The dangers and privacy issues of motion-based side-channel attacks

Milo Cesar
University of Twente
The Netherlands
m.cesar@student.utwente.nl

ABSTRACT

Motion data has long been available in smartphones, it is a staple of many games and it is still gaining traction through Augmented Reality web apps. The use of these motion sensors is, however, not without risk. This research examined if it is possible to extract motion data from web apps in a way that might violate users privacy. Different browsers were tested with regards to their security implementations for these sensors.

Keywords

Side-channel attack, Motion data, Sensors, Privacy, Browsers

1. INTRODUCTION

While an increasing number of sites is moving to two-factor authentication, passwords are still used as the most prevalent form of user authentication [19]. These passwords are frequently just simple, small and predictable pieces of text. This makes those passwords easy to crack. Originally passwords were mostly extracted through exploiting weaknesses in programs and websites that yield these passwords as part of larger data dumps. A less explored tactic is that of side-channel attacks. Such a side-channel attack is based on extracting data through indirect means, this might be emitted sound, power usage, magnetic fields, timing differentials or one of many other side-channels. The efficiency of such side-channel attacks has among others been shown by successfully extracting search history from Bing, and mail content from Gmail through timing differentials [14]. One of the main problems of side-channel attacks is their usability space; since new side-channels are found quite frequently it is hard for hardware and software manufacturers to defend against them.

This paper will consist of two main, albeit separate, parts. The first part will be about original research into abusing orientation and motion data to extract text from smartphones and the specific use cases of this side-channel attack, the second part will be about the privacy and security issues that arose from the implementation of these orientation and motion sensors in websites.

Motion data has in the past already been used to successfully extract PINs with a 70% accuracy[4] and to extract larger letter sequences which resulted in a median of 4.5 trials to extract a 6 character password[18]. This research tried to expand on these researches and especially the research of Bart Verkuil [20] on Extracting Passwords From Movement: Side-channel Attack On Smartphones Using MotionSensors. Research was done into the possibilities of extracting passwords or other input through motion sensor data. If it turns out that password extraction through these sensors is feasible, manufacturers should quickly limit the ability for websites and apps to use these sensors to protect their customers by eliminating a possible entry point for covert input detection and extraction.

While extracting unrestricted input from a smartphone might be a worst-case scenario, there are many other scenarios in which motion and/or orientation data can be used to breach a users privacy. This has already been shown in prior research such as [11] and [9], these researches give the impression that there is an increasing amount of websites that use motion and orientation data for advertising, tracking or identification purposes. Das et al. showed in 2016 [11] that at least 1% of the top 100.000 Alexa websites use these sensors for those purposes, this number increased to 3.6% in 2018 [9]. This trend yields questions such as "Who has influence over what processes can use motion and orientation sensors?" and "In what manner do websites utilize motion and orientation data".

The twofold in this paper, with the motion based input extraction on the one hand and the privacy concerns of said sensor on the other side, came to be due to the many roadblocks. These hindered the data collection necessary for motion-based text extraction to such an extent that there could not be a satisfying answer to the fist research question which in turn lead to the creation of the second research question. This paper aims to answer the proposed research questions while also documenting the progress made on the original research and to provide helpful insights into both parts of the research.

1.1 Roadblocks

A few roadblocks severely hindered the progress of the original research. First off the release of iOS 12.3 on May 6th, 2019 (during the 3rd week of this research) came with tremendous changes in the permission system in iOS Safari used to access accelerometer data. This lead to a delay of almost a week with regards to the schedule due to needing to find and implement a workaround for this new permission system. This required the users to switch a setting buried deep in the Settings app on their iPhone and subsequently to the need for an explanation for the users on how to find this switch. This also meant losing almost a week worth of work based on the old permission

31^{th} *Twente Student Conference on IT* July. 5^{nd}, 2019, Enschede, The Netherlands.

