*Manuscript

Sensor Defense In-Software (SDI): Practical Software Based Detection of Spoofing Attacks on Position Sensors

Kevin Sam Tharayil[a,*], Benyamin Farshteindiker[a], Shaked Eyal[a], Nir Hasidim[a], Roy Hershkovitz[a], Shani Houri[a], Ilia Yoffe (Iofedov)[a], Michal Oren[b], Yossi Oren[a]

[a] *Department of Software and Information Systems Engineering, Ben Gurion University, Israel*
[b] *Tomer Ltd., Israel*

Abstract

Position sensors, such as the gyroscope, the magnetometer and the accelerometer, are found in a staggering variety of devices, from smartphones and UAVs to autonomous robots. Several works have shown how adversaries can mount spoofing attacks to remotely corrupt or even completely control the outputs of these sensors. With more and more critical applications relying on sensor readings to make important decisions, defending sensors from these attacks is of prime importance.

In this work we present practical software based defenses against attacks on two common types of position sensors, specifically the gyroscope and the magnetometer. We first characterize the sensitivity of these sensors to acoustic and magnetic adversaries. Next, we present two software-only defenses: a machine learning-based single sensor defense, and a sensor fusion defense which makes use of the mathematical relationship between the two sensors. We performed a detailed theoretical analysis of our defenses, and implemented them on a variety of smartphones, as well as on a resource-constrained IoT sensor node. Our defenses do not require any hardware or OS-level modifications, making it possible to use them with existing hardware. Moreover, they provide a high detection accuracy, a short detection time and a reasonable power consumption.

Keywords: sensor spoofing, sensor fusion, machine learning

1. Introduction

Many electronic devices, such as smartphones and sensor nodes, are equipped with position sensors. These sensors are capable of measuring the position, orientation and motion of the device in three-dimensional space. We rely on these sensors for increasingly sensitive tasks including authentication [1, 2], navigation [3], and health monitoring [4]. This paper focuses on two widely used sensors: the gyroscope, which measures a device's angular momentum, or rate of rotation, and the magnetometer, which measures a device's orientation with respect to the magnetic field of the Earth.

Several recent works have shown how the readings of these sensors can be spoofed by applying an external acoustic stimulus to the device or its surroundings [5, 6]. The spoofed output of a sensor does not reflect the device's actual rotation or orientation; instead, the output is overwritten by artificial values which are either randomly corrupted or completely controlled by the attacker. Sensor spoofing attacks on smartphones are already being used for malicious purposes. For example, the online publication *Sixth Tone* reported on June 2018 that Chinese university students, who are required to reach at least 10,000 steps per day as part of their fitness requirement, use a variety of devices called "WeRun Boosters" to spoof the motion sensors on their smartphones, generating 6,000 to 7,000 steps on a smartphone per hour [7]. The risks associated with sensor spoofing will only grow as the amount of sensitive applications relying on these sensors increases. For example, Wang et al. [8] and Reinertsen et al. [9] proposed to use sensor measurements to assess the severity of illness of patients with schizophrenia. Sensor spoofing attacks, when applied to this scenario, may erroneously cause a person to be hospitalized in a psychiatric ward.

While several papers have discussed sensor spoofing, few of them have discussed the prevention of these attacks, a gap we wish to address in this work. One of the main lim-

*Corresponding author

Email addresses: tharayil@post.bgu.ac.il (Kevin Sam Tharayil), farshben@post.bgu.ac.il (Benyamin Farshteindiker), shakedey@post.bgu.ac.il (Shaked Eyal), hasidimn@post.bgu.ac.il (Nir Hasidim), royhersh@post.bgu.ac.il (Roy Hershkovitz), hourish@post.bgu.ac.il (Shani Houri), iofedov@post.bgu.ac.il (Ilia Yoffe (Iofedov)), michaloren78@gmail.com (Michal Oren), yos@bgu.ac.il (Yossi Oren)

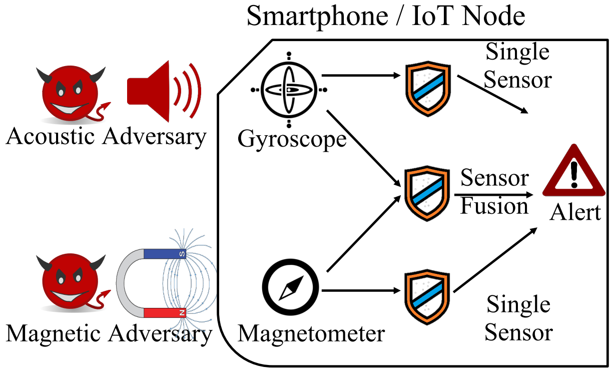

Figure 1: Overall Description of our Defenses

itations of many defenses against sensor spoofing is that they either require changes to the sensor hardware or to the low-level firmware used to interface it to the phone's CPU. Since position sensors are typically highly integrated low cost devices with a relatively long development cycles, such modifications are difficult to apply to hardware already deployed in the field, and are hard to justify from a system integration standpoint. While software-based anomaly detection mechanisms have been proposed for other types of sensor systems, such as wireless sensor networks [10], they typically did not consider a malicious adversary but only a random fault model.

Our Contribution: In this paper we propose two software-based defense methods against acoustic and magnetic attacks on a device's gyroscope and magnetometer. Our first defense method, SDI-1, uses machine learning to detect anomalies in the output of a single sensor. This defense method can detect sensor corruption attacks, but cannot detect cases where a more powerful adversary can force the sensor to output a spoofed but valid reading. Our second defense method, SDI-2, applies sensor fusion to compare the readings of multiple sensors measuring a similar type of motion. This method can potentially protect against a more powerful sensor spoofing adversary, as long as this adversary cannot control the entire set of sensors available on the device. Specifically, in this paper we present single-sensor defenses for acoustic attacks on the gyroscope and for magnetic attacks on the magnetometer. We also present a sensor fusion based defense combining the gyroscope and the magnetometer, as shown in Figure 1. We describe the physical and mathematical relationship between expected sensor readings, and show how the defender can measure deviations between the two sensors to detect an attack. We implemented our defenses on multiple smartphones from different vendors, as well as on a resource-constrained IoT node, in each case measuring the accuracy, detection time and power usage of our defenses. The main advantage of these defenses are that they are purely software based, and can therefore be deployed on many types of devices without any hardware modification.

Document Structure: We begin by describing the spoofing attacks on the MEMS gyroscope and magnetometer. In Section 2 we describe SDI-1, a machine learning-based single sensor defense, and SDI-2, a sensor fusion-based single sensor defense, and show how they can protect against acoustic and magnetic attacks on the gyroscope and on the magnetometer respectively. In Section 3 we perform a practical evaluation of our defense methods. Finally, in Section 4 we discuss defenses for another type of sensor, the accelerometer, and conclude by discussing further applications of sensor fusion and its improvements.

1.1. Types of Position Sensors

A smartphone's various position sensors are used to measure the phone's position and motion in space along the six axes of motion (or six degrees of freedom). The measurements of the device's sensors are generally provided in the device's frame of reference: a Cartesian coordinate system with coordinates attached to the device. This coordinate system is rotated with respect to the world's frame of reference, which is a standard static coordinate system. Of the six degrees of freedom, three coordinates (X, Y, and Z) are used to describe the phone's position and linear motion in space, while the three other coordinates (ρ,ϕ and θ, or pitch, roll and yaw) are used to describe the phone's Cartesian axes orientation with respect to the world's frame of reference and its rotational motion.

The **gyroscope** is a MEMS-based sensor which measures the device's angular velocity in units of radian per second. As described in [11], microelectromechanical systems (MEMS) gyroscopes typically contain a small mass moving back and forth at a constant frequency. As the phone is rotated, the Coriolis effect acts on this moving mass and causes it to vibrate with an amplitude that is directly related to the angular rotation rate. The modulated vibration amplitude is then converted to voltage, typically by a capacitive or piezo-electric sensor.

The **magnetometer**, or compass, measures the direction and magnitude of the ambient magnetic field around the device, in units of microtesla. As described in [12], virtually all smartphones use a Hall effect magnetometer, which works by detecting the voltage differential induced by the Hall effect across a thin metallic surface in response to a magnetic field perpendicular to the surface. The magnetic field measured by the phone field is typically a combination of the Earth's magnetic field, which points more or less to the north, and additional magnetic sources in the vicinity of the phone, such as iron beams, electric motors or induction coils. As long as the phone stays in the same place and the additional magnetic sources stay constant over time, the magnetometer's reading will point to the same direction in the world's reference frame, even when the phone is rotated. Other common position sensors include the accelerometer, which measures the linear acceleration of the device, and the GPS sensor, which measures the location of the device on Earth.

1.2. Spoofing Attacks on Position Sensors

As mentioned in the previous section, MEMS gyroscopes contain a small moving mass. As shown in [6] and [11], they are vulnerable to acoustic attacks, in which the sensor is subjected to external vibrations with the sensor's mechanical resonant frequency. When the moving mass inside the sensor is stimulated by this acoustic signal, it begins vibrating with a high amplitude. This prevents the sensor from interacting with the environment, allowing its reading to be controlled by the attacker. In other words, a high-frequency audio signal at a specific frequency can bring these sensors into a state of resonance, corrupting their outputs. The source of the disruptive signal can be an external device situated next to the phone, or even the phone's own speaker [13].

Acoustic attacks on MEMS-based gyroscopes and accelerometers were first presented by Son et al. in [11] in the context of drones, and later shown by [5, 14] to be applicable to smartphone sensors as well. Tu et al. in [6] performed a comprehensive evaluation of out-of-band signal injection methods to deliver adversarial control of embedded MEMS inertial sensors on a wide variety of devices including self balancing scooters, stabilizers, smartphones, VR headsets and other similar devices. Similarly, an adversary equipped with a magnetic coil is able to spoof the outputs of the magnetometer, an effect put to productive use in [12]. Recognizing the increasing risk caused by current and emerging sensor spoofing attacks, the Industrial Control Systems Cyber Emergency Response Team of the U.S. Department of Homeland Security (ICS-CERT) stated recently that it considers position sensor attacks as a "threat to critical infrastructure" [15].

