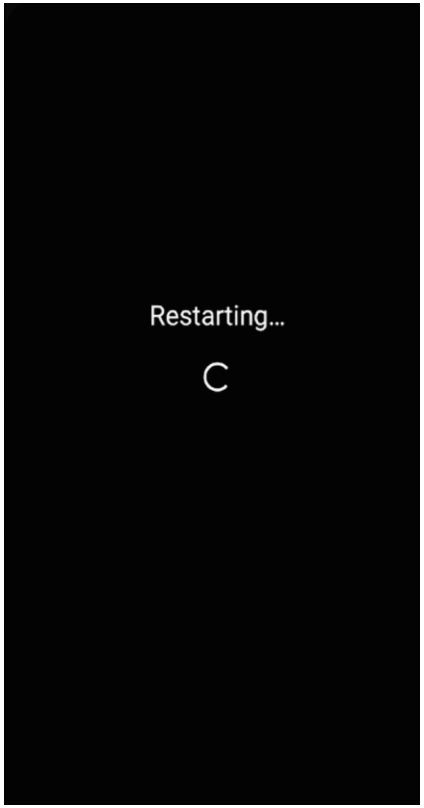

8:58
Google
Search or type web address
Articles for you
Learn more about suggested content

1. ANR and SNR

- □ In particular, ANR and SNR directly disrupt user experience
- □ Still unknown: their prevalence, characteristics, and root causes



ANR Example isn't responding



Restarting...

Application **N**ot **R**esponding (ANR)

Either kill the app or wait

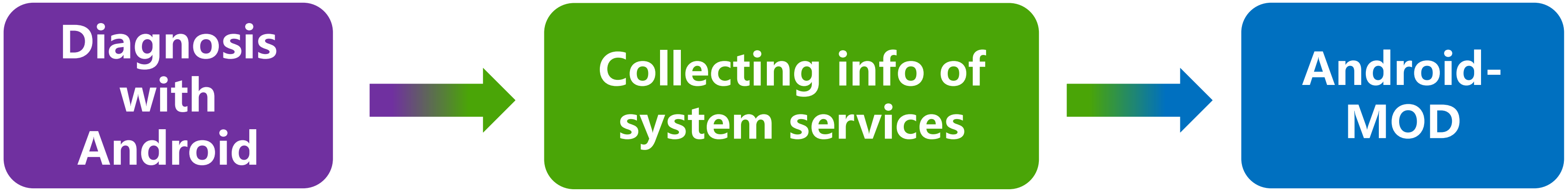
System **N**ot **R**esponding (SNR)

A system restart will be forced

2. Continuous Monitoring Infrastructure

- **Android's original diagnostic mechanism is not enough!**
  - Original: CPU, memory, call stacks of relevant processes
  - Lacking: visibility into critical system services

- **Our continuous monitoring infrastructure**
  - Android-MOD: a customized Android system
  - Modifying vanilla Android versions 7.0, 8.0, and 9.0
  - Lightweight: negligible runtime overhead



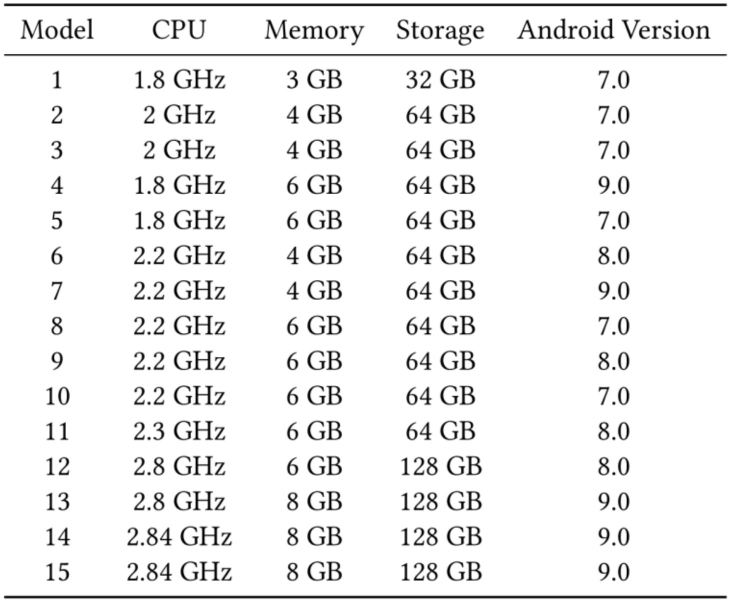
Lacking insights into important system services

