

1. Background

- Increase in **scam calls** lead to development of applications to **block** those calls
- Some studies were conducted on their **effectiveness**
- Little is known in **technical** perspective

Research question:
What Android APIs are commonly used to intercept and block calls?

2. Methodology

1. Use AndroGuard¹ to decompile DEX² files
2. Extract methods from classes that extend Android classes in `android.telecom` and `android.telephony` packages
3. Set methods that intercept calls as entry points of call graphs³
4. Build call graphs
5. Extract other Android APIs while traversing call graphs

3. Limitations

- Extracted APIs are not guaranteed to be called at runtime
- Android APIs are provided at runtime so there are no traces of Android APIs in DEX files
 - Android applications are developed by mainly implementing callbacks
 - If Android APIs are not explicitly referenced, AndroGuard has no information about them
 - Thus, it is hard to identify if methods are overridden or not
- Functionalities of APIs need to be manually checked from Android API reference⁴

4. Results

Android API	Number of applications
BroadcastReceiver#onReceive	10
CallScreeningService#onScreenCall	6
InCallService#onCallAdded	5
Call\$Callback#onStateChanged	4
PhoneStateListener#onCallStateChanged	4
ConnectionService#onCreateIncomingConnection	1
InCallService#onConnectionEvent	1

Table 1: Android APIs for intercepting calls

Android API	Number of applications
Call#reject	4
CallScreeningService#respondToCall	4
Call#disconnect	3
TelecomManager#endCall	1

Table 2: Android APIs for blocking calls

Android API	Number of applications
TelephonyManager#getSimCountryIso	3
TelephonyManager#getNetworkCountryIso	1
TelephonyManager#isNetworkRoaming	1
TelephonyManager#getNetworkOperatorName	1
SmsMessage#getMessageBody	1
Call\$Details#getCallerNumberVerificationStatus	1
Call\$Details#getHandlePresentation	1

Table 3: Other Android APIs found

5. Conclusion

- `BroadcastReceiver#onReceive` can be used for different purposes
 - `AndroidManifest.xml` needs to be inspected for its usage
- `CallScreeningService` manages both incoming and outgoing calls while `InCallService` manages calls when a device is in a call
- `PhoneStateListener` was deprecated in API level 31
- `ConnectionService` also manages VoIP⁵
- `TelecomManager#endCall` was deprecated in API level 29
- 3 applications require country of SIM
 - This could be a sign of different behaviour depending on location
- 1 application accesses SMS messages
- `Call$Details#getCallerNumberVerificationStatus` uses STIR process described in ATIS-1000082 to verify a phone number

6. Future works

- Decompile Android runtime JAR to automate checking if a method is overridden or not
- Explore other tools to build more accurate call graph with callback awareness
- Perform taint analysis to check information leaks

1 <https://github.com/androguard/androguard>
2 Dalvik executable format, bytecodes that run on Dalvik VM used by Android
3 Directed graph where methods are vertices and edges represent calling relationships between methods
4 <https://developer.android.com/reference>
5 Voice over Internet Protocol