Generally speaking, there are two types of spoofing attacks: corruption attacks, which we refer to as sensor rocking attacks (following the nomenclature of [11]) and rewriting attacks, which we refer to as sensor rolling attacks (for reasons of symmetry). Sensor rocking attacks replace the sensor readings with arbitrary corrupted values which are unrelated to the external environment. For example, the attacker can replace the sensor signal with a high frequency sine wave or random noise. While the attacker cannot control the shape of this corrupted signal, the attacker can turn the disruptive signal on and off at will. In fact, [14] and [12] used this ability as a data transmission mechanism. Sensor rolling attacks are a stronger class of attack, in which the attacker completely replaces the sensor readings with values of their choosing. Since the attacker can create any sensor readings including replaying previous readings, defense methods that detect anomalies will not be effective against rolling attacks.

In this work, we replicate two types of acoustic attacks on the gyroscope, as shown in [14] and [6], to collect data and test our defense methods. While [14] used a piezoelectric speaker kept in close proximity to the phone, [6] used regular speakers connected to an amplifier to attack the gyroscope from a distance. Both attacks work by using the sensor's mechanical resonant frequency. To spoof the magnetometer, we used a solenoid connected to a waveform generator as magnetic field source, similar to the methods of [16]. The high sensitivity of the magnetometer makes it extremely vulnerable to the presence of any external magnetic field, sometimes even to the magnet in the phone's own speaker [17].

2. Defense Methods

In this work we implement and evaluate two purely software-based approaches for sensor spoofing detection. The first approach, SDI-1, uses machine learning techniques applied to sensor output to detect anomalies. The second approach, SDI-2, is a novel fusion-based detector which works by examining multiple sensor outputs. Since these defenses apply signal processing and machine learning, it is important to examine the resource consumption of the defense methods, both in terms of processing time and of power consumption. It is also important to determine the response time of the countermeasures. If the countermeasure has a very slow response time, it may be possible for an attacker to evade detection by spoofing the outputs for just a very short amount of time. To demonstrate the generic nature of our defenses across all kinds of devices, we perform the attacks and test our defenses on various smartphones, as well as an IoT node, as listed in Table 1, representing a wide variety of electronic devices with different constraints in terms of CPU capabilities, memory and power consumption.

2.1. SDI-1: Machine Learning-Based Single Sensor Defense

The key idea behind SDI-1 is to train a machine learning model that can detect an anomaly (an attack) on the sensor output. To enable this defense, the defender generates many traces of benign sensor outputs and ideally traces of known attacks as well. Detection can either be performed by a two-sided classifier, which is trained both on benign and spoofed traces, or by a one-sided classifier, which is only trained on benign traces; detection takes place when a new trace deviates significantly from the benign traces. The advantage of the single sensor approach is that it requires no additional inputs other than the sensor readings themselves. Thus, it can be implemented inside the sensor hardware (or inside its manufacturer-provided driver) and does not require any high-level changes to the system. A possible short-coming of this defense is that the two-sided classifier must be trained on previously encountered and known attack traces. Any new spoofing method which results in different attacker characteristics will not be detected. This can be overcome by using a one-sided classifier which only needs to be trained on benign traces. In this case, any new trace which is significantly different from a benign trace will be identified as an attack. The

Device	Gyroscope	Magnetometer
Samsung Galaxy S5	InvenSense MPU-6500	AKM AK09911c
Samsung Galaxy S6	InvenSense MPU-6500	Yamaha YAS532
LG Nexus 5X	Bosch Sensortec BMI160	Bosch Sensortec BMM150
iPhone SE	InvenSense EMS-A	Alps Electric HSCDTD007
STM32L4 IoT Node	STMicroelectronics LSM6DSL	STMicroelectronics LIS3MDL

Table 1: Gyroscope and magnetometer sensors used in various test devices

disadvantage of the SDI-1 approach is that it only works for sensor rocking (corruption) attacks, and not for sensor rolling (overwriting) attacks; indeed, if an adversary can choose arbitrary values for the sensor, the attacker can simply replay values recorded by the sensor in the past which cannot be identified as anomalies.

Training a classifier directly on high-dimensional data, such as sensor readings over time, is inefficient and can cause over-fitting. Thus, before the learning algorithm operates on the traces, each trace must be reduced into a small set of succinct features. In [18, 19] the authors suggested a selection of features that are relevant for positional sensor readings, and we use this set in our work as well.

When designing our detector, we aimed to create a detector which is both effective and explainable. Non-explainable classifiers, such as ensemble-based methods or those based on deep learning, are less appropriate in a fraud detection setting, since they do not clearly indicate the reason for the detector's particular output. We were interested in selecting a classifier that has a simple internal structure and is therefore less sensitive to adversarial learning scenarios, where the attacker has some access to the training set. We looked for classifiers which had high accuracy and are less resource intensive, so that our defense method can be applied on a wide range of devices.

The single sensor defense can be implemented for all position sensors. In this work, we focus on defenses against acoustic attacks targeting the gyroscope and magnetic attacks targeting the magnetometer. We briefly discuss defenses against attacks targeting the accelerometer in Subsection 4.2.

2.2. SDI-2: Fusion-Based Multiple Sensor Defense

The key insight behind the second defensive approach is that the defender has an information advantage over the attacker, whereby instead of being limited to a single sensor, the defender can compare the current readings of multiple different sensors measuring the same physical phenomenon. If the sensors do not agree with each other, it can indicate that an attack is in progress. The advantage of this approach is that it works for both rocking and rolling attacks (i.e., even a completely valid sensor trace replayed by the attacker will be detected if other sensors on the system do not agree with it). Furthermore, this method is generic and future-proof in the sense that it does not depend on the characteristics of a specific attack method, but rather on the immunity of the gyroscope to magnetic attacks and, correspondingly, on the immunity of the magnetometer to acoustic attacks. To carry out fusion-based defense in practice, we first derive the mathematical relationships between the readings of different sensors, in this case the gyroscope and the magnetometer. To this end, we apply some basic Newtonian physics principles, as described below. Once the mathematical relationships are identified, it is possible to use the waveform output of one sensor to approximate the other sensor, or to use both sensors to calculate the same intermediate waveform. Then, we can measure the extent to which the two sensor readings agree, by applying some sort of distance measure between the two waveforms.

Sensor fusion has its own advantages and disadvantages as a countermeasure, as compared to single sensor detectors. Its main disadvantage is that it has to accommodate at least twice the amount of measurement noise, since it depends on multiple physical sensors. To highlight the difference between the methods, we first evaluate a threshold-based sensor fusion detector based on a simple distance measure, namely the mean squared error (MSE). We then show how this detector can be improved by combining both sensor fusion and machine learning methods.

The cornerstone of our fusion-based countermeasure is an equation relating the readings of two different position sensors. The device's sensor measurements are presented in a Cartesian coordinate system (X_d, Y_d, Z_d). This is the coordinate system (reference frame) attached to the device. This coordinate system can be rotated with respect to a fixed, Cartesian, or world coordinate system, (X, Y, Z) , in which the axes follow the North-East-Down (NED) convention: $X = north$, $Y = east$ and $Z = down$. The world frame is assumed to be inertial, ignoring the rotational motion of the Earth. Note that the origins of the two reference frames stay attached; translational degrees of freedom are not accounted for. At some time instance t, the altitude of the device frame with respect to the world frame is represented by a set of time dependent Tait-Bryan angles (ϕ, θ, ψ) . These are Euler angles where the sequence of rotations is x-y-z, known also as roll, pitch and yaw. The transformation from the inertial frame to the device frame is the rotation:

$$R(\phi, \theta, \psi) = R(\phi)R(\theta)R(\psi) \quad (1)$$

where:

$$R(\psi)=\begin{bmatrix}\cos(\psi) & \sin(\psi) & 0\\ -\sin(\psi) & \cos(\psi) & 0\\ 0 & 0 & 1\end{bmatrix} \tag{2}$$

is a rotation around the initial Z axis,

$$R(\theta)=\begin{bmatrix}\cos(\theta) & 0 & -\sin(\theta)\\ 0 & 1 & 0\\ \sin(\theta) & 0 & \cos(\theta)\end{bmatrix} \tag{3}$$

is a rotation around the intermediate Y axis, and

$$R(\phi)=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\phi) & \sin(\phi)\\ 0 & -\sin(\phi) & \cos(\phi)\end{bmatrix} \tag{4}$$

is a rotation around the final X axis. A general vector is represented in the rotated reference frame by:

$$\vec{G}_d = R(\phi,\theta,\psi)\vec{G}_w \tag{5}$$

To link the readings of the gyroscope and the magnetometer, we need to express the angular velocity $\vec{\omega}$ in terms of the rotation angles (ψ,θ,ϕ). The angular velocity components along the axes $\hat{\psi},\hat{\theta},\hat{\phi}$ perpendicular to the rotations are given by:

$$\omega_\psi=\dot{\psi},\ \omega_\theta=\dot{\theta},\ \omega_\phi=\phi \tag{6}$$

