Leave Me Alone: App-level Protection Against Runtime Information Gathering on Android

Nan Zhang, Kan Yuan, Muhammad Naveed, Xiaoyong Zhou and XiaoFeng Wang

Presented by Hitakshi Annayya

Contents

RIG Attacks

Runtime-Information-Gathering (RIG)

- Collect runtime information from target app (the victim)
- Directly steal or indirectly infer sensitive user information

1) Design weaknesses of the OS

   shared communication channels such as Bluetooth

2) Side channels

   memory and network-data usages

Android Permission Issues

A malicious app needs to run side-by-side with the target app (the victim) to collect its runtime information.

A malicious app can abuse the permission it gets "to directly collect sensitive data from the target app running in the foreground."

RIG attacks exploit apps to obtain sensitive user data "ranging from phone conversations to health information;"

A game app with the Bluetooth permission for connecting to its playpad can also download patient data from a Bluetooth glucose meter."

Android-based Internet of Things (IoT)

1. Belkin NetCam Wi-Fi Camera with Night Vision

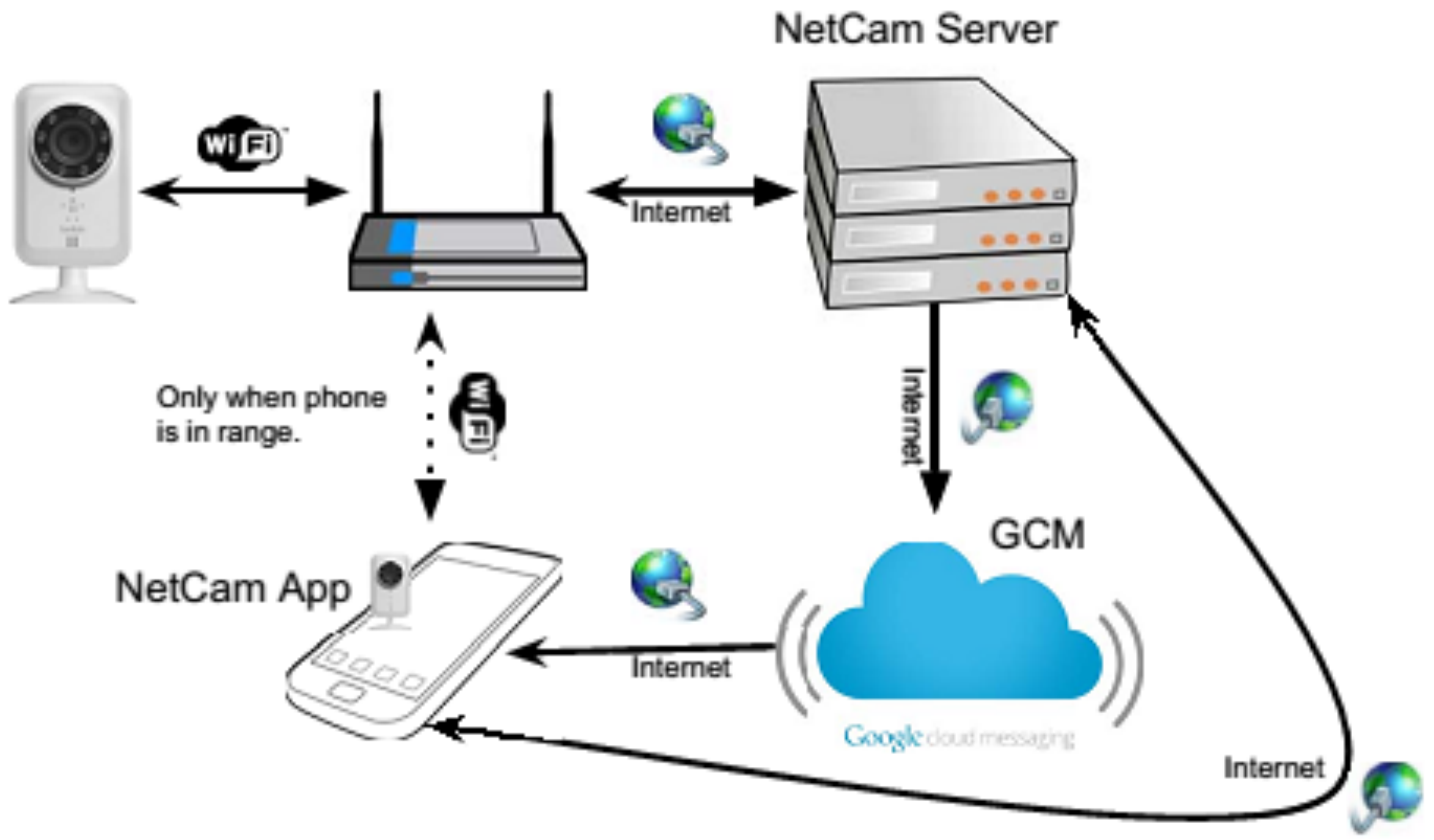
    Designed for home surveillance and motion detection

    Report to the house owner remotely

2. Nest Protect

 Shipped 440,000 of its smoke alarms in the United States between Nov. 2013 and Apr. 2014

NetCam Communication Model

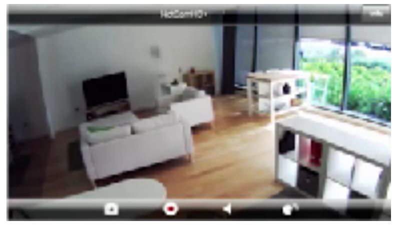


NetCam Attacks

- ➢ Utilize two side channels
  - Traffic statistics: tcp_snd and tcp_rcv
  - CPU usage: /proc/<pid>/stat

Three steps
- Infer if anybody is at home
- Mute alarm
- Infer anybody is watching surveillance

Motion Detection



https://sites.google.com/site/appguaridan/

How to Protect from RIG attack ?

