From Electromyogram to Password: Exploring the Privacy Impact of Wearables in Augmented Reality

RUIDE ZHANG, NING ZHANG, CHANGLAI DU, WENJING LOU, and Y. THOMAS HOU,
Virginia Polytechnic Institute and State University, Virginia, USA
YUICHI KAWAMOTO, Tohoku University, Sendai, Japan

With the increasing popularity of augmented reality (AR) services, providing seamless human-computer interactions in the AR setting has received notable attention in the industry. Gesture control devices have recently emerged to be the next great gadgets for AR due to their unique ability to enable computer interaction with day-to-day gestures. While these AR devices are bringing revolutions to our interaction with the cyber world, it is also important to consider potential privacy leakages from these always-on wearable devices. Specifically, the coarse access control on current AR systems could lead to possible abuse of sensor data.

Although the always-on gesture sensors are frequently quoted as a privacy concern, there has not been any study on information leakage of these devices. In this article, we present our study on side-channel information leakage of the most popular gesture control device, Myo. Using signals recorded from the electromyography (EMG) sensor and accelerometers on Myo, we can recover sensitive information such as passwords typed on a keyboard and PIN sequence entered through a touchscreen. EMG signal records subtle electric currents of muscle contractions. We design novel algorithms based on dynamic cumulative sum and wavelet transform to determine the exact time of finger movements. Furthermore, we adopt the Hudgins feature set in a support vector machine to classify recorded signal segments into individual fingers or numbers. We also apply coordinate transformation techniques to recover fine-grained spatial information with low-fidelity outputs from the sensor in keystroke recovery.

We evaluated the information leakage using data collected from a group of volunteers. Our results show that there is severe privacy leakage from these commodity wearable sensors. Our system recovers complex passwords constructed with lowercase letters, uppercase letters, numbers, and symbols with a mean success rate of 91%.

CCS Concepts: • **Security and privacy → Side-channel analysis and countermeasures**; *Access control*; *Embedded systems security*; *Information flow control*; Distributed systems security;

Additional Key Words and Phrases: Information leakage, augmented reality, EMG side-channel, PIN sequence inference, keystroke detection

This work was supported in part by the National Science Foundation under Grants CNS-1217889, CNS-1405747, CNS-1446478, CNS-1443889.

Authors' addresses: R. Zhang, N. Zhang, and C. Du, Room 314, 7054 Haycock Road, Falls Church, VA 22043, USA; emails: {rdzhang, ningzh, leondu}@vt.edu; W. Lou, Room 304, 7054 Haycock Road, Falls Church, VA 22043, USA; email: wjlou@vt.edu; Y. Thomas Hou, 302 Whittemore Hall, Blacksburg, VA 24061, USA; email: thou@vt.edu; Y. Kawamoto, Tohoku University, Sendai 980-8577, Japan; email: youpsan@it.ecei.tohoku.ac.jp.

https://doi.org/10.1145/3078844

ACM Reference format:
Ruide Zhang, Ning Zhang, Changlai Du, Wenjing Lou, Y. Thomas Hou, and Yuichi Kawamoto. From Electromyogram to Password: Exploring the Privacy Impact of Wearables in Augmented Reality. *ACM Trans. Intell. Syst. Technol.* 9, 1, Article 13 (September 2017), 20 pages.
https://doi.org/10.1145/3078844

1 INTRODUCTION

Augmented reality (AR) technology is a variation of virtual reality (VR). Unlike VR, AR promises to enhance our perception of and interactions with the real world, while VR completely immerse users inside a simulated one. AR has been researched extensively in the academic world since the 1960s. However, previous research mainly focused on the construction and application aspects of AR; there is little study on the security and privacy implications. With recent advancements in wireless networking and embedded devices, AR is no longer fancy equipment in sci-fi movies. Early generation AR products are already available commercially. Microsoft Hololens was just released at the start of 2016 (CNET 2016). People's enthusiasm on AR can be seen through the popular AR game Pokemon Go sweeping the world. Apart from this, Goldman Sachs Group has announced its prediction of an 80 billion dollar market by 2025 for AR and VR (Bloomburg 2016).

As AR systems are making their way into people's lives, we believe that it is now a pressing issue to study new security and privacy issues arising with AR. In order to identify new security problems, we ask ourselves: What new security and privacy concerns arise with AR systems? We observe that unlike most of today's desktop and smartphone applications, to provide their intended functionality, complex AR applications will require various, always-on sensing. It is crucial for AR systems to balance the access required for functionality with the risk of an application stealing data or misusing that access. The current common access control for mobile application is to ask for user permission for accessing specific sensor data. Some permissions, such as access to video or voice recording, can be easily identified as sensitive, while others can be subtle and difficult to know the implicit privacy risk. For example, two recent papers (Liu et al. 2015; Wang et al. 2016) leveraged insensitive accelerometer sensors on a smart watch to infer the PIN sequence that a user keyed in on an ATM machine.

In this article, we identify a new type of side-channel information leakage from the electromyography (EMG)-based gesture devices used in AR systems. EMG signals are subtle electric currents detectable from the skin due to muscle movements. When worn on the arm of the user, the signatures of current can be used to identify the hand gesture, such as holding a fist or waving hands. When the users are interacting with the AR system, gesture control input devices are used legitimately. However, in this work, we show that it is possible for malware listening in the background of an AR system to infer sensitive secrets from coarse-grained EMG signals, when the users are interacting with other computing systems in the physical world. To demonstrate the feasibility of attack, we use the leading EMG-based gesture control device, Myo, as the platform for the study. We consider two scenarios that happen almost every day in our daily lives.

The first one is tapping in a PIN sequence to unlock the screen on a mobile device. Nowadays, the authentication system on mobile devices like the iPhone relies on PIN sequence. If one can steal the PIN sequence, he is able to access all information (e.g., photo, text) on the mobile device. Previous research (Liu et al. 2015; Wang et al. 2016) has designed attacks for ATMs based on the fact that the user is moving his hand during the input process. However, when it comes to unlocking a screen, people often tap with both thumbs rather than a single one, so their hands keep still. The idea is to know which number each thumb is tapping from EMG sensor data. Our case study on this scenario shows that EMG signals can significantly reduce the search space for smart device PIN recovery.

The second scenario we consider is typing passwords on a keyboard. If one can deduce the password a victim types on a computer, he potentially may access the resources on the computer and even the victim's bank account. Liu et al. (2015) and Maiti et al. (2016) have provided methods to deduce words a user has input on a keyboard. But modern passwords are seldom words but a combination of signs, letters, and numbers. Therefore, recovering password needs accurate recovery of each symbol typed without the help of a dictionary. The idea is to combine the knowledge of which finger a user moves through raw EMG data and the track of a user's hand movement to recover the keystroke a user typed. Our experiments show promising results of recovering complex passwords with high probability. Furthermore, we observe that, even though the assumption of having a prior model of user's typing habit is widely used, it could be unrealistic in certain scenarios. We also perform further investigations to assess the possibility of employing unsupervised learning to develop user-specific models from his own typing. Even though the accuracy is lower than the supervised counterpart, it remains a serious threat to user privacy.

There are several challenges in our attack. First, it is challenging to detect the exact starting points for input events on the keyboard or touch screen device. Because of the pushing and releasing phase of a single keystroke or tapping being so close, the EMG signals appear as a whole rather than distinct signals. In addition, there are eight EMG signal channels, and not a single channel can have enough information to detect all of the starting points. Second, to track hand movement with low-fidelity sensors could be troublesome. The white noise due to the imperfection of sensors would make the estimation of direction and distance easily go wrong. Third, how to determine which finger has moved through the raw EMG sensor data segment is not clear.

To solve these challenges, we design and implement four subsystems based on the key insight of the signals. We borrow ideas from the digital signal processing field and machine-learning field to help build up the subsystems. Input event subsystem is capable of obtaining the starting points of the noisy EMG signal which solve the first challenge. The second challenge is tackled by the coordinate transformation subsystem which projects the track of hand movement to the keyboard plane. Number classification subsystem and finger classification system are designed to deal with the third challenge. The former one is able to infer the PIN sequences from EMG signal segments around the starting points, while the latter one can get the exact finger the user is moving.

