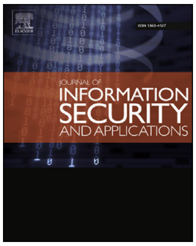
Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/jisa



TouchSignatures: Identification of user touch actions and PINs based on mobile sensor data via JavaScript [☆]

Maryam Mehrnezhad [*], *Ehsan Toreini, Siamak F. Shahandashti, Feng Hao*

School of Computing Science, Newcastle University, Newcastle, UK

ARTICLE INFO

Article history:
Available online 7 January 2016

Keywords:
Mobile sensors
JavaScript attack
Mobile browsers
Mobile security
Touch actions
PINs

ABSTRACT

Conforming to W3C specifications, mobile web browsers allow JavaScript code in a web page to access *motion and orientation* sensor data without the user's permission. The associated risks to user security and privacy are however not considered in W3C specifications. In this work, for the first time, we show how user security can be compromised using these sensor data via browser, despite that the data rate is 3–5 times slower than what is available in app. We examine multiple popular browsers on Android and iOS platforms and study their policies in granting permissions to JavaScript code with respect to access to motion and orientation sensor data. Based on our observations, we identify multiple vulnerabilities, and propose *TouchSignatures* which implements an attack where malicious JavaScript code on an attack tab listens to such sensor data measurements. Based on these streams, *TouchSignatures* is able to distinguish the user's touch actions (i.e., tap, scroll, hold, and zoom) and her PINs, allowing a remote website to learn the client-side user activities. We demonstrate the practicality of this attack by collecting data from real users and reporting high success rates using our proof-of-concept implementations. We also present a set of potential solutions to address the vulnerabilities. The W3C community and major mobile browser vendors including Mozilla, Google, Apple and Opera have acknowledged our work and are implementing some of our proposed countermeasures.

1. Introduction

1.1. *Access to mobile sensors within app*

Sensor-rich mobile devices are becoming ubiquitous. Smart phones and tablets are increasingly equipped with a multitude of sensors such as GPS, gyroscope, compass, and accelerometer. Data provided by such sensors, combined with the growing computation capabilities of modern mobile devices, enable richer, more personalised, and more usable apps on such devices. On the other hand, access to the sensor streams provides an app running in the background a side channel. Listening to mobile sensor data via a background process either for improving user security (Bo et al., 2013; Bojinov et al., 2014; De Luca et al., 2012; Halevi et al., 2012; Li et al., 2013; Riva et al., 2012; Shrestha et al., 2014; Velten et al., 2015) or attacking it (Cai and Chen, 2011; Michalevsky et al., 2014; Miluzzo et al.,

[☆] A preliminary version of the paper was presented at ASIACCS 2015 as a poster (Mehrnezhad et al., 2015).

[*] *Corresponding author.* School of Computing Science, Newcastle University, Newcastle, UK. Tel.: +44 1912085153; fax: +44 191 208 8788. *E-mail address:* m.mehrnezhad@ncl.ac.uk (M. Mehrnezhad).

http://dx.doi.org/10.1016/j.jisa.2015.11.007

Table 1 – Brief description of *TouchSignatures* and in-app sensor-based Password/PIN identifiers. Acc = accelerometer, Gyr = gyroscope, and Ori = Orientation. Motion streams are a set of measurements which are accessible within browsers and include accelerometer, accelerometer-including-gravity, and rotation rate (see Section 3.2).

Work	Sensor	Identification item	Access
PIN Skimmer (Simon and Anderson, 2013)	Camera, Mic	PINs	in-app
PIN Skimming (Spreitzer, 2014)	Light	PINs	in-app
Keylogging by Mic (Narain et al., 2014)	Mic	Keyboard, PINs	in-app
ACCessory (Owusu et al., 2012)	Acc	Keyboard, Area taps	in-app
Tapprints (Miluzzo et al., 2012)	Acc, Gyr	Keyboard, Icon taps	in-app
Acc side channel (Aviv et al., 2012)	Acc	PINs, Patterns	in-app
Motion side channel (Cai and Chen, 2012)	Acc, Gyr	Keyboard, PINs	in-app
TapLogger (Xu et al., 2012)	Acc, Ori	PINs	in-app
TouchLogger (Cai and Chen, 2011)	Ori	PINs	in-app
TouchSignatures	Motion, Ori	Touch actions, PINs	in-browser

2012; Owusu et al., 2012; Xu et al., 2012) has been always interesting for researchers.

