Antonio Ken Iannillo

Dependability Assessment of Android OS

Tutor: prof. Domenico Cotroneo

XXX Cycle - Third Year Presentation

DIETI UNINA

DEpendable Systems and Software Engineering Research Team

DESSERT

CRITIWARE s.r.l.
Innovating Critical Systems Engineering


bring
your own
device
INTERNET of
THINGS
Antonio Ken Iannillo

"Poor quality of software can result in serious damage to the brand value of an organization and often incurs huge repair costs"

Users cannot afford any
failure that could

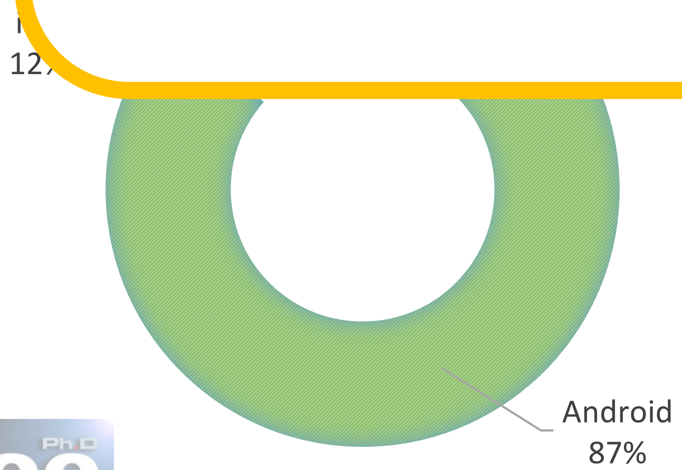
SMARTPHONE MUST BE
DEPENDABLE

financial capital they
own

Mobile OS and Android

- Managing services and re… o…
- M… c… te… to…

HOW CAN A MANUFACTURER **ASSESS THE DEPENDABILITY** OF ITS MOBILE DEVICES?



“The **dependability** of a system is the ability to avoid service failures that are more frequent and more severe than is acceptable”

A.Avizienis, J.Laprie, B.Randell, and C.Landwehr, “Basic Concepts and Taxonomy of Dependable and Secure Computing,” *IEEE Trans. on Dependable and Secure Computing*

Fault → **Error** → **Failure** → **Fault**

Fault

- an attribute of the system that leads to an error
- e.g., a missing event handler initialization instruction in the mobile OS code

Error

- an erroneous internal state that propagates within the system and eventually turns into a failure
- e.g., a mobile OS internal service has a missing event handler

Failure

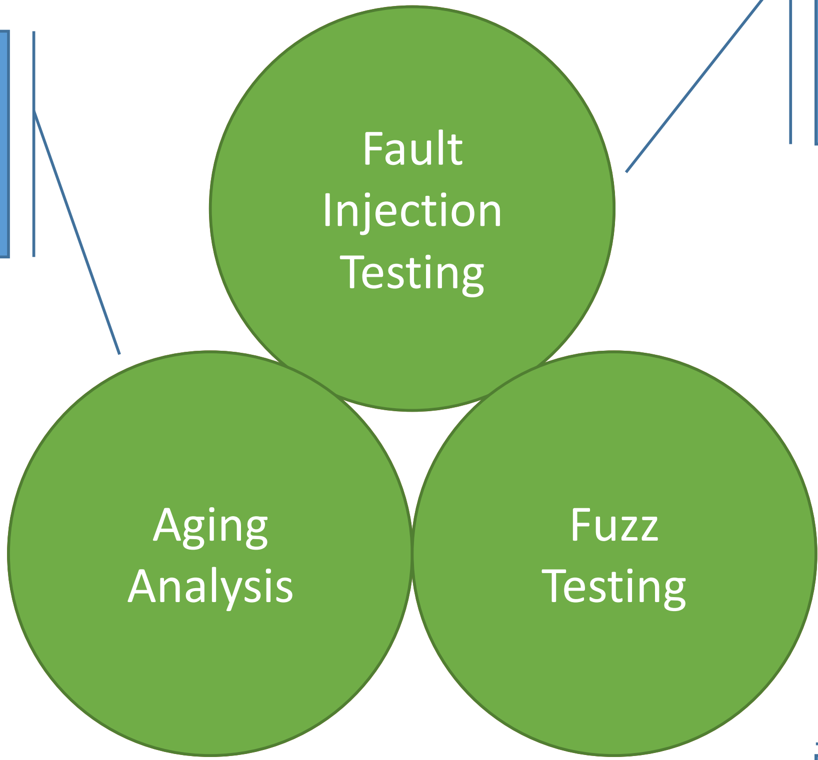
- an event that occurs when a system does not deliver the service as expected by its users
- e.g., the mobile OS crashes and the device can not be used

Dependability Threats of Mobile OS

- **Residual faults of the mobile OS**: they are hardware or software defects within the components of the mobile OS that, under special conditions (*i.e.*, triggers), leads to an internal error state. According to their propagation, they can be further divided in
  - **traditional faults**, when the errors, not correctly handled by FTMA, spread across other components in the mobile OS as component failures; or
  - **aging faults**, when the errors accumulate over time causing performance degradation and poor quality of service.
- **Misuses of the mobile OS**: they are the misuses of the mobile device system by users and applications. They are external faults that originate from the users of the system, including human users that interact with the device and applications that interact with the mobile OS interfaces or framework.

