Two Novel Defences against Motion-Based Keystroke Inference Attacks

Yihang (Frank) Song
Rahul Rawat
Madhur Kukreti
Urs Hengartner
Side Channels

Keyboard Acoustic Side Channel

Electro Magnetic Side Channel

Cold-boot Side Channel
Motion-Based Side Channel

Soft keyboard requires a tap for data input

A tap results in change of phone’s orientation and acceleration

Background app can infer user input from these sensors
Permissions to the Rescue?

Data collected by motion sensors is not considered sensitive by Android, iOS, and Blackberry

Apps can access the readings of accelerometer and orientation sensor without requiring any security permissions
Contributions

Two novel defences against motion-based keystroke inference attacks

1) Modified sensor readings
2) Randomized keyboard layout

Impact of these defences on usability and on apps with legitimate need for sensor data
Motion-Based Side Channel Attack Papers

TouchLogger, HotSec 2011
ACCessory, HotMobile 2012
**TapLogger, WiSec 2012**
Cai and Chen, Trust 2012
TapPrints, MobiSys 2012
Aviv et al., ACSAC 2012
Al-Haiqi et al., ICEEI 2013
TapLogger [Xu et al., WiSec 2012]

Trojan app

Training mode: app runs in foreground, gathers sensor data and tap events, trains model

Test mode: app runs in background, gathers sensor data, detects and infers tap events

50% accuracy for top 1 inference on number pad, 80% accuracy for top 4 inferences
Attack Outline

1) Tap Detection: find start and end of a tap in sensor data

2) Tap Inference: infer the pressed key based on the sensor readings in that period
Defences

Block tap detection or tap inference

Defence against tap detection: Accuracy reduction of sensor data

Defence against tap inference: Random keyboard layout
Accuracy Reduction of Sensor Data

Reduce accuracy such that legitimate apps remain useful but motion-based side channel attack fails.

Can be applied to tap detection or tap inference.

Different researchers have used different sensors for tap inference, requires comprehensive defence.

Tap detection generally relies on accelerometer, enables simpler defence.
Tap Detection based on Square Sum of Acceleration

[Xu et al., WiSec 2012]
Idea

Tapping causes relatively small changes in square sum of acceleration

Activities like walking or playing motion-oriented games cause much bigger changes

Keep square sum of reported acceleration constant if small change is detection and report unmodified value for large changes
Implementation

Modified SensorManager

Maintain direction of acceleration, but set square sum to \(9.8 \times 9.8 \text{ m}^2/\text{s}^4\) if square sum within \((80, 130)\) \(\text{m}^2/\text{s}^4\)

Currently done for all apps, but necessary only for background apps
Random Keyboard Layout Generation

Tap inference requires malicious app to know layout of number pad or keyboard

Generate random layout to hide this knowledge from malicious app

How often should we generate a new random layout?
Implementation

New random layout whenever keyboard pops up to avoid frequency analysis attacks
Evaluation – Impact of Accuracy Reduction on Tap Detection

User enters 300 keystrokes on stock Nexus One and modified Nexus One

Tap detection code from TapLogger

<table>
<thead>
<tr>
<th></th>
<th>True positives</th>
<th>False positives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stock N1</td>
<td>120</td>
<td>60</td>
</tr>
<tr>
<td>Modified N1</td>
<td>50</td>
<td>8</td>
</tr>
</tbody>
</table>

Accuracy reduction by > 50%, only one out of six taps is detected
Evaluation – Impact of Accuracy Reduction on Other Apps

10 participants play three stages of Crazy Labyrinth 3D on stock Nexus One and modified Nexus One
Evaluation – Impact of Accuracy Reduction on Other Apps

Participants took less time on modified device

<table>
<thead>
<tr>
<th></th>
<th>Stage 1</th>
<th>Stage 2</th>
<th>Stage 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stock Mean [s]</td>
<td>6.6</td>
<td>6.4</td>
<td>9.0</td>
</tr>
<tr>
<td>Mod. Mean [s]</td>
<td>6.5</td>
<td>5.5</td>
<td>7.3</td>
</tr>
</tbody>
</table>

3 participants preferred modified device
2 participants preferred stock device
5 participants had no preference
Evaluation – Impact of Accuracy Reduction on Other Apps

Run step-counting app on stock Nexus One and modified Nexus One and walk 100 steps

Result: 118 vs. 116 steps
Evaluation - Impact of Random Keyboard Layout on Input Time

10 participants entered two strings on normal and random keyboard layout

String 1: This is a message. I would like to type the message.
String 2: thisismypassword123

<table>
<thead>
<tr>
<th></th>
<th>Normal</th>
<th>Random</th>
<th>Normal</th>
<th>Random</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mean(s)</strong></td>
<td>00:23</td>
<td>01:02</td>
<td>00:10</td>
<td>00:28</td>
</tr>
</tbody>
</table>

Participants willing to use our defence
Limitations

Small number of apps and participants

One type of device with small screen
Other Defences

- Lowering the sampling rate
- Introducing new permissions
- Denying access to sensors while inputting (sensitive) data
- Vetting apps
- Using a leather or rubber case

Our two defences require no changes to apps and little security understanding
Future Work

Study reduction of sampling rate

Determine optimal frequency to shuffle keyboard layout

Introduce detection/notification mechanism for sensitive input to trigger shuffle of keyboard layout

Study defence for other sensors that could be used for tap detection
Conclusions

Two defences against motion-based keystroke inference attacks

Defence based on reducing accuracy has no impact on other apps

Defence based on random keyboard layout significantly slows down users