Is This the Lifecycle We Really Want?

Vincenzo Riccio

Domenico Amalfitano

Anna Rita Fasolino

Context

Android apps testing automation

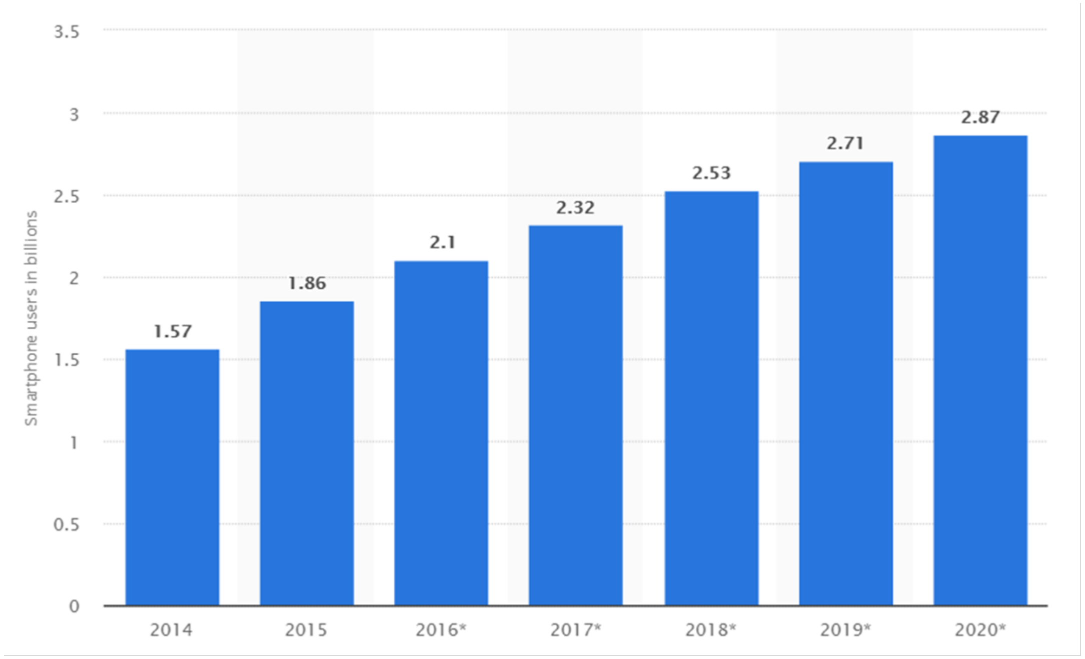
Goal

Valuable and effective solutions to support the Verification and Validation (V&V) activities for Android apps

Proposed Solution

Fully automated testing technique that explores an app for detecting issues tied to the Android Activity lifecycle

Number of smartphone users worldwide



https://www.statista.com/statistics/330695/

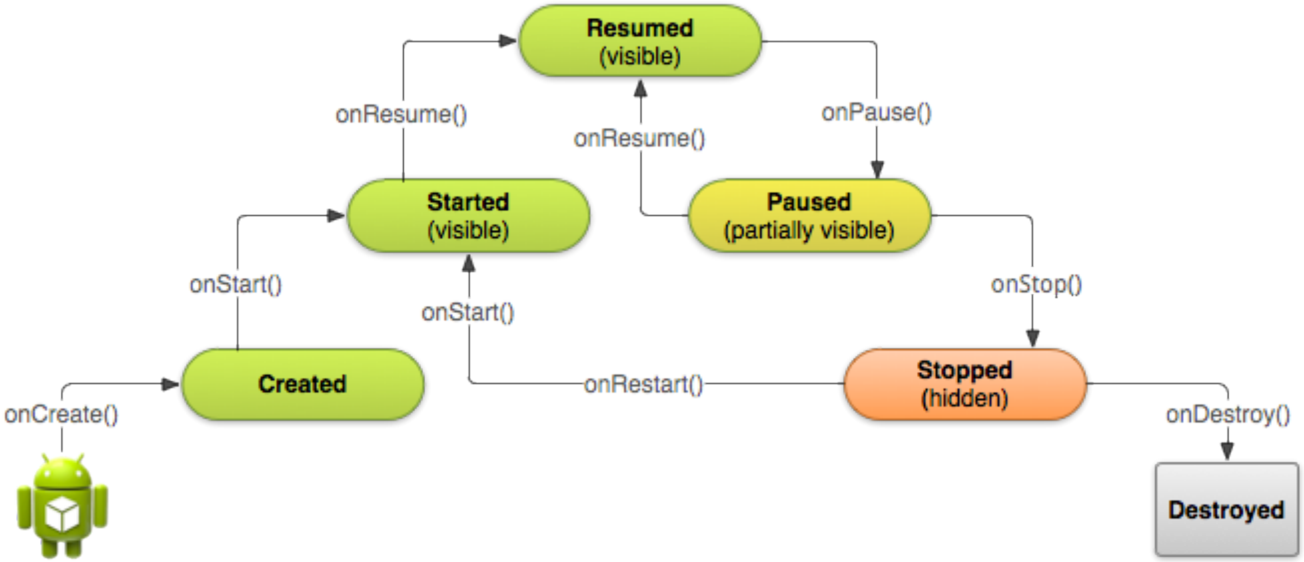
- There is a constant demand for new mobile apps
- The demand for app quality has grown together with their spread
- Android is today the world's most popular mobile operating system

Android Apps Testing

- Testing is a well-known approach for assuring the quality of software applications
- Test automation tools can facilitate software testing since they save humans from routine, time-consuming and error-prone manual tasks
- Mobile apps have several peculiarities compared to traditional software applications that have to be taken into account by testing techniques and tools

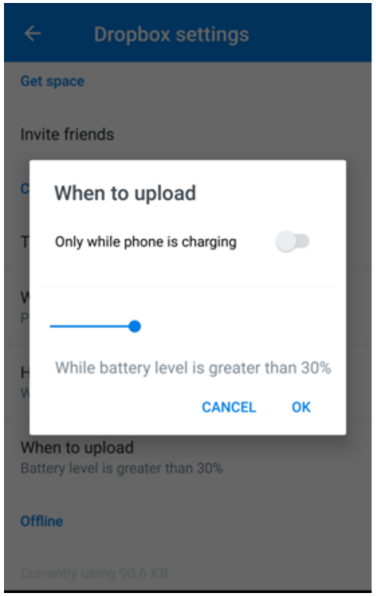
Android Activity Lifecycle

- An Android app is composed by one or more Activities
- Each Activity represents a single screen
- The Android Framework defines a peculiar lifecycle for Activity instances



Motivating Example: Crash

- **A crash occurs when an app stops functioning properly and exits unexpectedly**



Dropbox settings

Get space

Invite friends

When to upload

Only while phone is charging

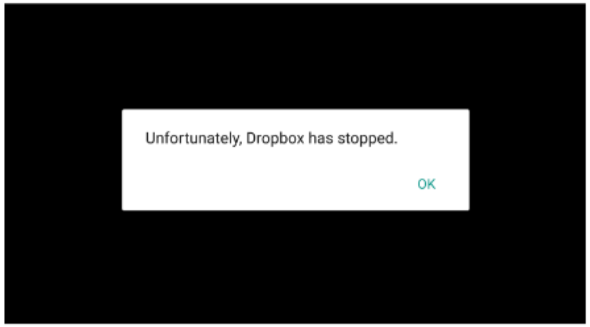
While battery level is greater than 30%

When to upload
Battery level is greater than 30%

Offline

Currently using 90.6 KB

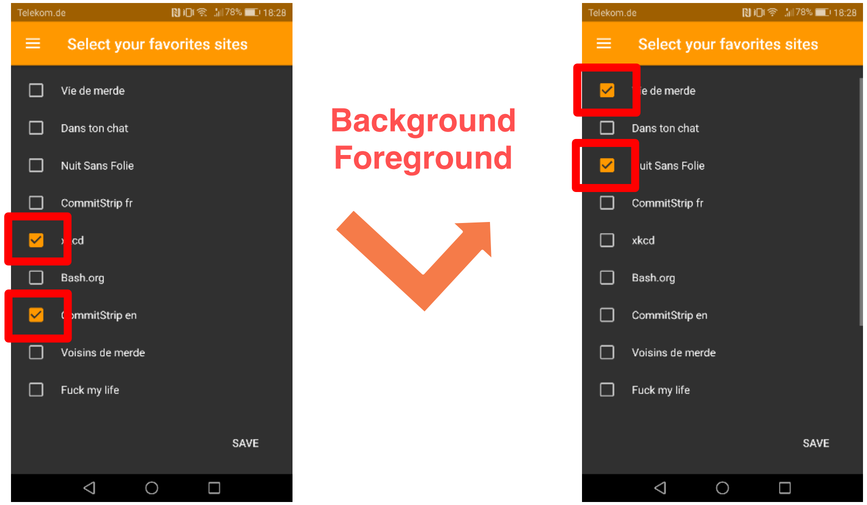
Orientation Change



Unfortunately, Dropbox has stopped.

Motivating Example: GUI Failure

- GUI failures consist in the manifestation of an unexpected GUI state
- The GUI state before the Activity is stopped, paused or destroyed is different from the GUI state displayed after the user returns to the Activity, whereas it is expected to be the same*



Telekom.de 78% 18:28

Select your favorites sites

□ Vie de merde
□ Dans ton chat
□ Nuit Sans Folie
□ CommitStrip fr
☑ xkcd
□ Bash.org
☑ CommitStrip en
□ Voisins de merde
□ Fuck my life

SAVE

Background
Foreground

Telekom.de 78% 18:28

Select your favorites sites

☑ Vie de merde
□ Dans ton chat
☑ Nuit Sans Folie
□ CommitStrip fr
□ xkcd
□ Bash.org
□ CommitStrip en
□ Voisins de merde
□ Fuck my life

