Relevance of Android Security

Android smartphone market share > 85%

Used for sensitive applications (e-banking, healthcare, etc.)
Ambitions

Easy to use JIT feedback tool

Detection of common ICC security smells

Evaluation on real world applications
Android Lint

```java
public class CustomWebViewClient extends WebViewClient {
    @Override
    public void onReceivedSslError(WebView view, final SslErrorHandler handler)
    {
        handler.proceed();
    }

    onReceivedSslError which always proceeds more... (Strg+F1)
}
```

Static JIT analysis framework

Integrated into Android Studio

Foundation of our extension
Android Lint Architecture

```
<interface> XMLScanner

Detector

<interface> UastScanner

Issue
+ id: String
+ briefDescription: String
+ explanation: String
+ implementation: Detector

Custom Detector

implemented by

1..*
```

**IssueRegistry**
## Detected ICC-Smells

<table>
<thead>
<tr>
<th>Permitted Dynamic Permission</th>
<th>Custom Scheme Channel</th>
<th>Incorrect Protection Level</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Unauthorized Intent</strong></td>
<td>Sticky Broadcast</td>
<td>Slack WebViewClient</td>
</tr>
<tr>
<td><strong>Broken Service Permission</strong></td>
<td>Insecure Path Permission</td>
<td>Broken Path Permission Precedence</td>
</tr>
<tr>
<td>Unprotected Broadcast Receiver</td>
<td>Implicit Pending Intent</td>
<td>Common Task Affinity</td>
</tr>
</tbody>
</table>
Android ICC Overview

**Intent:** Message object

**Activity:** Main user interface component

**Service:** Handles background tasks

**BroadcastReceiver:** Reacts on system or app events
Unauthorized Intent

Implicit Intent (Broadcast)

Intent i = new Intent("test.action");
i.putExtra("secret", privateData);
sendBroadcast(i);

Received by every matching receiver

Legit Receiver A

Legit Receiver B

Malicious Receiver
Mitigation

Receiver requires "custom" permission

```
Intent i = new Intent("test.action");
i.putExtra("secret", privateData);
sendBroadcast(i, "custom");
```

- Legit Receiver A
- Legit Receiver B
- Malicious Receiver
Demo

Demo 1: Unauthorized Intent
Evaluation

~700/100 apps analyzed in batch / manual mode

Fixed occurring crashes and improved detection

Studied smell prevalence and distribution
Smell Prevalence

- Insecure Path Permission: 2
- Persisted Dynamic Permission: 49
- Unprotected Broadcast Receiver: 60
- Custom Scheme Channel: 82
- Unauthorized Intent: 732
- All Apps: 732

# apps affected
Lessons Learned

Android Lint

Easy to extend effectively
(Documentation sometimes bad)
Lessons Learned

Wrong implementation of in-app communication (Intra-app communication rare, mostly opening browser)

Intent intent = new Intent(SHOW_PROGRESS);
intent.putExtra("show", show);
intent.putExtra("recording", recording);
intent.putExtra("sec", sec);
intent.putExtra("phone", phone);
context.sendBroadcast(intent);
Future Work

Quickfixes

Taint Analysis for Android Lint

Usability Study
Summary

Android Lint

ICC Security Smells

Demo

Evaluation

Lessons Learned

Future Work