Dependability Assessment of Mobile OS

- Experimental methodology to analyze aging
- Aging analysis campaign on Android OS

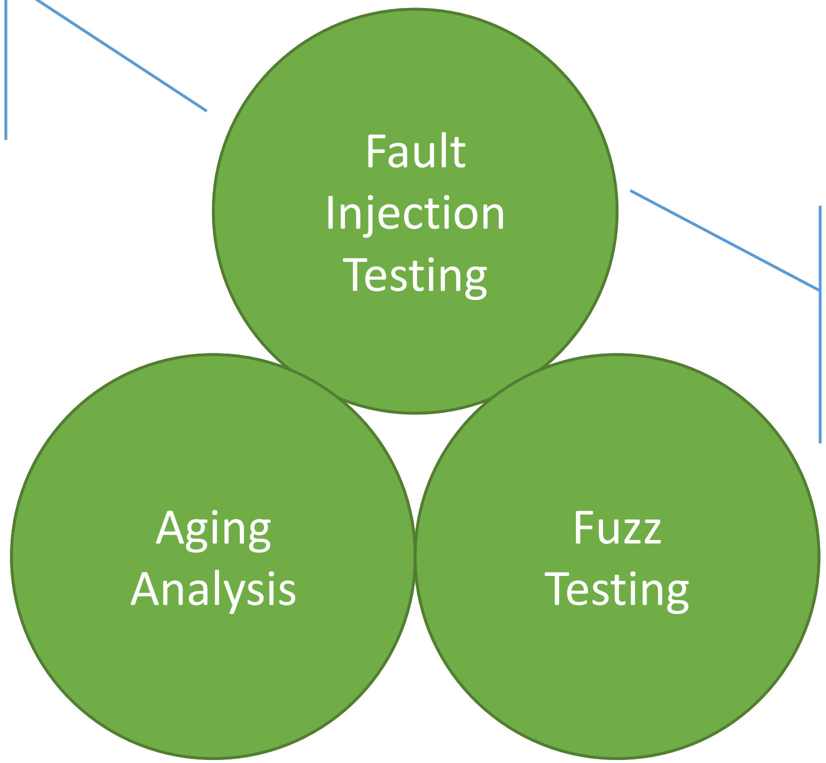


- Fault modeling methodology
- Fault Injection tool for Android OS
- Fault injection campaign

- Fuzzing tool for Android OS
- Fuzzing test campaign

Dependability Assessment of Mobile OS

"Improving Usability of Fault Injection" – Cotroneo, D.; De Simone, L. ; Iannillo, A.K. ; Lanzaro, A. ; Natella, R. Published in: Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on



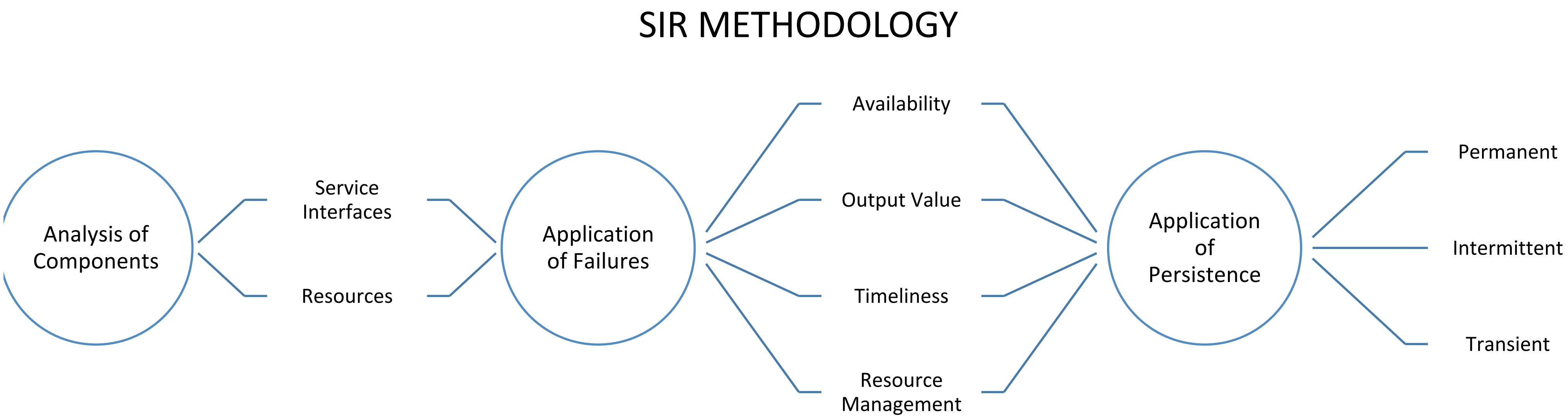
"AndroFIT: Software Fault Injection for the Android Mobile OS" – Cotroneo, D.; Iannillo, A.K.; Natella, R.; Rosiello S. under review for ICSE 2018