Previous Works

- ➢ **Enhancing access control causes compatibility issues**
  - + Prevent information leaks during security-critical operations such as phone calls
  - + Remove public resources that could be used for a side-channel analysis
  - - Inevitably make the system less usable
  - - Cause compatibility issues

Previous Works

- **Modify OS**

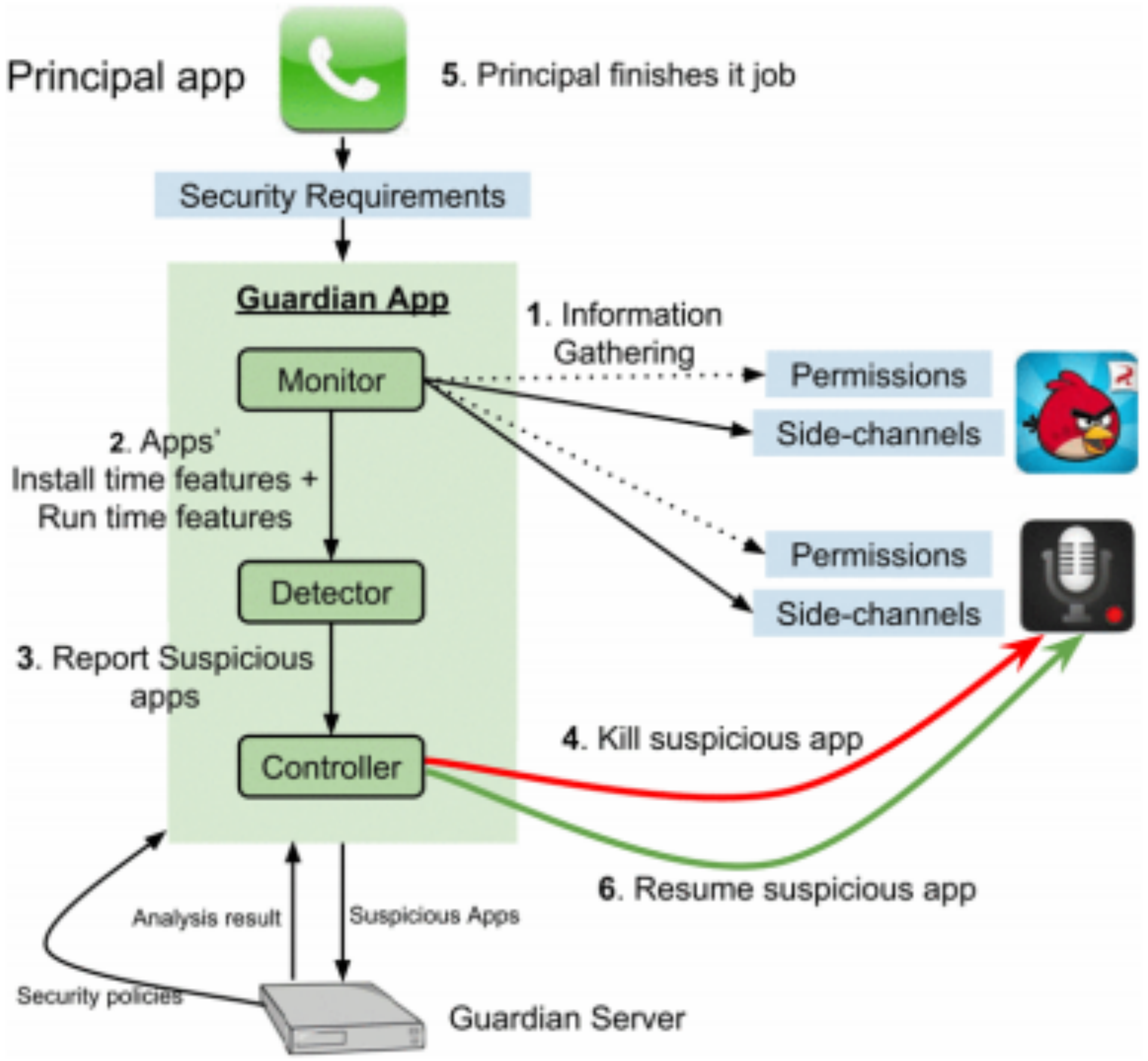
Complicated and painful (Android OS ecosystem: fragmentation)

-New protection takes a long time before it can reach Android devices worldwide;
-New RIG attacks continue to be brought to the spotlight;
-It is less clear what an app can do by itself to control its information exposed by the OS.