We summarize our main contributions as follows:

- —We are among the first to study privacy leakage from EMG signal in gesture-control devices, which is poised to be an essential component in next generation human-computer interaction in AR.
- —We design novel algorithms based on dynamic cumulative sum and wavelet transform to determine the exact time of finger movements. Furthermore, we adopt Hudgins feature set in a support vector machine to classify recorded signal segments into individual fingers or numbers. We also apply coordinate transformation techniques to recover fine-grained spatial information with low-fidelity outputs from the sensor in keystroke recovery.
- —Based on the experiments with one of the most popular gesture-control devices, Myo, we show that it is possible to recover sensitive user secrets, such as PIN sequence for unlocking smart devices and complex passwords typed on physical keyboards, using the coarsed-grained information EMG and accelerometer from the sensor.

The rest of the article is organized as follows. We begin with background introduction in Section 2, as well as an overview of our attacks in Section 3. Section 4 shows the system design of our attacks. An evaluation of our supervised scheme is presented in Section 5. Section 6 evaluates the performance of our supervised scheme, we discuss the limitations of this article and potential mitigation of information leakage based on the EMG signal. Finally, Section 8 concludes this article.

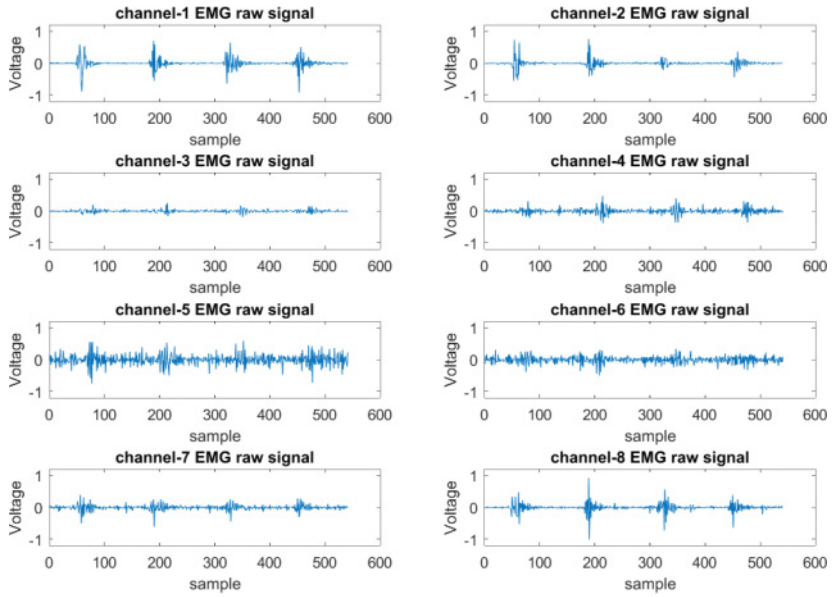


Fig. 1. Real-life EMG signals collected by Myo.

2 BACKGROUND

2.1 AR System and Gesture Controlled Device - Myo

AR has been researched broadly in the academic world since Ivan Sutherland described an AR prototype in 1968 (Sutherland 1968). From that time on, research on display technology, tracking, registration to properly align virtual and real objects, user interfaces, human factors, auxiliary sensing devices, and the design of novel AR application has been conducted (Caudell and Mizell 1992; Mann 1997; Feiner et al. 1997; Azuma et al. 2001; Costanza et al. 2009; Papagiannakis et al. 2008; Van Krevelen and Poelman 2010; Zhou et al. 2008).

Despite the long history of AR, the key components of AR systems remain similar. Commonly, an AR system is composed of a display device, input devices, and a computing device. The display device can be glasses or a head mounted device which is used to show a user virtual objects mixed with reality. Input devices are sensors including camera, GPS, motion sensor, etc. The computing device can be a computer or a mobile phone which provides computing power. Part of the objectives of input devices is to let the user interact with the AR system and our focus is in this category. And the last component, the computing device, can be a computer or a mobile phone which provides computing power.

In this article, we focus on a specific equipment, Myo (myo 2016), which is a gesture-control device that is designed to be worn on the forearms of a user. It is light-weight—only 93 grams. Jake Sims presented a demo of Myo in a real-life AR system in his blog (Myo Blog 2015). Multiple sensors are included on Myo to provide a seamless human-computer interaction. Myo is connected to computer desktop or mobile devices using Bluetooth. It is powered by an ARM Cortex M4 processor which enables it to be used for a full day with one charge. Within the slick design, it houses high-resolution medical grade sensors including eight EMG sensors and one three-axis accelerometer. Figure 1 shows some samples of EMG signals, which are recorded when a user is stroking the letter "s" on the keyboard. Although Myo enables many applications because of its high-resolution sensors, we observe that there is a potential privacy leakage caused by them. There

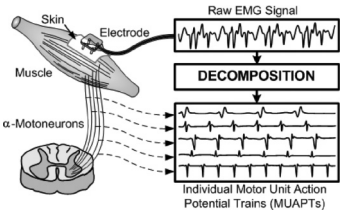


Fig. 2. EMG signals and their decomposition into MUAPTs (De Luca et al. 2006).

is no control of the access to the data streams generated by these sensors. Based on this observation, we believe this vulnerability can be leveraged to record passwords and PIN sequences.

The EMG signal collected by Myo reflects the body movement of a person. Body movement is a result of muscle contraction (Marieb and Hoehn 2007). A skeletal muscle is comprised of individual cells, or fibers, that are grouped into functional units called motor units. A single motor nerve can innervate the muscle fibers of a motor unit to make them contract together when receiving an electrical stimulus, called an action potential. The electrical stimulus is sent from the motor cortex of the brain to the muscle fibers via the motor nerve. When the motor unit fibers receive an action potential, they also generate action potentials by themselves, which are transient electrical signals that are conducted along the muscle fiber membranes. The motor unit action potential (MUAP) is the summation of the electrical stimulus in the single fibers of the motor unit and it can be elicited by a single action potential sent to a motor unit, which will lead to a transient contraction of the associated muscle fibers. Since muscle contraction results in electrical activity near the skin surface, it is possible to place sensors, called electrodes, onto the skin to detect the electrical activity. The area that an electrode is in direct contact with is referred to as the detection surface (Basmajian and De Luca 1985). Physiological data recorded by a surface electrode is called a surface EMG. Any portion of a muscle may contain muscle fibers belonging to 20–50 motor units. During a muscle contraction, multiple motor units are repeatedly stimulated. These stimulations typically occur asynchronously to facilitate smooth movements and delay muscle fatigue. This excitation pattern results in a sequence of MUAPs called a motor unit action potential train (MUAPT). Figure 2 shows that the myoelectric signal represents the temporal and spatial summation of MUAPTs within the pickup region of the recording electrode (De Luca et al. 2006). As we can see in Figure 2, EMG is a composite of different MUAPTs. The key insight here is that, when people are doing different motions, each MUAPT will contribute differently. This shows the possibility of classifying different finger actions.

2.2 Digital Key Stealing Systems and Privacy Leakage in Wearable Devices

There has been a long history of adversaries trying to steal the key entries of users on key-based security systems. A popular tool broadly employed by adversaries is key logger which can log all the keystrokes on the computer. The only drawback of this tool is that it leaves footage on the victim's computer. Some other traditional attacks in Balzarotti et al. (2008) and Maggi et al. (2011) rely on shoulder surfing and hidden cameras. In these kinds of attacks, the malicious code will gain access to the direct visual image of the key entry process. However, the capability to gain access to the camera is a strong assumption. In order to achieve stealth, we can see another line of work which focuses on developing novel side-channels to infer the key entries. For example, the sound produced by different keystrokes can be a valid side-channel to infer keys as described in Asonov and Agrawal (2004). Following this line of research, a bunch of other valid side-channels are discovered such as electromagnetic emanations (Vuagnoux and Pasini 2009), acoustic emanations (Zhu et al.

Table 1. Sensors On Different Off-the-Shelf Wearable Devices Nowadays

Smart Device Name	Sensor Type								
	D	G	A	Gy	H	Al	Am	T	E
Apple Watch Series 2 [Apple 2017]	No	Yes	Yes	Yes	Yes	No	Yes	No	No
Garmin Fenix 5[DC Rainmaker 2017]	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No
Fitbit [Fitbit 2017]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Samsung G3 [Samsung 2016]	No	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Myo [myo 2016]	Yes	No	Yes	Yes	No	No	No	No	Yes

D = Digital Compass, G = GPS, A = Accelerometers, Gy = Gyroscope, H = Heart Rate Sensor, Al = Altimeter, Am = Ambient light Sensor, T = Temperature Sensor, E = EMG Sensor.