Fault Injection Testing

- Software comes with residual faults that need to be tolerated by the system
- Failure tolerance mechanisms and algorithms (**FTMA**) should satisfy the requirement to obtain a dependable system by avoiding service failure in presence of faults
- **Fault injection** is the process of introducing faults in a system, with the goal of assessing the impact of faults on performance and on availability, and the effectiveness of fault tolerance mechanisms

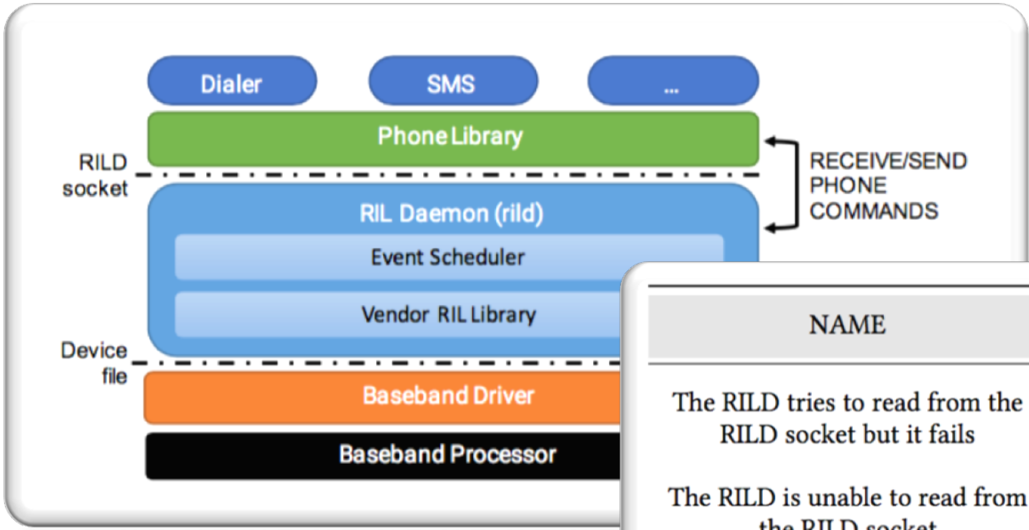
Fault Model

a set of realistic component failures that could be injected in the fault injection targets and act as a fault for the mobile OS



Android Fault Model

14 fault injection target components from 6 different subsystems (*i.e.*, phone, camera, sensors, activity, package, and storage subsystems) with their interfaces, and formalized more than 870 potential faults for the Android OS



NAME	FAILURE MODE	SERVICE/ RESOURCE	DESCRIPTION
The RILD tries to read from the RILD socket but it fails	availability	RECEIVE PHONE COMMAND ON RILD SOCKET	A read operation on the socket fails and returns an error code
The RILD is unable to read from the RILD socket	timeliness	RECEIVE PHONE COMMAND ON RILD SOCKET	A read operation on the socket receives no reply
The RILD reads a corrupted request from the RILD socket	output value	RECEIVE PHONE COMMAND ON RILD SOCKET	A read operation returns corrupted buffers
The RILD drops or cannot open the rild socket	resource corruption	SOCKET	The operation of the RILD on the socket fails