Need to modify the Android framework

2. Crowdsourcing Measurement

- □ We collaborate with Xiaomi, a major phone vendor
- □ We invite all its users (~250M) to participate, ≥30,000 opt in
- □ They upgrade to Android-MOD to record ANR/SNR data

- □ The measurement lasted for three weeks, involving a wide range of phones across 15 different models

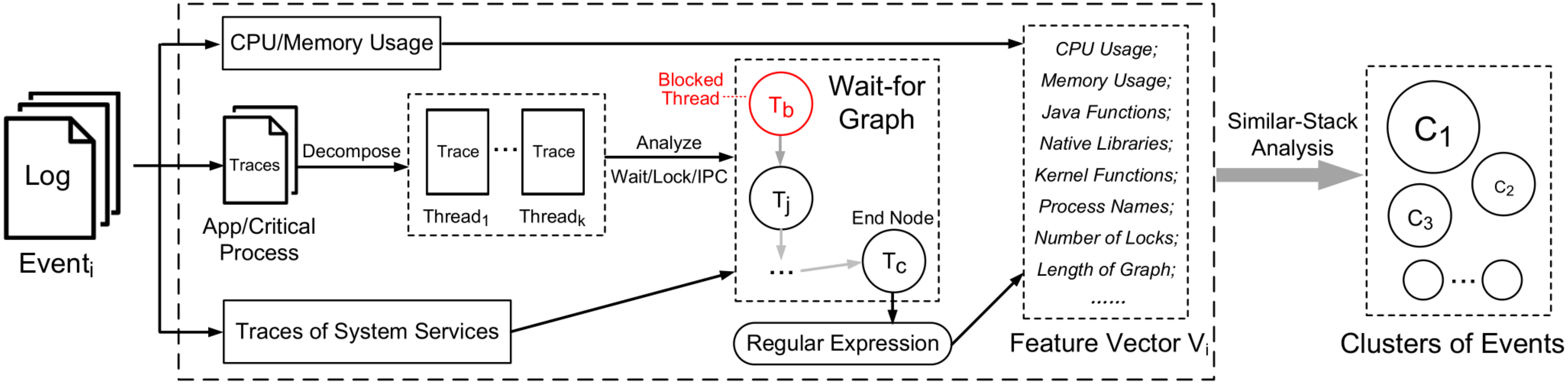


Model	CPU	Memory	Storage	Android Version
1	1.8 GHz	3 GB	32 GB	7.0
2	2 GHz	4 GB	64 GB	7.0
3	2 GHz	4 GB	64 GB	7.0
4	1.8 GHz	6 GB	64 GB	9.0
5	1.8 GHz	6 GB	64 GB	7.0
6	2.2 GHz	4 GB	64 GB	8.0
7	2.2 GHz	4 GB	64 GB	9.0
8	2.2 GHz	6 GB	64 GB	7.0
9	2.2 GHz	6 GB	64 GB	8.0
10	2.2 GHz	6 GB	64 GB	7.0
11	2.3 GHz	6 GB	64 GB	8.0
12	2.8 GHz	6 GB	128 GB	8.0
13	2.8 GHz	8 GB	128 GB	9.0
14	2.84 GHz	8 GB	128 GB	9.0
15	2.84 GHz	8 GB	128 GB	9.0

Hardware and OS configurations of our measured phone models

2. Root Cause Analysis Pipeline

- □ System developers usually analyze ANR/SNR logs by hand
- □ To scale, we develop an automated analysis pipeline
- □ It classifies ANR/SNR events with the same root cause to a cluster
- □ Validation: no false positives