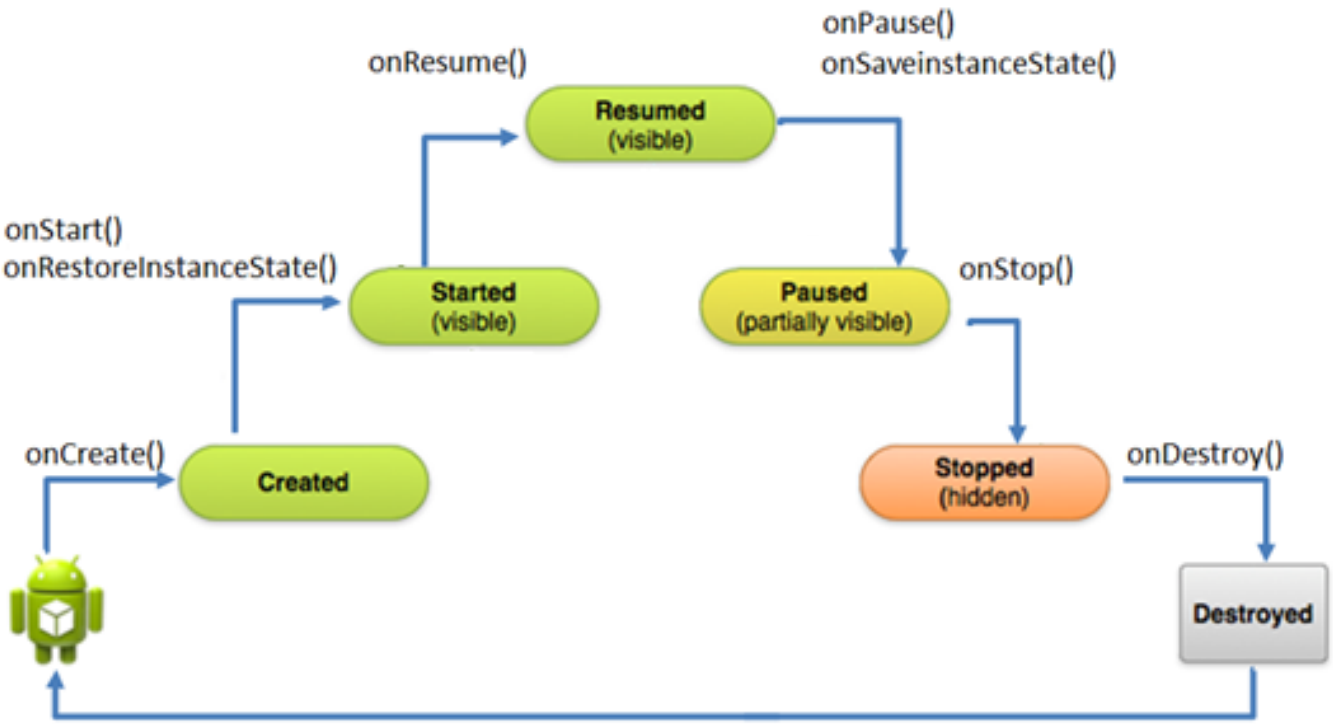
SAVE

*Amalfitano D, Riccio V, Paiva ACR, Fasolino AR. Why does the orientation change mess up my Android application? From GUI failures to code faults. *Softw Test Verif Reliab*. 2018;28:e1654.
https://doi.org/10.1002/stvr.1654

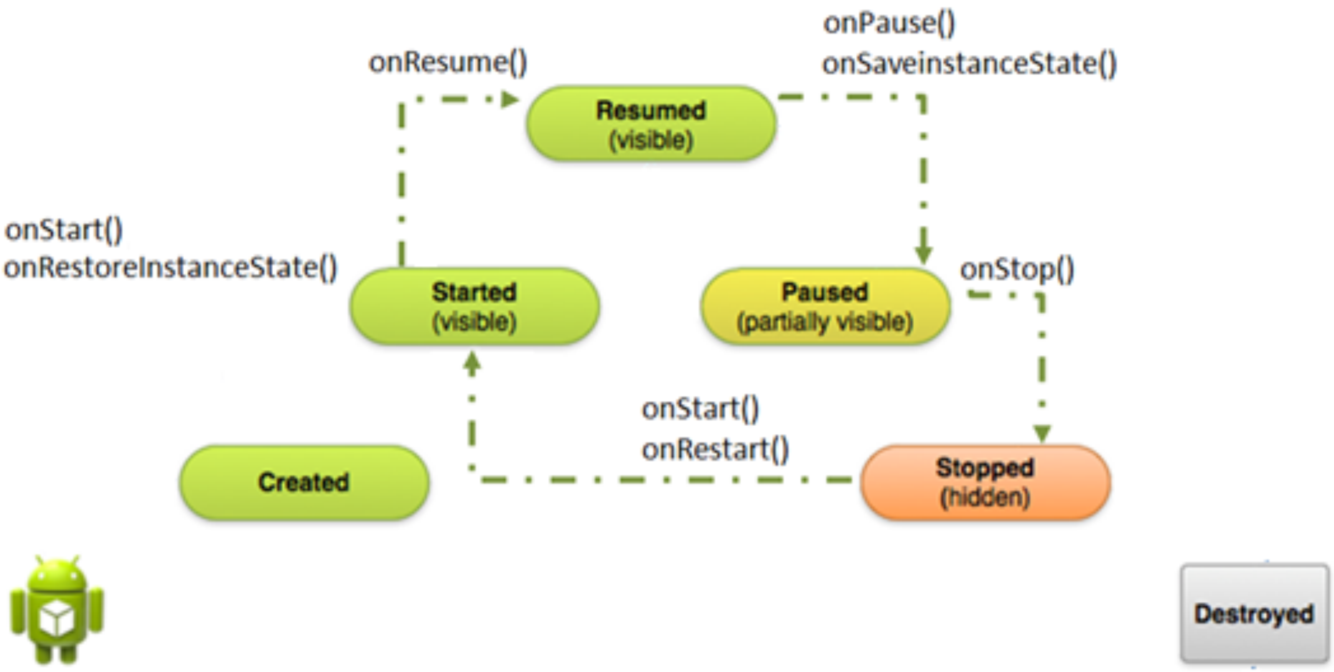
Proposed Solution

- ALARic (Activity Lifecycle Android Ripper), a novel fully automated Black-Box Event-based testing technique to detect issues tied to the Activity lifecycle
- It combines:
  - The traditional testing approaches based on dynamic app exploration
  - A strategy that systematically exercises the Activity lifecycle on each GUI state encountered during the exploration
- It relies on:
  - Lifecycle Event Sequences, mobile-specific events able to exercise the Activity lifecycle
  - Testing oracles to detect crashes and GUI failures tied to the Activity lifecycle