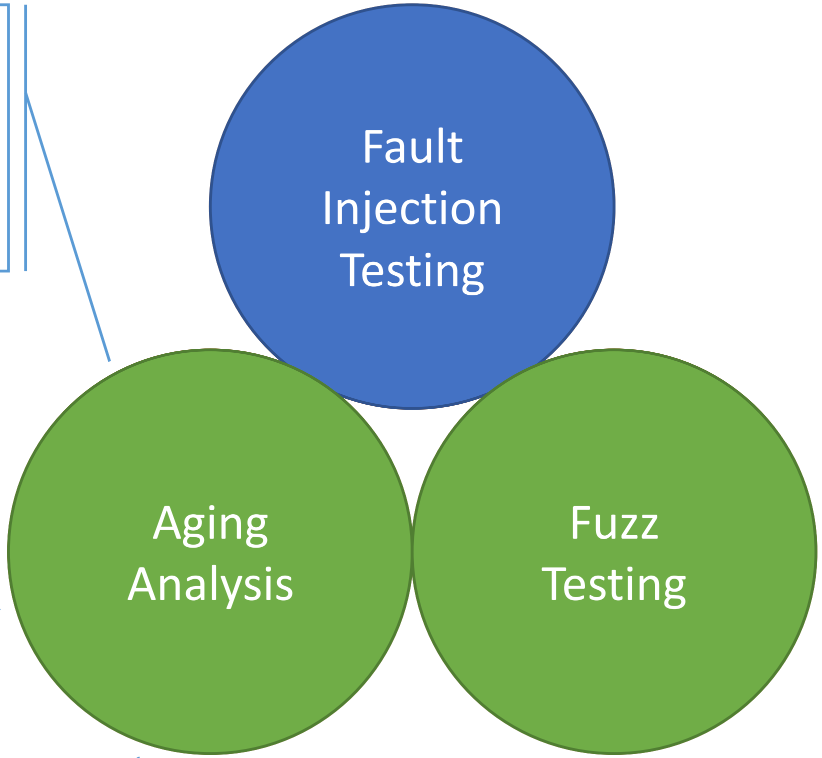
Fault Injection Campaign

- Android Fault Injection Tool (**AndroFIT**)
  - Design and implementation of all the necessary injection techniques
  - Design and implementation of an automatic experiment launcher
- 3 high-end smartphones
- Executing more than 2000 experiments, and each experiment lasts about 5 minutes
  - Testing time > 180hrs
- Outcomes from logs
- The close analysis of the experiments validates the accuracy of the AndroFIT suite

	subsystem	CRASH	ANR	FATAL	# of experiments
Samsung Galaxy S6 Edge	phone	0	0	22	309
	camera	31	5	3	111
	sensors	3	0	18	108
	activity	8	34	0	66
	package	3	27	0	63
	storage	33	3	0	75
		78	69	43	732
HTC One M9	phone	6	0	72	309
	camera	11	3	5	111
	sensors	7	0	9	108
	activity	32	18	1	66
	package	20	35	0	63
	storage	11	4	5	75
		87	60	92	732
Huawei P8	phone	6	0	108	309
	camera	56	0	4	111
	sensors	6	1	0	108
	activity	37	21	0	66
	package	55	5	0	63
	storage	8	1	3	75
		168	28	115	732

Dependability Assessment of Mobile OS

"**The Software Aging and Rejuvenation Repository**" – Cotroneo, D.; Iannillo, A.K.; Natella, R.; Pietrantuono, R.; Russo, S. Published in: *Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on*



"**Software Aging Analysis of the Android Mobile OS**" – Cotroneo, D.; Fucci, F.; Iannillo, A.K.; Natella, R.; Pietrantuono, R. To be published in: *Software Reliability Engineering (ISSRE), 2016 IEEE International Symposium on*

"**An Empirical Study of Software Aging in Android smartphones**" – Cotroneo, D.; Iannillo, A.K.; Natella, R.; Pietrantuono R. under review for IEEE Transactions on Reliability

Software Aging

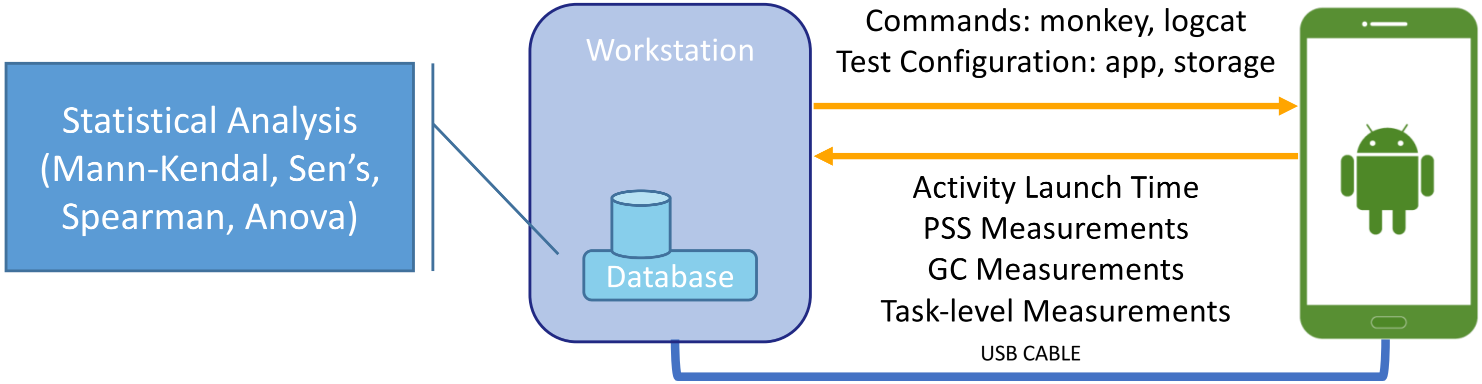
- Software aging can cause the device to **slowly degrade** its performance and to eventually fail, due to the **accumulation of errors** in the system state and to the incremental consumption of resources, such as physical memory.
- Software aging can be attributed to software faults that manifest themselves as memory leakage and fragmentation, unreleased locks, stale threads, data corruption, and numerical error accumulation.