Work flow of our developed automated root cause analysis pipeline

2. Root Cause Analysis Pipeline

Constructing Wait-for Graph

- Target: find the critical thread most related to the event
- Decompose the process-level call stacks into thread-level call stacks
- Only one thread is marked as *Blocked*, but is not the critical thread
- Wait-for graph based on the wait, lock, and IPC information in the call stack
- The end node in the graph is deemed as the critical thread

Similar-stack Analysis

- Remove irrelevant information such as memory address
- Reconstruct the call stack into a feature vector
- Calculate similarity in a “split-and-merge” manner
- Classify events into a root-cause cluster based on similarity

CPU Usage;
Memory Usage;
Java Functions;
Native Libraries;
Kernel Functions;
Process Names;
Number of Locks;
Length of Graph;
……

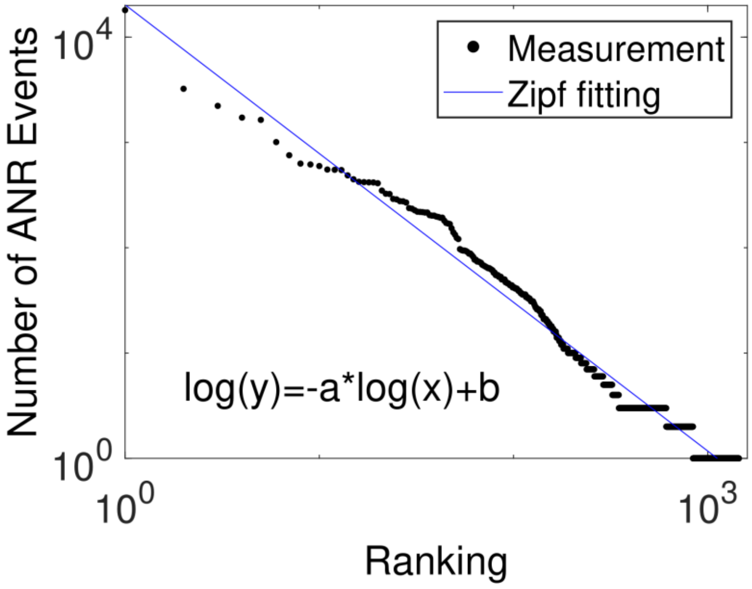
Feature Vector

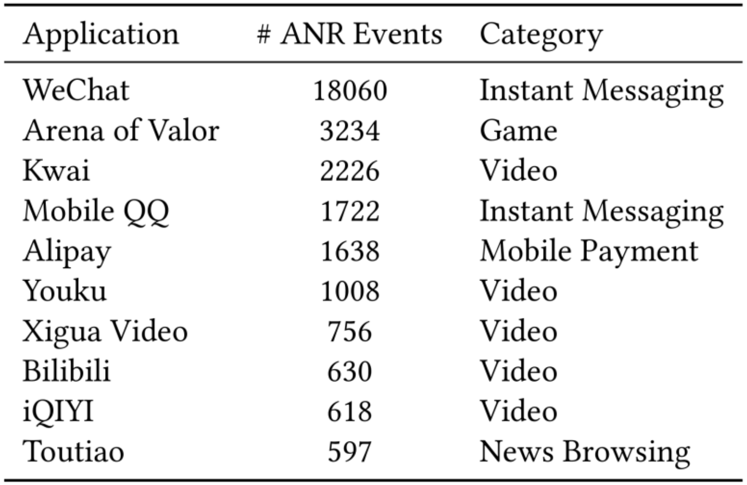
3. Key Findings: Prevalence & Mobile Apps