The directions of these components of $\vec{\omega}$ cannot constitute an orthogonal coordinate system; each rotation is made in a different reference frame. The transformation matrices $R(\psi),R(\theta),R(\phi)$ can be used to project the angular velocity components on the Cartesian coordinate system axes of the device frame [20]. We note that the first transformation is done by rotating around the original z axis. Therefore, $\hat{\omega}_\psi$ is directed along the original (world frame) z axis. In order to obtain the components of $\vec{\omega}$ in the device frame we should use the full rotation $R(\phi,\theta,\psi)$. The next rotation axis $\hat{\omega}_\theta$ coincides with the intermediate y axis and therefore should be transformed by $R(\phi)$. The third axis of rotation $\hat{\omega}_\theta$ coincides with the final x axis and therefore does not undergo a transformation. For each Cartesian component of $\vec{\omega}$ we can sum up the contributions of the projections. As a result of this procedure, the angular velocity in Cartesian coordinates of the device frame is given by:

$$\begin{aligned}\omega_{xd} &=\dot{\phi}-\dot{\psi}\sin(\theta)\\ \omega_{yd} &=\dot{\theta}\cos(\phi)+\dot{\psi}\cos(\theta)\sin(\phi)\\ \omega_{zd} &=-\dot{\theta}\sin(\phi)+\dot{\psi}\cos(\theta)\cos(\phi)\end{aligned} \tag{7}$$

We also note that the angular velocity transforms, as any other vector would do, from the world frame to the device frame:

$$\vec{\omega}_d = R(\phi,\theta,\psi)\,\vec{\omega}_w \tag{8}$$

In the remainder of this paper we exploit the fact that one can associate the angles of rotation (ϕ,θ,ψ) with the angular velocity of the device, together with the equation 5, in order to relate the angular velocity in the device frame (measurements of the gyroscope) to the rate of change in the magnetic field as measured by the rotating device (measurements of the magnetometer).

Magnetic Field Time Derivative in the Device Frame. Consider an arbitrary magnetic field, constant and uniform in the world frame:

$$\vec{B}_w=(B_x,B_y,B_z)$$

thus:

$$\frac{d\vec{B}_w}{dt}=0$$

In order to obtain the magnetic field in the reference frame of the device, the rotation matrix 1 is used:

$$\vec{B}_d(t)=R(t)\vec{B}_w \tag{9}$$

We take the derivative of 9 with respect to time to obtain:

$$\frac{d\vec{B}_d}{dt}=\frac{d\left(R(t)\vec{B}_w\right)}{dt}=\frac{d\left(R(t)\right)}{dt}\vec{B}_w$$

We use the fact that the rotation matrix is orthogonal, $R^{-1}=R^T$, and therefore $RR^T=R^TR=1$, and multiply the right-hand side by $R^T(t)R(t)=1$:

$$\frac{d\vec{B}_d}{dt}=\frac{d\left(R(t)\right)}{dt}R^T(t)R(t)\vec{B}_w$$

or:

$$\frac{d\vec{B}_d}{dt}=\left[\frac{d\left(R(t)\right)}{dt}R^T(t)\right]\vec{B}_d.$$

Calculations using 7 show that $\frac{d(R(t))}{dt}R^T(t)$ is a skew-symmetric matrix obeying:

$$\frac{d\left(R(t)\right)}{dt}R^T(t)=\begin{bmatrix}0 & \omega_{zd} & -\omega_{yd}\\ -\omega_{zd} & 0 & \omega_{xd}\\ \omega_{yd} & -\omega_{xd} & 0\end{bmatrix}$$

and multiplication of the matrix $\frac{d(R(t))}{dt}R^T(t)$ with the magnetic field vector is equivalent to the negative of the cross product of angular velocity with the magnetic field vector. Thus, the final mathematical relationship between the measurements of the magnetometer and the gyroscope is given by:

$$\frac{d\vec{B}_d}{dt}=-\vec{\omega}_d\times\vec{B}_d \tag{10}$$

When the readings of the magnetometer and gyroscope are in agreement, this equality should hold, regardless of the orientation of the phone. Therefore, any difference between the two sides of Equation 10 should indicate that either the gyroscope or the magnetometer is being spoofed. Translating this into practice, we first calculate the values

$\vec{\zeta} = -\vec{\omega} \times \vec{B}$ and $\vec{\eta} = \frac{d\vec{B}}{dt}$, approximating $\frac{d\vec{B}}{dt}$ by the finite difference $\frac{\vec{B}(t)-\vec{B}(t-\Delta t)}{\Delta t}$. All three components (x,y,z) of both $\vec{\zeta}$ and $\vec{\eta}$ are vectors of length N for the given measurement period $T = N\Delta t$. The mean square error (MSE) between the two signals is then given by:

$$MSE = \frac{1}{T}\sum_{i=1}^{T}\left((\zeta_i^x - \eta_i^x)^2 + (\zeta_i^y - \eta_i^y)^2 + (\zeta_i^z - \eta_i^z)^2\right) \tag{11}$$

A similar equation can be derived for the accelerometer-Doppler sensor pair as well (see Section 4.2).

3. Evaluation

We evaluated the defenses for the gyroscope by first reproducing the two acoustic attacks on the gyroscope as mentioned in [14] and [6]. To reproduce the attack of [14], we used a PUI Audio APS2509S-T-R piezoelectric transducer connected to a Picoscope 2206BMSO supported by Picoscope software v6.13.7.707 used as a waveform generator. To reproduce the attack in [6], we used a 4x2 dual channel PUI Audio AS06608PS-2-R speaker array with 8Ω impedance, connected to a Lepy LP-2051 audio amplifier which received input from the same Picoscope 2206BMSO. To perform automated frequency sweeps and frequency switches, we wrote a series of Python scripts which could control the Picoscope using the libraries provided by Picotech.

To evaluate the defenses for the magnetometer, we used an air-core solenoid with a 50Ω impedance, connected to the same Picoscope 2206BMSO waveform generator through an amplifier. Our test devices, as listed in Table 1, include a variety of smartphones from multiple vendors, as well as an STM32L475VG IoT node manufactured by STMicroelectronics. To collect the traces from the phone, we wrote a custom Android application that timestamped the sensor readings and uploaded them to an experiment server. The server is capable of controlling various components like the frequency of the wave, the number of traces to be collected and the duration of each trace. The IoT node was running custom C++ code written using the Mbed framework.

3.1. Methodology

The benign traces from all the phones were collected while the phone was being subjected to typical user activities like walking, running, at rest on the table, at rest in a pocket, and while the phone was being being shaken in motions similar to thosed used to play mobile video games.

To carry out the acoustic attack, we first had to identify the resonance frequency of the device. Tu et. al. in [6] listed the resonance frequency range of devices using the same sensor model as our test devices. We used our experiment server to sweep through the resonance frequency range and plotted the frequency against the variance of the sensor reading to pin-point the resonance frequency of the gyroscope. We determined the resonance frequency of the MPU - 6500 series family of gyroscope used in our test smartphones to be 27.243 kHz, and that of the LSM6DSL chip used in the IoT node to be 19.718 kHz.

Since [14] uses a piezoelectric speaker attached to the phone, the attack traces were collected not only when the phone was at rest, but also when it was being moved (walking, running, shaking etc.). On the other hand, the attack mentioned in [6] was carried out with a stationary speaker array and the device at rest. The distance between the speaker array and the test device was 0.3 m. The IoT node was programmed to replicate the functioning of a self-balancing scooter, one of the main test devices in [6]. A servo FS5103R motor was connected to the IoT node which rotated based on the feedback of the gyroscope. The benign traces from the IoT gyroscope included the sensor data when the device is at rest, when subjected to a repetitive to and fro motion, and when subjected to random shaking motions.

To carry out the magnetic attack on the gyroscope, we found that we were able to achieve maximum intensity and range of attack when the frequency of the wave was 1 Hz. The magnetometer attack traces included sensor traces collected as the solenoid was directed at the phone from different directions and orientations; these variations affect different axes differently. To simulate a rolling attack, in which the sensor reading is arbitrarily determined by the attacker, we created random pairings of benign gyroscope and magnetometer readings, each from a different, independent measurement session.

In total, from each phone we obtained 500 benign traces (100 traces each of walking, running, at rest on the table, at rest in a pocket, and random shaking) of each sensor, 500 acoustic attack traces of the gyroscope (250 for each of the two types of acoustic attacks, as described above), and 500 magnetic attack traces of the magnetometer. From the IoT node we obtained 1500 benign traces (500 traces each of at rest, under to and fro motion, and random shaking) and 1500 acoustic attack traces (all traces collected using the acoustic attack setup described in [6]). The numbers of benign and malicious traces collected were kept equal, to provide balanced classes for the machine learning training algorithms. The sensors were sampled at the highest possible sampling rate: 200 Hz for the gyroscope and 100 Hz for the magnetometer.

3.2. SDI-1: Single Sensor Defense

As mentioned earlier, training a classifier directly on high-dimensional data, such as sensor readings over time, is inefficient and can cause over-fitting. Thus, before the learning algorithm operates on the traces, each trace must be reduced into a small set of succinct features. Das et al. in [19] identified a list of features relevant for smartphone sensors in a different context. The data collected

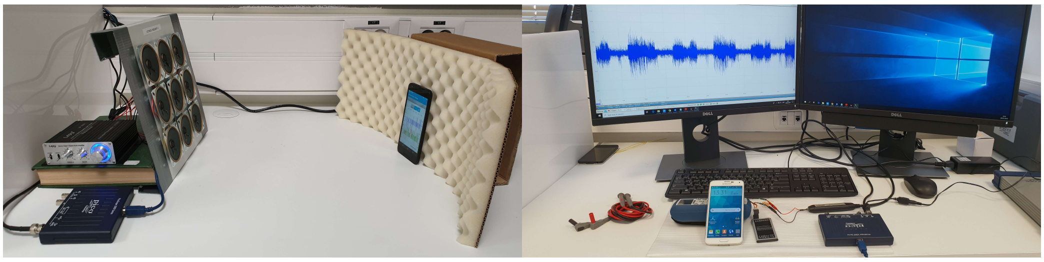

Figure 2: (a) Attacking the gyroscope (b) Carrying out power analysis on Galaxy S5

from the gyroscope is a stream of timestamped real values. Since we obtain the values from the three axes, the value is a vector consisting of x, y and z values associated with a specific point in time. The vector can be converted to a scalar by calculating the L_2 norm, which is equal to $L_2 = \sqrt{x^2 + y^2 + z^2}$. Another approach would be to look at the readings of only one axis. Das et al. summarise the characteristics of a sensor data stream by exploring a set of 25 features consisting of 10 temporal and 15 spectral features. With the help of a domain expert we also identified a new feature to represent the sensor data: max_val_fft, which is the maximum value of the fast Fourier transform of the sensor data stream. To analyze the relative importance of each feature, we used MATLAB's implementation of the *Relieff* algorithm [21], with k =20. The top ranking features and their corresponding weights are listed in Table 2.