Design of Experiments

- A first set of experiments covers all of the DEV levels, and keeps the *VER* factor to *ANDROID6*, since Android 6 Marshmallow is the only version that can be installed on all the devices, allowing us to study the impact of **software aging across devices from different vendors** (and all other factors with the same level).
- The second set of experiments fixes the *DEV* factor to *HUAWEIP8*, and varies the *VER* factor between *ANDROID5* and *ANDROID6*. The third set instead, fixes *DEV* to *SAMSUNGS6EDGE* and the *VER* to either *ANDROID6* or *ANDROID7*. These last two sets are used to study the impact of **software aging across different versions** of the Android OS

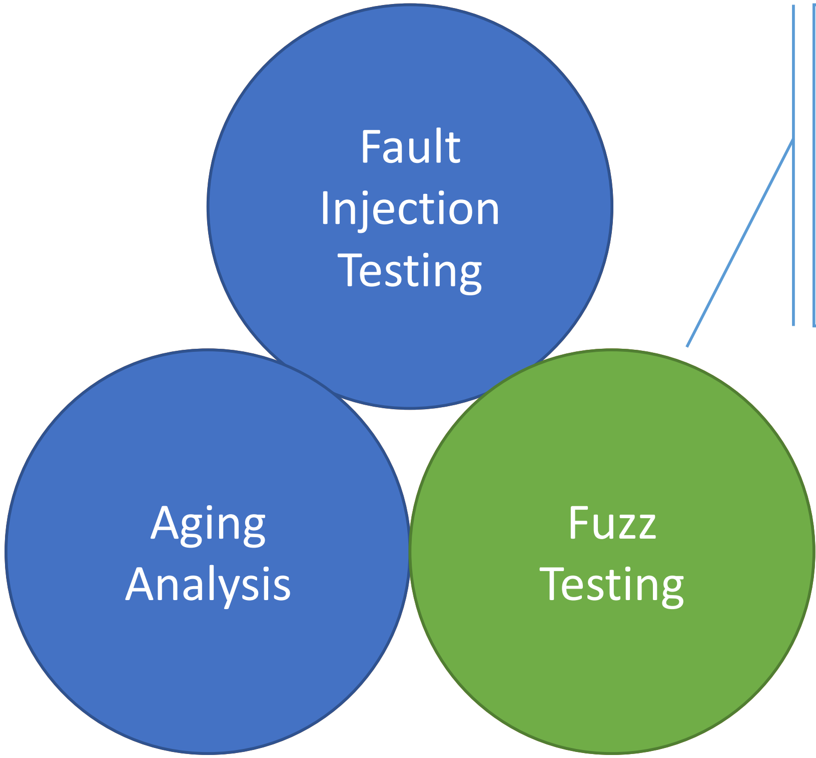
Factor	Level	Description
DEV	HTCONEM9 HUAWEIP8 LGNEXUS SAMSUNGS6EDGE	HTC One M9 device Huawei P8 device LG Nexus device Samsung S6 Edge device
VER	ANDROID5 ANDROID6 ANDROID7	Android 5 (Lollipop) Android 6 (Marshmallow) Android 7 (Nougat)
APP	EU	com.google.android.videos com.*.camera com.android.browser com.android.email com.android.contacts com.google.android.apps.maps com.android.chrome com.google.android.play.games com.android.calendar com.google.android.music com.google.android.youtube
	CHINA	com.tencent.mm com.sina.weibo com.qiyi.video com.youku.phone com.taobao.taobao com.tencent.mobileqqi com.baidu.searchbox com.baidu.BaiduMap com.UCMobile com.moji.mjweather
EVENTS	MIXED1 MIXED2 MIXED3	mostly switch events mostly touch events mostly navigation events
STO	FULL NORMAL	90% of storage space usage default storage space usage

Software Aging Analysis



- **not limited to specific Android devices**
  - the software aging effects are exacerbated by the specific Android vendors
- **not limited to specific Android versions**, but that the problem permeates the Android OS.
  - tests did not show an improvement of the Android OS over time
- accompanied by a statistically-significant increase of the **memory consumption of key Android processes**
  - e.g., system_server

Dependability Assessment of Mobile OS



"**Chizpurfle: A Gray-Box Android Fuzzer for Vendor Service Customizations** " – Cotroneo, D.; Iannillo, A.K.; Natella, R.; Pietrantuono, R.
To be published in: *Software Reliability Engineering (ISSRE), 2017 IEEE International Symposium on*
BEST RESEARCH PAPER AWARD