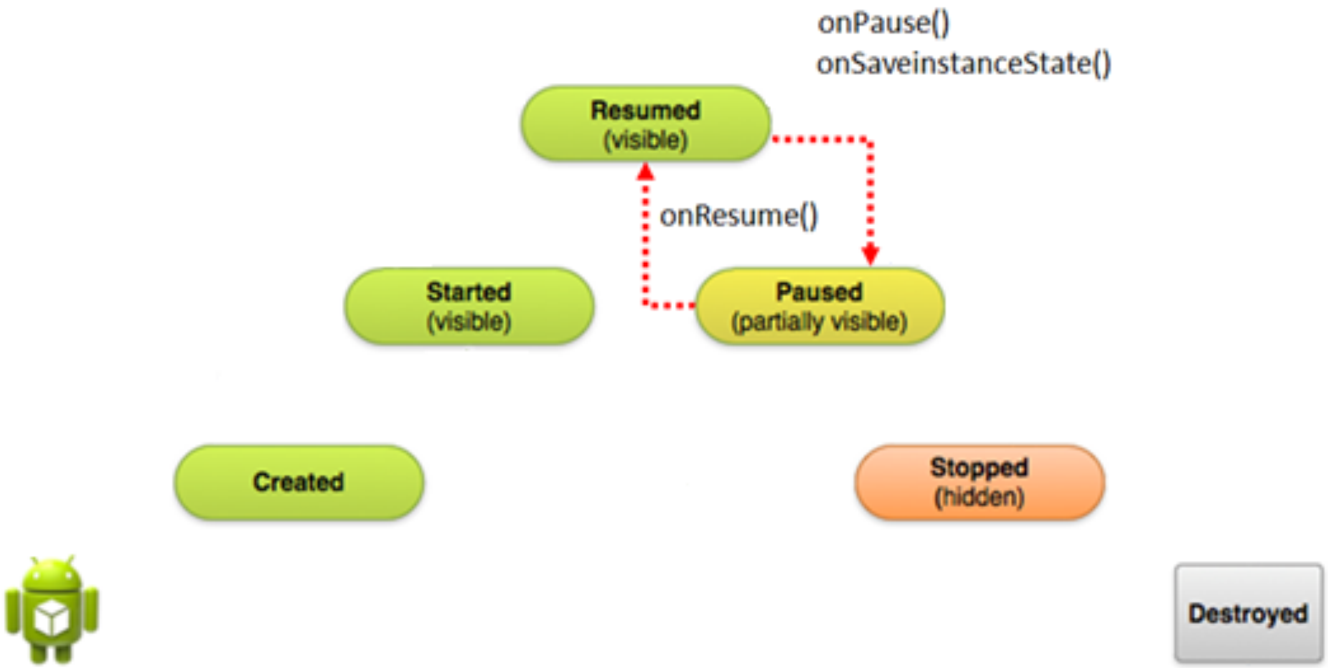
Lifecycle Event Sequences - DOC



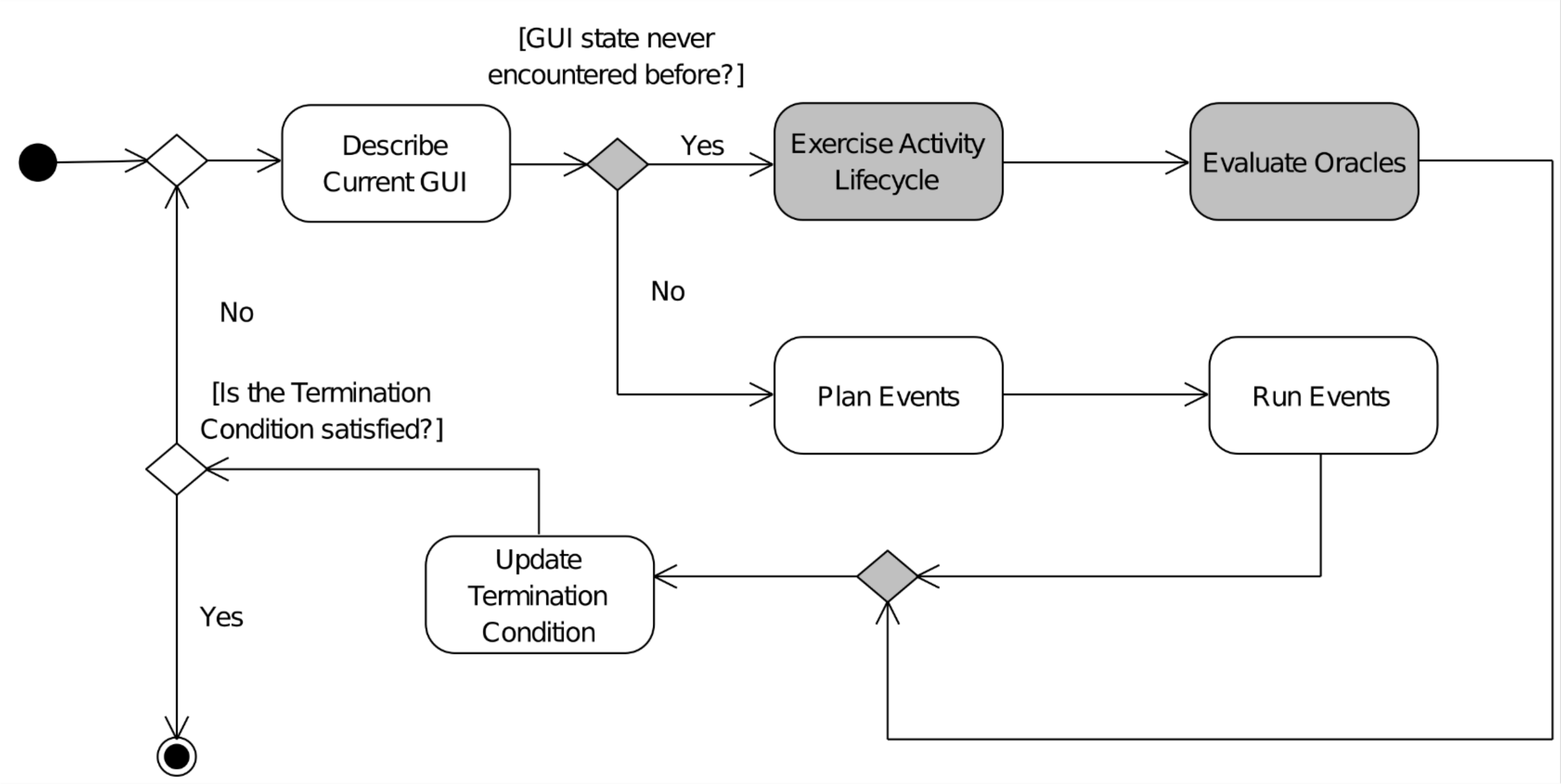
Lifecycle Event Sequences - BF



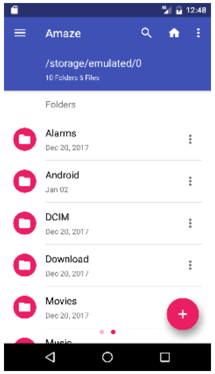
Lifecycle Event Sequences - STAI



The ALARic approach


[GUI state never encountered before?]
Describe Current GUI
Yes
Exercise Activity Lifecycle
Evaluate Oracles
No
No
Plan Events
Run Events
[Is the Termination Condition satisfied?]
Update Termination Condition
Yes