Rank	Feature	Feature Importance Weight
1	Max val fft	0.0520
2	Max	0.0514
3	Mean	0.0409
4	Min	0.0396
5	Average Deviation	0.0341
6	RMS	0.0329
7	Standard Deviation	0.0282
8	ZCR	0.0052

Table 2: Importance of each feature, according to the *Relieff* algorithm

3.2.1. Detecting Attack on Gyroscope on Smartphone

After extracting the features from the raw traces, we used MATLAB's Classification Learner tool to train and test various machine learning models using a 10-fold cross validation scheme. The performance of the various classifiers we evaluated is presented in Table 3. As shown in the table, SDI-1 achieves a very high detection rate for all of the devices we implemented.

To evaluate the effectiveness of SDI-1 on the smartphone in an online setting, we selected the classification tree algorithm due to its consistently high accuracy and simple internal structure. We exported the structure of the trained tree from MATLAB, and developed an app in Android studio which implements the classification tree to detect the attack on the phone. The app also made it possible to explore different sampling window sizes, while keeping track of the true positives, true negatives, false positives and false negatives so that we can calculate the detection accuracy of the model. The app was initially installed on a Galaxy S5.

On initial testing we found that, despite the high accuracy shown when tested in MATLAB, our model had a very high false positive and false negative rate, especially when the sampling window was small when detecting in real-time. Upon inspecting the scatter plot which plots the various features used by the classification tree, we identified that the features we used were not able to separate between attack and normal user activity. This indicated that the features were not able to effectively separate between various acoustic attacks and typical user activities in real-time.

To overcome this shortcoming, instead of extracting the features from the L_2 norm, we extracted the features from the individual axes. This required the calculation of eight features (Table 2) on data from three axes. To reduce the number of calculations, we decided to remove two features: ZCR (lowest rank) and max_val_fft (calculation complexity). This leaves us with a total of six features for each of the three axes, for a total of 18 features.

The classification tree was trained again using Classification Learner in MATLAB and the model was implemented in the app. To calculate the real time accuracies, each attack detected when the phone was actually under was considered as a true positive (TP) and each attack our defense failed to detect was considered as a false negative (FN). During typical user activity (no-attack) each falsely detected attack was considered a false positive (FP), and rest of the events classified as true negatives (TN). Accuracy was calculated using the formula $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$. On testing, this updated model showed good performance irrespective of the sampling window, as shown in Table 4.

Type	Classifier	Galaxy S5	Nexus 5X	Galaxy S6	iPhone SE
Tree	Simple	98.9	92.9	96.7	100
	Medium	98.9	96.2	96.7	100
	Complex	98.9	96.2	96.7	100
Regression	Logistic Regression	86.9	76.7	99.0	100
SVM	Linear	85.7	73.8	98.6	100
	Quadratic	97.0	91.0	99.0	100
	Cubic	99.3	96.7	99.0	100
	Fine Gaussian	99.6	97.6	99.5	99.7
	Medium Gaussian	95.1	90.5	98.6	99.9
KNN	Fine	99.0	96.2	100.0	99.9
	Coarse	86.1	70.0	77.6	97.7
	Medium	96.4	91.4	98.6	99.9
	Cosine	94.4	92.9	97.6	99.9
	Cubic	95.5	91.9	96.7	99.9
	Weighted	97.4	96.7	99.5	99.9
Ensemble	Bagged Tree	99.8	98.6	99.5	100
	Subspace KNN	99.4	96.2	98.6	99.9

Table 3: Offline accuracy (%) of SDI-1 machine learning classifiers for gyroscope using 10-fold cross validation

Phone	Sensor	Sampling Window (sec)	Accuracy (%)
Galaxy S5	Gyroscope	1	98.42
		2	98.18
		5	98.33
	Magnetometer	1	98.20
		2	98.94
		5	97.64
Nexus 5X	Gyroscope	1	97.77
		2	99.04
		5	98.18
	Magnetometer	1	99.08
		2	98.75
		5	98.33

Table 4: Real time accuracy (%) of SDI-1 with different sampling windows

One-Sided Classification: As discussed earlier, in one-sided classification the classifier is trained only using the benign data and is tested on both the benign and malicious data. The main advantage of this method is that, in contrast to a two-sided classification model which must anticipate all attacks ahead of time, a one-sided classifier will be effective against new attacks, as long as they sufficiently deviate from the training data.

The *fitcsvm* function in MATLAB was used for one-sided classification, based on the S5 data set. The data table consisting of the features and the labels were divided into two parts (train and test). The benign instances from the first part were used to generate the model using the *fitcsvm* function. The data from the second part was used to test the model. The labels from the test table were removed and the model was made to predict each instance as one or zero representing an attack and no-attack respectively. The classifier gave 99.20% accuracy in this case.

We also tested one-sided classification of the Nexus 5X and the Galaxy S6, albeit with smaller data sets, resulting in accuracies of 71.69% and 75.47% for the Nexus 5X and Galaxy S6 respectively.

3.2.2. Detecting Attack on Gyroscope on IoT node

As mentioned in Section 3.1, we collected 1500 benign and 1500 acoustic attack traces from the gyroscope. Considering the severely constrained resources of the IoT node, we selected the five simplest features from Table 2: max, mean, min, standard deviation and average deviation. The features were extracted from the L_2 of the traces and then used to train a simple tree using the Classification Learner tool of MATLAB, resulting in a detection accuracy of 99.8% in the offline model after 5-fold cross validation. The tree which was trained using MATLAB was implemented on the IoT node. We programmed an LED to turn on every time an attack was detected. We also wrote a program to keep track of the true positives, true negatives, false positives and false negatives. After extensive testing under attack and under normal conditions, we obtained an accuracy of 98.03% with a sampling window of 5 ms. This proves that this defense method is efficient and effective in a wide range of devices, even under high resource constraints. In this case, unlike when using the Galaxy S5, we were able to obtain high accuracy using the features extracted from L_2 norms.

3.2.3. Detecting Attack on Magnetometer on Smartphone

Similar to the implementation of the single sensor defense on the gyroscope, the single sensor defense was implemented on the magnetometer. The accuracies of various

Type	Classifier	Accuracy (%)
Tree	Simple	91.4
	Medium	90.3
	Complex	80.4
Regression	Logistic Regression	89.8
SVM	Linear	87.4
	Quadratic	90.4
	Cubic	95.0
	Fine Gaussian	93.8
	Medium Gaussian	92.3
KNN	Fine	93.1
	Medium	91.2
	Coarse	76.4
	Cosine	91.2
	Cubic	90.5
	Weighted	93.0
Ensemble	Bagged Tree	94.9
	Subspace KNN	94.3

Table 5: Offline accuracy (%) of SDI-1 machine learning classifiers for the magnetometer using 10-fold cross validation

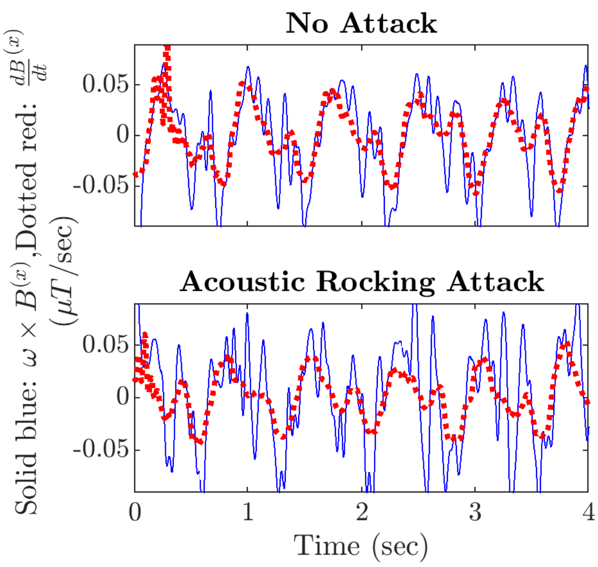

Figure 3: A rocking attack can be detected by the sensor fusion mechanism

machine learning models under K-fold cross validation using the Classification Learner tool in MATLAB are shown in Table 5. The trained classification tree was implemented on the phone using our Android app. Unlike the gyroscope, the model provided good accuracy (Table 4) using features extracted from L_2 norms. This shows that for *Zero-order sensors* like the magnetometer which measures the phone's static position or orientation, features extracted from L_2 are sufficient to differentiate between an attack and a no-attack scenario. However for *First-order sensors* such as the gyroscope, which measures the phone's rotation, features extracted from individual axes might be better to differentiate between an attack and a no-attack scenario. Single sensor defense for magnetometer was not carried out on the IoT node.

3.3. SDI-2: Gyroscope-Magnetometer Sensor Fusion Defense

In contrast to the machine learning defense presented in the previous subsection, the sensor fusion countermeasure works by comparing the output of the magnetometer, $\vec{B}$, to that of the gyroscope, $\vec{\omega}$, as described in Subsection 2.2. It is important to note that the two sensors have different physical characteristics. Specifically, the inexpensive Hall effect magnetometer used on most phones has a slower response time, lower sensitivity, and a higher noise level than the gyroscope.