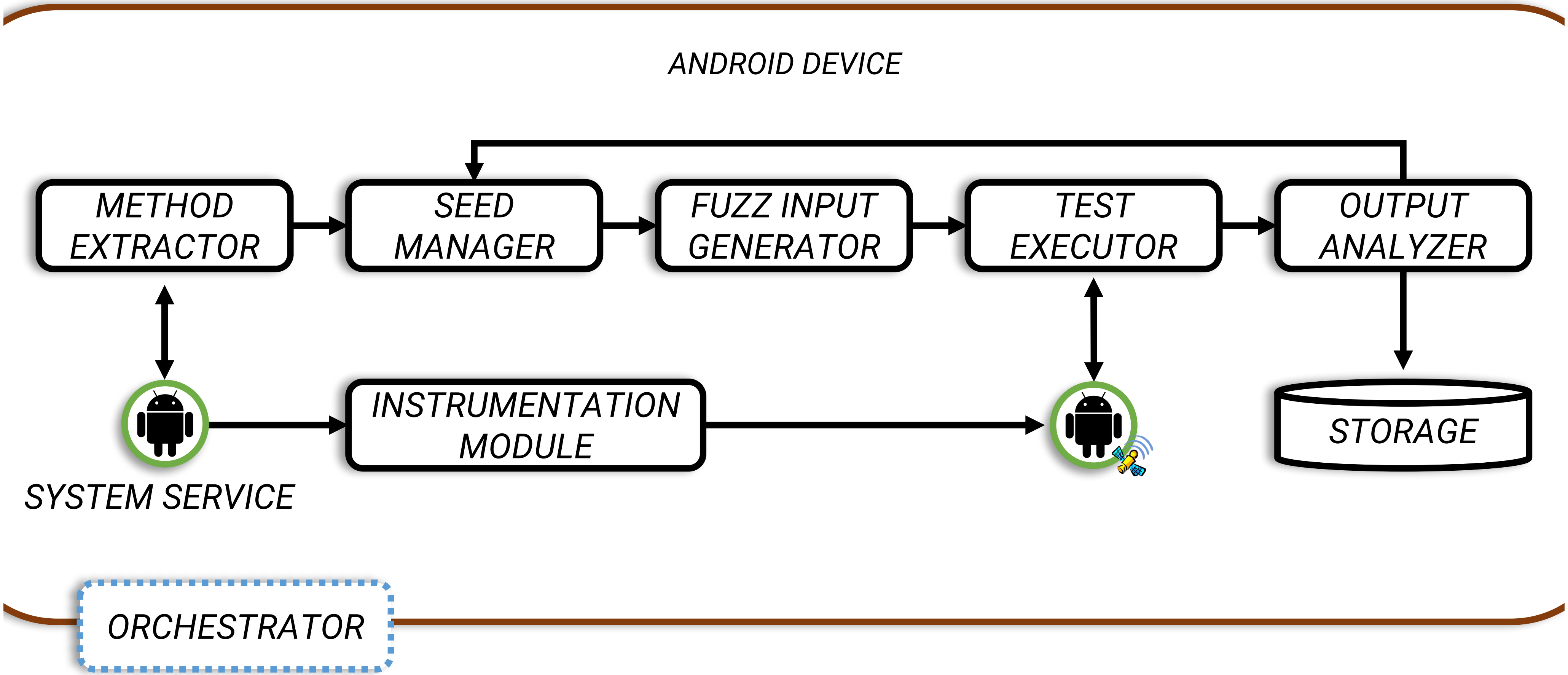
Fuzz Testing

- Vendor customizations often introduce new vendor-specific software defects
  - **do not benefit** from the feedback loop of the Android ecosystem
- **Fuzzing** is a well-established and effective software testing technique to identify weaknesses in fragile software interfaces by injecting invalid and unexpected inputs
- Black-box vs White-box

Chizpurfle: a gray-box approach

- It leverages test coverage information, while avoiding the need for recompiling the target code, or executing it in a special environment
- Use of dynamic binary instrumentation techniques (such as software breakpoints and just-in-time code rewriting) to obtain information about the block coverage
- It guides fuzz testing only on the vendor customizations, by automatically extracting the list of vendor service interfaces from an Android device