□ Both ANR and SNR occur prevalently on all the 15 models

□ ANR occurrences on apps are skewed

□ 60% ANR events are attributed to only the top-10 (0.7%) apps



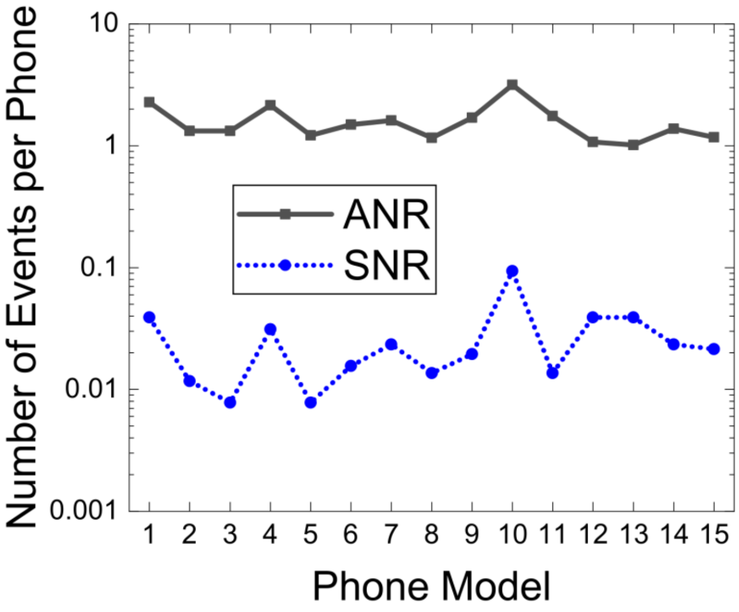


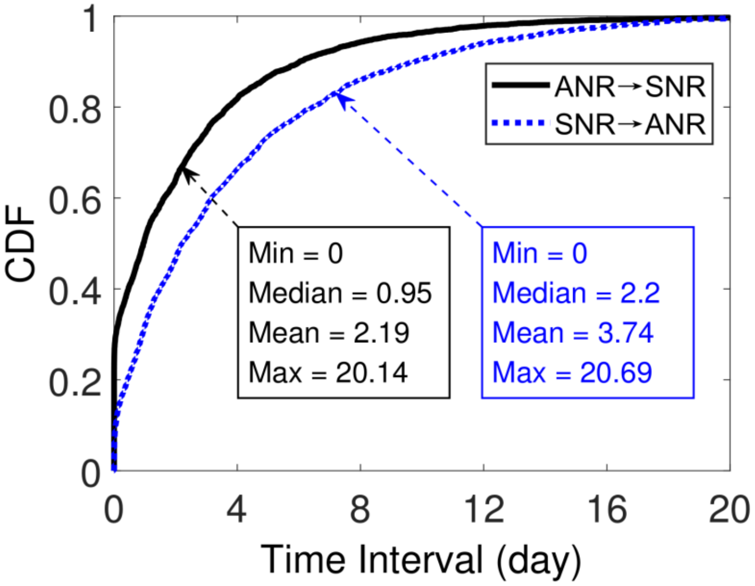
Application	# ANR Events	Category
WeChat	18060	Instant Messaging
Arena of Valor	3234	Game
Kwai	2226	Video
Mobile QQ	1722	Instant Messaging
Alipay	1638	Mobile Payment
Youku	1008	Video
Xigua Video	756	Video
Bilibili	630	Video
iQIYI	618	Video
Toutiao	597	News Browsing

Top-10 apps ordered by number of ANR events.

3. Key Findings: Correlations

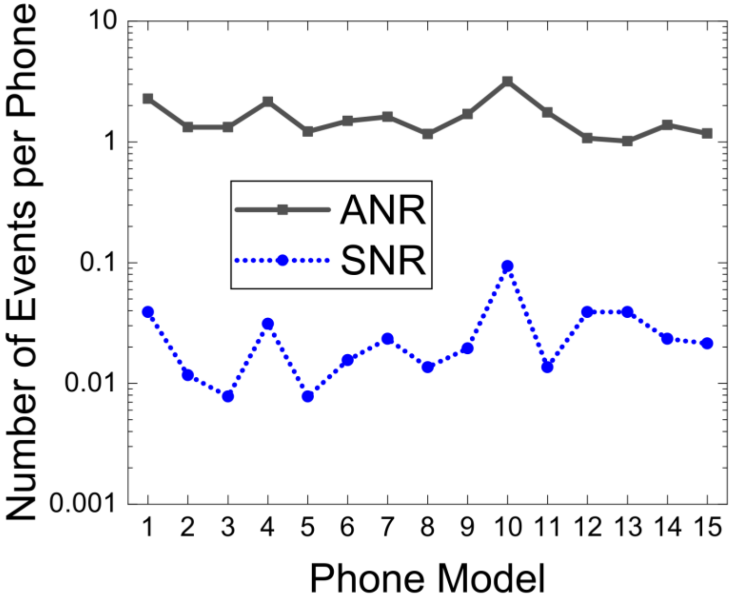
- □ ANR and SNR events are highly correlated in terms of occurrences
- □ But there is no causality between ANR and SNR events
- □ It suggests that ANR/SNR tend to be caused at the system level