ALARic Workflow Example



12:48

Amaze

/storage/emulated/0

10 Folders 5 Files

Folders

Alarms
Dec 20, 2017

Android
Jan 02

DCIM
Dec 20, 2017

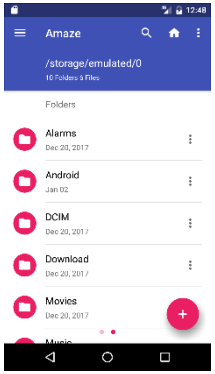
Download
Dec 20, 2017

Movies
Dec 20, 2017

Music

A

ALARic Workflow Example



12:48

Amaze

/storage/emulated/0

10 Folders 5 Files

Folders

Alarms
Dec 20, 2017

Android
Jan 02

DCIM
Dec 20, 2017

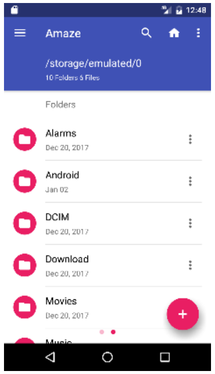
Download
Dec 20, 2017

Movies
Dec 20, 2017

Music

A

DOC



12:48

Amaze

/storage/emulated/0

10 Folders 5 Files

Folders

Alarms
Dec 20, 2017

Android
Jan 02

DCIM
Dec 20, 2017

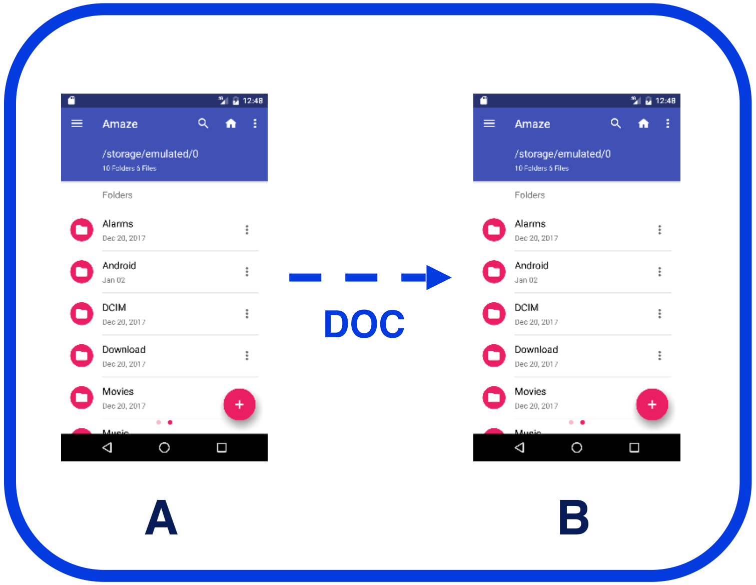
Download
Dec 20, 2017

Movies
Dec 20, 2017

Music

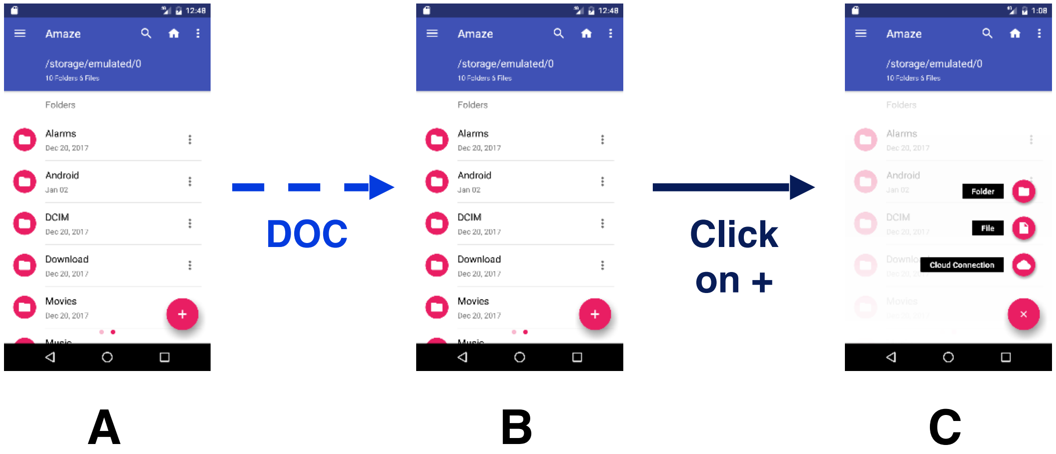
B

ALARic Workflow Example



B = A

ALARic Workflow Example



DOC

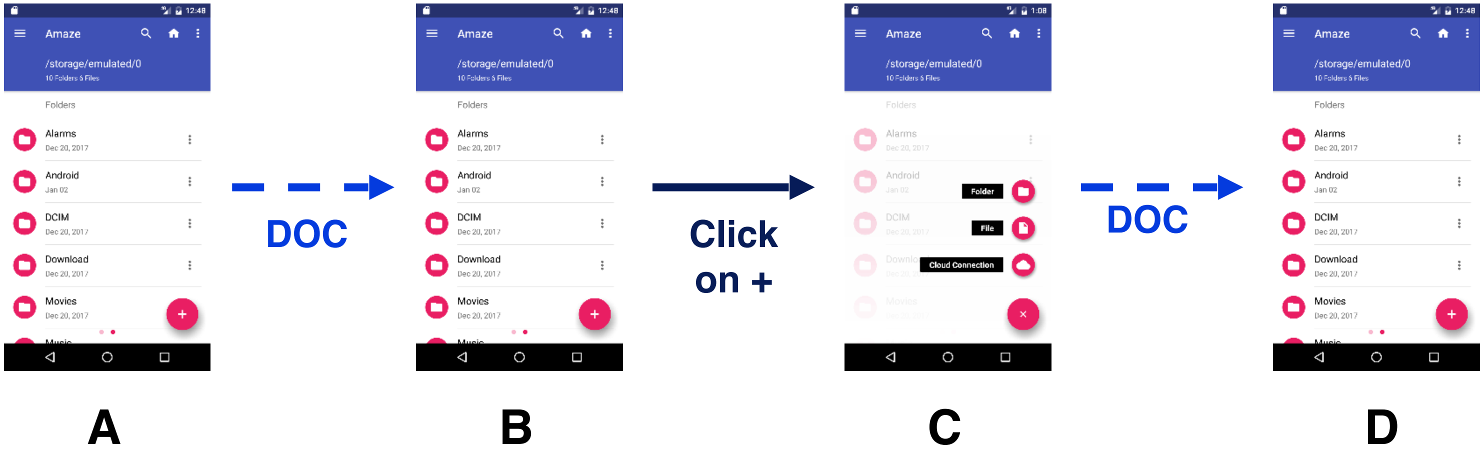
Click on +

A

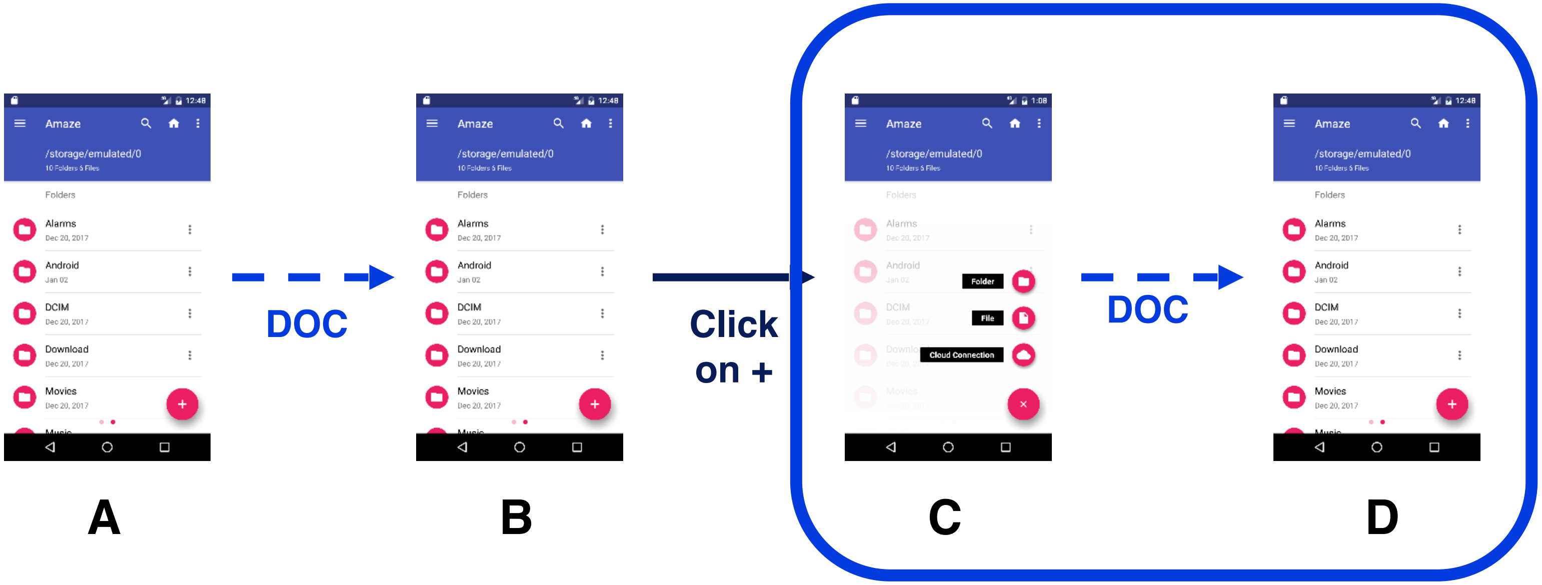
B

C

ALARic Workflow Example



ALARic Workflow Example



DOC

Click on +

DOC

A

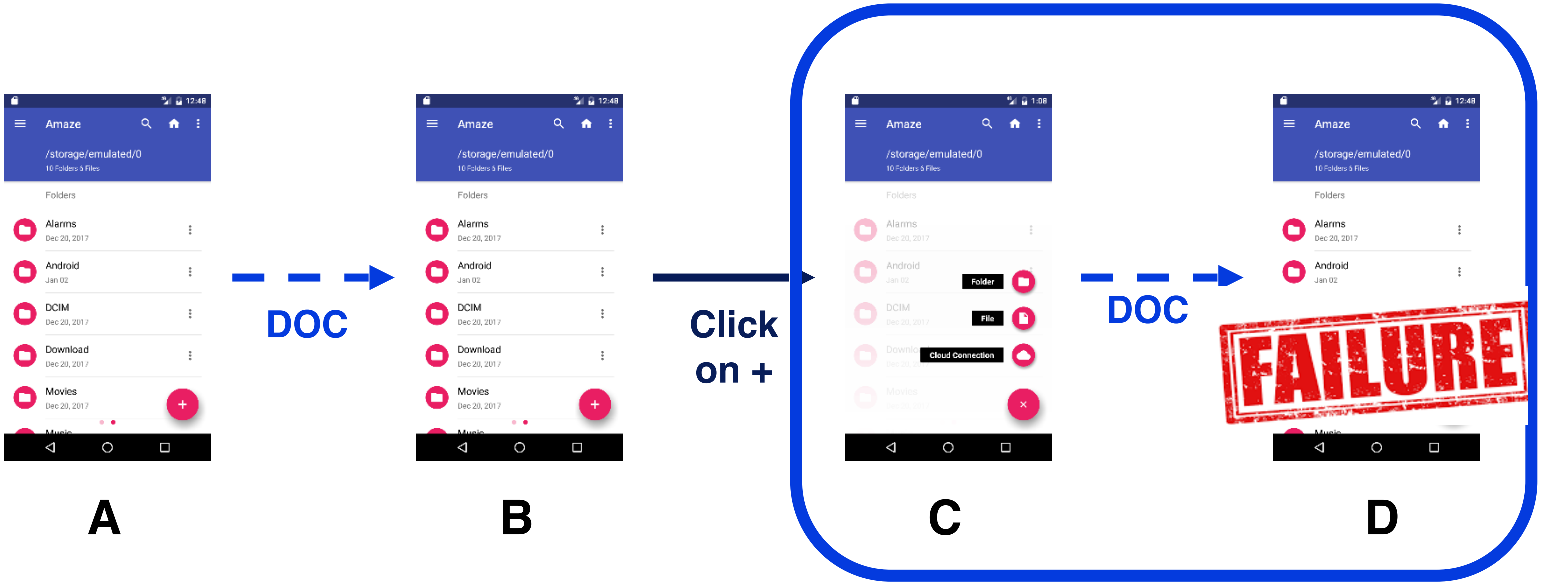
B

C

D

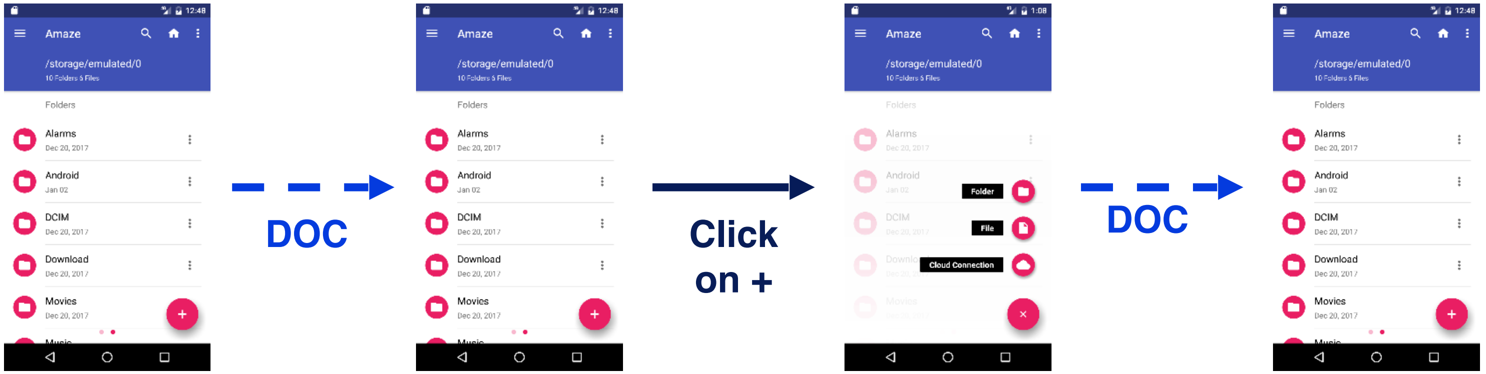
D ≠ C

ALARic Workflow Example



D ≠ C

ALARic Workflow Example



DOC

Click on +

DOC

A

B

C

D

D = A

Experimental Evaluation

- GOAL: Evaluate the ability of ALARic to automatically detect crashes and GUI failures tied to the Activity lifecycle
  - RQ1: How effective is the ALARic tool in detecting issues tied to the Activity lifecycle in real Android apps?
  - RQ2: How does the effectiveness of the ALARic tool in detecting crashes tied to the Activity lifecycle in real Android apps compare to the state-of-the-practice tool, Monkey?

Objects

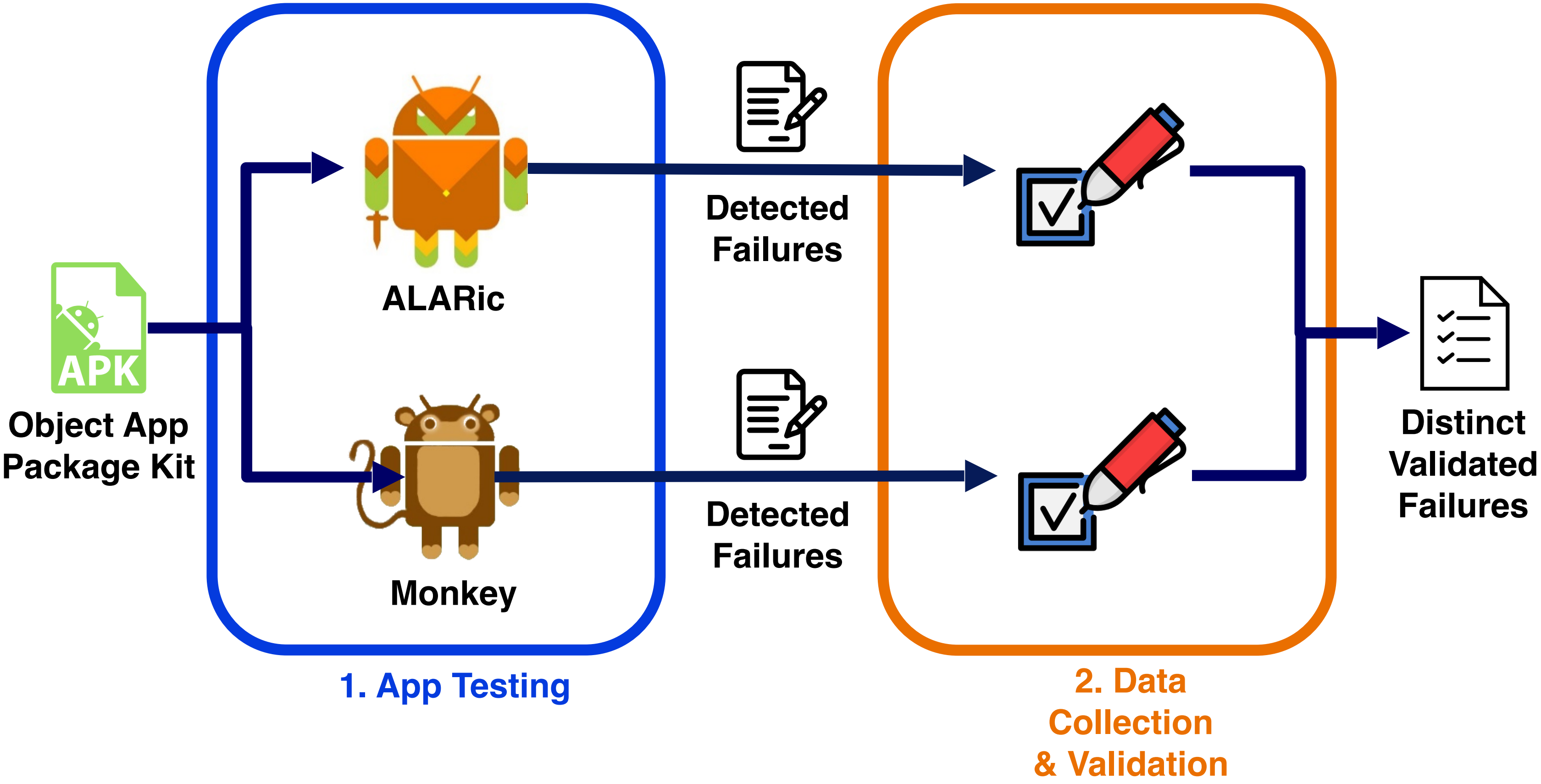
- **15 apps that are distributed by Google Play Store whose source code is available in the F-Droid repository**

ID	App	Version	Activities
A1	A Time Tracker	0.21	5
A2	Port Knocker	1.0.9	6
A3	Who Has My Stuff?	1.0.27	4
A4	Agram	1.4.1	5
A5	Alarm Klock	1.9	5
A6	Padland	1.3	10
A7	Syncthing	0.9.1	12
A8	Anecdote	1.1.2	3
A9	Amaze File Manager	3.1.2 RC4	5
A10	Google Authenticator	2.21	5
A11	BeeCount	2.3.9	8
A12	FOSDEM companion	1.4.6	8
A13	Periodical	0.30	6
A14	Taskbar	3.0.2	23
A15	SpaRSS	1.11.8	8