Figure 3 shows the output of a sensor fusion calculation, captured on the Samsung Galaxy S5 phone, both under natural conditions (top) and under a rocking attack (bottom). In both cases the phone was placed in the researcher's pocket while the researcher was walking around the lab. As seen in the figure, the values over time of the x components of $-\vec{\omega} \times \vec{B}$ (solid blue) and $\frac{d\vec{B}}{dt}$ (dotted red) are much closer on the top half of the figure than on the bottom half. Nevertheless, the two values plotted on the top graph are still not entirely identical, due to the effects of the magnetometer's high measurement noise and variations in external magnetic sources.

As it is clear from the figure, even when the device is not under attack, there is still a small difference in the gyroscope and magnetometer reading. To mitigate these issues, we need to specify a threshold value and assume that any deviations below this threshold are normal. To identify the threshold, we calculated the MSE between the two sensor signals under typical user activity, under acoustic attack and under magnetic attack. Then, by using the sensor fusion MSE as a single feature, we trained a single split binary classification tree on MATLAB. Doing so we effectively instructed MATLAB to create a threshold-based detector, choosing an ideal threshold. When implementing the sensor fusion defense on the device, we can use the same exact threshold which was identified by MATLAB. To implement this method, we used a sampling window approach. An attack is identified if within the sampling window, 80% of the MSE's are above the threshold.

We implemented the sensor fusion on the Galaxy S5 using our Android app. After extensive testing under normal conditions and in the face of acoustic and magnetic adversary, the accuracy of sensor fusion was discovered to be above 95% for all sampling windows we evaluate, as shown in Table 6. We also carried out an offline threshold based sensor fusion defense on an iPhone SE based on the data collected from its gyroscope and magnetometer, obtaining an accuracy of 74.4%.

Device	Sampling window (sec)	Accuracy (%)
Salaxy S5	1	96.98
	2	99.04
	5	98.82
Nexus 5X	1	98.68
	2	98.38
	5	97.95

Table 6: Real time accuracy of of SDI-2 with different sampling windows

Classifier	Accuracy (%)	
	5-fold	10-fold
Fine tree	97.4	97.2
Medium tree	97.4	97.2
Quadratic SVM	96.8	97.2
Fine KNN	97.4	97.2
Bagged tree	97.9	98.0

Table 7: Offline accuracy of SDI-2 using multiple features extracted from MSE on Galaxy S5 traces

3.3.1. Sensor-Fusion on IoT node

Similarly to the case of the smartphone, sensor fusion was also implemented on the IoT node. Initially MSE was collected under normal conditions and under attack conditions. A single split binary classification tree using MSE as a single feature on MATLAB was used to identify the threshold. Then, we applied the threshold-based sensor fusion mechanism on the IoT node in real time. After testing under both normal and attack (acoustic and magnetic) conditions, we obtained a detection accuracy of 95.70%.

3.3.2. Improving Sensor Fusion Using Machine Learning

The advantage of the MSE threshold-based sensor fusion is its simple structure. Once we identify the threshold, every MSE above the threshold will be classified as an attack. Though our experiments on both the Galaxy S5, Nexus 5X and the IoT node showed that sensor fusion is highly effective as it is, it can be reinforced by using machine learning. Similar to calculating the features from the L_2 norm in single sensor defense, we can calculate the same set of features from the MSE's within a sampling window. These MSE based features were used to train various machine learning models using the Classification Learner tool in MATLAB.

The accuracies were calculated using different schemes of K-fold cross validation. The accuracies of various machine learning models trained and cross validated using Galaxy S5 data are provided in Table 7. As shown in the table, these accuracies are equivalent to those of the threshold-only defense, but we consider that this design may be more robust to intentional disruption.

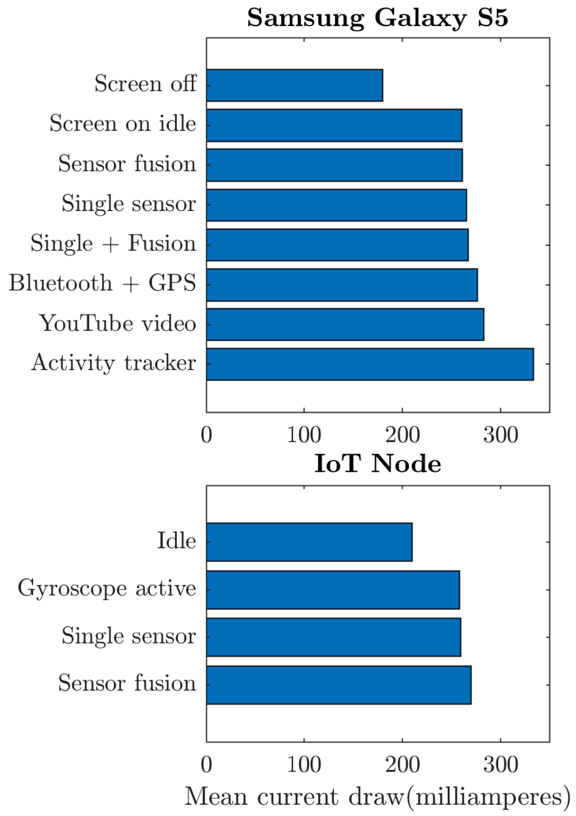

Figure 4: The effect of enabling SDI on the instantaneous power consumption of a smartphone (top) and an IoT node (bottom)

3.4. Real-Time Power Consumption and Performance Evaluation

Since our targeted devices include smartphones and other low power devices which have a limited energy reserve, any practical defense must consume only a minimal amount of power. To show that our methods provide this property, we measured their real-time power consumption using an extrernal lab setup, as illustrated in Figure 2. To carry out the power analysis, we disconnected the battery from the Galaxy S5 phone and routed it through a 0.2Ω resistor connected in parallel to a high-sensitivity Picotech TA046 800 MHz Differential probe. The voltage drop on the probe was sampled and stored on a PicoScope 2206BMSO oscilloscope. The traces were then imported to MATLAB for analysis.

As shown in Figure 4, our defenses consume a very small amount of power in excess to the phones normal activities. To put matters in proportion, assuming that the battery of the Galaxy S5 is at its full capacity of 2800 mAh and that the phone is constantly turned on but left idle, a phone in which our defense is always powered on will run out of battery 1.6 minutes sooner than a phone without our defense, a difference hardly noticeable by users.

Our generic fusion-based solution has very practical computational requirements, making it feasible to implement on a variety of software and hardware targets. Even though the mathematical model looks complex, per sample calculation of the gryoscope-magnetometer fusion relationship requires only a finite difference calculation (three subtractions), a cross product (nine multiplications), a Euclidean distance calculation (three multiplications), and finally a comparison to a threshold. A performance evaluation was carried out on the Galaxy S5 running Android version 5.0. Using Android Studio, we measured the real-time CPU consumption of our countermeasure at a high resolution.

Carrying out the calculations for our defenses took between 70 and 150 microseconds. Interestingly, this value was uncorrelated with the size of the sampling window we selected, leading us to believe that most of this time was actually spent on inter-process communications and UI updates, and not on the calculation itself. The total detection time is the sum of the sampling window size and the time taken for calculation. Thus, by using a sampling window size of 1 second, the time for detection will be approximately 1.0001 seconds. When running at real-time at a sensor sampling rate of 200 Hz, the highest rate possible on native Android applications, our fusion calculations consumed only 0.5% of the phone's CPU. On our test device the app only used 1.7% of RAM, showing that the countermeasure is both effective and feasible. We note that our application did not require special user permissions nor modifications to the underlying operating system. We believe that integrating our countermeasures into the device kernel will cause its resource consumption to be even lower.

4. Discussion

We presented two effective software-only methods for detecting acoustic and magnetic attacks on the gyroscope and the magnetometer. We developed and implemented our defenses, and performed detailed analysis on various devices under various circumstances. One of the major advantages of our defense methods is that they can be used for all kinds of devices. Although the machine learning models require data collection and training, this can be done externally, irrespective of the device, and only the trained model need to be implemented on the device. In addition, all of our defenses were independent of the size of the sampling window. We were able to achieve good accuracy with a sampling window as small as 5 ms on the IoT node, as mentioned in the previous section.

As we saw from the previous section, one of the main components determining the accuracy is sufficient 'good' data. A reduction in the size of the training data-set caused the reduction in accuracy when experimenting with one-sided single sensor defense on the Nexus 5X and Galaxy S6. Increasing the size of the data set used to train the model will have a significant effect on the performance of the classifier. Manufacturers who wish to implement these defenses can use a larger data-set, including additional user activities from multiple users, to train the models externally before implementing the defenses on the device. In addition, more feature engineering can be done to create new features that can better differentiate between an attack and a normal use.

4.1. Related Works

Machine-learning based methods for detecting sensor malfunctions based on a single sensor have already been considered in other domains, such as the field of environmental sensor networks. For example, in [22] the authors demonstrated the use of four data-driven methods for creating a one-step-ahead prediction model to create a sensor anomaly detection system, based on order q Markov models for different values of q. Even though this method can fit many kinds of streaming data sets, it is not appropriate for use in our scenario, where the characteristics of the signal can change dramatically between consecutive samples, even in benign situations. In [23] the authors reviewed a number of proposed machine learning solutions pertaining to network layer DoS attacks in wireless sensor networks.