Chizpurfle: a gray-box approach



Fuzz Testing Campaign

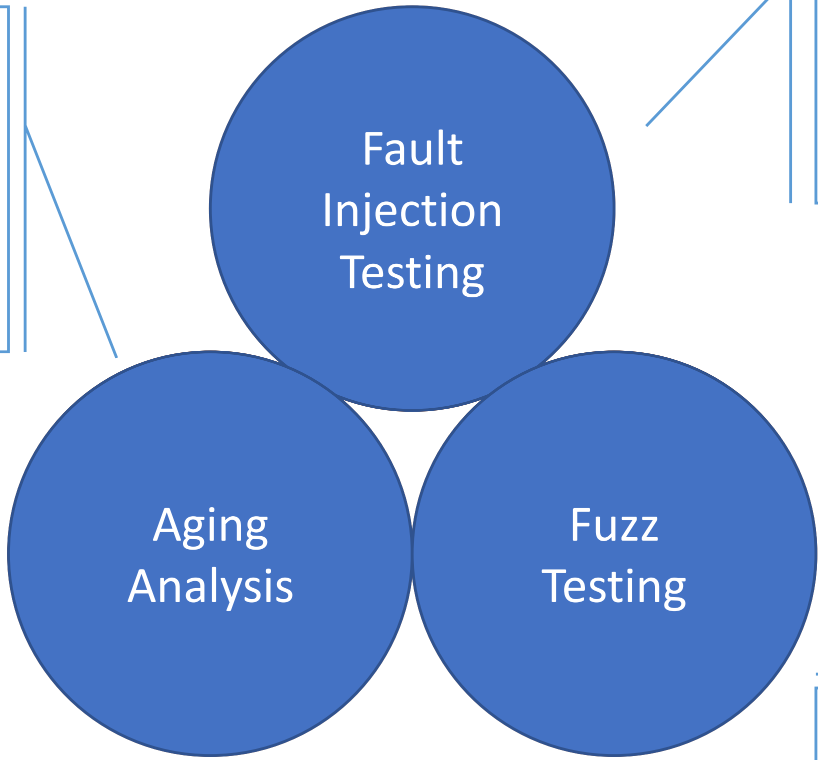
- *Chizpurfle* detected 2,272 service methods from Samsung customizations
- *Chizpurfle* performed 34,645 tests on these methods
- Found failures were **critical**
- 9 tests failed, due to 2 distinct bugs

Comparison with Black-box approach

- Compare throughput and code coverage against Chizpurfle black-box configurations
- On average, Chizpurfle covers **2.3x more code**
  - more effective on methods with complex inputs
- On average, Chizpurfle has a performance **slow-down of 11.97x**
  - Valgrind slowdows is 4.3x, and 22.1x when performing memory leak analysis

Dependability Assessment of Mobile OS

- Experimental methodology to analyze aging
- Aging analysis campaign on Android OS
- The components identified in our analysis are potential candidates for applying software rejuvenation



- SIR methodology
- AndroFIT
- Fault injection campaign with reliability improvements and lesson learned

- CHIZPURFLE
- Fuzzing test campaign
- Valuable opportunity for research on fuzzing

Products – Conference Papers

1. "**Improving Usability of Fault Injection**" – Cotroneo, D.; De Simone, L. ; Iannillo, A.K. ; Lanzaro, A. ; Natella, R.
   Published in: *Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on*
   Date of Conference: 3-6 November 2014

2. "**Network Function Virtualization: Challenges and Directions for Reliability Assurance**" – Cotroneo, D.; De Simone, L.; Iannillo, A.K.; Lanzaro, A.; Natella, R.
   Published in: *Software Reliability Engineering (ISSRE), 2014 IEEE International Symposium on*
   Date of Conference: 3-6 November 2014

3. "**Dependability Evaluation and Benchmarking of Network Function Virtualization Infrastructures**" – Cotroneo, D.; De Simone, L.; Iannillo, A.K.; Lanzaro, A.; Natella, R.
   Published in: *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*
   Date of Conference: 13-17 April 2015
   BEST CONFERENCE PAPER AWARD

4. "**The Software Aging and Rejuvenation Repository**" – Cotroneo, D.; Iannillo, A.K.; Natella, R.; Pietrantuono, R.; Russo, S.
   Published in: *Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on*
   Date of Conference: 2-5 November 2015

5. "**Software Aging Analysis of the Android Mobile OS**" – Cotroneo, D.; Fucci, F.; Iannillo, A.K.; Natella, R.; Pietrantuono, R.
   To be published in: *Software Reliability Engineering (ISSRE), 2016 IEEE International Symposium on*
   Date of Conference: 23-27 October 2016