Listening to the sensor data through a malicious background process may enable the app to compromise the user security. Here, we present Table 1 and briefly describe the existing in-app sensor-based password/PIN identifiers. Some of the existing works in Table 1 try to identify PINs and Passwords by using sensors such as light, camera and microphone (Narain et al., 2014; Simon and Anderson, 2013; Spreitzer, 2014). In this paper, we are interested in the use of accelerometer and gyroscope sensors as a side channel to learn about user's PINs and Passwords (Aviv et al., 2012; Cai and Chen, 2011; Miluzzo et al., 2012; Owusu et al., 2012; Xu et al., 2012).

1.2. *Access to mobile sensors within browser*

All these attacks suggest to obtain sensor data through a background process activated by a mobile app, which requires installation and user permission. By contrast, *TouchSignatures* suggests to record the sensor measurements via JavaScript code without any user permission. This is the first report of such a JavaScript-based attack. This attack is potentially more dangerous than previous app-based attacks as it does not need any user permission for installation to run the attack code.

Mobile web applications are increasingly provided access to more mobile resources, particularly sensor data. Client-side scripting languages such as JavaScript are progressively providing richer APIs to access mobile sensor data. Currently, mobile web applications have access to the following sensor data: geolocation (W3C, 2014), multimedia (video cameras, microphones, webcams) (W3C, 2015a), light (W3C, 2015b), and device motion and orientation (W3C, 2011).

W3C specifications discuss security and privacy issues for some mobile sensors such as GPS and light. For example, the working draft on ambient light events explicitly discusses security and privacy considerations as follows (W3C, 2015b): "The event defined in this specification is only fired in the top-level browsing context to avoid the privacy risk of sharing the information defined in this specification with contexts unfamiliar to the user. For example, a mobile device will only fire the event on the active tab, and not on the background tabs or within iframes". The geolocation API, on the other hand, requires explicit user permission to grant access to the web app due to security and privacy considerations.

Table 2 – Maximum in-app sampling frequencies on different mobile OSs.

Device/mOS	Accelerometer freq. (Hz)	Gyroscope freq. (Hz)
Nexus 5/Android 5.0.1	200	200
iPhone 5/iOS 8.2	100	100

On the other hand, security and privacy issues regarding motion and orientation sensor data have not been as readily evident to the W3C community and browser vendors as those of the sensors discussed above. Interestingly, in contrast to geolocation and ambient light sensors, there is no *security and privacy considerations* section in the W3C working draft on motion and orientation sensors (W3C, 2011). JavaScript code in a web page is given full access to motion and orientation sensor streams on mobile devices without needing to ask for user permission. This opens the door for attackers to compromise user security by listening to the motion and orientation sensor data as we present in this paper.

1.3. *Access to mobile sensors within app vs. browser*

The in-browser sensor data access that the W3C specification allows is heavily restricted in multiple ways. First, the access is restricted to only two types of streams: the *device orientation* which supplies the physical orientation of the device, and the *device motion* which represents the acceleration of the device. Motion data include sequences from accelerometer, accelerometer-including-gravity, and rotation rate (W3C, 2011). The orientation sensor, on the other hand, derives its data by processing the raw sensor data from the accelerometer and the geomagnetic field sensor.[1]

More importantly, access is also restricted to *low-rate* streams which provide data with slower frequencies as compared to those provided in-app. Here, we present two tables (Tables 2 and 3) on sampling frequencies on different platforms and popular browsers. The in-app frequency rates in Table 2 for Android are obtained from running an open source program

[1] http://developer.android.com/guide/topics/sensors/sensors_position.html#sensors-pos-orient.