Sensor fusion was first discussed as a defense against corrupted sensor readings by Chew et al. in [24]. In this work, the authors presented a methodology for transforming a process control program in a way that allows it to tolerate sensor failure. In this methodology, a reliable abstract sensor is created by combining information from several real sensors that measure the same physical value. Based on this work, Ivanov et al. [25] discussed an optimal schedule for sampling from abstract sensors in the presence of a spoofing adversary, and performed an experimental validation of their methods on a simulation based on the Landshark unmanned ground vehicle. In their work, Ivanov et al. sought to minimize the intervals in which the system relies on sensor fusion by choosing an optimal schedule in which the various sensors are sampled. While this work discusses the best detection and counter detection strategies for an abstract sensor, it does not implement a concrete sensor fusion algorithm, as we present in our work.

Delporte et al. made use of positional sensor fusion in a constructive context in [26]. In this work, a world frame approximation of the gyroscope was obtained while using a system equipped with only a magnetometer and an accelerometer. Our system uses a simpler algorithm than that used by Delporte et al. and makes fewer assumptions, since it is only interested in detecting incongruities in the sensor reading and not in explicitly estimating the sensor reading. In [27], sensor fusion algorithms were used for sensor bias estimations and adaptive strategies. Nashimoto evaluated in detail the security of sensor fusion by considering a sensor fusion scenario that involves measuring inclination, with a combination of an accelerometer, gyroscope, and magnetometer using Kalman filter in [28]. In [29], Kune et. al. used a software based method to mitigate EMI signal injection attacks against analog sensors.

Shoukry et. al. in [30] developed a physical challenge-response authentication scheme designed to protect active sensing systems against physical attacks occurring in the analog domain, while Shin et. al. developed a method to bypass these timing based sensor spoofing detection mechanism in [31].

In [32], the authors proposed a context-aware intrusion detection system which uses machine learning to detect sensor-based attacks on smartphones. The evaluation was done using adversaries constructed in a lab environment. It is important to stress that the attack model considered by Sikder et al. is different than ours – while their work protects against malicious apps that try to leak or steal information through malicious use of sensors, our work protects against attackers that try to corrupt the behaviour of innocent apps through manipulation of their sensor input.

On the experimental side, several works investigated hardware-based approaches for preventing sensor attacks. In [11], the authors tested physical isolation of sensors using paper, aluminum, acrylic and foam. They were able to decrease the effect of sound noise on the gyroscope up to 60% on the z-axis. Physical isolation of sensors, however, is not practical in many cases, since it increases the size and weight of the device and can also raise its internal temperature, which may cause other sensors and components to malfunction. Serrano et al. [33] and Pagani et al. [34] discuss new design for MEMS chips that can resist sonic attacks by increasing the driving frequency of the chips or by changing their internal geometries. Though effective, this updated hardware is not an option for the plethora of sensors already deployed in the field, especially since MEMS sensor chips are highly integrated devices with relatively long development cycles. Unlike hardware based solutions, our proposed countermeasures only require changes to the device software and firmware, which is relatively quick to develop and deploy. Moreover, our countermeasures can be deployed in devices which are currently in use.

Trippel et al. in [5] experimented and evaluated randomized sampling and 180^0 out-of-phase sampling as defenses against spoofing attacks. These defenses do not require the sensor hardware itself to be modified, but do assume that the defender has precise control over the sampling scheme of the sensors. If a device has such a capability, for example through a software-controlled ADC, this makes it possible to apply these defenses through low-level firmware upgrades. Most other devices, however, cannot be upgraded to include this defense without hardware modifications. Another limitation of these attacks is that they are very specifically tailored to the WALNUT attack, and do not protect against other methods of sensor spoofing. Our software-based defense mechanism, on the other hand, is generic and has no low-level requirements from the sensor control interfaces.

As explained in previous sections, the attacks shown in [14] and [6] are the acoustic attacks we reproduced to test our defenses. The list of features used by Das et. el. in [19] to develop sensor fingerprints served as the foundation during the feature creation stage for our machine learning based single sensor defense.

4.2. Protecting Against Attacks on Other Types of Sensors

Our paper shows how to protect against attacks on the gyroscope and the magnetometer. There are, however, also attacks on the accelerometer, another common type of MEMS motion sensor which measures the linear acceleration of the phone [5]. To determine whether SDI can protect against attacks against the accelerometer, we reproduced an acoustic rocking attack on the accelerometer on the Galaxy S5 based on [5], and deployed the SDI-1 defense against attack using the same methodology we used to protect against gyroscope attacks. Upon analyzing the data, we identified that the attack was very effective and the sensor readings were clearly separable from the readings from other user activities like running, walking, shaking etc. This led to most of the machine learning classifiers having perfect accuracy of 100% in identifying an attack. However, there are spoofing attacks for the accelerometer which completely control it (e.g. [5]). Defending against this type of attack requires the sensor fusion mechanism of SDI-2, which means we must correlate the accelerometer reading with another sensor which measures linear motion.

Doppler sensors are now being implemented in mobile phones, and they will fortunately be capable of addressing this need. The Doppler sensor measures the linear speed of the device, in meters per second (meter/sec), by analyzing instantaneous shifts in the frequency of signals the phone receives from stationary radiation sources such as Wi-Fi access points or cellular base station, as a result of the Doppler effect. Similar to the gyroscope-magnetometer sensor fusion, a similar mathematical relationship can be derived for accelerometer-Doppler sensor fusion.

Assume that the transmitter and the receiver are moving in an instantaneous relative velocity $\vec{v}(t)$, such that $|\vec{v}(t)| \ll c$, where c is the speed of light, while the transmitter emits an EM wave with frequency f_0. Without loss of generality, we assume that the traveling wave has some wave vector $\vec{k}$ that forms an angle $\theta(t)$ with the direction of the motion of the device. Using these considerations, the Doppler shift is given by:

$$\begin{aligned}\Delta f =& f' - f_0 = \\ =& \frac{v(t)}{c}\cos(\theta(t))\, f_0\end{aligned}$$

where $v = |\vec{v}|$, v_r is the radial velocity (the velocity in the direction of the line connecting the emitter and receiver) and f' is the shifted frequency. The sign of the radial velocity, i.e. the velocity times the cosine of the angle between $\vec{v}$ and $\vec{k}$, indicates the sign of the shift. If the transmitter and the receiver are moving towards each other, the sign of $v_r = v\cos(\theta)$ is positive and the detected EM wave is blueshifted. If, on the other hand the transmitter and the receiver are moving away from each other, the sign of $v_r = v\cos(\theta)$ is negative and the detected EM wave is redshifted. Note that this calculation requires the receiver to know the exact shape of the transmitted waveform. In a classical Doppler radar setup it is trivial to recover this waveform, since both the transmitter and the receiver are in the same physical circuit. Previous work in the radar research community has shown that it is also possible to recover the precise transmitted waveform of an external Wi-Fi receiver [35].

Once we reconstruct the velocity of the device from the Doppler shift in the frequency of the wave emitted by the transmitter and detected by the device, **it is possible to compare the readings from the Doppler radar to those of the accelerometer**. In order to compare the two measurements, one can either take the derivative of the velocity reading output by the Doppler radar, or, in order to avoid the noise added in the process of differentiation, one can use the integral of the acceleration (measured by the accelerometer) and apply a high-pass filter on the result to eliminate the constant part of the integral.

If the movement of the device is in the radial direction, i.e.: $\vec{v} \parallel \vec{k}$; $\cos(\theta) = \pm 1$, and the acceleration is parallel to the velocity, we can derive a simplified form of the instantaneous acceleration, in which the same convention about the direction of movement holds:

$$a(t) = \frac{c}{f_0}\frac{d(\Delta f)}{dt}$$

Note that by using the Doppler effect, one can only detect the relative velocity in the direction of the line connecting the transmitter and receiver. Reconstructing the acceleration in the case of an arbitrary θ requires additional processing. One must also take into account the gravitational acceleration added to the measurements of the accelerometer.

Evaluating a full Doppler-based defense would require us to make low-level modifications to the phone's closed-source radio baseband stack. We therefore leave this research as a direction for future work.

4.3. Is the Gyroscope Truly Invulnerable to Magnetic Attacks?

The initial phases of our research included identifying the effect of magnetic and acoustic adversaries on the gyroscope and magnetometer of the Galaxy S5. We did this by performing frequency sweeps using the PicoScope with our experimental setup as explained in Section 3. The magnetometer was immune to an acoustic adversary and vulnerable to magnetic adversary, as expected. Also, as shown in many previous works, the gyroscope showed disturbance in the face of an acoustic adversary. The gyroscope showed maximum variance in its readings at its resonance frequency range at 27 kHz. Interestingly, we identified that the gyroscope was showing disturbance under a magnetic field as well. We were able to observe a spike in the variance of the gyroscope readings under a magnetic adversary which coincides exactly with the resonance frequency of the gyroscope under the acoustic attack at 27 kHz, as seen in fig. 5. The magnitude of variance under a magnetic adversary is much smaller than that of an acoustic adversary, but still significantly above the noise level of a phone at rest. On inspecting the tear-down of the device, we found that many of the important chips like the CPU, RAM package, power management IC, gyroscope and accelerometer chip etc are housed under a metallic covering. This metallic covering might be causing the magnetic field to be converted to the corresponding acoustic vibrations. The fact that the maximum variance under the magnetic field coincides with the resonance frequency of the gyroscope supports this hypothesis. In our attempts, we were only able to use the magnetic adversary as a rocking attack on the gyroscope. Unlike the acoustic attack, due to its properties, the magnetic field is difficult to direct and control as needed for a rolling attack. Generating a directed magnetic field which can precisely control both the magnetometer and the gyroscope can, in theory, cause our sensor fusion to fail.

4.4. Responding to an Attack

As explained in Section 3, our defense can detect attacks but are unable to prevent them. This leads to the natural question: what should the phone do when there is major disagreement between its various sensor readings?