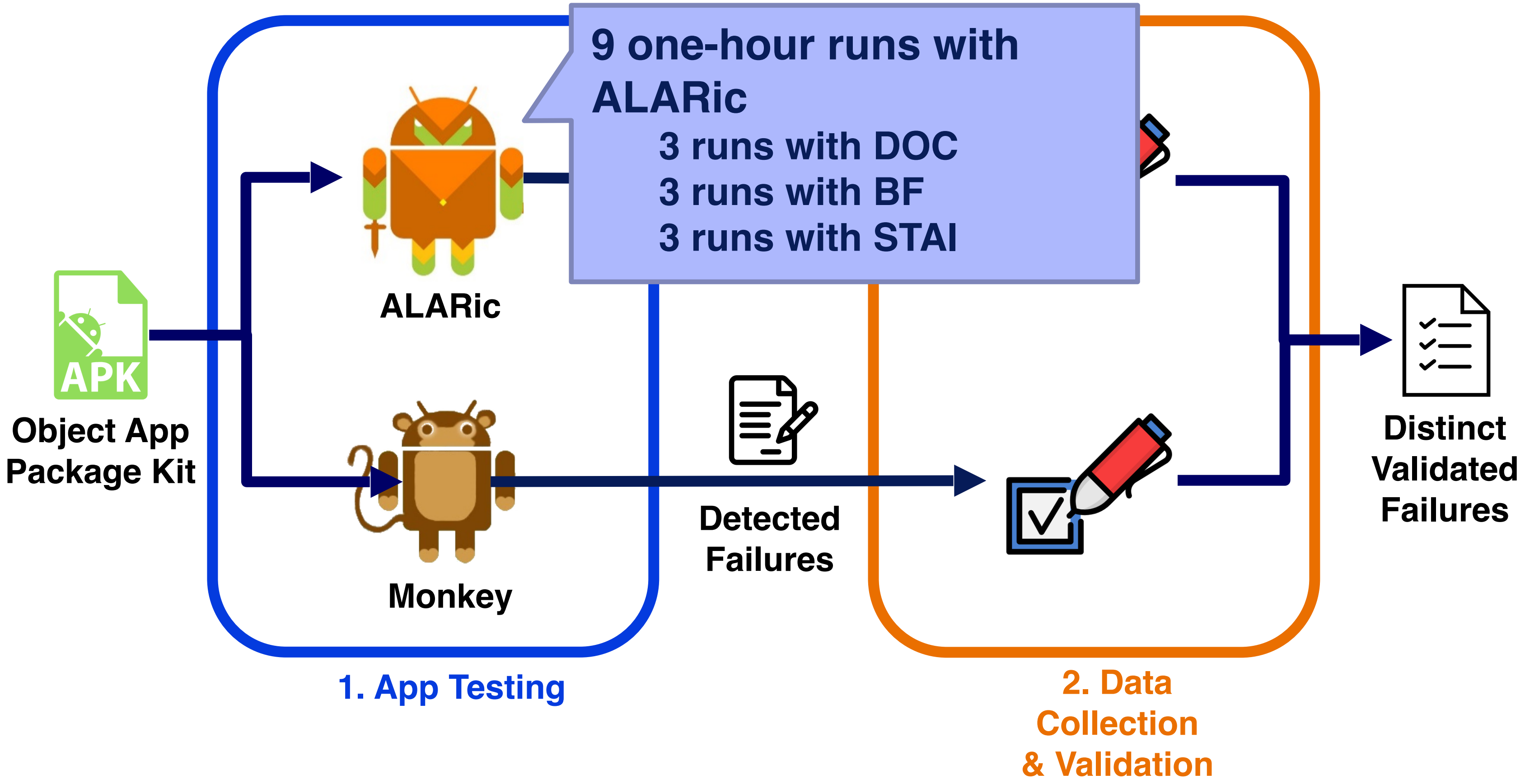
Metrics

- To evaluate the effectiveness of ALARic in detecting GUI failures:
  - $\#DGF_{DOC}$ number of distinct GUI Failures triggered by DOC
  - $\#DGF_{BF}$ number of distinct GUI Failures triggered by BF
  - $\#DGF_{STAI}$ number of distinct GUI Failures triggered by STAI
  - $\#DGF_{TOTAL}$ number of distinct GUI Failures triggered by the DOC, BF, STAI
- To evaluate the effectiveness of both the tools in finding Crashes:
  - $\#DC_{DOC}$ number of distinct crashes triggered by DOC
  - $\#DC_{BF}$ number of distinct crashes triggered by BF
  - $\#DC_{STAI}$ number of distinct crashes triggered by STAI
  - $\#DC_{TOTAL}$ number of distinct crashes triggered by the DOC, BF, STAI

Experimental Procedure



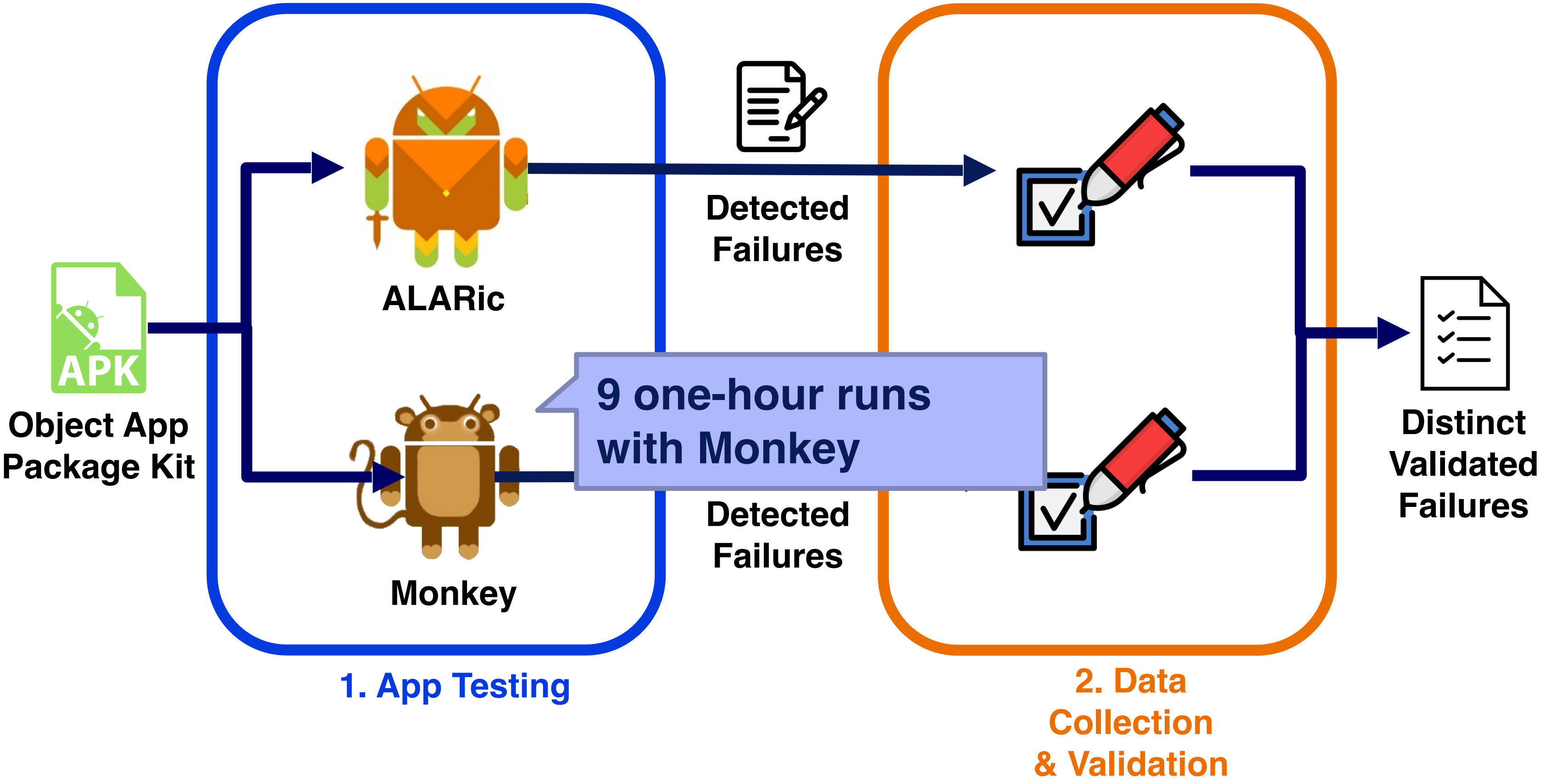
Experimental Procedure



9 one-hour runs with ALARic

- 3 runs with DOC
- 3 runs with BF
- 3 runs with STAI

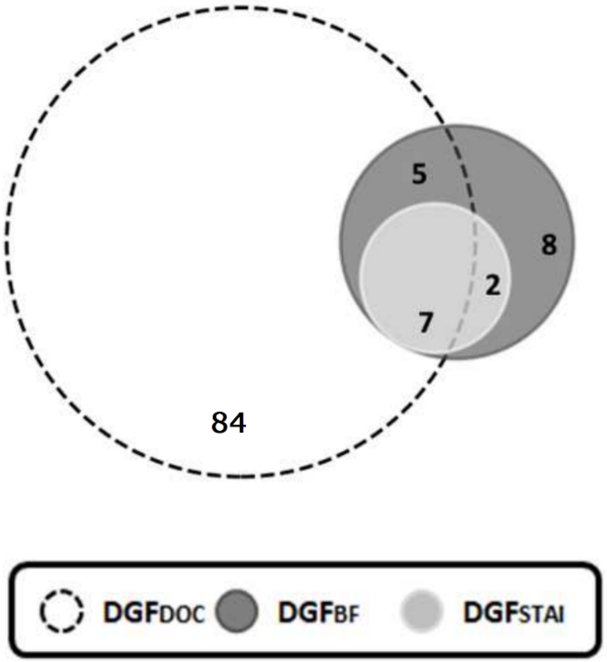
Experimental Procedure

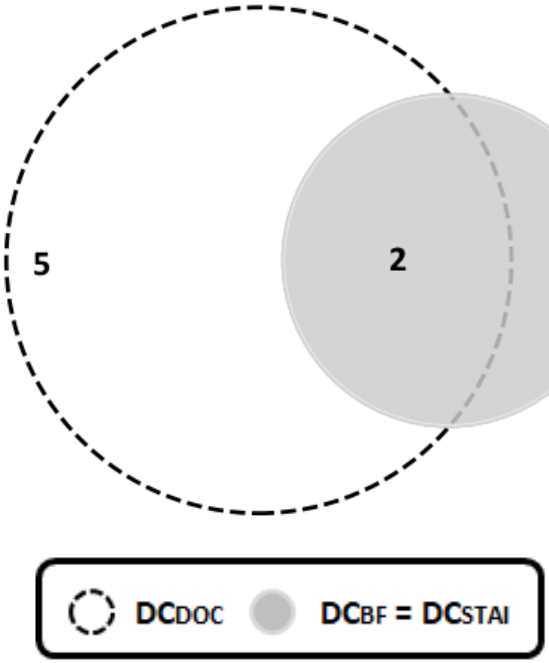


9 one-hour runs with Monkey

Experimental Results: RQ1

- ALARic detected 106 distinct GUI failures and 8 crashes tied to the Activity lifecycle in all the analyzed apps