system. Almost exactly one month later, the same issues arose again with the beta release of iOS 13, Apple Inc. changed the permission system build into the iPhone once more which had the same consequences with regards to time constraints. This further limited the available time and testing possibilities. The last major roadblock was hit when the decision was made to switch to an Android-only testing group, it turned out that both the Chrome and Samsung internet browsers did no longer support the *DeviceMotionEvent*, they instead relied on the new *Gyroscope* API for developers to collect this data. At the time of research possible topics, the documentation did not yet reflect this change. This resulted in the need for another rewrite of the most crucial part of the experiment. In the meantime information began to surface with regards to Google Inc.'s intention to add a permission system to the sensor API. At this point, it did no longer seem reasonable to continue on investing more time into those tracks and therefore the switch was made to research into the trend of privacy implications of motion and orientation sensor access control systems thus leading to the twofold in this paper.

2. RESEARCH QUESTIONS

If we want to be able to correctly determine the impact of the expected problem, we need to be able to quantify the ability to extract text or other input based on motion sensor data. To do this, this research will try to answer the following research question.

RQ1 To what extent can motion sensor data combined from all motion sensors be used to extract text entered through the onscreen keyboard of a smartphone?

If it is possible to extract text, RQ1 will be answered by a percentage of correct first guesses and by an average number of guesses needed to come to the correct input.

The further research questions will focus on privacy problems as projected before.

RQ2 What access control systems are in place to protect users privacy with regards to motion and orientation data in modern devices?

RQ2 will be answered by examining popular browsers across popular operating systems.

3. RELATED WORK

Due too the aformentioned twofold nature of this research, there will be two parts related work. The first part will show the related work done with regards to extracting input through movement, the second part will focus on research done into the privacy considerations with regards to motion and orientation access.

3.1 Extracting input through movement

Back in 2004 research has already been done by Asonov and Agrawal[2] on extracting keyboard presses, they differentiated between the sounds of key-presses on a physical keyboard with which they could extract key-presses with 79% accuracy. One of the earliest pieces of research done into motion-based side-channel attacks was done by Marquardt et al. [15] in which they used the motion sensors of an iPhone 3GS to extract words typed on a physical keyboard by laying the iPhone on a desk next to the keyboard. When combining their input data with a dictionary they were able to extract words with an 80% accuracy. Individual letter recognition, however, was severely limited with a 25.89% vs 78.85% accuracy between Marquardts, and Asonov and Agrawal's researches.

One of the earliest studies on extracting data through motion sensors while typing on software keyboards has been done by Cai and Chen in 2011 [4] here an accuracy of 70% is reached on a numeric keyboard, the use of a numeric-only keyboard, however, increased the area per key and decreases the total number of keys with regards to a standard qwerty keyboard. Owusu et al. performed the earliest motion-based detection on a full-size software keyboard, they reached a key-press accuracy of 24.5%[18] this research was, however, limited to passwords of a fixed length of 6 characters. Last year Verkuil performed research on motion-based password extraction on a full-size software keyboard in which he managed to get a 70% accurate prediction for 25 input passwords of different lengths.

The aim of this research was to closely represent these studies, which means the research stats by asking subjects to input specific codes, passwords or other text during which the mobile phone will register the motion, this data is then used to train a machine learning algorithm. Said algorithm will be frozen after the training session. When the data is processed, participants can test the predictability of the algorithm. Keeping the same methodology as other researches makes it easier to compare the results between the different researches. This research might therefore most closely resemble the research done by Owusu et al.[18], this is a 7 years old research, in this time a lot of improvements have been made in the areas relevant to this research. This research deviates by the other researches with its intention to detect individual letters out of input. Furthermore, this research can utilize new techniques that might lead to improved results.

3.2 Privacy considerations with regards to motion data

Users have long been aware of their privacy and what might impact their expectation of privacy. Back in 2009 Cai et al. [5] did research into the privacy implications of adding sensors to the then relatively new smart-phones (the first smartphones were introduced around 2007). This research focused mainly on apps, Cai et al. tried to categorize apps by there legitimate use cases for sensor data. They propose the idea of a hardware switch to disable sensors while also explaining why this might not be the best idea. They furthermore propose a solution which closely resembles the well-known permission pop-ups currently in use.