Researchers proposed solution

App Guardian

App Guardian

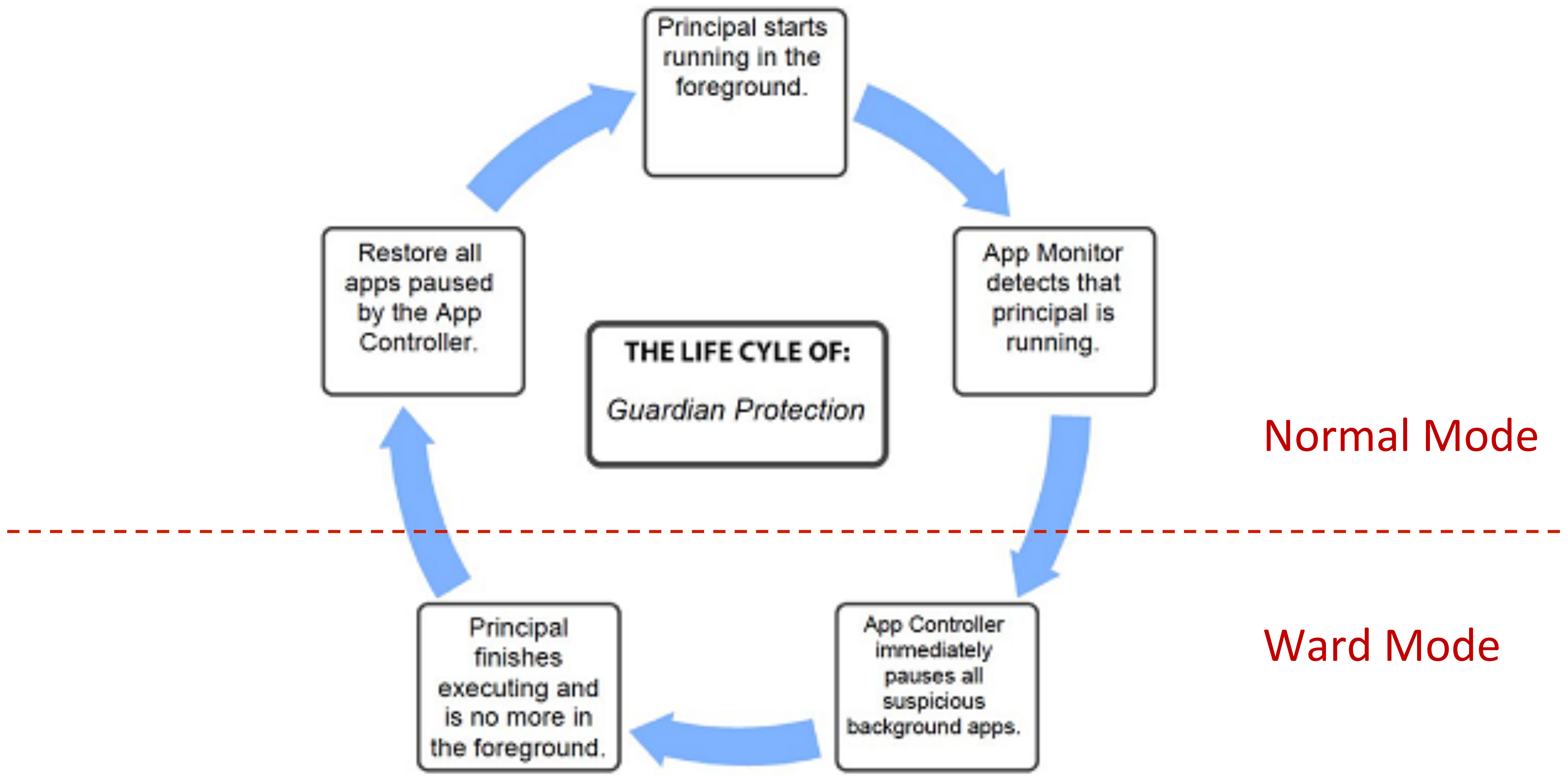


1. Information Gathering
   - Permissions, side-channels
2. Install / Run time features
3. Report suspicious apps
4. kill suspicious app
5. Principal finished
6. Resume suspicious app

Grant Guardian a set of permissions

- KILL_BACKGROUND_PROCESSES – for closing other third-party apps
- SYSTEM_ALERT_WINDOW - for popping up an alert to the user
- INTERNET – to access internet
- GET_TASK - for getting top activity
- BIND_NOTIFICATION_LISTENER_SERVICE - for controlling notifications

