# Vulcan: Lessons on Reliability of Wearables through State-Aware Fuzzing

**Edgardo Barsallo Yi, Heng Zhang, Kefan Xu, Amiya Maji, Saurabh Bagchi**

Dependable Computing System Lab (DCSL)
Purdue University






## Outline

- **Motivation**
- Vulcan Design
- State-aware Study Results
- System Reboots Analysis
- Lessons Learned
- Conclusion

## Motivation

- Reliability of Android is well explored but wearables come with a new set of challenges
  - Wearable devices are **sensor rich** and have **limited resources**
  - User Interface (UI) is designed to require **minimal human interaction** (micro transactions)
  - **Unique communication pattern** where many apps are tethered with a mobile counterpart
- A popular use-case is monitoring, accumulating and disseminating of **health and fitness** data
- Existing testing approaches overlook the above unique features of the wearable ecosystem



PURDUE UNIVERSITY

## Outline

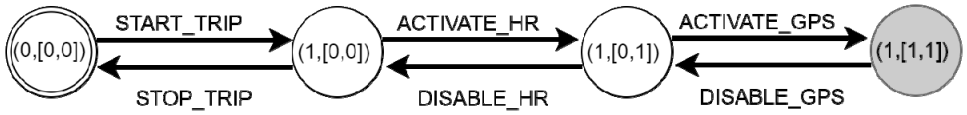
- Motivation
- **Vulcan Design**
- State-aware Study Results
- System Reboots Analysis
- Lessons Learned
- Conclusion



PURDUE UNIVERSITY

# State Model Parameters

- **Sensor activity:** Status of a sensor (activated or deactivated)
- **Inter-device communication:** Presence of active communication between mobile and wearable




**Vulnerable State**: States with maximum concurrent activity

INIT

COMM = 1
SENSORS = [0, 0]

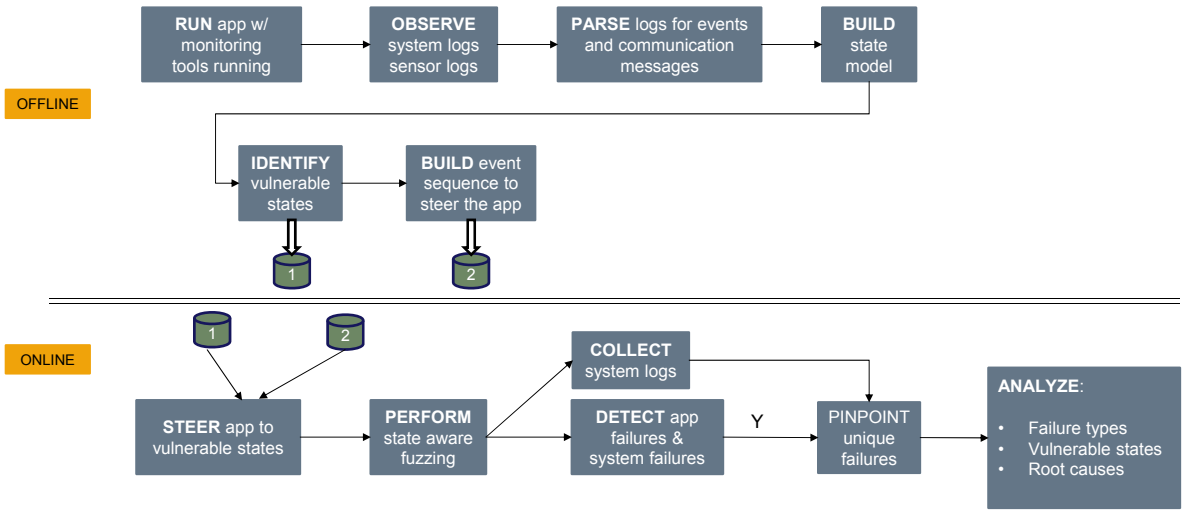
COMM = 1
SENSORS = [0, 1]

COMM =1
SENSORS = [1, 1]