2014; Berger et al. 2006; Zhuang et al. 2009), optical emanations (Raguram et al. 2011), and even the vibration of a wooden desk (Marquardt et al. 2011). The main drawback of these side-channels is that special equipment needs to be deployed beforehand. Researchers noticing this drawback try to install malicious applications on smartphones to exclude the strong assumption. However, the experiment results in Marquardt et al. (2011) and Zhu et al. (2014) indicates that smartphones need to be placed close enough to the keyboard by the victim, which is not the case in most scenarios. Shukla et al. (2014) employs a camera to capture the user's hand and the back side of the touch screen to recover the smartphone lock PIN. This method has a low inference accuracy and it assumes the capability for malicious code to access sensitive sensors. Researchers also explore the possibility of using channel state information of WiFi signals to infer smartphone PIN sequences or keystrokes nowadays (Li et al. 2016; Ali et al. 2015).

In recent years, wearable devices are becoming part of modern life. From Table 1, we can see various sensor types on off-the-shelf wearable devices nowadays. Researchers who notice the trend start to exploit sensors on wearable devices to do keystroke or PIN sequence inference. Two recent papers (Wang et al. 2016; Liu et al. 2015) leverage sensors on smart watches to infer PIN sequence. Explicitly, both of them take advantage of the accelerometers to measure the distance between two different inputs on an ATM machine to recover a PIN sequence. Wang et al. (2016) takes one more step to get rid of the training phase which is required in Liu et al. (2015). Liu et al. (2015) and Maiti et al. (2016) provide methods which employ accelerometers and audio recorders on smartwatches to infer keystrokes on a keyboard. However, their methods can only recover the user's typed words and will not work with non-contextual inputs. In comparison, in order to recover complex passwords which are probably not words, our work requires letter-granularity precision.

3 INFORMATION LEAKAGE OVERVIEW

In this section, we first make clear the assumptions. Then we demonstrate the steps of how our supervised implementations could infer passwords and PIN sequences. In addition, at the end of each part, we demonstrate the potential advantages of information leakage based on our methods.

3.1 Threat Model

For our supervised attacks, we assume an application has been installed beforehand on a user's computer or mobile device depending on which Myo armbands are connecting. This application can access insensitive EMG and accelerometer sensor data and the application can interact with a remote server. We also assume our application includes an initialization phase. During the initialization phase, a user is instructed to do a series of tapping actions. Applications with these abilities are common. For example, a health monitoring application would serve all the needs.

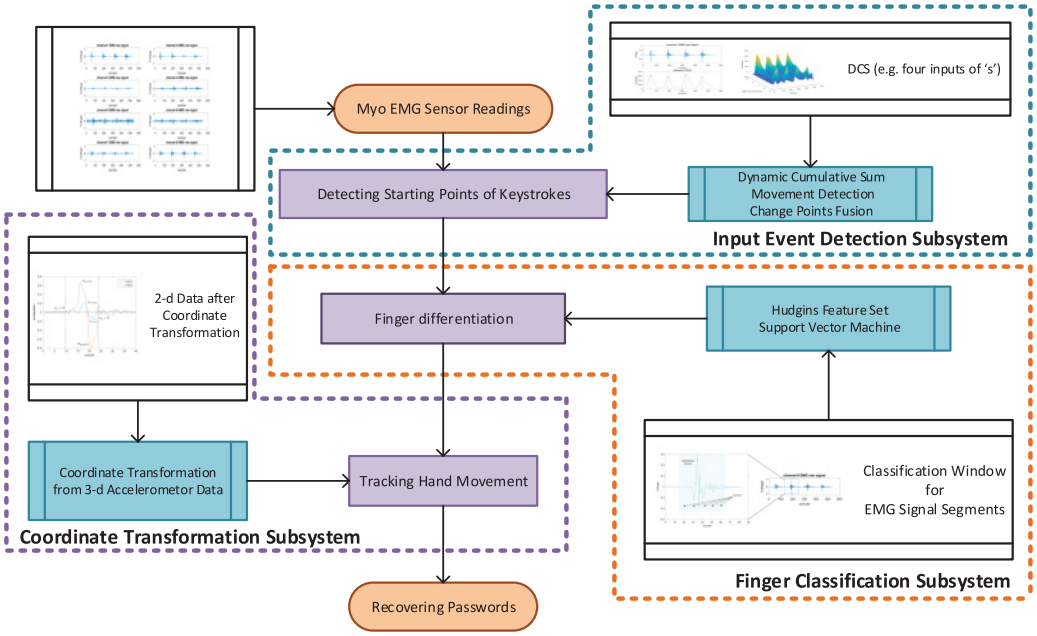


Fig. 3. System overview of deducing passwords.

3.2 Steps in Deducing Passwords

We implement a realistic system to clarify the possibility of using side-channels provided by gesture control devices to recover passwords. Figure 3 presents the system framework for deducing passwords. In the system, we first detect the starting points of each keystroke. Then we extract the EMG signal around the starting point. With machine-learning schemes, we map the EMG signal to the finger. At the same time, we extract the trace of hand movement relative to the keyboard. Combining the finger and trace of hand, we deduce the exact keystroke a user has input. The methods applied in the system are elaborated below.

Detecting keystrokes. The sensor data for a user's finger movements need to be separated first when he types on a keyboard. By the methods implemented in the input event detection subsystem, we are able to separate the signals. The methods include dynamic cumulative sum (DCS) in Khalil and Duchêne (2000), Al-Assaf (2006), El Falou et al. (2000), and Mustapha et al. (2008a, 2008b), and our new algorithms, movement detection algorithm and change points fusion algorithm.

Finger differentiation. With the timestamps from the previous step, we now direct them into the finger classification subsystem. The finger classification subsystem uses the Hudgins feature set (Hudgins et al. 1993) and adopts a supervised machine-learning method, support vector machine (SVM) (Keerthi et al. 2001), to generate the classifier.

Track hand movement. Utilizing the starting points from the first step, we employ the coordinate transformation subsystem to obtain the distance and direction of hand movement relative to keyboard between consecutive keystrokes.

Recovering passwords. Combining the information from the second and third steps, we can now infer which key the user has typed each time. Furthermore, by recording the key sequences when a user is typing passwords, we successfully recover a user's passwords.

Compared with previous works on keystroke inferring methods, our work has multiple advantages as follows:

—Non-intrusive. In Asonov and Agrawal (2004), Marquardt et al. (2011), and Zhu et al. (2014), they have to deliberately put external devices like a microphone or touch screen device close to the keyboard. Otherwise, they cannot access signals with enough signal noise ratio to do the inference. However, in our scheme, Myo is on the user's forearm. So we do not have to set up any specified scenario.
—No access to highly sensitive sensors. In Liu et al. (2015), they assume the application can gain access to the audio recorder of the mobile device and in Balzarotti et al. (2008) and Maggi et al. (2011), they assume the application can obtain the camera data. In our method, we only need access to an accelerometer and EMG sensor, which are pretty common for any gesture-control applications.
—Capability of recovering non-contextual inputs. In Wang et al. (2015) and Liu et al. (2015), linguistic models or dictionaries are employed to infer the words. Their methods cannot recover non-contextual inputs like passwords. Nonetheless, with our scheme, we can achieve letter-granularity precision which is necessary for recovering passwords.
—High accuracy. We adopt accelerometer and EMG sensor data which can generate high entropy. And this leads to the high letter-granularity accuracy of our scheme.

3.3 Steps in Recovering PIN Sequences

The other scenario we consider is that the victim is holding his touch screen device with both hands when unlocking the screen. In this case, our application only requires the access of an EMG sensor. At first, we detect the starting points of each tapping action. Then we extract the EMG signal around the starting point and map the EMG signal to the number. The methods applied in the demonstration are elaborated below.

Detecting thumb movement. We slightly modify the parameters in our input even detection subsystem to adapt to this scenario because the patterns of the EMG signal are similar to the keyboard scenario. The outcome of this step is the starting point for every thumb movement.

Classifying thumb movement. With the starting point of every thumb movement, we direct EMG signal segments into the number classification subsystem. SVM and Hudgins feature set are adopted in the number classification subsystem.

Previous works (Cai and Chen 2011; Miluzzo et al. 2012; Xu et al. 2012) utilize data through the accelerometer embedded in the targeted device, which is the touch screen device in our case. They cannot be used if the touch screen device is free of malware. On the contrary, the method we adopt does not need direct access to the target device. In addition, our attack in this scenario is also non-intrusive and does not need access to highly sensitive sensors.

4 SYSTEM DESIGN

In this part, we discuss the detailed technologies applied in our supervised attacks. We start with introducing the modeling of the EMG signal and then we introduce the four subsystems to accomplish the demonstrations.

