Static Analysis of spam call blocking applications on Android

Author: Colin Busropan

Supervisors: Dr. Yury Zhauniarovich & Dr. Apostolis Zarras

1. Background

- People receive 14 spam calls per month on average [1]
- Many spam call blocking applications exist, with millions of installations
- Existing research exists on the user experience [2] and on spam call blocklists [3]
- No research on how these applications work technically

2. Terminology

- **AndroidManifest.xml file** of an app describes essential information to build and run the application
- **Activities** are essentially the different views of an application
- **Services** can perform (long-runnning) operations in the background and don't provide an user interface
- **Broadcast receivers** are components that want to be notified when a certain broadcast is sent
- **Content providers** encapsulate data and can provide other applications access to it

3. Research questions

- What do spam call blocking applications have in common in their AndroidManifest.xml files and what differences are there?
- What is the purpose of these elements?

4. Methodology

- 10 applications from Google's Play Store were selected:
  - Must be free
  - Must have more than 100K downloads
  - Must not be exclusive to a phone carrier
  - Must not solely have a manual blocklist
- **Androguard** to parse information from the AndroidManifest.xml file

5. Findings

- 9 permissions were required by all apps, most notably to:
  - Use the internet
  - Make calls
  - Read call history
  - Read contacts
  - Monitor call status
  - Monitor cellular network
- 2 apps did **not** require permission to answer phone calls
- 5 apps required the user's location
- 6 apps implemented CallScreeningService
  - To screen calls before they arrive at the user
- 5 apps implemented InCallService
  - To manage phone calls
  - Is required to become the default phone app
- Most frequent system event subscriptions by broadcast receivers
  - **BOOT_COMPLETED** - when the system has finished booting up
  - **CONNECTIVITY_CHANGE** - a change in the network connection
  - **PHONE_STATE** - a change in the phone call state
- Some common content providers don't actually provide content but are used to initialize their SDK, because:
  - Content providers are the first components initialized during app startup
  - Developers don't have to do much to initialize it
- 1 app declared a content provider that provides information about phone numbers
  - Other apps did not use this
- 8 apps declared an activity to listen for when the system wants to perform the *DIAL* action
  - This starts the dialer activity
  - Optionally enters a phone number

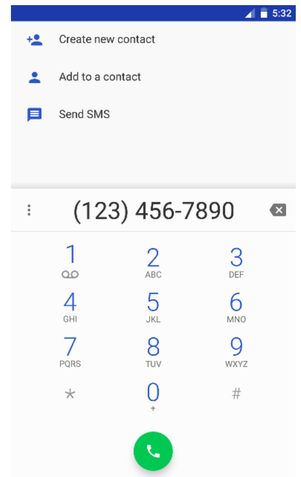

Figure 1: Example of a dialer activity, with a phone number entered.

6. Conclusions & Future work

- Sensible commonalities were found
  - Call permissions and call services
  - Broadcast receivers to monitor call state
  - Declarations to start a dialer activity
- However, some apps did not declare elements that others did declare, such as the CallScreeningService and permission to answer calls
  - (How) do they implement similar functionality?
- Internet and location permissions were required by some, but what exactly are they used for?

TU Delft Delft University of Technology

References:

[1] Hiya. (2022). State of the call - 2021 [Online; accessed 22.Apr. 2022]. https://f.hubspotusercontent30.net/hubfs/6751436/2022/Reports-and-Studies/Stateof-the-Call-2022/2022_SOTC_report.pdf

[2] Sherman, I. N., Bowers, J., McNamara Jr, K., Gilbert, J. E., Ruiz, J., & Traynor, P. (2020). Are you going to answer that? measuring user responses to anti-robocall application indicators. NDSS.

[3] Pandit, S., Perdisci, R., Ahamad, M., & Gupta, P. (2018). Towards measuring the effectiveness of telephony blacklists. NDSS.