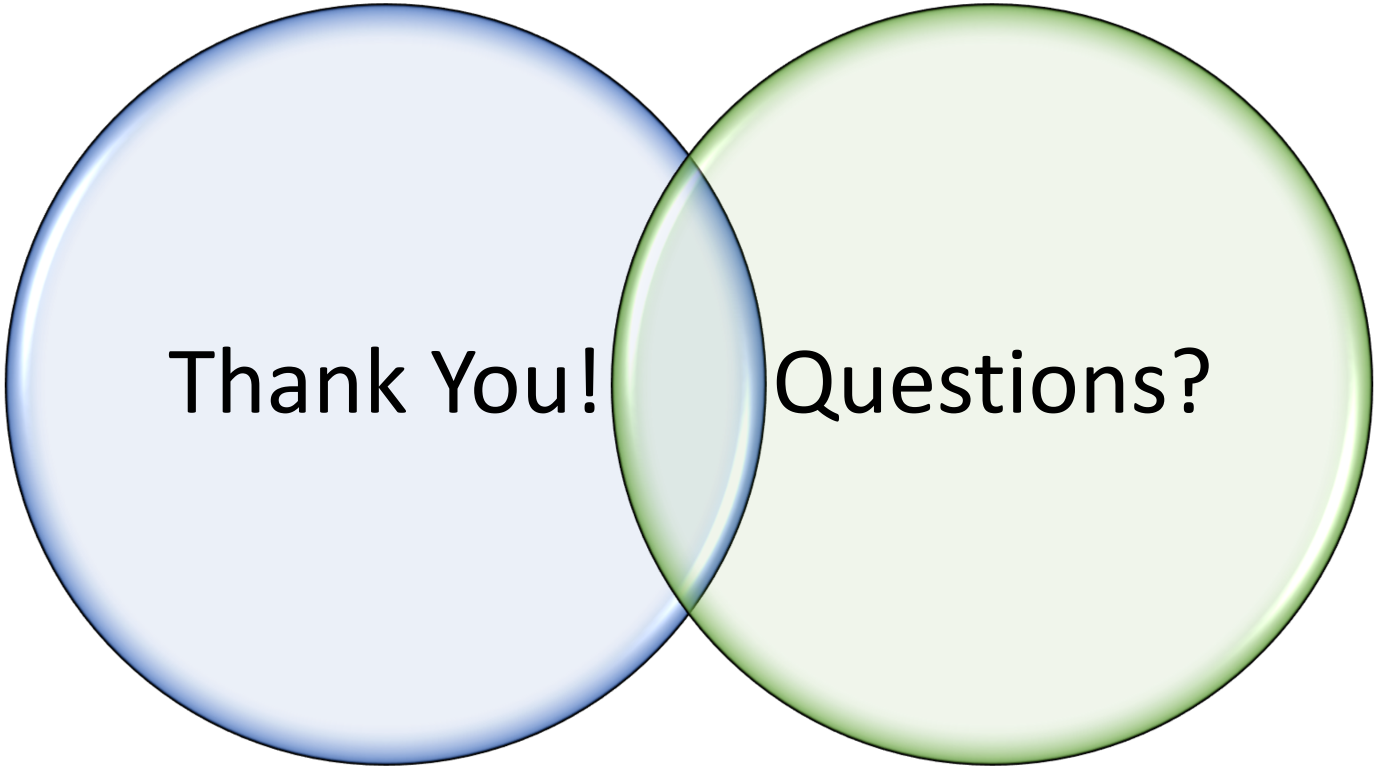
6. "**Chizpurfle: A Gray-Box Android Fuzzer for Vendor Service Customizations** " – Cotroneo, D.; Iannillo, A.K.; Natella, R.; Pietrantuono, R.
   To be published in: *Software Reliability Engineering (ISSRE), 2017 IEEE International Symposium on*
   Date of Conference: 23-26 October 2017
   BEST RESEARCH PAPER AWARD

7. "**AndroFIT: Software Fault Injection for the Android Mobile OS**" – Cotroneo, D.; Iannillo, A.K.; Natella, R.; Rosiello S.
   under review for ICSE 2018

Products – Journal

8. **“An Empirical Study of Software Aging in Android smartphones”** – Cotroneo, D.; Iannillo, A.K.; Natella, R.; Pietrantuono R. under review for IEEE Transactions on Reliability

	Credits year 1								Credits year 2								Credits year 3								
Bimonthly Period		1	2	3	4	5	6			1	2	3	4	5	6			1	2	3	4	5	6		
	Estimated	nov-dec 2014	jan-feb 2015	mar-apr 2015	may-jun 2015	jul-aug 2015	sep-oct 2015	**Summary**	**Estimated**	nov-dec 2015	jan-feb 2016	mar-apr 2016	may-jun 2016	jul-aug 2016	sep-oct 2016	**Summary**	**Estimated**	nov-dec 2016	jan-feb 2017	mar-apr 2017	may-jun 2017	jul-aug 2017	sep-oct 2017	**Summary**	**Total**
Modules	**20**	0	3	3	15	0	0	**21**	**10**	0	12	0	3	0	0	**15**	**0**	0	0	0	0	0	0	**0**	**36**
Seminars	**5**	0.5	0.7	1	3	0	0	**5.2**	**5**	1	8	0	7	0	0	**16.1**	**0**	0	0	0	0	0	0	**0**	**21.3**
Research	**35**	10	8	6	6	10	8	**48**	**45**	7	7	7	7	7	9	**44**	**60**	10	10	10	10	10	10	**60**	**152**
	60	**10.5**	**11.7**	**10**	**24**	**10**	**8**	**74.2**	**60**	**8**	**27**	**7**	**17**	**7**	**9**	**75.1**	**60**	**10**	**10**	**10**	**10**	**10**	**10**	**60**	**209**


Thank You!
Questions?