4.1 EMG Signal Modeling

An input event detection subsystem is designed for detecting the starting points of the motions. Detection of finger movement is based on the analysis and characterization of forearm EMG during an action. In our case, the recorded electromyographic signals can be modeled by a random process

$$x(t) = \sum_{i=1}^{n} C_i(t) + \sum_{i=1}^{n} R_i(t) + n(t). \tag{1}$$

This equation is a composite of multiple types of signals collected by the EMG sensor like activity burst and noise. $\sum_{i=1}^{n} C_i(t)$ are our target signals which are caused by the pushing actions while $\sum_{i=1}^{n} R_i(t)$ are caused by the releasing actions. The superscript n means the number of keystrokes performed. Both the pushing and releasing actions follow a pattern of short potentials which appear with the acts of fingers. Lastly, $n(t)$ is the white noise caused by multiple factors like environmental conditions or thermal noise.

4.2 Input Event Detection

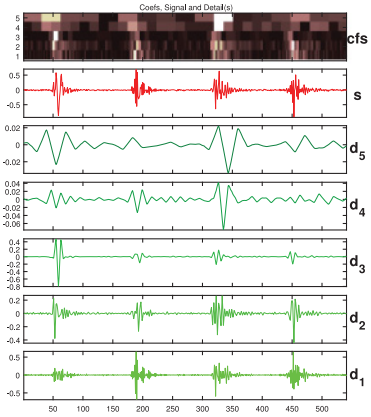
Detection of human movement by simple threshold methods and simple energy comparison between neighbor signal windows has been presented in Chan et al. (2000), Englehart et al. (1999), Tsenov et al. (2006), and in Farry et al. (1996). However, those methods are not suitable for cases where signals appear dynamic and noisy. Also, to obtain the starting point of movement visually as in Jorgensen et al. (2003) is neither accurate nor efficient in our case. In addition, no unique database can be set up for any person (Khalil and Duchêne 2000). Fortunately, the generalized likelihood ratio test in Fancourt and Principe (2000) and Barkat (2005) can be utilized to build DCS which can be used to detect human movement (Khalil and Duchêne 2000; Al-Assaf 2006; El Falou et al. 2000; Mustapha et al. 2008b). Nonetheless, their method cannot be directly applied to our case because our EMG signal has eight channels and requires distinguishing between two similar patterns (i.e., pushing and releasing of keys). In order to construct an accurate movement detection algorithm, we first obtain the DCS of the collected EMG signals. Then we design novel algorithms to obtain the starting point of each target action from DCS. The key insight is that, DCS will reach maximum during the motion as proved in Khalil and Duchêne (1999).

4.2.1 Dynamic Cumulative Sum. DCS is an improvement of CUSUM (or cumulative sum control chart) (Grigg et al. 2003). However, CUSUM is only suitable for situations where the priori knowledge of what change will happen to the signal after the point of change is known. Thus, we adopt DCS which suits circumstances where the priori knowledge is not required. One prerequisite of applying DCS is that signals must follow Gaussian distribution. We will show that in the following part of this section. Basically, DCS calculates local cumulative sum of likelihood ratios between segments before and after time point t_m. Let us assume the two segments are $S_b^{(t_m)}$ (before t_m) and $S_a^{(t_m)}$ (after t_m) and the width of these two segments is W. $S_b^{t_m} : x_{i,i=t_m-W,\ldots,t_m-1}$ follows a pdf $f_{\theta_b}(x_i)$ and $S_a^{t_m} : x_{i,i=t_m+1,\ldots,t_m+W}$ follows a pdf $f_{\theta_a}(x_i)$. The parameters $\hat{\theta}_b$ and $\hat{\theta}_a$ are estimated using $S_b^{(t_m)}$ and $S_a^{(t_m)}$. The DCS is defined as the sum of the logarithm of likelihood ratios from the beginning of the signal to the time t_m:

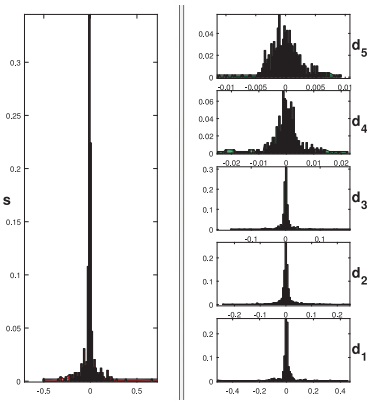
$$DCS^{(t_m)}\left(S_a^{(t_m)}, S_b^{(t_m)}\right) = \sum_{i=1}^{t_m} Ln \frac{f_{\hat{\theta}_a}{}^{(t_m)}(x_i)}{f_{\hat{\theta}_b}{}^{(t_m)}(x_i)}, \tag{2}$$

where the θ can be estimated by the variance of each segment.

We further adopt wavelet transform (WT) (Chui 2014) to improve the movement detection accuracy. The prerequisite of using WT in DCS is that the WT decompositions of EMG signals are multidimensional Gaussian. Figure 4(b) presents an example of the histograms of randomly selected 600-sample and its WT decomposition at five scales. It shows that our case meets the prerequisite. WT is applied to both the before and after segments. The choice of motherlet is crucial when adopting WT in signal processing. In Al-Assaf (2006), they conclude that the best wavelet for human movement EMG signal processing is the second-order Coiflet associated with the first five decomposition scales obtained by Shannon entropy criterion. The results of our experiment reinforce their conclusion. Figure 4.1 illustrates multiscale decomposition of the EMG signal in

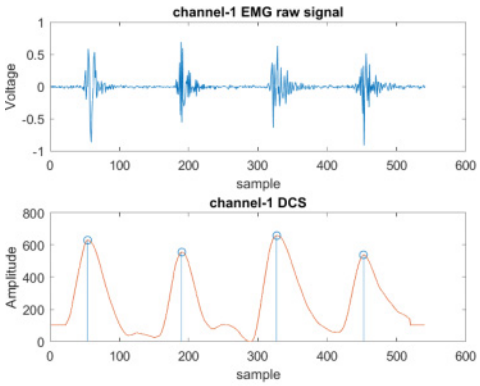


Multicscale decomposition using orthogonal wavelet.

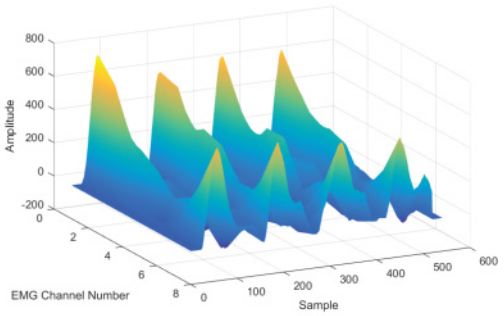


5-levels detailed WT decompositions histograms.

Fig. 4. Analysis of EMG signals.



DCS and point of change for one channel.



DCS of all eight EMG channels.

Fig. 5. DCS of EMG signals.

Figure 1 using a second-order Coiflet orthogonal wavelet. The time interval between samples in is 5 milliseconds.

The DCS corresponding to signals in Figure 4(a) is depicted in Figure 5(a) and (b). We observe that DCS in some channels has larger maximum than others and larger maximum can make the movement detection more accurate. From Figure 5(b), channel 1 and channel 8 which are next to each other have greatest maximum. This is because the muscles used to perform these actions majorly sit close to each other. The detection decision is included in Figure 5(a) as blue circles. We easily observe that the turning point of the DCS indicates the existence of an action as expected. In addition, we could find two bumps which indicate the releasing movement in the DCS of the first two keystrokes.

4.2.2 Algorithms. Our next task is to extract the timestamps of starting points for movements from the DCS described above. We develop two algorithms to accomplish the task. A movement detection algorithm is developed to calculate the DCS and detect the pushing movements in the EMG signals. In the movement detection algorithm, we set up threshold T to get rid of the releasing movements. The value of T is set to 350 empirically. Then we redirect the output timestamps to

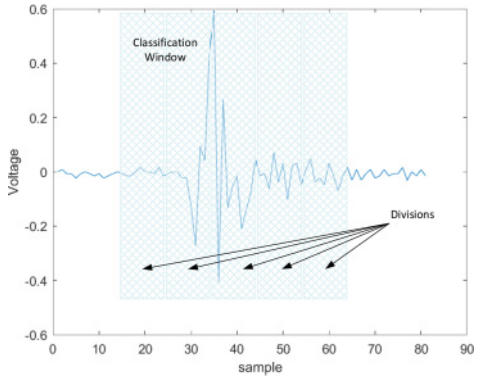


Fig. 6. Classification window and its divisions.