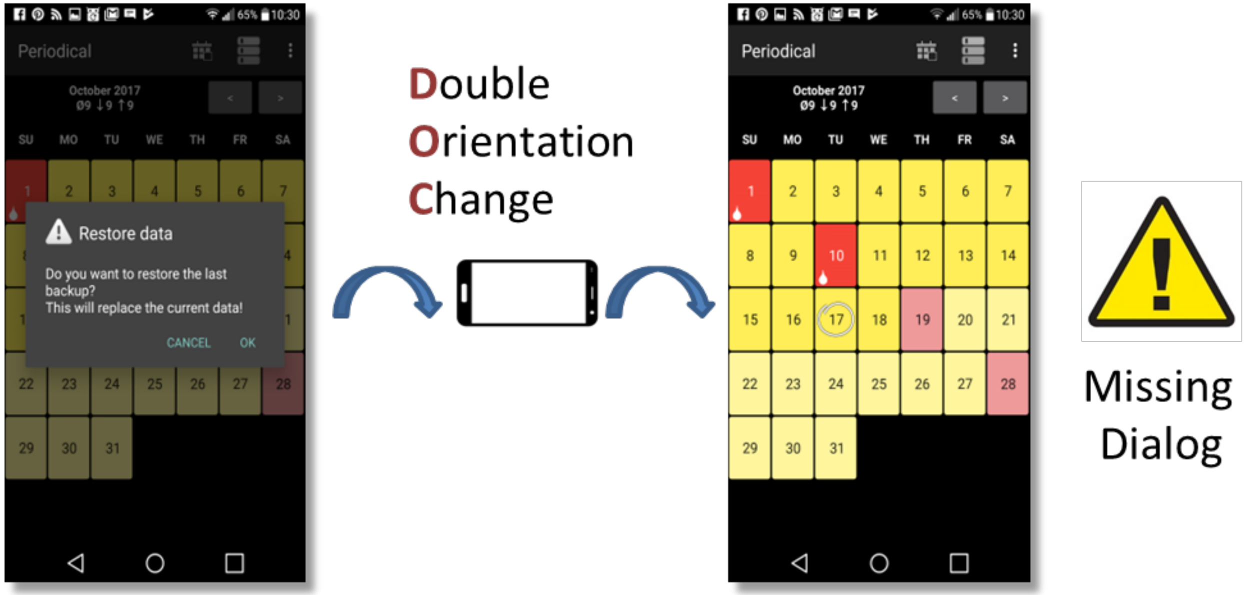
Experimental Results: RQ2

- ALARic outperformed Monkey in the ability to detect issues tied to the Activity lifecycle
  - In total ALARic triggered more crashes than Monkey
  - Monkey seeds events that exercise the Activity lifecycle, e.g. orientation changes, back button press, but it applies them without a proper strategy
- Both the tools detected an additional crash in A9 that was not tied to the Activity lifecycle

App	#DCALARic	#DCMonkey
A4	1	1
A6	1	0
A7	1	0
A9	2	0
A11	1	0
A15	2	1
Total	8	2

Lesson Learned

- **57 out of the 106 detected GUI failures involved a Dialog object disappearing from the GUI after the execution of a Lifecycle Event Sequence**



Restore data

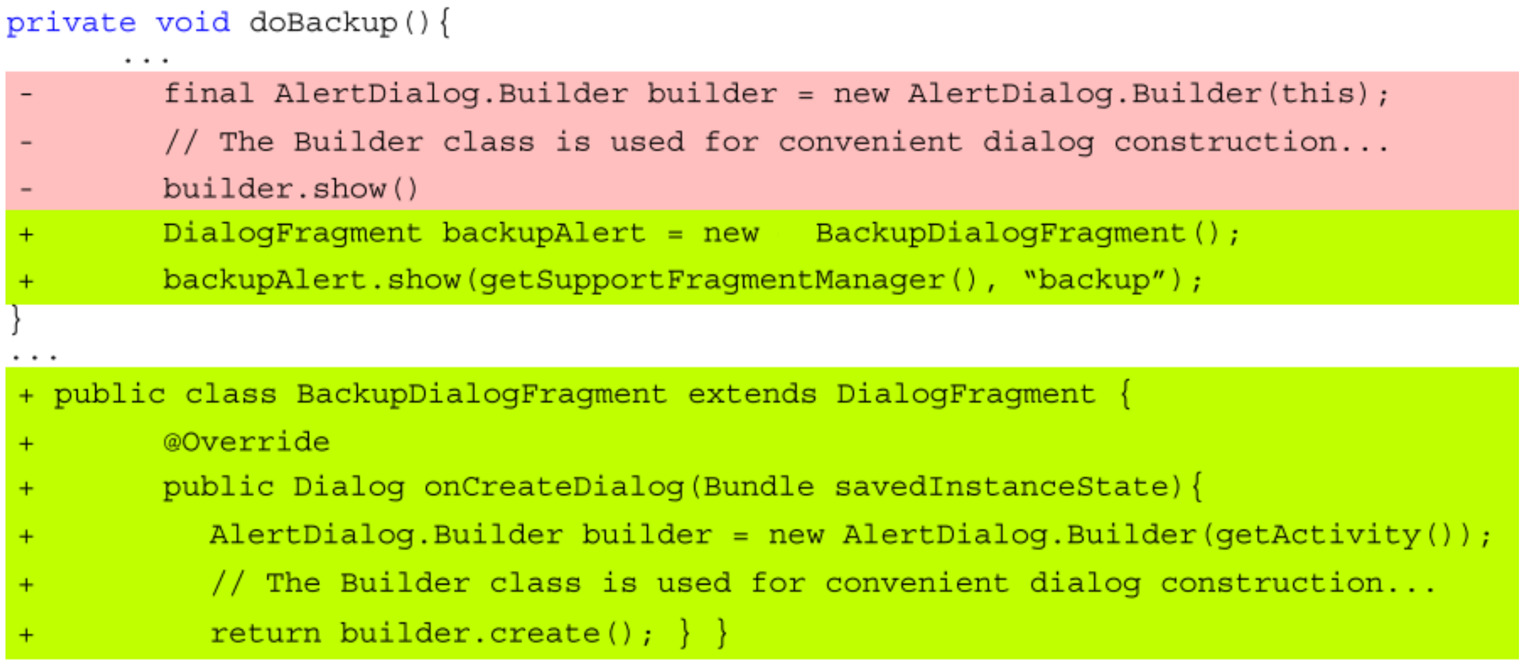
Do you want to restore the last backup?
This will replace the current data!

CANCEL OK

Double
Orientation
Change

Missing
Dialog

Lesson Learned



```
private void doBackup(){
       ...
-       final AlertDialog.Builder builder = new AlertDialog.Builder(this);
-       // The Builder class is used for convenient dialog construction...
-       builder.show()
+       DialogFragment backupAlert = new   BackupDialogFragment();
+       backupAlert.show(getSupportFragmentManager(), “backup”);
}
...
+ public class BackupDialogFragment extends DialogFragment {
+       @Override
+       public Dialog onCreateDialog(Bundle savedInstanceState){
+          AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
+          // The Builder class is used for convenient dialog construction...
+          return builder.create(); } }
```

Lesson Learned

- The debugging activity we performed in the failure validation step showed us that the faults causing the failures were mostly located outside the code that overrides the lifecycle callback methods
  - Testers should look for faults that may affect the lifecycle of the Activities also outside the methods that override the lifecycle callbacks
- Developers should correctly use the Android framework components since they may cause inconsistencies in the app behavior at runtime when Lifecycle Event Sequences occur

Conclusion

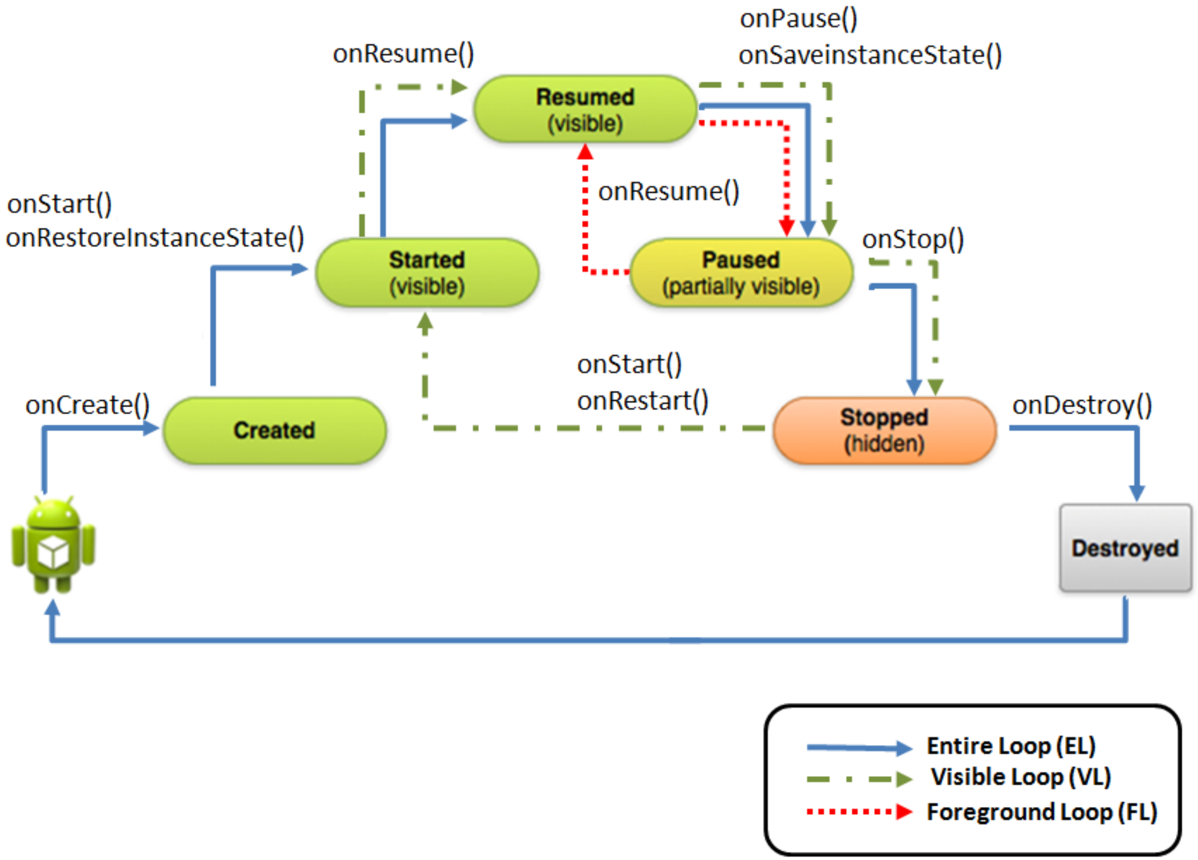
- We presented an Android automated testing technique that systematically exercises the lifecycle of app Activities to detect GUI failures and crashes
- We performed an experimental study involving 15 real world apps that showed the ability of the tool to automatically detect issues tied to the Activity lifecycle

