DRD06. Do not act on malicious intents

(THIS CODING RULE OR GUIDELINE IS UNDER CONSTRUCTION)

A malicious application may send an intent to an exported component that is not expecting to receive an intent from that application. This may trick the receiving application into taking some inappropriate action with undesirable results. In particular, if the component is not meant to be public but is nonetheless exported then such an attack may be possible.

Chin, et al., [Chin 2011] says: "*If an exported Broadcast Receiver blindly trusts an in-coming broadcast Intent, it may take inappropriate action or operate on malicious data from the broadcast Intent. Receivers often pass on commands and/or data to Services and Activities; if this is the case, the malicious Intent can propagate throughout the application.*"

Components that register to receive broadcast intents with *system actions* (such as `BATTERY_LOW`, `ACTION_POWER_CONNECTED`, `DEVICE_STORAGE_LOW`, `ACTION_SHUTDOWN`, etc.) are particularly vulnerable. Intents to inform applications about system events are broadcast by the operating system. Some action strings contained in such intents may only be added by the operating system. By registering to receive system broadcasts a broadcast receiver become publicly accessible, so a malicious application could send an intent explicitly addressed to the receiver (but not including the system action string). However, if the receiver does not check the caller's identity then it may perform some action that only the system should be able to trigger.

Chin, et al., [Chin 2011] describes malicious activity launch: "*Exported Activities can be launched by other applications with either explicit or implicit Intents. This attack is analogous to cross-site request forgeries (CSRF) on websites*" and malicious service launch: "*A malicious Service launch is similar to a malicious Activity launch, but Services typically rely on input data more heavily than Activities. Consequently, a malicious launch attack where the Intent contains data is more likely to put a Service at risk. Additionally, there are more opportunities for a bound Service to return private data to its caller because Services often provide extensive interfaces that let their binders make many method calls.*"

Component exposure should be limited by setting permission requirements in the manifest or by dynamically checking the caller's identity.

Noncompliant Code Example

This noncompliant code example shows an application that acts on receiving an intent without checking the caller's identity:

```
TBD
```

Compliant Solution (Checking Caller's Identity)

In this compliant solution the caller's identity is checked before any action is taken:

```
TBD
```

Compliant Solution (Setting Permission Requirements)

This compliant solution shows the permissions set in the manifest that prevent a malicious application from triggering an inappropriate action:

```
TBD
```

Risk Assessment

Acting on receipt of an intent without validating the caller's identity may lead to sensitive data being revealed or to denial of service.

Rule	Severity	Likelihood	Remediation Cost	Priority	Level
DRD06-J	High	Probable	Medium	P12	L1

Automated Detection

Automatic detection of the receipt of an intent is straightforward. It is not feasible to automatically determine whether appropriate checks are made of the caller's identity or whether appropriate permission requirements have been set in the manifest.

Bibliography

[Chin 2011]	Analyzing Inter-Application Communication in Android