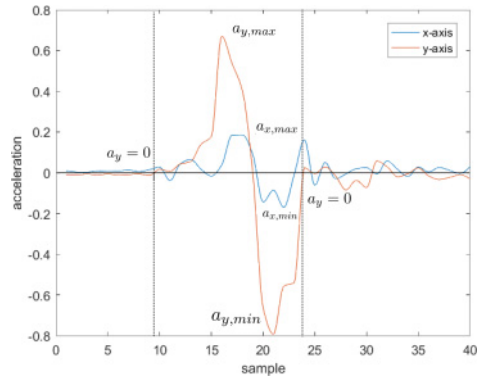


Fig. 7. Acceleration after projection.

the change point fusion algorithm. Empirically, we use two channels and the threshold x in the change point fusion algorithm is set to 20, which is 100 milliseconds.

4.3 Finger and Number Classification

Finger and number classification subsystems are similar to each other, so we introduce them together in this part. The finger classification subsystem is about classifying which finger a user is using from a part of the EMG signal. While the job of the number classification subsystem is to determine which number has been tapped from a segment of EMG signals. With the input event detection subsystem, we now have the starting point for each finger action. Here we set up a window for the EMG signal at each starting point. Empirically, classification achieves decent performance when the size of the sliding windows is 45 samples which is 225 milliseconds for the finger classification subsystem and 60 samples which is 300 milliseconds for the number classification subsystem. The insight here is that the action of tapping is larger than the action of stroking a key. In addition, we add offset to the starting point so that the sliding window could include the signal for the whole action.

4.3.1 Feature Extraction and Classification. We extract Hudgins feature set (Englehart et al. 1999) from each motion. The Hudgin's time-domain features are comprised of five different features for a given classification window. Here we divide the classification window into five equally divided segments as in Figure 6 and each of the segments will have five features. So there will be a total of 30 features per channel (including the undivided classification window). These features include mean absolute value (MAV), difference MAV, zero crossing, slope sign changes, and waveform length. Then we take advantage of the labeled samples collected in the initialization phase to do a supervised learning using a SVM classifier in our implementation. After training, the SVM classifier could give us which finger or number a new signal segment is related to.

4.4 Hand Movement Tracking

The last subsystem is a coordinate transformation subsystem which calculates projection of distance and direction of the hand movement between every two successive keystrokes onto the keyboard plane. The distance and direction derivation sections are similar to the technique used in Wang et al. (2016) and we follow their symbol and sign in our description of this subsystem. Our scheme and theirs differ on the coordinate alignment part. Wang et al. (2016) assume the adversary has placed other accelerators on the target plane (which is the keyboard plane in our case).

ALGORITHM 1: Movement Detection Algorithm

At each sample, the DCS is calculated according to (3) using the two segments $S_b^{t_m} : x_{i,i=t_m-W,\ldots,t_m-1}$ and $S_a^{t_m} : x_{i,i=t_m+1,\ldots,t_m+W}$
if The DCS has a turning point at that sample which indicates that it may be a finger pushing movement or a finger releasing movement **then**
    if There is a releasing movement before **then**
        This is a pushing movement, record the timestamp
        Move to the next sample
    else
        if The difference between this movement and the former pushing movement exceeds a threshold T **then**
            This is a releasing movement
            Move to the next sample
        else
            This is a pushing movement, record the timestamp
            Move to the next sample
        end if
    end if
else
    Move to the next sample
end if
Output the timestamps recorded

ALGORITHM 2: Change Point Fusion Algorithm

Get the recorded timestamps from the output of movement detection algorithm for selected channels and sort them into list $L1$ ascendingly.
Generate an empty list $L2$
Start from the first element l_m in $L1$ and do the following.
if Any timestamp from other selected channels are close to l_m within threshold x **then**
    Add l_m into list $L2$
    Delete timestamps close to l_m within threshold x in list $L1$
    Delete l_m in list $L1$
    Go to the next element in list $L1$
else
    Go to the next element in list $L1$
end if
Output the list $L2$

However, in our case, we only assume the keyboard is placed on a flat plane and all users' forearms have similar initial position toward the keyboard.

4.4.1 Projection Matrix. In order to calculate the displacement of hand moving on keyboard, we need to perform coordinate system transformation. So our goal is to obtain a projection matrix. According to our assumption, it only needs to be calculated once. The first coordinate system we build is the keyboard coordinate and the second coordinate system is the device coordinate. The job required is to transform the displacement in the device coordinate to the keyboard coordinate. This way, we can observe the finger movement projected to the keyboard plane directly. For the sake of calculating the displacement between two consecutive keystrokes, we assume the origins of the two coordinate systems overlap. To calculate the P matrix, we need to have correspondences between three different points in the two coordinate systems. According to our assumptions, the gravity is parallel with the z axis of the keyboard coordinate. Thus, we assume the coordinate for gravity is $(0, 0, 1)$ for convenience, which will not affect the

construction of P. The initial reading of accelerometers in Myo is caused by the gravity, so let us assume the initial readings are (x_1, y_1, z_1). And (x_1, y_1, z_1) is according to the device coordinate. This vector in the keyboard coordinate will be $(0, 0, 1)$ according to our assumption. We obtain the other two points by asking a user to type in "f" "r" and "f" "g," respectively. This way, we get the vector $(0, 1, 0)$ and $(1, 0, 0)$ on the keyboard coordinate and at the same time, we record the readings of Myo. The readings of Myo can be used to construct the displacement of hand in every two consecutive keystrokes. The method applied here to get the displacement is integration. It is well known that the integral of acceleration is velocity and the integral of velocity is displacement. Let us assume the recorded vectors are (x_2, y_2, z_2) and (x_3, y_3, z_3). So we get the linear algebra formula $\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{pmatrix} * M$. By solving these linear equations, we can obtain the linear transformation matrix M. And the projection matrix P can be constructed by extracting the first two columns of M.

4.4.2 Distance Estimation and Direction Derivation. Because we obtain the starting points of the movements from movement detection, we can extract the accelerometer sensor within the time interval between two consecutive keystrokes out accurately. In addition, the movement of the forearm is before the stroking of keys, so we add an offset here to capture the whole interval of the movements. Figure 7 shows an example of acceleration data after projection between two keystrokes "v" and "t." To get rid of the noise brought on by hand vibration, we can easily observe that acceleration captured during the two consecutive motions has unique patterns on the x and y axes (i.e., either up-and-down or down-and-up shapes due to different moving directions). Thus, we follow the technology in Wang et al. (2016) to get the starting point and ending point by first zero-crossing point occurring before and after the unique acceleration pattern. So the acceleration is always like a pattern of $[0, a_{k,max}(a_{k,min}), 0, a_{k,min}(a_{k,max}), 0]$ (k could be x or y).

Therefore, our strategy can be separated into the following parts: (1) extract the three-axis acceleration between the releasing and pressing points of two consecutive keystrokes; (2) project the three-axis accelerometer data to the keyboard plane; (3) examine the two-dimensional data to find $[a_{x,max}, a_{x,min}, a_{y,max}, a_{y,min}]$; (4) find the starting point of the movement by searching the first time that acceleration crosses the axis (i.e., zero-crossing point) before $a_{k,max}$ or $a_{k,min}$, whichever comes first; (5) similarly, find the ending point by searching the zero-crossing point after $a_{k,max}$ or $a_{k,min}$, whichever comes later. The two accelerations after projection within the range of starting and ending points correspond to forearm movement and are employed to calculate the distance and direction of the forearm movement. And we set the hand position at the starting point as the origin of both coordinate systems.

The distance calculation is trivial. We consider the movements in both x and y axes bounded by their starting and ending points. As the distance is two times integration of accelerations, we apply the trapezoidal rule to approximate the distance on each axis.