Future Work

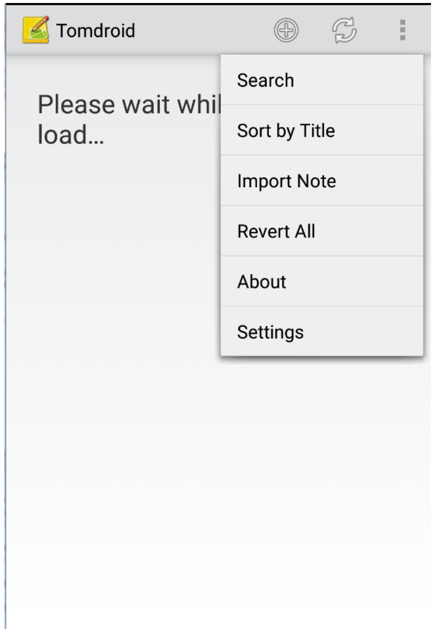
- Extension of the ALARic tool by adding other Lifecycle Event Sequences
- Design and implementation of a set of oracles able to detect other issues tied to the Activity lifecycle, such as memory leaks and threading issues
- Wider experimentation of the approach

Backup Slides

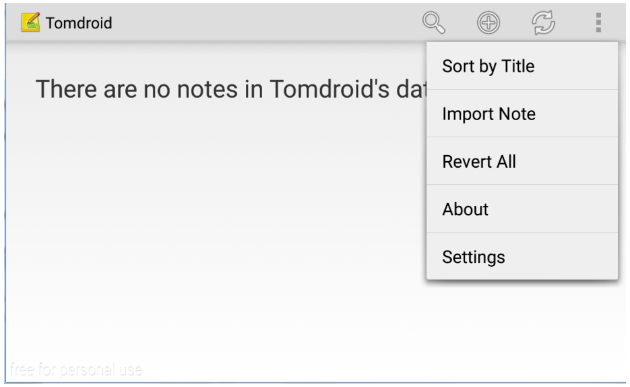
Lifecycle Event Sequences



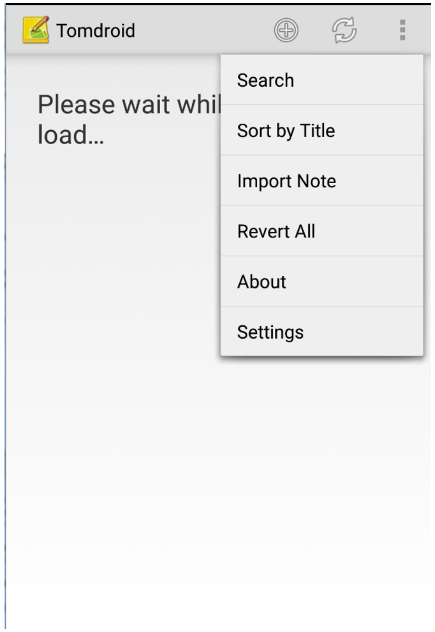
Lifecycle Event Sequences - DOC



Please wait whil
load...

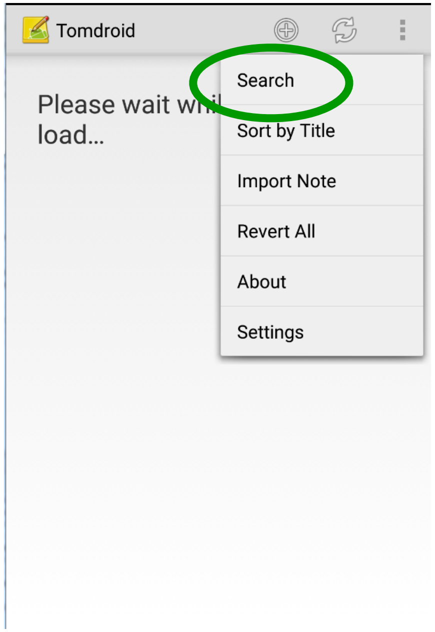


There are no notes in Tomdroid's dat

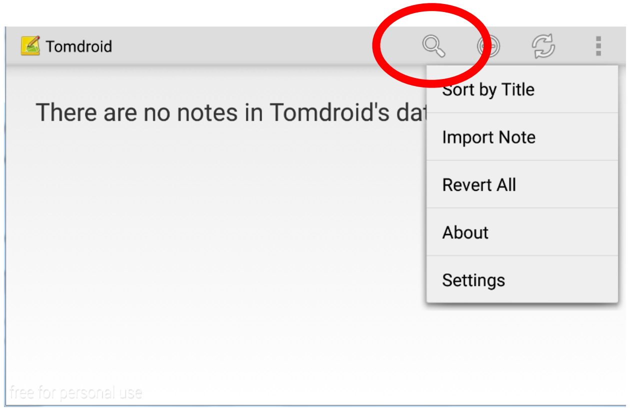


Please wait whil
load...

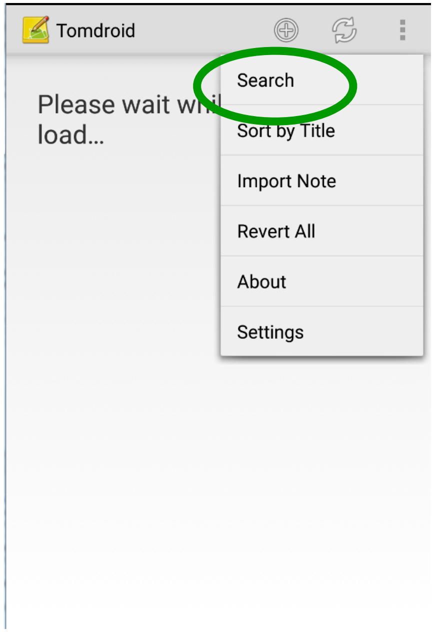
Lifecycle Event Sequences - DOC



Please wait whil
load...

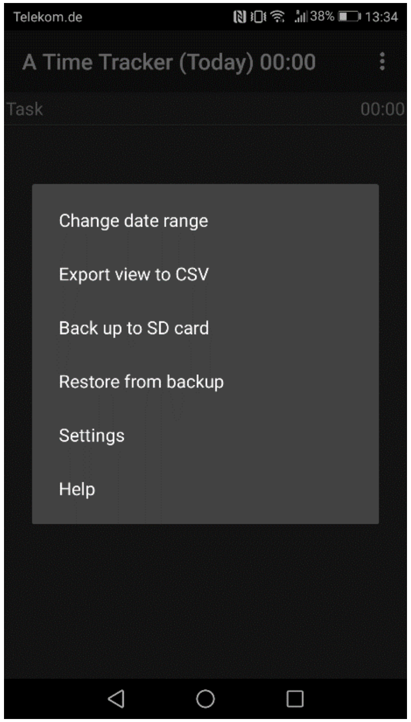


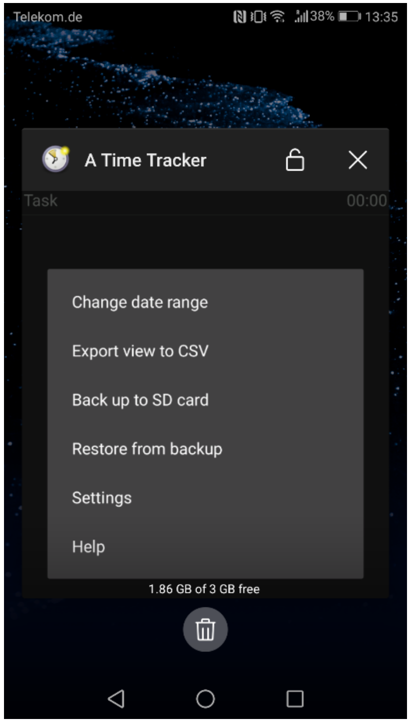
There are no notes in Tomdroid's dat

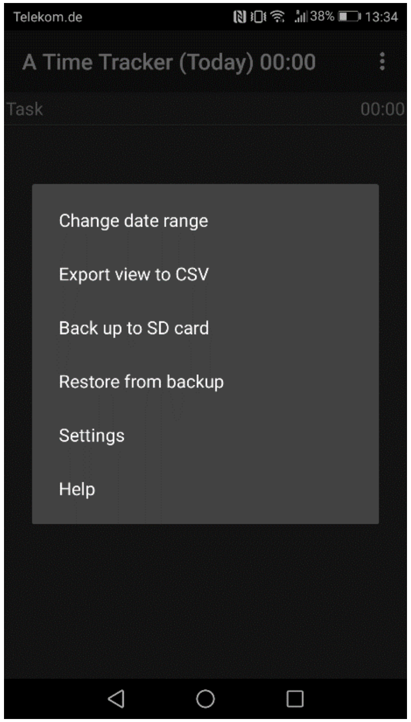


Please wait whil
load...