In 2011 Beresford et al. tried to create a privacy-aware version of the Android operating system [3] in which they mock all sensor data. They created a layer between the sensors and apps, in this layer the user could decide on a per-app basis if said app should get the real data, fake "mock" data or no data at all. They proposed this as a solution to keep existing apps working [1] while increasing the users' privacy. This research has been expanded on by Cappos et al. in 2014 [6], this latter research gives more specifics on how to implement such a permission system with more fine-grained controls. Both pieces of research do not opt for permission pop-ups as are now widespread but they would allow all apps access to the sensors while limiting the amount of actual use-full data the app receives,

[1] All apps should already handle situations in which the sensors may be unavailable for other reasons i.e. no GPS signal to obtain a location.

they propose to do this by sending the aforementioned mock data or by introducing errors or rounding on actual data to reduce the accuracy.

Research has also been done into what the requirements from users are for permission systems [16]. They interviewed 22 participants and found what data they stored on their phones and how valuable and sensitive they thought this data to be. The participants said that their location and motion information was very sensitive to them although not too valuable, this was mainly due to the fact that location and motion data could lead to identifying once home and place of work while in the meantime providing access to the information when somebody left their home. This research further states that users had little trust in the security of their smart-phones and would rather not store sensitive information on these devices. This information is very relevant to the current research for users should trust that their data is safe and used as they expect, otherwise they might be anxious to allow apps with legitimate use cases access to their sensors which severely decreases the usability of smart-phones and undermines the uses of permission systems.

In 2016 Das did research into the possibilities of using acoustic and motion data to fingerprint smart-phones. He states that "our model provides a conservative estimate of at least 10% classification accuracy with 100 000 devices." which is using the motion and orientation sensors. His research further focuses on methods to mitigate such attacks, these mitigation techniques had already been published by Das et al. [10]. This research further emphasizes the risks associated with unrestricted access to motion and orientation data. One of Das' later works is also very relevant and gets even closer to the topics discussed in this paper. In his 2018 paper, Das et al. do research into the actual real-world usages of sensors. They research what websites use which sensors and they try to classify the usages [9], the results produced by this paper will be more in-depth discussed in the discussion section.

4. RESEARCH METHODS

As already discussed in the Introduction, there are two main parts to this research, they both deserve their own explanations as to what research has been performed.

4.1 Data collection for motion-based text extraction

The research questions as asked in this research are used to determine the likelihood of an app or website, that might have bad intentions or be hacked, to be able to extract input from other apps or websites through the proposed motion-based side-channel attack. An illustration of which is given in Figure 1. The app or website might start collecting data for legit use cases such as a social-media app. This app might then send the collected data to a backend where it can be processed and create a user-specific neural network, this does not need to use resources in the app and can thus be done outside the phone, without any context. This app or website might later be covertly monitoring the aforementioned sensors while in a background state, the collected data in this background state might then be processes once again by the earlier created neural network to extract the predicted input. Another issue with this attack scenario is that such an app or website might also have access to data such as the user's date of birth, data which is often used in human password creation. Such a neural network might, therefore, be very adequately trained and thus be able to predict passwords or other input with high accuracy.

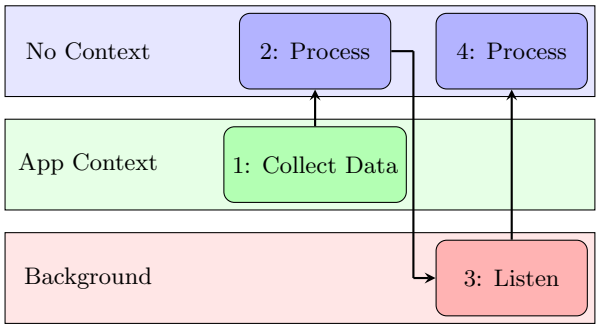

Figure 1. Context for each step of the attack

The data set as required for this research is not yet (publicly) available, therefore this data will need to be collected from participants. For this a website/web-app has been created in which the *DeviceMotion* and *DeviceOrientation* events where to be captured and send to a server. *DeviceMotion* would be used to collect information on the acceleration (m/s^2 in each direction) and rotation rate (degree per second) of the device, *DeviceOrientation* would be used to get the devices orientation (degrees). On the aforementioned server, the data should have been processed to create a participant specific fully connected neural network using the *TensorFlow.js* library. This neural network could later have been used by the same participant to test the predictive capabilities of the neural network.