We employ the acceleration data for the x and y axes and the results of distance calculation to do direction derivation. From the comparison of the positions of $[a_{x,max}, a_{x,min}, a_{y,max}, a_{y,min}]$, we can derive the direction range. The whole 360° can be split into eight direction ranges (start from the x axis). 0° to 45° is range 1, 45° to 90° is range 2, and so on. So if the position of $a_{y,max}$ is before $a_{y,min}$, it means the direction angle sits in 0° to 180°. If the position of $a_{x,max}$ is before $a_{x,min}$, it means the direction angle sits in 270° to 90°. With these two comparisons, we can locate the direction angle into a 90° range. Then we compare the distance obtained from distance calculation. If the absolute value of distance along the x axis is larger than the absolute value of distance along the y axis, it means the direction angle is either in 315° to 45° or 135° to 225°. Combining this

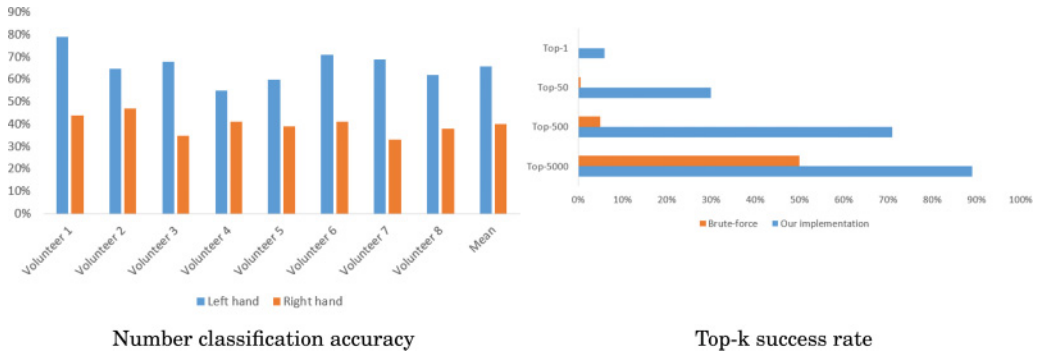


Fig. 8. Evaluation on inferring PIN sequence.

comparison with former comparisons, we can locate the direction angle into a 45° range, which can be used to differentiate between consecutive keystroke pairs like "f" "v" and "f" "b."

5 EXPERIMENTS AND RESULTS

We conduct experiments on eight volunteers, and all the participants are between 20 and 40 years old, including three women and five men. All the volunteers have the ability to type in words following the standard type method (Ratatype 2016) fluently. All participants are instructed to type or tap as they usually do. The participants are also instructed to avoid huge body movements and keep the wrist always above the desk when typing words. They are instructed to hold the iPad with both hands and tap with thumbs when unlocking the screen.

5.1 Evaluation Matrices

We develop the following metrics to evaluate our system.

Classification accuracy. To evaluate the performance of the classifier, we define classification accuracy as the possibility of correct classification. The ground truth is recorded by us during the experiments.

Top-k success rate. Given an experimental run of a password or PIN activity, our algorithm could return multiple top candidates of password or PIN sequence. We define that a top-k success hit if the password or PIN resides in the returned k candidates list.

5.2 Implementation and Evaluation

In the experiment to infer PIN, we ask the volunteers to tap in "0" to "9" each for 20 times for two rounds. The data gathered from the first round is used to train the SVM classifier while the data collected in the second round is used to do the testing. Then, the volunteers are instructed to type in different length−4 PIN sequences. We adopt the input event detection subsystem to extract the timestamps of starting points and it fully detected all the movements. We start by evaluating the performance of the number classification subsystem. There is one SVM classifier for each volunteer trained by the their own labeled samples. The classification accuracy for each volunteer is shown in Figure 8(a). We have also explored whether it is possible to generate a general classifier for all volunteers. However, the general classifier achieves classification accuracy close to random guess. The reason is that the EMG signals collected from volunteers relate to the structure of the volunteers' muscle and every volunteer has a different muscle structure.

We can observe that a big difference in the number detection accuracy between the left hand and right hand exists. This is because people tend to type in "1" "4" "7" "8" "0" with the left hand

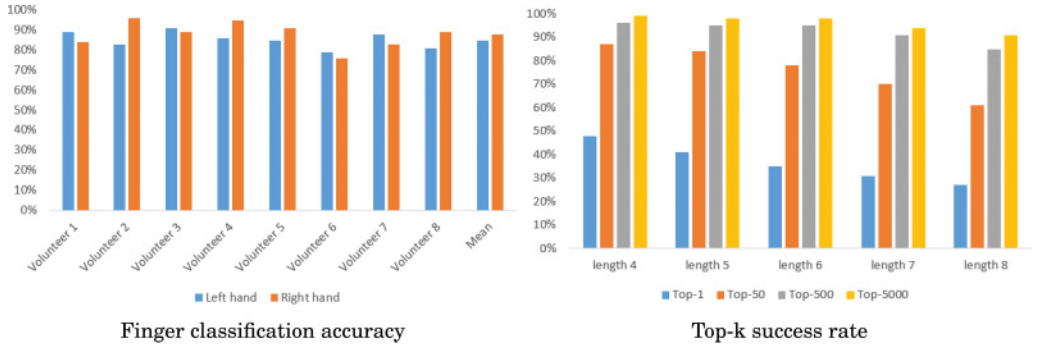


Fig. 9. Evaluation on inferring passwords.

and the others with the right hand. And the numbers touched by the right hand are close to each other. The way we use to generate a candidate is basically by replacing classification outcomes one by one. For example, if the ground truth is "5709" but the classifier give us "5749," then the top-4 candidate list will be "0749" "5049" "5709," and "5740." So we have recovered the right typing in the top-3 candidate list. Figure 8(b) presents the top-k success rate of the PIN sequence reconstruction compared to simple brute-force.

In the experiment to recover passwords, we ask the volunteers to type in "a" "s" "d" "f" "j" "k" "l" ";" each for 20 times for classification, and another 20 times for testing. Then, we ask volunteers to type in multiple passwords. The construction of the passwords includes lowerase letters, uppercase letters, numbers, and symbols. We first evaluate the performance of the finger classification subsystem. An input event detection subsystem is employed here and it has 100% accuracy. There is one SVM classifier for each volunteer trained by their own labeled samples. The classification accuracy for each volunteer is shown in Figure 9(a).

Basically, one way we use to generate a candidate is to replace one classification outcome one by one. For example, if the ground truth is "see," which is {ring finger, middle finger, middle finger} but the classifier gives us {ring finger, index finger, middle finger}, then the top-3 candidate list will be {pinky finger, index finger, middle finger}, {ring finger, middle finger, middle finger}, and {ring finger, index finger, index finger}. So we have recovered the right typing in the top-3 candidate list. The other factor we use to generate the candidate list is from the coordinate transformation subsystem. The coordinate transformation subsystem can make it easy to differentiate situations like "r" and "v." Figure 9(b) presents the success rate of the password reconstruction.

6 FURTHER EXPLORATION: PASSWORD EXTRACTION WITHOUT PRIOR INFORMATION

In the supervised implementation, we assume our application includes an initialization phase, during which a user is instructed to do a series of tapping actions. We make a further effort to make our attack even more stealthy and practical. We get rid of the training phase in our unsupervised implementation. The assumption reduces to that the application instead only needs to have the ability to record the EMG sensor data of the user typing in an article.

6.1 Extracting User Muscle Models without Labeled Data

The difference between supervised and unsupervised implementation lies in the finger classification subsystem. Other than that, the procedure is the same. In the finger classification subsystem for unsupervised implementation, it uses a model obtained from unsupervised learning.

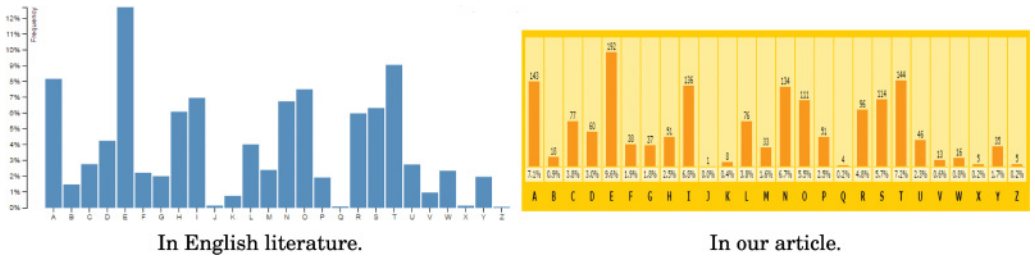


Fig. 10. Comparison of relative frequencies in English literature and in our article.

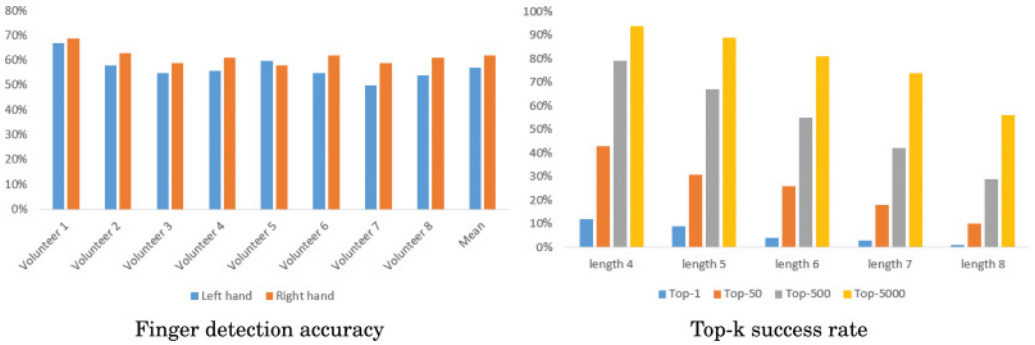


Fig. 11. Evaluation on unsupervised typing model.