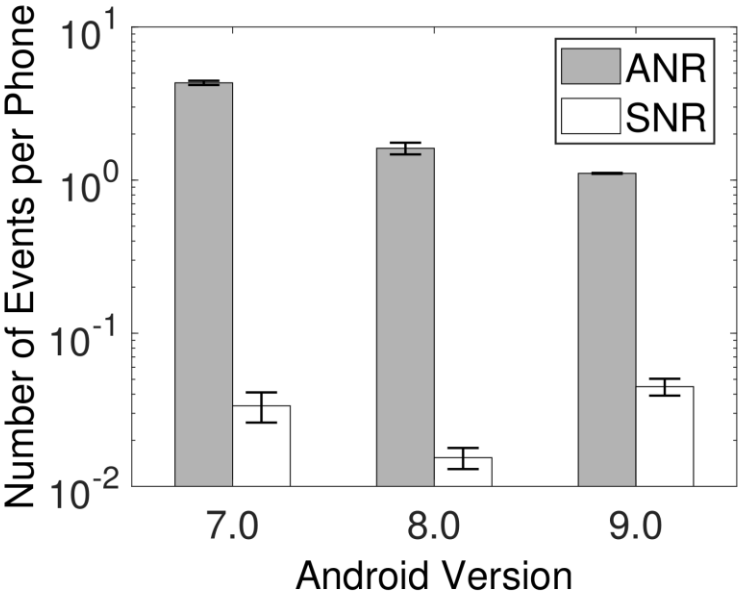




3. Key Findings: Hardware/OS

- □ No correlations between hardware and the prevalence of ANR/SNR
- □ Better hardware even appears to aggravate SNR
- □ Newest OS: less ANR, more SNR