The neural network was to be trained to extract a two-dimensional vector, which together with its certainty represents the tapped area as shown in Figure 2, from a nine-dimensional input vector. This input vector is created by capturing data on acceleration, rotation and rotation rate, each of these provides values on three axes, thus providing nine values per measurement. This tapped area can be mapped to an individual key or an array of possible keys after which this set of tapped keys can be used to deduct the typed text.

The web app has been created but has not been used to collect actual data due too the technical issues outlined in the introduction. This lead to no collected data and thus the inability to create the neural networks.

Figure 2. Showing the tapped area after a user typed a letter 'p'. The neural network yields a high certainty of letter 'p'

4.2 Motion and orientation access through browsers

Most information in this regard has been collected through literature research as already discussed in the Related Work

section of this paper. Data has mainly been collected from [3], [5], [6], [8], [9], [10], [12] and [16]. Previous research done by Das et al. [9] showed that some browsers allowed insecure access to motion and orientation access as well as cross-origin access. The current research tries to replicate these results with current generation browsers on modern OS'es. This research further wants to collect information on what type access is necessary to get the motion and orientation data. To get this data a webapp was setup, this webapp performed multiple checks it first fetched information about the device, thereafter check if the device populates the *DeviceMotion* even, following that it check if the device can handle permissions and lastly the webapp checks if it does actually receive motion data.

```
alert('Useragent' + (window.navigator || {}).userAgent)
if(window.DeviceMotion) {
  if(window.DeviceMotion.requestPermission){
    alert('Device supports permission pop-ups');
  } else {
    window.addEventListener('devicemotion', () => {
      alert('Device sends motion data');
    });
  }
} else {
    alert('Event not defined');
}
```

Listing 1. Simplified version of code used to detect necessary access

The code from Listing 1 was loaded 3 times for each browser. It ran once in the main context, once in an iframe on the same origin and lastly in an iframe cross-origin. This test ran on devices from the researcher, his friends and his relatives, all tests had been run by the researcher himself. The tests were run on the most up-to-date versions of the browsers for their respective platforms except for the cases which are later on clearly marked with beta markings, these where the most up-to-date pre-release versions of these browsers.

5. RESULTS

The earlier mentioned roadblocks led to no data collection with regards to the motion and orientation sensors. These results will, therefore, focus solely on the research done on access control for motion and orientation access in major browsers and the corresponding literature research into access control and its implications

5.1 Access control

The methodology as discussed in Section 4.2 was used to collect information on the access control for motion and orientation access in 7 browsers across 5 devices running/emulating 8 different OS versions. Table 1 shows if access is granted to motion and orientation data and if this is allowed over an insecure (unencrypted) connection. The triplets show the status for (1,2,3) 1: Access in the origin window, 2: Access through an iframe on the same origin and 3: Access through an iframe cross-origin. Please note that the last 4 test devices do not possess a physical on-device gyroscope and do not expose a gyroscope to the JavaScript, their behavior here is still relevant as to provide information on the behavior of the browser towards permissions.

Table 1 shows interesting discrepancies between versions, Apple Inc.'s switch from Allow to Deny to Pop-up is very evident in this table. The major Android browsers behave as expected an intended, they evidently do not follow the W3Cs's recommendations [22] which advices implementing a permission system for "sensitive sensor data". The four final devices have other interesting results, primarily due to the fact that none of these devices actually have a Gyroscope or Accelerometer on-board. Safari handles this by not defining the *DeviceMotion* property, the other browsers do provide this property. Firefox and Edge outright refuse to give any data on Windows devices if no hardware sensors are present, on the same device Google Chrome was happy to provide the webapp with data, all sensors, however, gave "0" on all their axis, this does require the user's permission in the latest beta build. Chrome on Ubuntu shows the last interesting discrepancy, for some unclear reason, it falsely considered all iframes to be cross-origin and thus preventing motion and orientation data transmission. Lastly, it becomes clear that no browser allows sensor access over an unsecured connection. As a further decrease in privacy violation background access to data readings has also been disable in all of the tested brosers. This severy helps decrease the dangers sketched in the previous section. It is now impossible for a rogue webapp to collect data while the user is typing/-tapping in a different browser tab. This means that step 3 and by extension step 4 from Figure 1 can no longer be execute and this thus closes the proposed attack vector. Please note that it is not relevant to discuss if the webapp could theoretically use motion to extract input from its own windows, it could do this regardless by capturing the data directly instead of using this convoluted motion based methodology.

