### An Investigation of the Android Kernel Patch Ecosystem

Zheng Zhang<sup>1</sup>, Hang Zhang<sup>1</sup>, Zhiyun Qian<sup>1</sup>, Billy Lau<sup>2</sup> <sup>1</sup>UC Riverside, <sup>2</sup>Google Inc

#### **Openness of Linux and Android**



\* Syzbot and the Tale of Thousand Kernel Bugs - Dmitry Vyukov, Google, 2018

#### Android kernel ecosystem



by Richi Jennings on October 7, 2019

- The relationship between the upstream and downstream e.g., initial patch? how does it propagate?
- The timeliness of patch propagation e.g., typical delay in each step? bottleneck?
- The factors that influence the patch propagation e.g., current best practices? how to improve?

# Challenges

- Challenge 1: Not completely open source
  - OEM provides mostly binary kernel images
    Fiber: H. Zhang and Z. Qian. Precise and accurate patch presence test for binaries. USENIX Security, 2018
  - Good accuracy with some improvements (~95%)

- Challenge 2: Decision making a blackbox
  - Complex ecosystem/Poor documentation
  - Information from multiple sources
  - Measurements/Inference

#### Measurements: overall

Dataset:

- 20 phone models, 500+ instances
- 400 Linux/Qualcomm CVEs: Android/Pixel security bulletin From the beginning to 5/2019

Bottleneck (Linux CVEs):

Qualcomm mainline -> Qualcomm stable/OEMs



# **Knowledge of security patches**

- Lack of knowledge
  - Patches == fixing serious security flaws?
- Knowledge generation
  - Time-consuming triage to determine security impact
- Knowledge propagation
  - Not efficient















#### Improvements

- Automated patch triage (security or not)
  - Extracting security knowledge
  - Useful for all kernels
- More efficient knowledge propagation
  - Linux: develop its own notification mechanisms
  - OEM: Pay attention to Qualcomm notifications
- Redesign the mechanism: more merges/forks
  - Follow Linux LTS directly in downstreams (Ongoing effort @ Google)
  - still require more efforts

# Thanks for your attention!