3. Key Findings: Root Causes

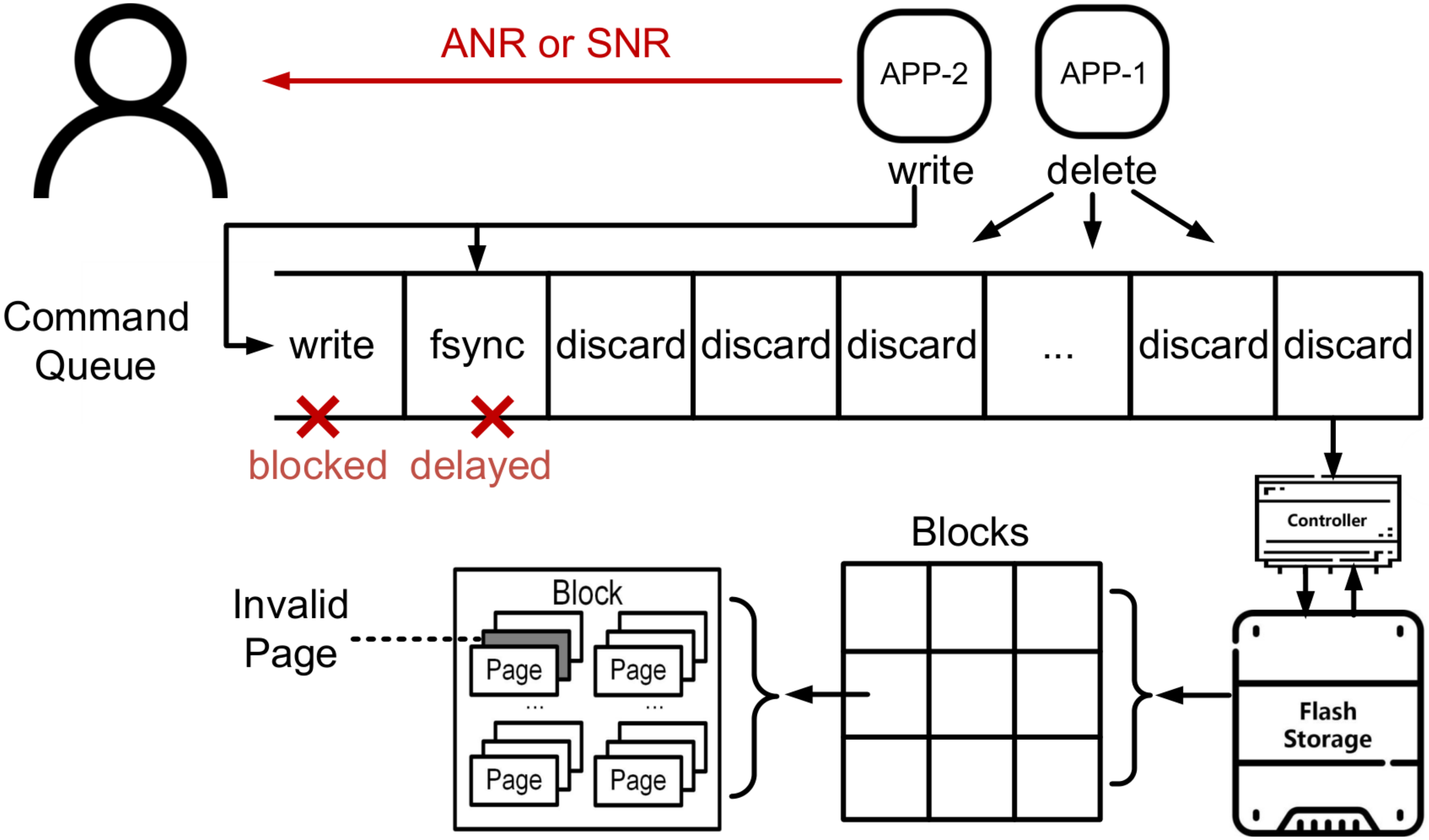
□ Four major root causes of ANR/SNR

- □ Pathological Write Amplification Mitigation (WAM, 35%)
- □ Lock contention among system services (21%)
- □ Insufficient memory (18%)
- □ App-specific defects (26%)

Pathological WAM can be fundamentally eliminated!

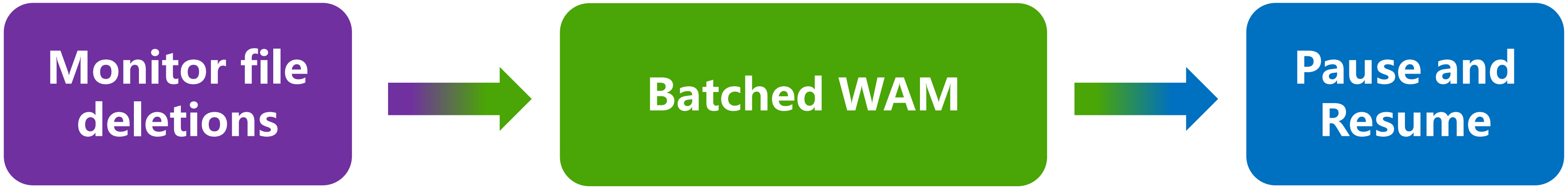
3. Measurement Findings: Root Causes

- The issue with Android's implementation of WAM (real-time)



4. Eliminating the Largest Root Cause

- □ Batched WAM: once a day, trim the entire storage
- □ Performing practical on-demand WAM
- □ Support for pausing and resuming