Table 1. Access to orientation and motion data in major browsers[3]. ✓ Data is granted, ❑ A popup requesting permission is shown, ✗ Data is- and will not granted

OS	Browser	Access	HTTP	BG
iOS 12.1	Safari	(✓, ✓, ✗)	✗	✗
iOS 12.4	Safari	(✗, ✗, ✗)	✗	✗
iOS 13 (Beta)	Safari	(❑, ❑, ✗)	✗	✗
Android 8.0.0	Chrome	(✓, ✓, ✗)	✗	✗
	Firefox	(✓, ✓, ✗)	✗	✗
	Firefox Focus	(✓, ✓, ✗)	✗	✗
	Opera	(✓, ✓, ✗)	✗	✗
macOS 10.14	Safari	(✗, ✗, ✗)	✗	✗
macOS 10.15 (Beta)	Safari	(✗, ✗, ✗)	✗	✗
Windows 10	Chrome	(✓, ✓, ✗)	✗	✗
	Chrome Canary 75 (Beta)	(❑, ❑, ✗)	✗	✗
	Firefox	(✗, ✗, ✗)	✗	✗
	Edge	(✗, ✗, ✗)	✗	✗
Ubuntu 19.04	Chrome	(✓, ✗, ✗)	✗	✗
	Firefox	(✓, ✓, ✗)	✗	✗

5.2 Realworld sensor usage

As has already been shown, it was trivial for websites to actually track this data this becomes even more evident from

[3]Browsers in this list are estimated to have sent over 85% of the internet traffic in May 2019[1]

the fact that almost 3700 from the 100.000 top Alexa websites[4] actually tracked some sensor data of their visitors [9]. Almost 2700 sites tracked the accelerometer, well over 2000 tracked the gyroscope[5]. At least 500 of these websites actually send this data to data-collection agencies. This is an increase from 2016 where Das et al. performed another research, also utilizing the top 100.000 Alexa websites, in which he shows that "over 1%" of these websites utilize motion tracking [11]. There are many reasons why websites would use this sensor data, some more legitimate than others, 789 of the sites researched by Das et al. have a legitimate use-case. However, 1198 where used exclusively for user detection [9]. The data as split into their separate categories can be found in Table 2, this data is an aggregation of the data collected by Das et al. [9].

Table 2. Usage of sensor data

# of Sites	Description
4	Use sensor data to add entropy to random numbers
114	Checks what HTML5 features are offered
413	Differentiating bots from real devices
35	Parallax Engine that reacts to orientation sensors
103	Automatically resize contents in page or iframe
533	Reacting to orientation, tilt, shake
1198	Tracking, analytics, fingerprinting and audience recognition

5.3 Privacy

The usage of these sensors and their data is very diverse, as already outlined above. The "problematic" usages are the once that invade the privacy of their users, earlier categorized as "Tracking, analytics, fingerprinting and audience recognition". These fingerprinting techniques are very effective as can be seen by [12] and [13], not only are they very effective, this effectiveness also makes them very lucrative for this industry [17].