To respond to an attack, we first have to identify which sensor has been compromised. A device equipped with our single sensor defense for both the gyroscope and magnetometer will be able to detect which sensor is compromised, but will not be able to detect a rolling attack. A system with the gyroscope-magnetometer sensor fusion defense will be able to detect both rocking and rolling attacks, but will be unable to identify the compromised sensor. An ideal system would have both the single sensor and sensor fusion defenses implemented, allowing it to detect both rocking and rolling attacks, and next to identify the sensor that has been compromised.

Once we know which sensor has been compromised, one possible solution is to attempt to simulate the corrupted sensor using the non-corrupted one. While the performance of this simulated sensor will be degraded compared to the original sensor (i.e. lower sensitivity, longer response time, etc.), it will still be useful in many situations. In fact, Delporte et al. [26] were able to use only accelerometer and magnetometer readings to create a "virtual gyroscope". Another possible solution in the event of a sensor disagreement would be to tweak the sensor readings until they both agree, effectively halving the power of the attacker.

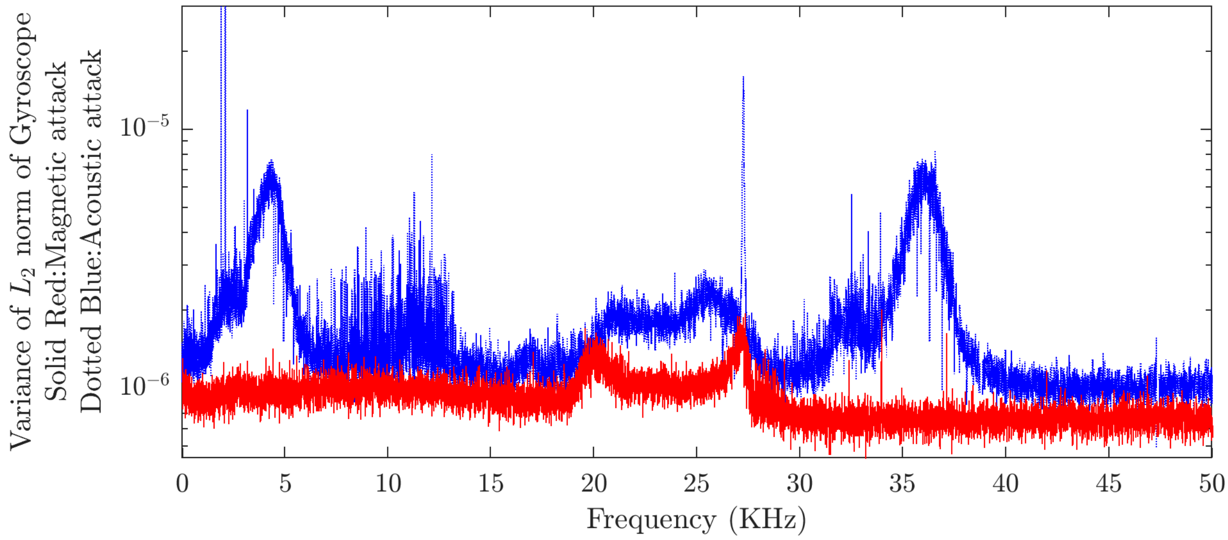

Figure 5: The effect of magnetic and acoustic adversaries on the variance of a Samsung Galaxy S5 smartphone's gyroscope

It seems that the optimal behavior in the case of sensor disagreement which cannot be corrected would be to report an error condition to the calling application, and leave the decision of how to respond to the application developers. This will allow the application to decide how to alert the user, and how to safely and intelligently carry out at least parts of its original intended functionality, even though it has low confidence in the readings of the sensor. How to provide this degraded functionality in a usable and generic way remains an open question.

4.5. Improving Sensor Fusion

In this work, we showed how to improve the reliability of one sensor reading by comparing it to another sensor. We can generalize this notion by comparing the sensor not just to other sensors, but to higher order state indicators known to the phone. One such indicator that might be combined with gyroscope readings in a sensor fusion algorithm is the timing and location of touches on the phone's touch screen. As shown in [36] and follow-up works, the phone's position sensor readings are so highly correlated with touches on the touchscreen that the gyroscope's output alone can serve as a keylogger. We can reverse the direction of inference, and consider what sort of gyroscope outputs should be detected whenever a key is pressed. Incongruence could indicate that the gyroscope is under a spoofing attack, or alternatively, that the touch screen is under a touch injection attack [37].

In a wider sense, even higher-order notions, such as the activity and general context of the phone, can be incorporated as inputs to the sensor fusion algorithm. For example, when the screen's display is off and its proximity sensor is active, one can reasonably assume that the phone is in the user's pocket. As Unger et al. have shown, data from the phone's myriad sensors can determine many fine-grained user contexts, such as periods when the user is eating, smoking, or listening to music [38]. Once the user context is established, the phone can apply a sensor spoofing detection model fine-tuned to this context, thereby achieving better performance. With more and more new sensors being integrated into the devices (e.g., GPS, barometer, sonar, lidar etc.), higher order sensor fusion has huge potential.

4.6. Applicability to Recent Phones

To apply our results to additional phone models, we need to establish that their sensors have a similar hardware design, and that they retain the same OS API for sensor access. On the hardware front, we have verified that a selection of modern phones are still vulnerable to rocking attacks on their sensors based on ultrasonic interference, including a Google Pixel 3 XL, an HTC OnePlus 6T, and a Samsung Galaxy S9. On the software front, we have verified that the most recent shipping version of Android (Android 10) still uses the SensorListener and SensorEventListener APIs used for our generic fusion-based solution. These two factors indicate that our experimental results can be immediately applied to more recent phones.

Since our defense method is purely software-based, implementing it on existing devices requires only a simple software update and does not require any expensive hardware modification. In addition, it should be noted that not only smartphones utilize these sensors -- a wide range of electronic devices use these sensors to act as a bridge to the outside physical world, which makes our work all the more important in securing today's cyber physical systems. Our implementation on the resource-constrained IoT node shows that even these resource-constrained devices can be made safer against sensor spoofing attacks with no additional hardware costs.

4.7. Conclusion

In this work, we developed, implemented and analyzed two new defenses against acoustic and magnetic adversaries affecting the gyroscope and magnetometer. Leveraging the information advantage the defender has over the attacker, we applied sensor fusion methods to detect when different sensor readings on the phone disagreed with each other.

We showed how fusion-based defenses can be applied to the magnetometer and gyroscope. Our software-only defense method can protect against attacks which cannot be detected by other methods, including sensor replay attacks (rolling attack). Sensor fusion defense can be augmented by machine learning based single sensor defense methods. Most significantly, our method has very realistic resource requirements and does not require changes to the phone's hardware, drivers, or operating system. Thus, it can be immediately put to use by phone manufacturers as well as smartphone application developers.

In future work, it would be interesting to flesh out the Doppler countermeasure, especially as Doppler-equipped phones and 5G networks become more prevalent. Future work could also focus on the evaluation of sensor fusion defenses based on high-level context and touch events.

References

[1] M. Conti, I. Zachia-Zlatea, B. Crispo, Mind how you answer me!: transparently authenticating the user of a smartphone when answering or placing a call, in: B. S. N. Cheung, L. C. K. Hui, R. S. Sandhu, D. S. Wong (Eds.), Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2011, Hong Kong, China, March 22-24, 2011, ACM, 2011, pp. 249–259. doi:10.1145/1966913.1966945.

[2] W. Lee, R. B. Lee, Implicit sensor-based authentication of smartphone users with smartwatch, in: Proceedings of the Hardware and Architectural Support for Security and Privacy 2016, HASP@ICSA 2016, Seoul, Republic of Korea, June 18, 2016, ACM, 2016, pp. 9:1–9:8. doi:10.1145/2948618.2948627.

[3] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, F. Zhao, A reliable and accurate indoor localization method using phone inertial sensors, in: A. K. Dey, H. Chu, G. R. Hayes (Eds.), The 2012 ACM Conference on Ubiquitous Computing, Ubicomp '12, Pittsburgh, PA, USA, September 5-8, 2012, ACM, 2012, pp. 421–430. doi:10.1145/2370216.2370280.

[4] R. J. Ellis, Y. S. Ng, S. Zhu, D. M. Tan, B. Anderson, G. Schlaug, Y. Wang, A validated smartphone-based assessment of gait and gait variability in parkinson's disease, PLOS ONE 10 (2015) 1–22. doi:10.1371/journal.pone.0141694.

[5] T. Trippel, O. Weisse, W. Xu, P. Honeyman, K. Fu, WALNUT: waging doubt on the integrity of MEMS accelerometers with acoustic injection attacks, in: 2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017, IEEE, 2017, pp. 3–18. doi:10.1109/EuroSP.2017.42.

[6] Y. Tu, Z. Lin, I. Lee, X. Hei, Injected and delivered: Fabricating implicit control over actuation systems by spoofing inertial sensors, in: W. Enck, A. P. Felt (Eds.), 27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018., USENIX Association, 2018, pp. 1545–1562.

[7] X. Yujie, 2018, The gadget that boosts your step count while you nap: From celebrity fans to lazy university students, there are plenty of people willing to pay for fake fitness stats on wechat.

[8] R. Wang, M. S. H. Aung, S. Abdullah, R. Brian, A. T. Campbell, T. Choudhury, M. Hauser, J. Kane, M. Merrill, E. A. Scherer, V. W. S. Tseng, D. Ben-Zeev, Crosscheck: toward passive sensing and detection of mental health changes in people with schizophrenia, in: P. Lukowicz, A. Krüger, A. Bulling, Y. Lim, S. N. Patel (Eds.), Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016, Heidelberg, Germany, September 12-16, 2016, ACM, 2016, pp. 886–897. doi:10.1145/2971648.2971740.