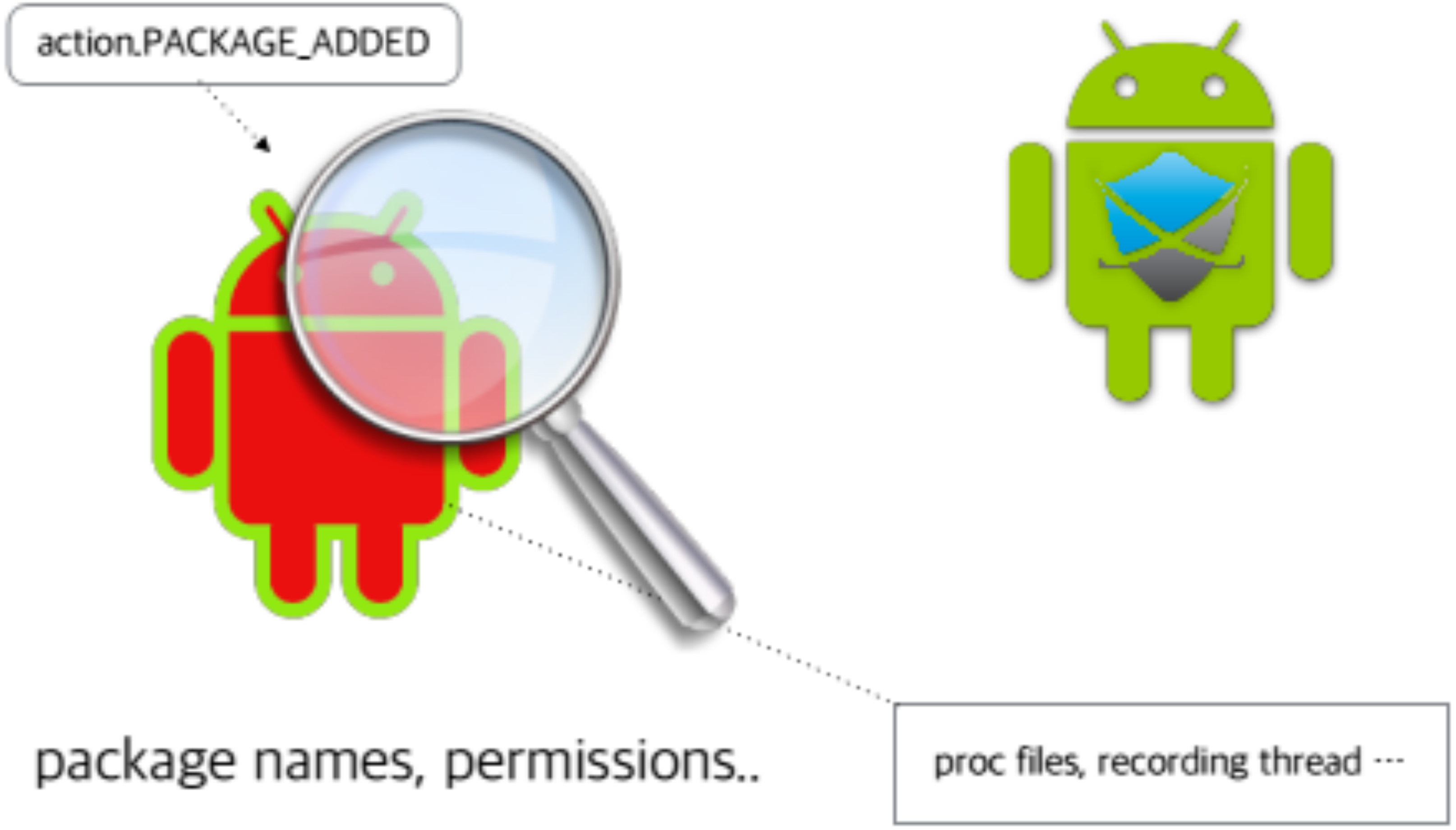
Life cycle of Guardian Protection


Principal starts running in the foreground.
App Monitor detects that principal is running.
App Controller immediately pauses all suspicious background apps.
Principal finishes executing and is no more in the foreground.
Restore all apps paused by the App Controller.
THE LIFE CYLE OF:
Guardian Protection