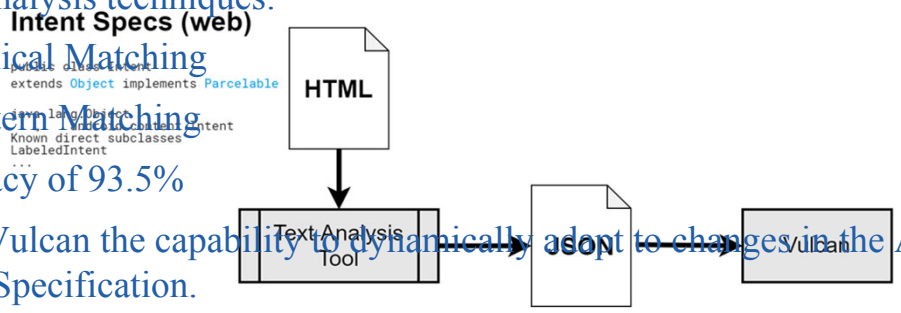


# Vulcan Workflow



# Automated Intent Specification Generation

- We designed a text analysis tool to parse the Android Specification to generate semi-valid Intents
- Core analysis techniques:
  - Lexical Matching
  - Pattern Matching
- Accuracy of 93.5%
- Adds Vulcan the capability to dynamically adapt to changes in the Android Intent Specification.








# Fuzzing Strategies

- Intent Injection Fuzzing

| | |
|---|---|
| `{act=ACTION.RUN, cmp=some.component.name}` | `{act=fitness.TRACK, cmp=some.component.name}` |

- Communication Fuzzing: empty

| | |
|---|---|
| `[/getOffDismissed, (SomeMessage)]` | `[/getOffDismissed, (null)]` |

- Communication Fuzzing: random

| | |
|---|---|
| `[/getOffDismissed, (SomeMessage)]` | `[/getOfDismissed, (11000111)]` |

## Evaluation through State-aware Fuzzing

- We evaluated 100 apps and sent over 1M Intents to fuzz the Wear OS apps
- Baselines: Ape [ICSE '19], QGJ [DSN '18], Monkey
- We designed the experiments based on the following goals:
  - Evaluate the effectiveness of a state-aware fuzzing strategy.
  - Evaluate the degree of concurrency on application reliability. We developed a Manipulator app to stress the apps and the device by activating sensors externally



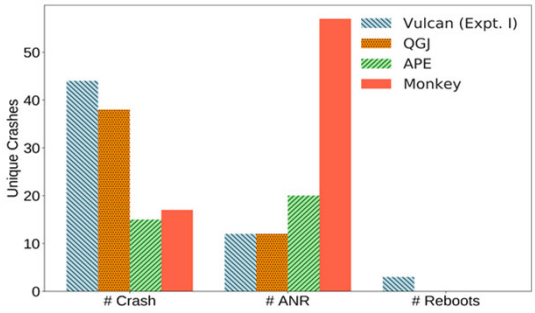


## Outline

- Motivation
- Vulcan Design
- **State-aware Study Results**
- System Reboots Analysis
- Lessons Learned
- Conclusion

## State-aware Injections are More Effective

- State-aware injections are more effective than state-agnostic tools in triggering crashes and reboots
- Sensor activation has a negative effect in the overall reliability of the system




| State | #ANR | #Crashes | #Reboots |
|---|---|---|---|
| Vulcan (Expt. I) | 12 | 44 (39, 5) | 3 (3, 0) |
| Vulcan (Expt. II) | 20 | 45 (40, 5) | 12 (12, 0) |
| QGJ | 12 | 38 | 0 |
| Monkey | 57 | 17 | 0 |
| APE | 20 | 15 | 0 |

**Impact of Device State in the Reliability**

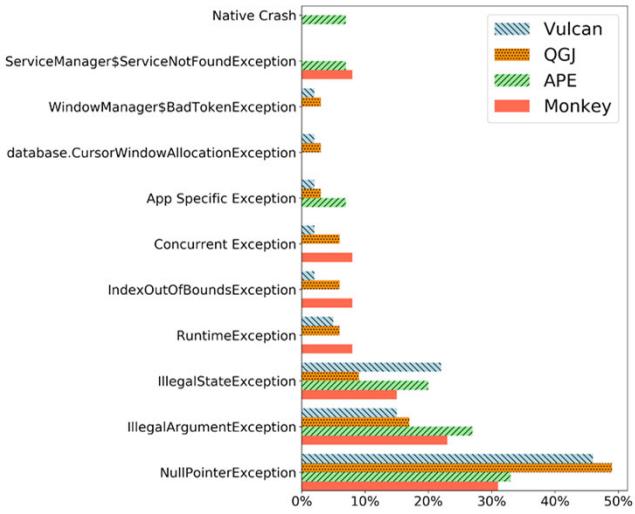
- ANR and System Reboots were more frequent on states with higher sensor activity