[9] E. Reinertsen, M. Osipov, C. Liu, J. M. Kane, G. Petrides, G. D. Clifford, Continuous assessment of schizophrenia using heart rate and accelerometer data, Physiological Measurement 38 (2017) 1456.

[10] E. M. de Lima Pinto, R. Lachowski, M. E. Pellenz, M. C. Penna, R. D. Souza, A machine learning approach for detecting spoofing attacks in wireless sensor networks, in: L. Barolli, M. Takizawa, T. Enokido, M. R. Ogiela, L. Ogiela, N. Javaid (Eds.), 32nd IEEE International Conference on Advanced Information Networking and Applications, AINA 2018, Krakow, Poland, May 16-18, 2018, IEEE Computer Society, 2018, pp. 752–758. doi:10.1109/AINA.2018.00113.

[11] Y. Son, H. Shin, D. Kim, Y. Park, J. Noh, K. Choi, J. Choi, Y. Kim, Rocking drones with intentional sound noise on gyroscopic sensors, in: J. Jung, T. Holz (Eds.), 24th USENIX Security Symposium, USENIX Security 15, Washington, D.C., USA, August 12-14, 2015., USENIX Association, 2015, pp. 881–896.

[12] W. Jiang, D. Ferreira, J. Ylioja, J. Gonçalves, V. Kostakos, Pulse: low bitrate wireless magnetic communication for smartphones, in: A. J. Brush, A. Friday, J. A. Kientz, J. Scott, J. Song (Eds.), The 2014 ACM Conference on Ubiquitous Computing, UbiComp '14, Seattle, WA, USA, September 13-17, 2014, ACM, 2014, pp. 261–265. doi:10.1145/2632048.2632094.

[13] K. Block, S. Narain, G. Noubir, An autonomic and permissionless android covert channel, in: G. Noubir, M. Conti, S. K. Kasera (Eds.), Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks, WiSec 2017, Boston, MA, USA, July 18-20, 2017, ACM, 2017, pp. 184–194. URL: `https://doi.org/10.1145/3098243`. doi:10.1145/3098243.3098250.

[14] B. Farshteindiker, N. Hasidim, A. Grosz, Y. Oren, How to phone home with someone else's phone: Information exfiltration using intentional sound noise on gyroscopic sensors, in: [39], 2016. URL: `https://www.usenix.org/conference/woot16`.

[15] U. CERT, 2017, ICS-ALERT-17-073-01A: MEMS accelerometer hardware design flaws (update a).

[16] Y. Shoukry, P. D. Martin, P. Tabuada, M. B. Srivastava, Non-invasive spoofing attacks for anti-lock braking systems, in: G. Bertoni, J. Coron (Eds.), Cryptographic Hardware and Embedded Systems - CHES 2013 - 15th International Workshop, Santa Barbara, CA, USA, August 20-23, 2013. Proceedings, volume 8086 of *Lecture Notes in Computer Science*, Springer, 2013, pp. 55–72. URL: `https://doi.org/10.1007/978-3-642-40349-1`. doi:10.1007/978-3-642-40349-1_4.

[17] W3C, 2018, Magnetometer.

[18] A. Das, N. Borisov, E. Chou, M. H. Mughees, Smartphone fingerprinting via motion sensors: Analyzing feasiblity at large-scale and studing real usage patterns, CoRR abs/1605.08763 (2016).

[19] A. Das, N. Borisov, M. Caesar, Tracking mobile web users through motion sensors: Attacks and defenses, in: 23rd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016, The Internet Society, 2016.

[20] H. Goldstein, C. P. P. Jr., J. L. Safko, Classical Mechanics (3rd Edition), Pearson, 2001.

[21] I. Kononenko, E. Simec, M. Robnik-Sikonja, Overcoming the myopia of inductive learning algorithms with RELIEFF, Appl. Intell. 7 (1997) 39–55. doi:10.1023/A:1008280620621.

[22] D. J. Hill, B. S. Minsker, Anomaly detection in streaming environmental sensor data: A data-driven modeling approach, Environmental Modelling and Software 25 (2010) 1014–1022. doi:10.1016/j.envsoft.2009.08.010.

[23] S. Gunduz, B. Arslan, M. Demirci, A review of machine learning solutions to denial-of-services attacks in wireless sensor networks, in: T. Li, L. A. Kurgan, V. Palade, R. Goebel, A. Holzinger, K. Verspoor, M. A. Wani (Eds.), 14th IEEE International Conference on Machine Learning and Applications, ICMLA 2015, Miami, FL, USA, December 9-11, 2015, IEEE, 2015, pp. 150–155. doi:10.1109/ICMLA.2015.202.

[24] P. Chew, K. Marzullo, Masking failures of multidimensional sensors, in: Tenth Symposium on Reliable Distributed Systems, SRDS 1991, Pisa, Italy, September 30 - October 2, 1991, Proceedings, IEEE Computer Society, 1991, pp. 32–41. doi:10.1109/RELDIS.1991.145402.

[25] R. Ivanov, M. Pajic, I. Lee, Attack-resilient sensor fusion for safety-critical cyber-physical systems, ACM Trans. Embedded Comput. Syst. 15 (2016) 21:1–21:24. doi:10.1145/2847418.

[26] B. Delporte, L. Perroton, T. Grandpierre, J. Trichet, Accelerometer and magnetometer based gyroscope emulation on smart sensor for a virtual reality application, Sensors & Transducers 14 (2012) 32–47.

[27] B. Fan, Q. Li, T. Liu, How magnetic disturbance influences the attitude and heading in magnetic and inertial sensor-based orientation estimation, Sensors 18 (2018) 76. doi:10.3390/s18010076.

[28] S. Nashimoto, D. Suzuki, T. Sugawara, K. Sakiyama, Sensor con-fusion: Defeating kalman filter in signal injection attack, in: J. Kim, G. Ahn, S. Kim, Y. Kim, J. López, T. Kim (Eds.), Proceedings of the 2018 on Asia Conference on Computer and Communications Security, AsiaCCS 2018, Incheon, Republic of Korea, June 04-08, 2018, ACM, 2018, pp. 511–524. URL: `https://doi.org/10.1145/3196494`. doi:10.1145/3196494.3196506.

[29] D. F. Kune, J. D. Backes, S. S. Clark, D. B. Kramer, M. R. Reynolds, K. Fu, Y. Kim, W. Xu, Ghost talk: Mitigating EMI signal injection attacks against analog sensors, in: 2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013, IEEE Computer Society, 2013, pp. 145–159. doi:10.1109/SP.2013.20.

[30] Y. Shoukry, P. Martin, Y. Yona, S. N. Diggavi, M. B. Srivastava, Pycra: Physical challenge-response authentication for active sensors under spoofing attacks, in: I. Ray, N. Li, C. Kruegel (Eds.), Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015, ACM, 2015, pp. 1004–1015. URL: `http://dl.acm.org/citation.cfm?id=2810103`. doi:10.1145/2810103.2813679.

[31] H. Shin, Y. Son, Y. Park, Y. Kwon, Y. Kim, Sampling race: Bypassing timing-based analog active sensor spoofing detection on analog-digital systems, in: [39], 2016. URL: `https://www.usenix.org/conference/woot16`.

[32] A. K. Sikder, H. Aksu, A. S. Uluagac, 6thsense: A context-aware sensor-based attack detector for smart devices, in: E. Kirda, T. Ristenpart (Eds.), 26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017., USENIX Association, 2017, pp. 397–414. URL: `https://www.usenix.org/conference/usenixsecurity17`.

[33] D. E. Serrano, R. Lipka, D. Younkin, P. Hrudey, J. Tovera, A. Rahafrooz, M. F. Zaman, S. Nagpal, I. Jafri, F. Ayazi, Environmentally-robust high-performance tri-axial bulk acoustic wave gyroscopes, in: 2016 IEEE/ION Position, Location and Navigation Symposium (PLANS), 2016, pp. 5–8.

[34] L. G. Pagani, S. Dellea, G. Bursi, M. Brunetto, L. Falorni, G. Langfelder, Enhancing vibration robustness and noise in automotive gyroscope with large drive motion and levered sense mode, in: 2019 IEEE 32nd International Conference on Micro Electro Mechanical Systems (MEMS), 2019, pp. 708–711.

[35] Q. Chen, B. Tan, K. Chetty, K. Woodbridge, Activity recognition based on micro-doppler signature with in-home wi-fi, in: HealthCom, IEEE, 2016, pp. 1–6.

[36] L. Cai, H. Chen, Touchlogger: Inferring keystrokes on touch screen from smartphone motion, in: P. D. McDaniel (Ed.), 6th USENIX Workshop on Hot Topics in Security, HotSec'11, San Francisco, CA, USA, August 9, 2011, USENIX Association, 2011.

[37] O. Shwartz, A. Cohen, A. Shabtai, Y. Oren, Shattered trust: When replacement smartphone components attack, in: W. Enck, C. Mulliner (Eds.), 11th USENIX Workshop on Offensive Technologies, WOOT 2017, Vancouver, BC, Canada, August 14-15, 2017., USENIX Association, 2017.

[38] M. Unger, L. Rokach, A. Bar, E. Gudes, B. Shapira, Contexto: lessons learned from mobile context inference, in: A. J. Brush, A. Friday, J. A. Kientz, J. Scott, J. Song (Eds.), The 2014 ACM Conference on Ubiquitous Computing, UbiComp '14 Adjunct, Seattle, WA, USA - September 13 - 17, 2014, ACM, 2014, pp. 175–178. doi:10.1145/2638728.2638781.

[39] N. Silvanovich, P. Traynor (Eds.), 10th USENIX Workshop on Offensive Technologies, WOOT 16, Austin, TX, USA, August 8-9, 2016, USENIX Association, 2016. URL: `https://www.usenix.org/conference/woot16`.