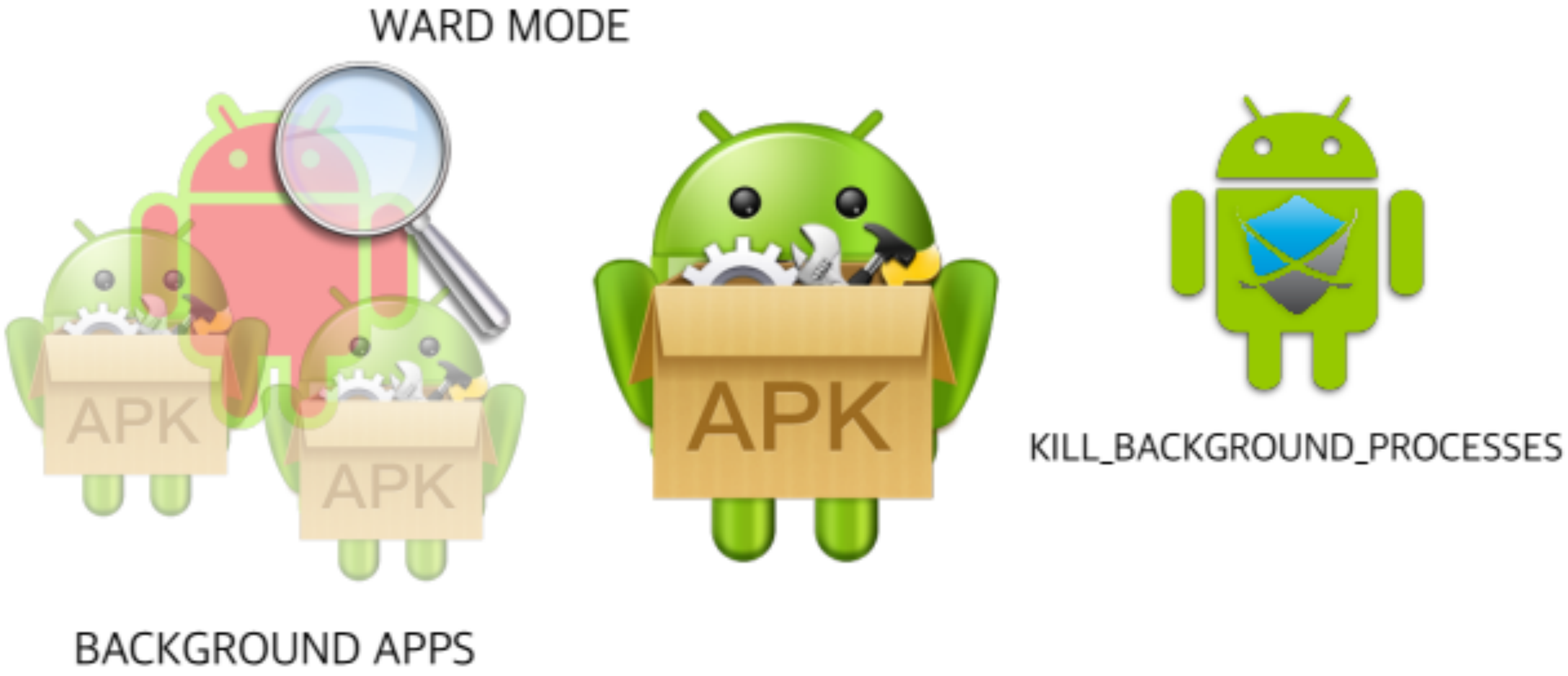
Normal Mode

Ward Mode

Monitoring


action.PACKAGE_ADDED
package names, permissions..
proc files, recording thread …

Entering the ward



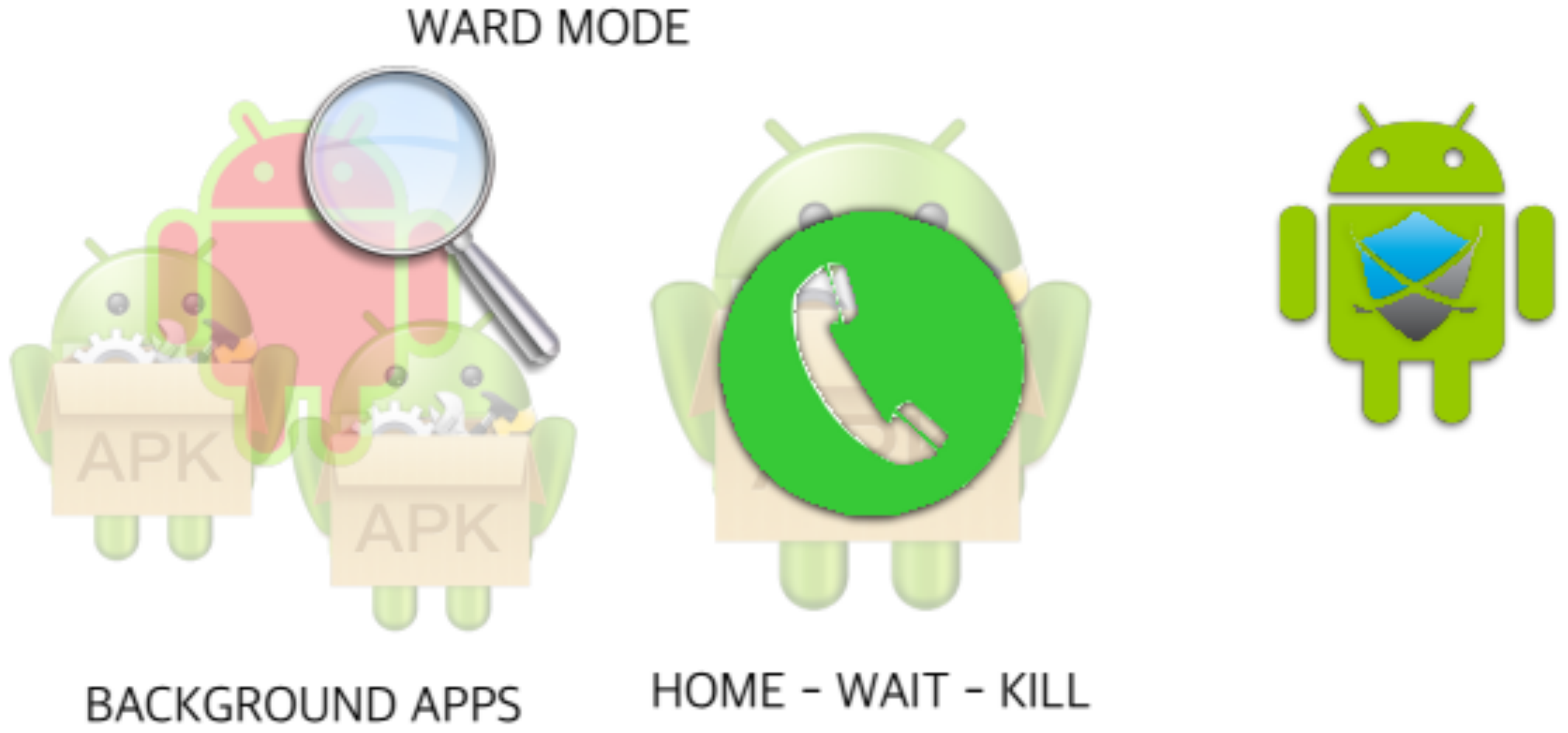
Entering the ward

oom_adj score (-17 ~ 15)