PURDUE UNIVERSITY

## Distribution of Exception Types

- `NullPointerException` dominates across all tools as the main cause of failure
- Most crashes can be avoided by doing exception handling in the apps

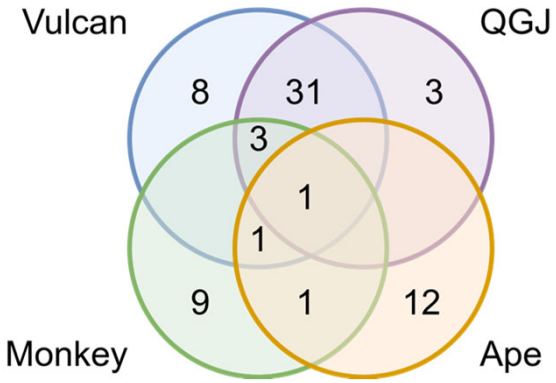





PURDUE UNIVERSITY

## Unique Crashes across Tools

- QGJ and Vulcan have a large degree of overlap primarily because they use similar Intent injection campaigns
- Vulcan is able to trigger 8 crashes not triggered by QGJ






## QGJ vs Vulcan

- **Efficiency**: Vulcan is 5.5X more efficient than QGJ in inducing unique crashes through Intent fuzzing with less Intents
- **Failure Types**: Vulcan was able to identify 5 failures related to inter-device communication. These failures were mostly due to `IllegalStateException`
- **Deterministic System Reboots**: In our experiments, QGJ did not trigger any system reboots. We identified that apps with high concurrency often trigger system reboots using Vulcan – *Vulcan was able to trigger system reboots deterministically on these apps*

## Outline

- Motivation
- Vulcan Design
- State-aware Study Results
- **System Reboots Analysis**
- Lessons Learned
- Conclusion



PURDUE UNIVERSITY

## System Reboots due to Resource Starvation

- Our results show that it is possible to trigger system reboots on Wear OS **without any system-level or root privileges**
- ***Watchdog*** is a protection mechanism to prevent the wearable from becoming unresponsive
- The root cause of the System Reboot is related to **lock handling** in the OS
- ***Watchdog*** kills the System Server process if any monitored component is hung, triggering a **system reboot**

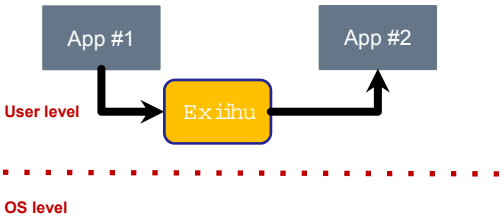


PURDUE UNIVERSITY

## Mitigation of System Reboots

- Use an Intent buffer to alleviate resource starvation and thereby prevent system reboots
- Intents sent from one app to another are stored in the buffer. Then a **Fetcher process** fetches one Intent every time
- System Intents (**trusted**) can bypass our buffer
- We tested our solution in a user study with 15 users and only one noted a significant difference in the performance due latency introduced by the Intent Fetcher

**Implementation**








## Outline

- Motivation
- Vulcan Design
- State-aware Study Results
- System Reboots Analysis
- **Lessons Learned**
- Conclusion

## Lessons Learned

- **Input Handling**: Improper input validation of Intents is still a **major cause for crashes** in Wear OS
- **Android – Wear OS Code Transfer**: Legacy code in Wear OS makes wearable apps vulnerable to the injection of `KEYCODE_SEARCH` key
- **Error Propagation**: Vulnerabilities related to ongoing synchronization between mobile and wearable can lead to **error propagation**
- **System Reboots:** Resource starvation on Wear OS can lead to system reboots





## Outline

- Motivation
- Vulcan Design
- State-aware Study Results
- System Reboots Analysis
- Lessons Learned
- **Conclusion**

## Conclusion

- State-aware fuzzing leads to more app crashes compared to state-agnostic fuzzing
- It is possible to deterministically reboot a wearable device from a user app, no system-level or root privileges, by targeting specific states. Besides, our POC solution based on an Intent buffer helps to prevent the system reboot
- Lessons for improving the wearable ecosystem are *better exception handling*, *type checking of inter-device communication messages*, and *diagnosing and terminating components that starve sensor resources*



PURDUE UNIVERSITY

## Q&A

Thank you!