5.3.1 Fingerprinting

Due to too minute differences created during the manufacturing process of gyroscopes and accelerometers, there are differences in how each of these sensors handles specific movements. These differences are very small and do not impact the usability of these sensors, they are however very use-full in fingerprinting smart-phones. The other well-known fingerprinting techniques such as detecting video and audio formats that the browser can playback or the fonts that the browser can render are easily spoofed by the browser. However, physically changing a gyroscope or accelerometer is, of course, a totally different story. This makes it easy to track smart-phones over longer periods of time, limited mostly by people switch to another smart-phone[17]. The effectiveness of fingerprinting gyroscopes and accelerometers has been demonstrated by Das et al. [12], in a study on 400 smart-phones they reached a 90% accuracy when increasing this to 100.000 devices "at least 12-16% accuracy can be realized". In the same study, Das et al. researched countermeasures by reducing the accuracy of the sensor data. Their research did not show that reducing the accuracy impacts other performance, they, however, do state that this cannot lead to the conclusion that reducing accuracy does not impact performance.

5.4 Improvements

Since January of 2018 steady improvements have been made. In 2018, all major browsers[6] allowed insecure access to the motion and orientation sensors as well as access through iframes [9], even though this was discouraged by the W3C as far back as 2015 [21]. The same major browsers no longer allowed access through insecure context or iframes at the time of this research as can be seen in Table 1. Safari has implemented a new permission system for the orientation and motion sensors as of iOS 13 [7] as did Chrome in version Canary 75, this extends the default permission system already in place in these major browsers and gives a pop-up prompting the user to give or decline permission. This permission is on a website and session bases i.e. every website has to request this permission individually and when a new session starts the user has to give permission again. It is important to note that Mozilla is made aware of the privacy implications of sensor access without permission as becomes evident from their bug tracker [7], they, however, did not yet provide an update to their Firefox browser to resolve this issue as of June 30, 2019.t

6. DISCUSSION

As discussed, the information gained from both literature and experimental research done into the access control for motion and orientation sensors was very enlightening. The experiments performed into cross browser and cross OS implementations of these access control systems show interesting differences. They give a good overview on the progress made in the last few years and they sketch an image on how fast the industry as a whole might change in some respect when the stakes are as high as they are with these privacy issues.

7. CONCLUSION

The focus of this research was twofold, this is also portrayed by the research questions. As became evident, it has been very hard to collect data for motion-based text extraction. This results in an inconclusive answer to the first research question. A view has been sketched about previous work into this topic, from this point the reader is free to interpret those results and come to a conclusion for themselves. As became clear in section 4. This question was asked to estimate the dangers that a rogue website could pose, this is a question that *can* be answered with the outcome of this research. With the proposed upgrade path of browsers in mind, we can come to the conclusion that it is very hard if not impossible for a rogue website or app to collect the motion and orientation data. This logically leads us to conclude that, given the lack of data, it is impossible to utilize this non-existing data to create a neural network that could identify inputs made on the screen of a smartphone.

The second research question does have a concrete answer; all major browsers are currently working on implementing permission systems to restrict access to motion and orientation data. Furthermore, browsers are protecting users by disallowing access through insecure context and by disabling background access to these sensors. This all leads

[4]Top 100K websites as of May 12th 2018.

[5]The other tracked sensors were amongst others the GPS and ambient light sensors

[6]Google Chrome, Safari and Firefox

[7]iOS 13 is as of 03/06/19 in Developer Beta

to increased privacy for the users while still allowing access for legitimate use cases.

Given the fact that nearly all software has flaws, it would still be very interesting to get an actual qualitative answer to RQ1. While the threat posed in this research has been proven to have very little chance of actually gaining traction; it is interesting to know the possible extent of the damage done if a bug in browsers could lead to motion and orientation data extraction. Lastly, given the enormous amount of users for each of the browsers discussed in this paper, independent researchers should periodically audit these browsers to check their security, this can lead to new side-channel attacks being found by researchers who would responsibly disclose these issues instead of them being abused.

8. REFERENCES

[1] statcounter GlobalStats, June 2019.

[2] D. Asonov and R. Agrawal. Keyboard acoustic emanations. In *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*, pages 3–11, May 2004.

[3] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan. MockDroid: trading privacy for application functionality on smartphones. In *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications - HotMobile '11*, page 49, Phoenix, Arizona, 2011. ACM Press.