(typically) 9

2

Exiting the ward

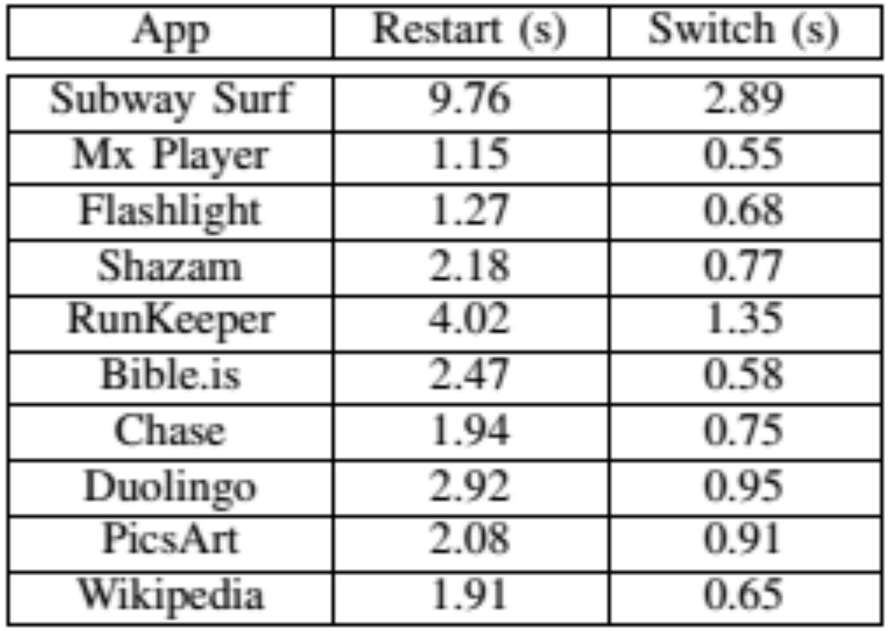


Impacts on Performance

- **Close an app which might be restarted later**

  + App states are well preserved

  - Take longer time than Switch to foreground



App	Restart (s)	Switch (s)
Subway Surf	9.76	2.89
Mx Player	1.15	0.55
Flashlight	1.27	0.68
Shazam	2.18	0.77
RunKeeper	4.02	1.35
Bible.is	2.47	0.58
Chase	1.94	0.75
Duolingo	2.92	0.95
PicsArt	2.08	0.91
Wikipedia	1.91	0.65

Finding suspicious App

- Use malicious app’s side channel

Finding suspicious App (Cont.)

Data Stealing Attacks

1. RECORD_AUDIO permission
2. Start Audioin_X process to record audio (/proc/<pid>/task/<tid>/status)

Side-channel Attacks

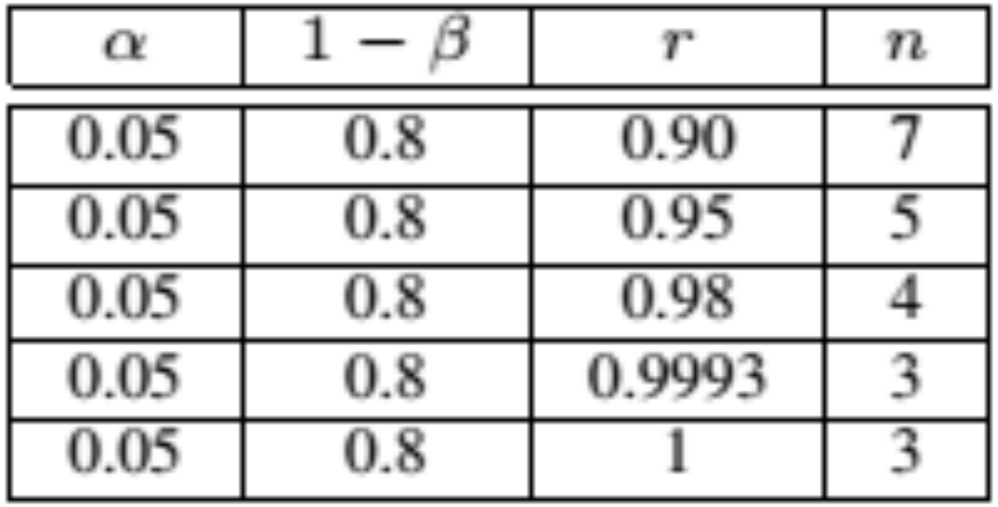
- How frequently app uses the CPU resources
- Number of times schedule to use CPU

Behavior change

- **Challenge:**
  - keep low profile before the principal show up
  - act aggressively afterwards

- **Solution:**

  Pearson correlation coefficient (r)



α	$1-\beta$	r	n
0.05	0.8	0.90	7
0.05	0.8	0.95	5
0.05	0.8	0.98	4
0.05	0.8	0.9993	3
0.05	0.8	1	3