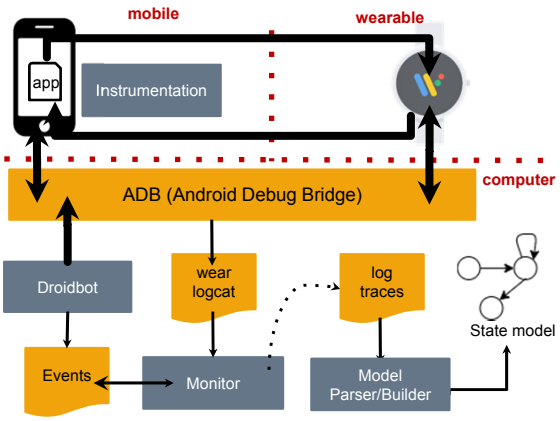
PURDUE UNIVERSITY

# Extra Slides

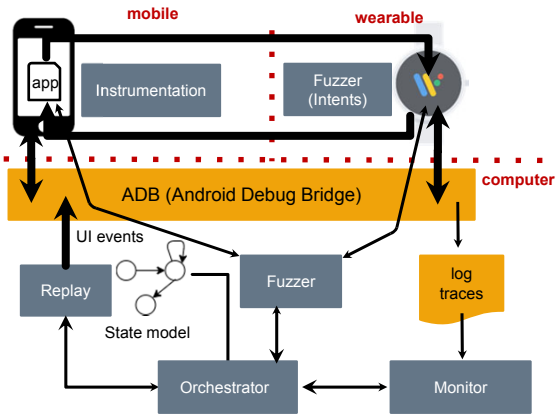


## Vulcan Architecture

### Offline Training




### Fuzz Application



## Device State affects the Reliability

| State | #ANR | #Crashes | #Reboots |
|---|---|---|---|
| Vulcan (Expt. I) | 1 | 10 | 2 |
| Vulcan (Expt. II) | 12 | 10 | 3 |
| Vulcan (Expt. III) | 9 | 10 | 3 |
| QGJ | 2 | 8 | 0 |
| Monkey | 18 | 5 | 0 |
| APE | 10 | 0 | 0 |

Failure manifestation for apps that **use sensors**.

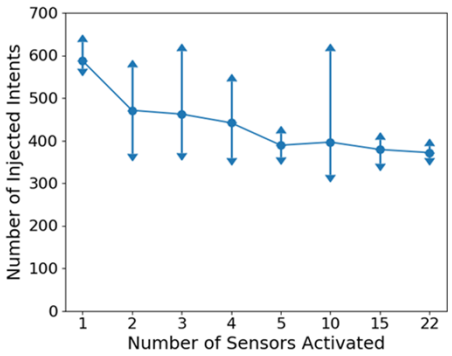
| State | #ANR | #Crashes | #Reboots |
|---|---|---|---|
| Vulcan (Expt. I) | 12 | 44 (39, 5) | 3 (3, 0) |
| Vulcan (Expt. II) | 20 | 45 (40, 5) | 12 (12, 0) |
| QGJ | 12 | 38 | 0 |
| Monkey | 57 | 17 | 0 |
| APE | 20 | 15 | 0 |

Failure manifestation for apps that do **not use sensors**. In parenthesis (Intent Fuzzing Result, Communication Fuzzing Result).



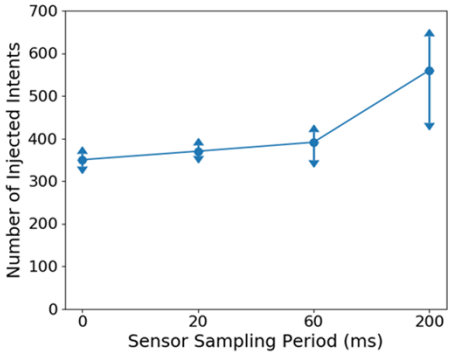


## Effect of Load on System Reboots




As the number of sensors activated increases, we need fewer number of Intents to trigger a system reboot.

The faster the sensors are sampled, the more resources they consume and therefore, it requires fewer Intents to trigger system reboots.



## Contributions

- **State-aware Fuzzing**: We present a state-aware fuzzing tool for the Wear OS ecosystem
- **Higher Failure Activation**: We show that the stateful fuzzing can increase the fault activation rate on Wear OS compared to prior works
- **System Reboot**: We demonstrate that is possible to trigger system reboots deterministically through Intent injection, without system-level or root privileges. We designed and implemented a POC solution to prevent these system reboots