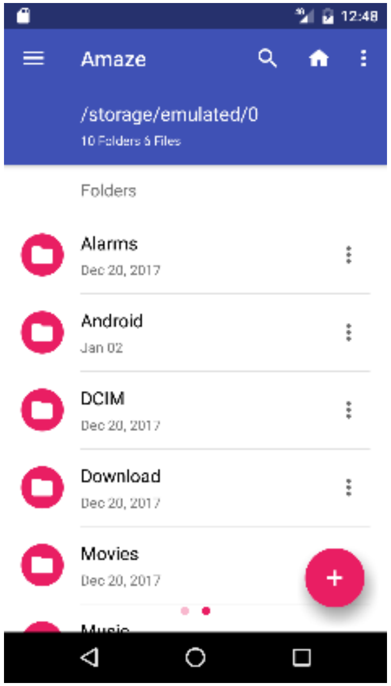
Lifecycle Event Sequences - BF

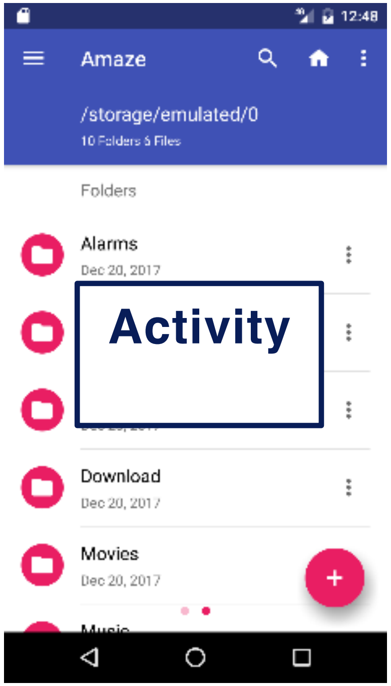




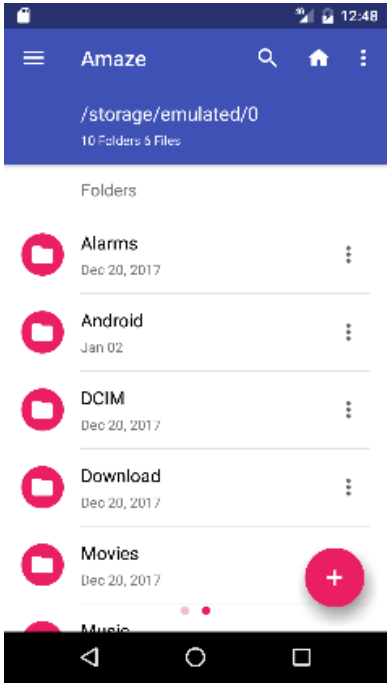


Lifecycle Event Sequences - STAI

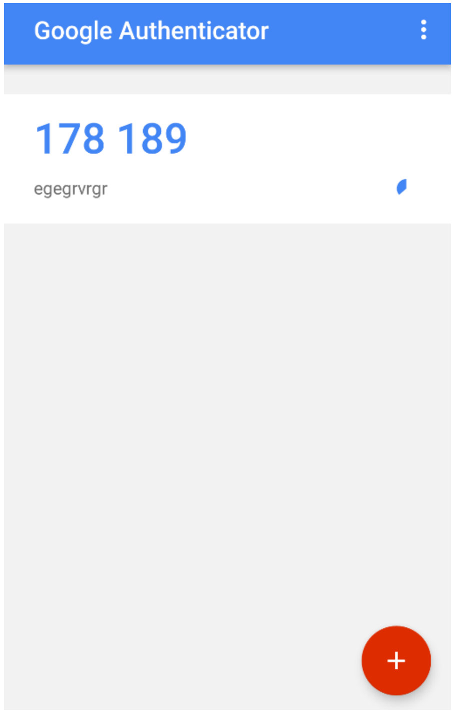




Activity



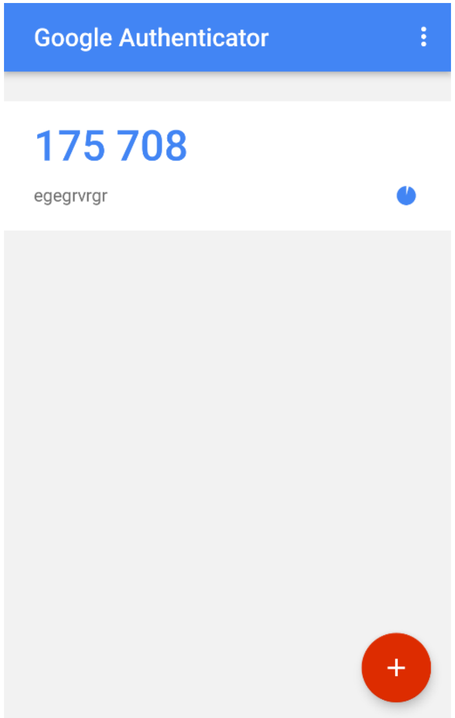
False Positive Example #1


Google Authenticator

178 189

egegrvrgr

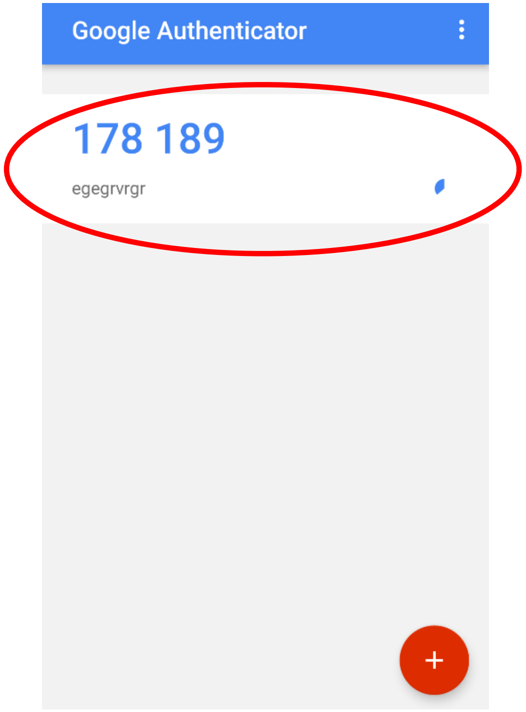
DOC
BF
STAI


Google Authenticator

175 708

egegrvrgr

False Positive Example #1

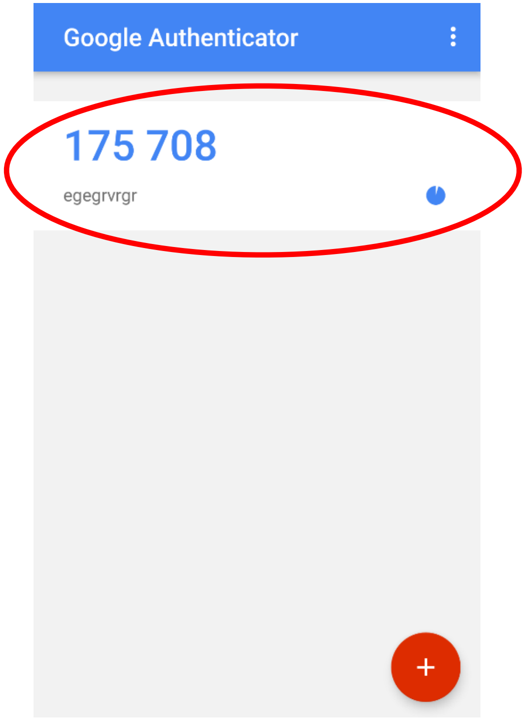


Google Authenticator

178 189

egegrvrgr

DOC
BF
STAI

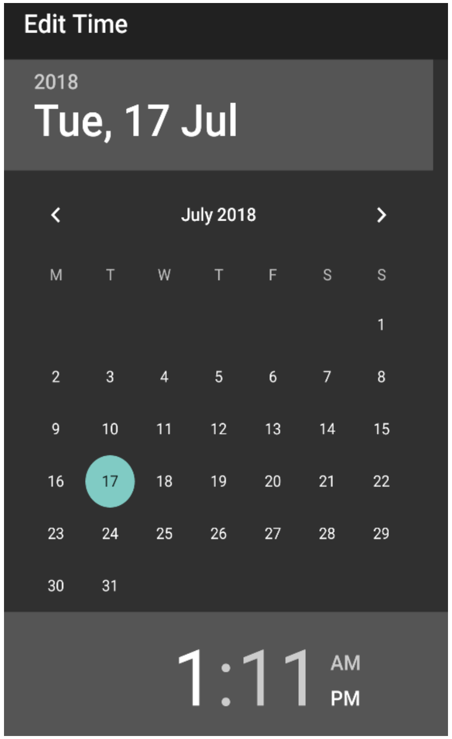


Google Authenticator

175 708

egegrvrgr

False Positive Example #2



Edit Time

2018

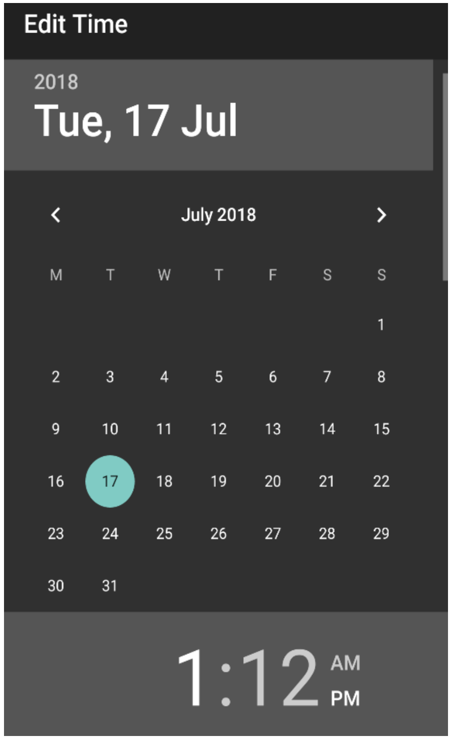
Tue, 17 Jul

July 2018

M	T	W	T	F	S	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

1:11 AM PM

BF



Edit Time

2018

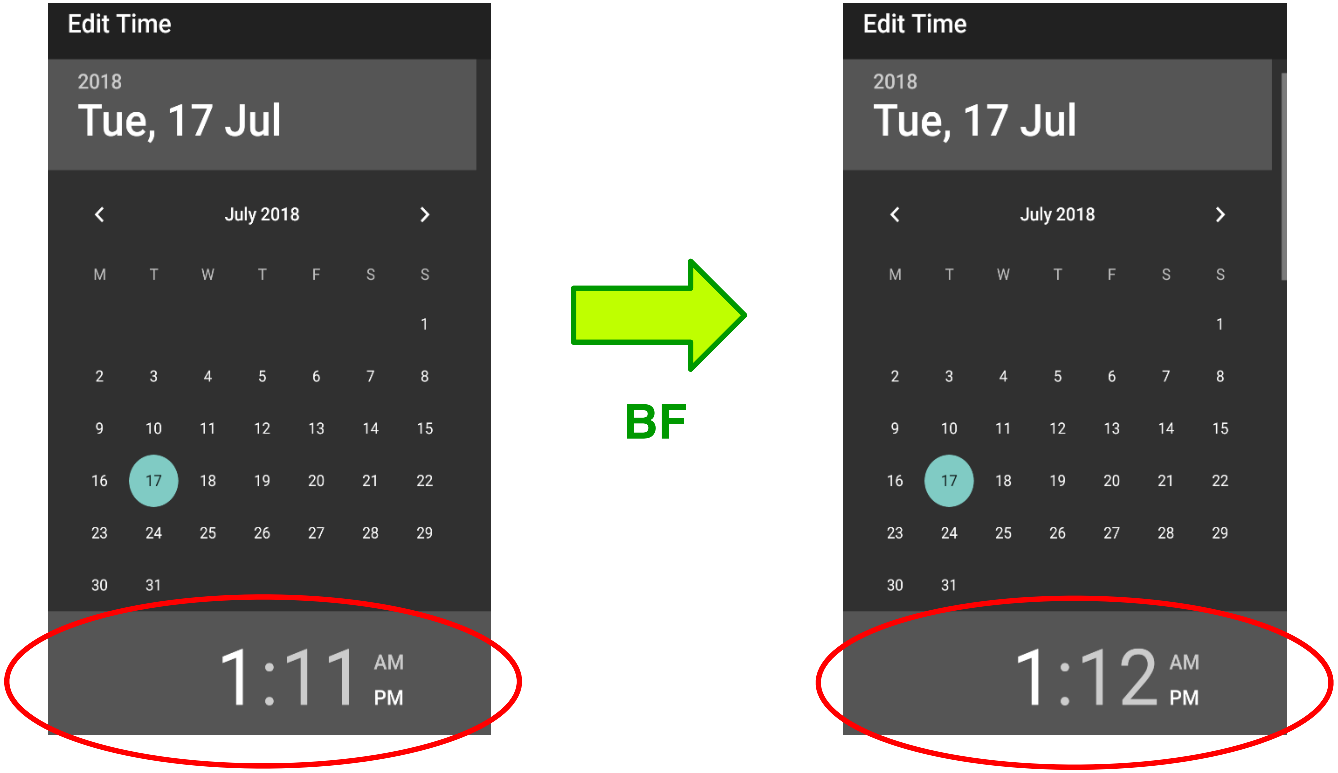
Tue, 17 Jul

July 2018

M	T	W	T	F	S	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

1:12 AM PM

False Positive Example #2



Edit Time

2018

Tue, 17 Jul

July 2018

M	T	W	T	F	S	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

1:11 AM PM

Edit Time

2018

Tue, 17 Jul

July 2018

M	T	W	T	F	S	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

1:12 AM PM