Collusion

- **Challenge:**

  Multiple apps sample at a lower rate but still collect sufficient information

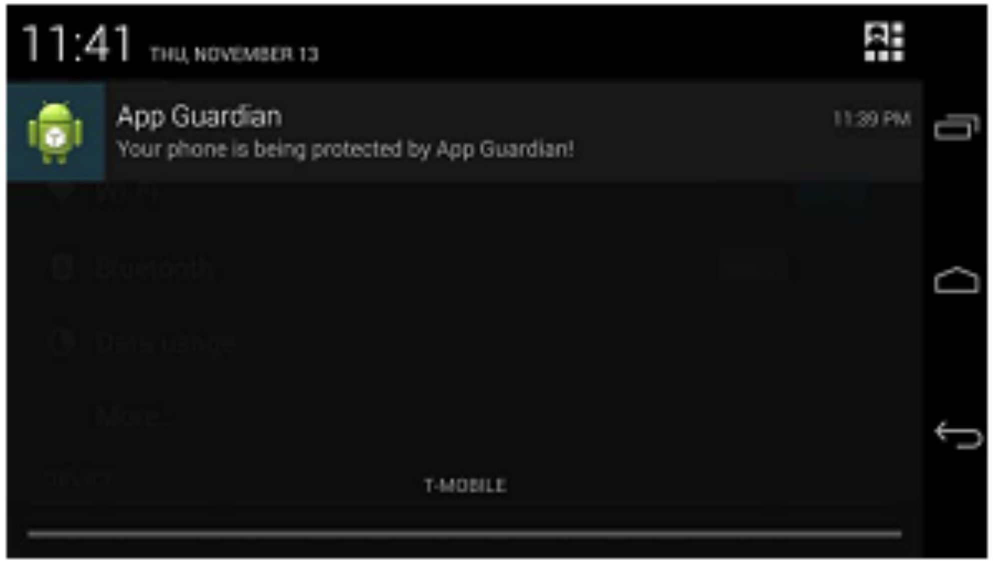
- **Solution:**
  - ➢ Grouping apps with same signature
  - ➢ Detect link-installed apps
  - ➢ Ask user if less obvious recommenendation

Self Protection

- **Use startForceground to start a service**

  Prevent it from killed by

  KILL_BACKGROUND_PROCESSES



11:41 THU, NOVEMBER 13

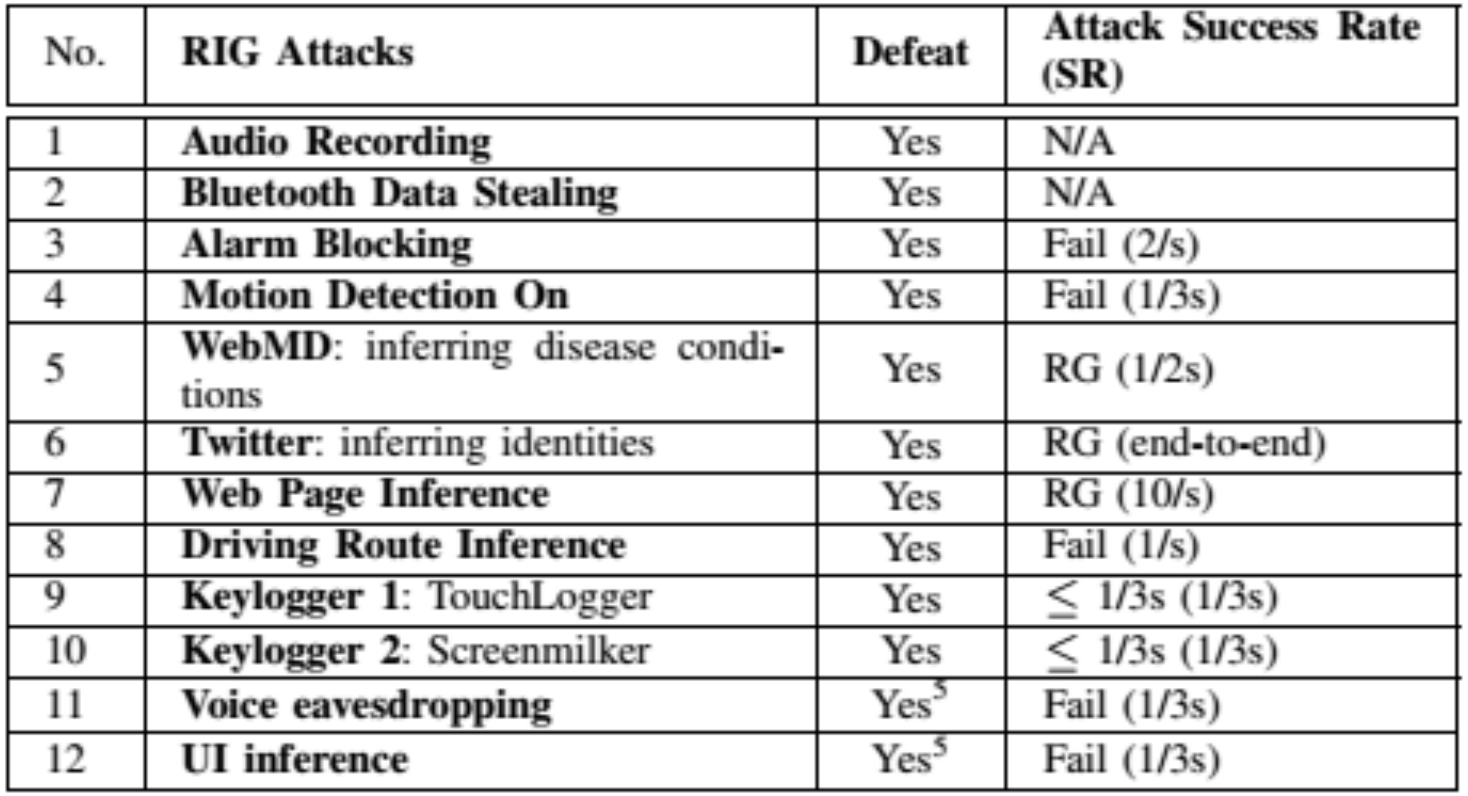
App Guardian 11:39 PM

Your phone is being protected by App Guardian!