[4] L. Cai and H. Chen. TouchLogger: Inferring Keystrokes on Touch Screen from Smartphone Motion. In *HotSec*, 2011.

[5] L. Cai, S. Machiraju, and H. Chen. Defending against sensor-sniffing attacks on mobile phones. In *Proceedings of the 1st ACM workshop on Networking, systems, and applications for mobile handhelds - MobiHeld '09*, page 31, Barcelona, Spain, 2009. ACM Press.

[6] J. Cappos, L. Wang, R. Weiss, Y. Yang, and Y. Zhuang. BlurSense: Dynamic fine-grained access control for smartphone privacy. In *2014 IEEE Sensors Applications Symposium (SAS)*, pages 329–332, Queenstown, New Zealand, Feb. 2014. IEEE.

[7] M. Corporation. Implement requestPermission() for DeviceOrientationEvent and DeviceMotionEvent, 2019.

[8] A. Das. *UNDERSTANDING AND MITIGATING THE PRIVACY RISKS OF SMARTPHONE SENSOR FINGERPRINTING*. PhD thesis, 2016.

[9] A. Das, G. Acar, N. Borisov, and A. Pradeep. The Web's Sixth Sense: A Study of Scripts Accessing Smartphone Sensors. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security - CCS '18*, pages 1515–1532, Toronto, Canada, 2018. ACM Press.

[10] A. Das, N. Borisov, and M. Caesar. Exploring Ways To Mitigate Sensor-Based Smartphone Fingerprinting. *arXiv:1503.01874 [cs]*, Mar. 2015. arXiv: 1503.01874.

[11] A. Das, N. Borisov, and M. Caesar. Tracking Mobile Web Users Through Motion Sensors: Attacks and Defenses. In *Proceedings 2016 Network and Distributed System Security Symposium*, San Diego, CA, 2016. Internet Society.

[12] A. Das, N. Borisov, E. Chou, and M. H. Mughees. Smartphone Fingerprinting Via Motion Sensors: Analyzing Feasibility at Large-Scale and Studying Real Usage Patterns. *arXiv:1605.08763 [cs]*, May 2016. arXiv: 1605.08763.

[13] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi. AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable. In *Proceedings 2014 Network and Distributed System Security Symposium*, San Diego, CA, 2014. Internet Society.

[14] N. Gelernter and A. Herzberg. Cross-Site Search Attacks. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15*, pages 1394–1405, Denver, Colorado, USA, 2015. ACM Press.

[15] P. Marquardt, A. Verma, H. Carter, and P. Traynor. (sp)iPhone: decoding vibrations from nearby keyboards using mobile phone accelerometers. In *Proceedings of the 18th ACM conference on Computer and communications security - CCS '11*, page 551, Chicago, Illinois, USA, 2011. ACM Press.

[16] I. Muslukhov, Y. Boshmaf, C. Kuo, J. Lester, and K. Beznosov. Understanding Users' Requirements for Data Protection in Smartphones. In *2012 IEEE 28th International Conference on Data Engineering Workshops*, pages 228–235, Apr. 2012.

[17] Å. Olejnik, C. Castelluccia, and A. Janc. Why Johnny Can't Browse in Peace: On the Uniqueness of Web Browsing History Patterns. page 16, 2012.

[18] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang. ACCessory: password inference using accelerometers on smartphones. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications - HotMobile '12*, page 1, San Diego, California, 2012. ACM Press.

[19] T. Petsas, G. Tsirantonakis, E. Athanasopoulos, and S. Ioannidis. Two-factor authentication: is the world ready?: quantifying 2fa adoption. In *Proceedings of the Eighth European Workshop on System Security - EuroSec '15*, pages 1–7, Bordeaux, France, 2015. ACM Press.

[20] B. Verkuil. Extracting Passwords From Movement: Side-channel Attack On Smartphones Using Motion Sensors. 2018.

[21] W3C. Generic Sensor API W3c First Public Working Draft, 15 October 2015, Oct. 2015.

[22] W3C. Generic Sensor API W3c Working Draft, 7 March 2019, Mar. 2019.