We record the whole process of a user typing an article and then we apply k-means clustering method with principal component analysis (PCA) to all the unlabeled samples collected from the article. With the letter frequency analysis of English text, the subsystem can know which cluster corresponds to which finger. For example, if one of the letters "a" "q" "z" appears in the article, it means one keystroke with the little finger. And according to the letter frequency ranking of the sum of "a" "q" "z" in English text, we can know the correspondence between finger and cluster because the cluster with the greatest amount of samples relates to the finger used most frequently in English text typing. The frequency analysis of English text is shown in Figure 10(a) (Lewand 2000). In comparison, Figure 10(b) is the frequency analysis of the article we used in our experiment. After training, the classifier could give us which finger a new signal segment is related to.

6.2 Experiments and Results

As usual, we evaluate the performance of the finger classification subsystem first. We ask the volunteers to type in an article with 2,000 letters. Then, we ask volunteers to type in the same passwords as in the first implementation. The EMG data gathered is put into the input event detection subsystem. We implement a small experiment here to test the performance of our input event detection subsystem. We extract the signal segment of 20 letters from each volunteer and combine the signal segments together. Then we put the composite signal into our input event detection subsystem. It turns out that if we set the maximum false-positive rate to be 5%, our subsystem could detect 84% of the events. With the timestamps, we can extract the keystroke samples. We adopt k-means clustering to the unlabeled samples to find the centroids. The finger classification accuracy for each volunteer is shown in Figure 11(a). It is worth mentioning here that thumb movement is not required when recovering passwords. So whenever the sample

segment to be classified is close to the centroid for the thumb, we label it as index finger. We apply the same way as in the first implementation to generate a candidate list and Figure 11(b) presents the top-k success rate of the password reconstruction.

7 MITIGATION AND LIMITATION

7.1 Mitigation

Given the severity of these information leakages, it would be important for future AR system designers to take in the information leakage of input devices as a consideration in actual deployment. There are two research directions to mitigate the side-channel information leakage.

First, we can enforce the access control of the Myo raw signal. We can apply the FlowFence programming framework proposed in Fernandes et al. (2016). FlowFence presents a new information flow model. A data-publishing app (which controls the raw EMG signal source) tags the EMG data with a taint label. Developers write data-consuming apps (like a health monitoring app) so that EMG data is only processed within designated functions that run in FlowFence-provided sandboxes for which taints are automatically tracked. Therefore, an app consists of a set of designated functions that compute on EMG data, and code that does not compute on EMG data. FlowFence only makes EMG data available to apps via functions that they submit for execution in FlowFence-provided sandboxes. So an app cannot directly access the raw EMG data but through functions and the track of its usage of processed EMG data will be recorded by taint labels. A trusted service process in the background will terminate any suspicious information flow of the app according to the taint analysis. In the example provided in their paper, they show they can prevent raw camera data from flowing to the Internet. With light modification, it can be applied to prevent raw EMG data from flowing to the remote server.

The second research direction aims to obfuscate the finger movement from the signal. A naive approach is to lower the sampling rate. Currently, the EMG signals are sampled and broadcast in maximum sampling frequency for the equipment (myo 2016). However, depending on gestures to distinguish, it might not be necessary to sample at such frequency to provide basic gesture differentiation. Furthermore, it is also possible to add calculated noise to the signal to make finger movement undetectable while in the meantime, gesture detection is still available. Privacy-preserving techniques such as differential privacy (Dwork 2006) can also be applied to the problem to quantitatively degrade the signal.

7.2 Limitation

The information leakage demonstrations presented in this work present the feasibility of exploiting the EMG side-channel information to recover sensitive passwords and PINs of AR users. However, there remains some practical challenges when our method is applied in different environments. For example, our scheme is affected by the surface of the skin of the user. If there is a significant amount of hair on the user's arm, the signal-to-noise ratio could still be good enough for coarse-grained information, which is necessary for Myo armband to function. But the addition of this noise could significantly hinder the modeling of the finger movement and finger detection. Despite the reduction in the information of the channel, our demonstrations can still be applied to reduce the space of brute-force attack. In addition, the performance of our proposed scheme would increase with the development sensor technologies.

8 CONCLUSION

In this work, we study a new kind of side-channel information leakage on gesture-control devices in AR. By taking advantage of the EMG and acceleration signals available on the wearable device,

we are able to recover passwords from keyboard and user login PIN on a touch screen device. To demonstrate the information leakage, we address unique challenges in using the EMG signal. More specifically, we invent new movement detection algorithms based on DCS to reliably detect movement events of fingers. In our demonstration with unsupervised learning, we avoid the assumption of labeled sensor data which makes our implementation stealthy. At the same time, it is able to recover complex passwords constructed with lowercase letters, uppercase letters, numbers, and symbols with a mean success rate of 56% in the first 5,000 trials. Furthermore, our method with supervised learning is able to achieve a mean success rate pf 91% in the same settings. Through experiment data recorded with volunteers, we show that our implementations can be applied to users from different age and gender groups. Lastly, we provide a discussion on the possible mitigation to this kind of information leakage. Through the successful demonstrations based on the new EMG sensor, we further emphasize the importance of understanding potential abuse of sensor data type with the coming of various kinds of sensors.

REFERENCES

Apple. 2017. Apple Watch Series 2—Technical Specifications. (2017). https://support.apple.com/kb/SP746?locale=en_US.

Yousef Al-Assaf. 2006. Surface myoelectric signal analysis: Dynamic approaches for change detection and classification. *IEEE Transactions on Biomedical Engineering* 53, 11 (2006), 2248–2256.

Kamran Ali, Alex X. Liu, Wei Wang, and Muhammad Shahzad. 2015. Keystroke recognition using WiFi signals. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 90–102.

Dmitri Asonov and Rakesh Agrawal. 2004. Keyboard acoustic emanations. In *Proceedings of the IEEE Symposium on Security and Privacy*, Vol. 2004. 3–11.

Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. 2001. Recent advances in augmented reality. *IEEE Computer Graphics and Applications* 21, 6 (2001), 34–47.

Myo Blog. 2015. Jake Sims is a straight-up wizard. Retrieved from http://developerblog.myo.com/featured-dev-jake-sims/.

Bloomburg. 2016. Goldman Sachs has four charts showing the huge potential in virtual and augmented reality. Retrieved from http://www.bloomberg.com/news/articles/2016-01-13/goldman-sachs-has-four-charts-showing-the-huge-potential-in-virtual-and-augmented-reality.

Davide Balzarotti, Marco Cova, and Giovanni Vigna. 2008. Clearshot: Eavesdropping on keyboard input from video. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE, 170–183.

Mourad Barkat. 2005. *Signal Detection and Estimation*. Artech House.

John V. Basmajian and C. J. De Luca. 1985. Muscles alive. *Muscles Alive: Their Functions Revealed by Electromyography* 278 (1985), 126.

Yigael Berger, Avishai Wool, and Arie Yeredor. 2006. Dictionary attacks using keyboard acoustic emanations. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*. ACM, 245–254.

CNET. 2016. Microsoft's HoloLens is super limited—and hella magical. Retrieved from https://www.cnet.com/products/microsoft-hololens-hands-on/.

Liang Cai and Hao Chen. 2011. TouchLogger: Inferring keystrokes on touch screen from smartphone motion. *HotSec* 11 (2011), 9–9.

Thomas P. Caudell and David W. Mizell. 1992. Augmented reality: An application of heads-up display technology to manual manufacturing processes. In *Proceedings of the 25th Hawaii International Conference on System Sciences, 1992*, Vol. 2. IEEE, 659–669.

Francis H. Y. Chan, Yong-Sheng Yang, F. K. Lam, Yuan-Ting Zhang, and Philip A. Parker. 2000. Fuzzy EMG classification for prosthesis control. *IEEE Transactions on Rehabilitation Engineering* 8, 3 (2000), 305–311.

Charles K. Chui. 2014. *An Introduction to Wavelets*. Vol. 1. Academic Press.