Evaluation and analysis

Effectiveness

- Defeat all 12 RIG Attacks

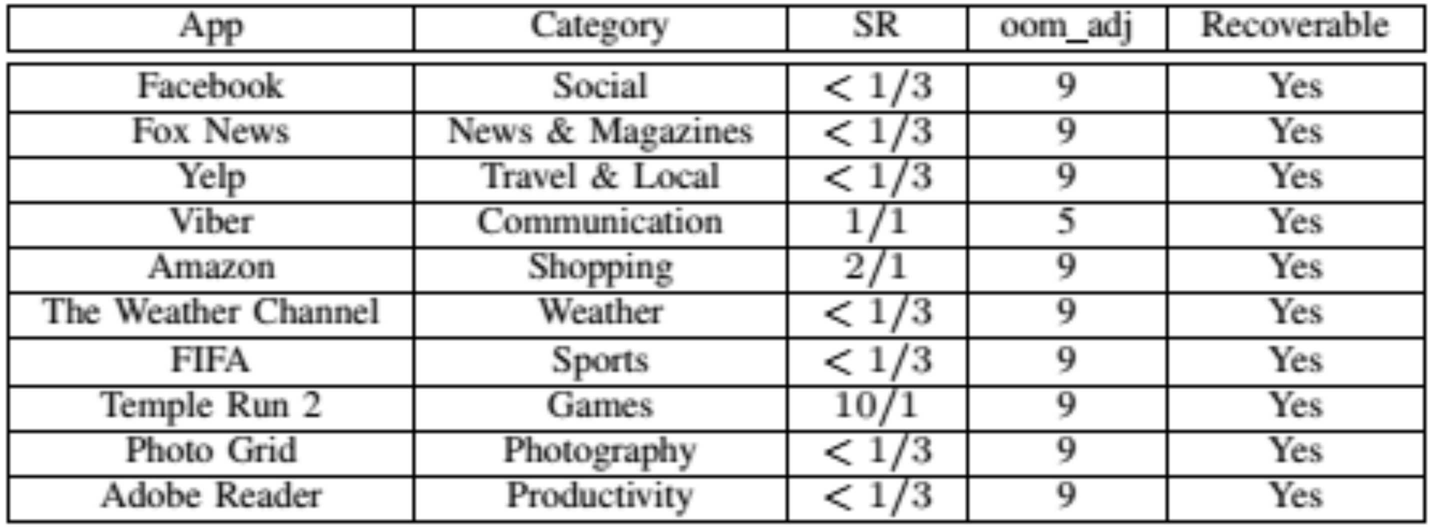


No.	**RIG Attacks**	**Defeat**	**Attack Success Rate (SR)**
1	**Audio Recording**	Yes	N/A
2	**Bluetooth Data Stealing**	Yes	N/A
3	**Alarm Blocking**	Yes	Fail (2/s)
4	**Motion Detection On**	Yes	Fail (1/3s)
5	**WebMD**: inferring disease conditions	Yes	RG (1/2s)
6	**Twitter**: inferring identities	Yes	RG (end-to-end)
7	**Web Page Inference**	Yes	RG (10/s)
8	**Driving Route Inference**	Yes	Fail (1/s)
9	**Keylogger 1**: TouchLogger	Yes	$\leq$ 1/3s (1/3s)
10	**Keylogger 2**: Screenmilker	Yes	$\leq$ 1/3s (1/3s)
11	**Voice eavesdropping**	Yes[5]	Fail (1/3s)
12	**UI inference**	Yes[5]	Fail (1/3s)

Utility Impacts and Performance

- 475 popular Apps from 27 categories on Google Play Store

  - 92 apps (19.3%) apps potentially needs to be closed
  - 8 apps (1.68%) may affect phone users' experience



App	Category	SR	oom_adj	Recoverable
Facebook	Social	$< 1/3$	9	Yes
Fox News	News & Magazines	$< 1/3$	9	Yes
Yelp	Travel & Local	$< 1/3$	9	Yes
Viber	Communication	1/1	5	Yes
Amazon	Shopping	2/1	9	Yes
The Weather Channel	Weather	$< 1/3$	9	Yes
FIFA	Sports	$< 1/3$	9	Yes
Temple Run 2	Games	10/1	9	Yes
Photo Grid	Photography	$< 1/3$	9	Yes
Adobe Reader	Productivity	$< 1/3$	9	Yes

Overhead

- **CPU & Memory usage**

Two Nexus5 phones with 250 apps installed on each

- In ward mode, 5% CPU Resource, 40MB Memory
- Out of ward mode, < 1% CPU

- **Battery Usage**

Two Nexus5 phones with 50 apps installed on each

- In ward mode, 0.12% ~ 0.18% per hour
- Out of ward mode, 0.75% ~ 1.05% per day
- Estimate a day, 0.84~ 1.18% per day

Discussion and future work

- **Detection and Separation**
  A more accurate identification of malicious activities will he

- **Background process protection**
  Protect background process at minimal cost

- **Sanitization**
  Thoroughly clean up the principals' execution
  environment after the program stop running

- **Possible side-channel attack on iOS / WatchOS**

Conclusion

- **Serious of RIG attacks on Android**

  IoT systems are also vulnerable

- **App Guardian**
  - App level protection
  - Uses side channel to protect principle

Thank you !