Monitor file deletions

Batched WAM

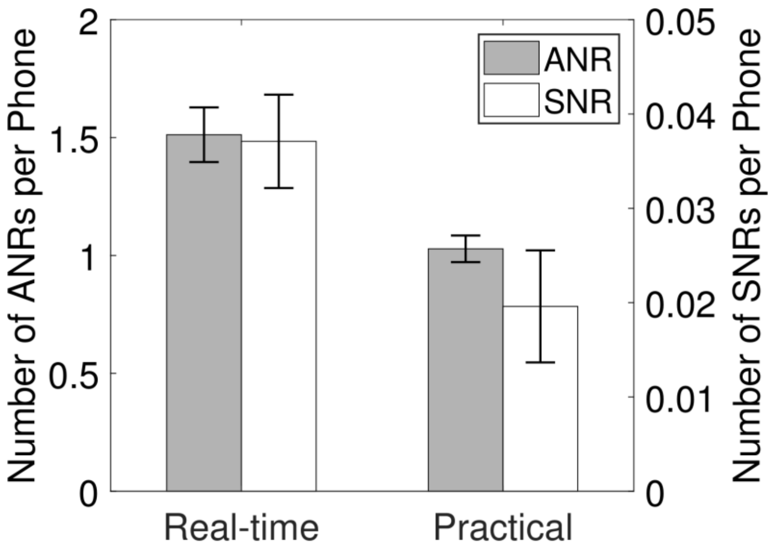
Pause and Resume

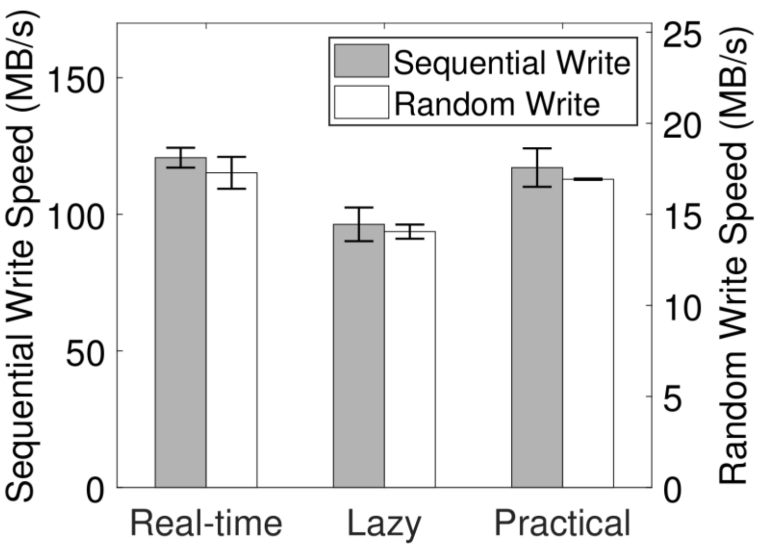
When the amount reaches a data-driven threshold

Screen lock/unlock

4. Commercial Deployment & Evaluation

- □ Patched our proposed WAM mechanism to Android-MOD
- □ Invited the original 30,000 users to upgrade (14,000 opted in)
- □ Reduce 32% of ANR 47% of SNR per phone
- □ Almost all of the WAM-induced ANR/SNR have been avoided
- □ The random (sequential) write speed decreases by an average of merely 2% (3%), compared to real-time WAM





4. Commercial Deployment & Evaluation

□ Adopted by 5 stock Android systems whose corresponding hardware models are as listed below, benefiting ∼20M Android users

- Sagit
- Cepheus
- Lavender
- Rosy
- Jasmine

□ Besides Xiaomi’s MIUI OS, Huawei EMUI OS has also optimized the native WAM mechanism in a similar way

5. Summary of Contributions

- We conduct the first large-scale measurement of ANR and SNR of Android in the wild. We discover that ANR and SNR are more of a software issue than a hardware issue

- We present our end-to-end data collection and analysis pipeline for deeply understanding ANR and SNR.

- We diagnose and address the largest root cause of ANR and SNR. After real-world deployment, our solution reduces 32% ANR and 47% SNR events while only decreasing 3% of the data write speed

- Code and data at https://Android-Not-Respond.github.io