TouchSignatures: Identification of User Touch Actions based on Mobile Sensors via JavaScript

Maryam Mehrnezhad, Ehsan Toreini, Siamak F. Shahandashti, Feng Hao
School of Computer Scinece, Newcastle University
Newcastle upon Tyne, United Kingdom
{m.mehrnezhad, ehsan.toreini, siamak.shahandashti, feng.hao}@ncl.ac.uk

Introduction. Conforming to the recent W3C specifications (www.w3.org/TR/orientation-event), modern mobile web browsers generally allow JavaScript code in a web page to access *motion and orientation* sensor data without the user's permission. The associated risks to user privacy are however not considered in W3C specifications. In this work, for the first time, we show how user privacy can be compromised using device motion and orientation sensor data available in-browser, despite the fact that the data rate is 5 to 10 times slower than what is attainable in-app. We examine different browsers on the Android and iOS platforms and study their policies in granting permissions to JavaScript code with respect to access to motion and orientation sensor data and identify multiple vulnerabilities. Based on our findings, we propose *TouchSignatures*, implementation of an attack in which malicious JavaScript code on an inactive tab listens to such sensor data measurements. Based on these streams, *TouchSignatures* is able to distinguish the user's touch actions (e.g., tap, scroll, hold, and zoom) on an active tab, allowing the remote website to learn the client-side user activities. Finally, we demonstrate the practicality of this attack by collecting real-world user data and reporting high success rates using our proof-of-concept implementation.

Vulnerabilities. We have developed JavaScript code to record the sensor data streams, and have carried out tests on different combinations of mobile operating systems and browsers. As shown in Table 1, all major browsers allow JavaScript code to access the sensor data if the user is interacting with the *same tab*. Furthermore, this access is provided to all separate frames of a web page. This means that malicious code within an `iframe` (e.g., showing a third-party ad) is able to get access to the sensor data available to the main web page despite the fact that the iframe has a different web origin from the main page. We show this in the table under *intra tab* and identify it as a privacy leakage vector. Even worse, in some cases, access to the sensor data is provided even to JavaScript code in a tab other than the one with which the user is interacting. This is shown under *other tab* in the table and obviously demonstrates a flaw in browser security policies. In some browsers, e.g., Maxthon on Android and UC Browser on iOS, (not included in the table due to space constraint), the code has access to the sensor data even if the browser is in the background.

Device mobile OS	Browser (version)	same tab	intra tab	other tab
Nexus 5 Android 4.4.4	Chrome (38.0.2125.102)	yes	*yes*	—
	Firefox (33)	yes	*yes*	—
	Opera (21.0.1619.86037)	yes	*yes*	—
iPhone 5 iOS 8.1	Safari (8.0)	yes	*yes*	—
	Chrome (38.0.2125.67)	yes	*yes*	*yes*
	Opera Mini (8.0.5)	yes	*yes*	*yes*

Table 1: Mobile browser access to the motion and orientation sensor on Android and iOS. A *yes* (in italics) indicates a potential privacy leakage vector.

Each user touch action, such as click, scroll, zoom, and hold, induces a distinctive device motion and orientation trace. *TouchSignatures* tries to identify these touch actions given access to the sensor data through malicious JavaScript code embedded either intra-tab within an `iframe` or within another tab. This reveal potentially sensitive information about the user's interaction with other webpages or apps.

Implementation and Experiments. To collect labelled data, we have implemented JavaScript code for the client side which asks the user to follow certain steps and collects touch actions and their associated sensor streams. On the server side, we have developed a NodeJS-based server to handle communications, and a NoSQL database to handle the storage of the captured sensor data continuously. To implement the attack, we apply two classifiers: the first one classifies *click, hold, scroll, zoom in, and zoom out* and the second one identifies the type of scroll: *up, down, right, and left*. We identify appropriate time- and frequency-domain features that are extracted from the eavesdropped sensor trace. The classifiers then apply the k-nearest neighbour (k-NN) algorithm to identify user's touch actions.

Results. Our two classifiers achieve total identification rates of 87.39% and 61.59%, respectively. Our attacks highlight major flaws in the access control policy of both mobile OSs and mobile browsers with respect to user privacy. As a countermeasure which strikes a balance between security and usability, we suggest that device motion and orientation data be treated similarly to GPS, and comparable user notifications and control mechanisms are implemented in mobile OSs and browsers. (**Acknowledgement**: the last three authors are supported by ERC Starting Grant 106591.)

ASIA CCS '15, Apr 14-17, 2015, Singapore, Singapore
ACM 978-1-4503-3245-3/15/04.
http://dx.doi.org/10.1145/2714576.2714650