Enrico Costanza, Andreas Kunz, and Morten Fjeld. 2009. Mixed reality: A survey. In *Human Machine Interaction*. Springer, 47–68.

Carlo J. De Luca, Alexander Adam, Robert Wotiz, L. Donald Gilmore, and S. Hamid Nawab. 2006. Decomposition of surface EMG signals. *Journal of Neurophysiology* 96, 3 (2006), 1646–1657.

Cynthia Dwork. 2006. Differential privacy. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming, Part II (ICALP'06)*.

Wassim El Falou, Mohamad Khalil, and Jacques Duchene. 2000. AR-based method for change detection using dynamic cumulative sum. In *Proceedings of the 7th IEEE Internattional Conference on Electronics, Circuits and Systems (ICECS)*, Vol. 1. 157–160.

Kevin Englehart, Bernard Hudgins, Philip A. Parker, and Maryhelen Stevenson. 1999. Classification of the myoelectric signal using time-frequency based representations. *Medical Engineering & Physics* 21, 6 (1999), 431–438.

Fitbit. 2017. Fitbit Specs. Retrieved from https://www.fitbit.com/surge#specs.

Craig L. Fancourt and Jose C. Principe. 2000. On the use of neural networks in the generalized likelihood ratio test for detecting abrupt changes in signals. In *IJCNN (2)*. 243–252.

Kristin A. Farry, Ian D. Walker, and Richard G. Baraniuk. 1996. Myoelectric teleoperation of a complex robotic hand. *IEEE Transactions on Robotics and Automation* 12, 5 (1996), 775–788.

Steven Feiner, Blair MacIntyre, Tobias Höllerer, and Anthony Webster. 1997. A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. *Personal Technologies* 1, 4 (1997), 208–217.

Earlence Fernandes, Justin Paupore, Amir Rahmati, Daniel Simionato, Mauro Conti, and Atul Prakash. 2016. Flowfence: Practical data protection for emerging IoT application frameworks. In *Proceedings of the USENIX Security Symposium*.

Olivia A. Grigg, V. T. Farewell, and D. J. Spiegelhalter. 2003. Use of risk-adjusted CUSUM and RSPRTcharts for monitoring in medical contexts. *Statistical Methods in Medical Research* 12, 2 (2003), 147–170.

Bernard Hudgins, Philip Parker, and Robert N. Scott. 1993. A new strategy for multifunction myoelectric control. *IEEE Transactions on Biomedical Engineering* 40, 1 (1993), 82–94.

Chuck Jorgensen, Diana D. Lee, and Shane Agabont. 2003. Sub auditory speech recognition based on EMG signals. In *Proceedings of the International Joint Conference on Neural Networks, 2003*, Vol. 4. IEEE, 3128–3133.

S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karuturi Radha Krishna Murthy. 2001. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation* 13, 3 (2001), 637–649.

Mohamad Khalil and Jacques Duchêne. 1999. Dynamic cumulative sum approach for change detection. *IEEE Transactions on Signal Processing* 47, 4 (1999), 1205.

Mohamad Khalil and Jacques Duchêne. 2000. Uterine EMG analysis: A dynamic approach for change detection and classification. *IEEE Transactions on Biomedical Engineering* 47, 6 (2000), 748–756.

Robert Lewand. 2000. *Cryptological Mathematics*. MAA.

Mengyuan Li, Yan Meng, Junyi Liu, Haojin Zhu, Xiaohui Liang, Yao Liu, and Na Ruan. 2016. When CSI meets public WiFi: Inferring your mobile phone password via wifi signals. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1068–1079.

Xiangyu Liu, Zhe Zhou, Wenrui Diao, Zhou Li, and Kehuan Zhang. 2015. When good becomes evil: Keystroke inference with smartwatch. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1273–1285.

Myo. 2016. Homepage. Retrieved from https://www.myo.com/.

Federico Maggi, Simone Gasparini, and Giacomo Boracchi. 2011. A fast eavesdropping attack against touchscreens. In *Proceedings of the 2011 7th International Conference on Information Assurance and Security (IAS)*. IEEE, 320–325.

Anindya Maiti, Oscar Armbruster, Murtuza Jadliwala, and Jibo He. 2016. Smartwatch-based keystroke inference attacks and context-aware protection mechanisms. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. ACM, 795–806.

Steve Mann. 1997. Wearable computing: A first step toward personal imaging. *Computer* 30, 2 (1997), 25–32.

Elaine Nicpon Marieb and Katja Hoehn. 2007. *Human Anatomy & Physiology*. Pearson Education.

Philip Marquardt, Arunabh Verma, Henry Carter, and Patrick Traynor. 2011. (sp) iPhone: Decoding vibrations from nearby keyboards using mobile phone accelerometers. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*. ACM, 551–562.

Emiliano Miluzzo, Alexander Varshavsky, Suhrid Balakrishnan, and Romit Roy Choudhury. 2012. Tapprints: Your finger taps have fingerprints. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*. ACM, 323–336.

Oussama Mustapha, Dimitri Lefebvre, Ghaleb Hoblos, Houcine Chafouk, and Mohamad Khalil. 2008a. *Fault Detection Algorithm Based on Filters Bank Derived from Wavelet Packets*. INTECH Open Access Publisher.

Oussama Mustapha, Dimitri Lefebvre, Mohamad Khalil, Ghaleb Hoblos, and Houcine Chafouk. 2008b. Filters bank derived from the wavelet transform for real time change detection in signal. In *Proceedings of the 3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA'08)*. IEEE, 1–6.

George Papagiannakis, Gurminder Singh, and Nadia Magnenat-Thalmann. 2008. A survey of mobile and wireless technologies for augmented reality systems. *Computer Animation and Virtual Worlds* 19, 1 (2008), 3–22.

Ratatype. 2016. Learn how to touch type. Retrieved from http://www.ratatype.com/learn.

DC Rainmaker. 2017. Hands-on: Garmins New Fenix 5 Multisport GPS Series with mapping! Retruecved from https://www.dcrainmaker.com/2017/01/hands-on-garmins-new-fenix-5-multisport-gps-serieswith-mapping.html.

Rahul Raguram, Andrew M. White, Dibyendusekhar Goswami, Fabian Monrose, and Jan-Michael Frahm. 2011. iSpy: Automatic reconstruction of typed input from compromising reflections. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*. ACM, 527–536.

Samsung. 2016. [In-Depth Look] The Parts and Pieces that Make the Gear S3 Tick. Retrieved from https://news.samsung.com/global/in-depth-look-the-parts-and-pieces-that-make-the-gear-s3-tick.

Diksha Shukla, Rajesh Kumar, Abdul Serwadda, and Vir V. Phoha. 2014. Beware, your hands reveal your secrets!. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 904–917.

Ivan E. Sutherland. 1968. A head-mounted three dimensional display. In *Proceedings of the December 9–11, 1968, Fall Joint Computer Conference, Part I.* ACM, 757–764.

G. Tsenov, A. H. Zeghbib, F. Palis, N. Shoylev, and V. Mladenov. 2006. Neural networks for online classification of hand and finger movements using surface EMG signals. In *Proceedings of the 2006 8th Seminar on Neural Network Applications in Electrical Engineering*. IEEE, 167–171.

D. W. F. Van Krevelen and R. Poelman. 2010. A survey of augmented reality technologies, applications and limitations. *International Journal of Virtual Reality* 9, 2 (2010), 1.

Martin Vuagnoux and Sylvain Pasini. 2009. Compromising electromagnetic emanations of wired and wireless keyboards. In *Proceedings of the USENIX Security Symposium*. 1–16.

Chen Wang, Xiaonan Guo, Yan Wang, Yingying Chen, and Bo Liu. 2016. Friend or foe?: Your wearable devices reveal your personal PIN. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. ACM, 189–200.

He Wang, Ted Tsung-Te Lai, and Romit Roy Choudhury. 2015. Mole: Motion leaks through smartwatch sensors. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 155–166.

Zhi Xu, Kun Bai, and Sencun Zhu. 2012. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 113–124.

Feng Zhou, Henry Been-Lirn Duh, and Mark Billinghurst. 2008. Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, 193–202.

Tong Zhu, Qiang Ma, Shanfeng Zhang, and Yunhao Liu. 2014. Context-free attacks using keyboard acoustic emanations. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 453–464.

Li Zhuang, Feng Zhou, and J. Doug Tygar. 2009. Keyboard acoustic emanations revisited. *ACM Transactions on Information and System Security (TISSEC)* 13, 1 (2009), 3.

Received November 2016; revised February 2017; accepted April 2017