VR-Spy: A Side-Channel Attack on Virtual Key-Logging in VR Headsets

Abdullah Al Arafat*
Electrical and Computer Engineering
University of Central Florida

Zhishan Guo†
Electrical and Computer Engineering
University of Central Florida

Amro Awad‡
Electrical and Computer Engineering
NC State University

Abstract

In Virtual Reality (VR), users typically interact with the virtual world using virtual keyboard to insert keywords, surfing the webpages, or typing passwords to access online accounts. Hence, it becomes imperative to understand the security of virtual keystrokes. In this paper, we present VR-Spy, a virtual keystrokes recognition method using channel state information (CSI) of WiFi signals. To the best of our knowledge, this is the first work that uses WiFi signals to recognize virtual keystrokes in VR headsets. The key idea behind VR-Spy is that the side-channel information of fine-granular hand movements associated with each virtual keystroke has a unique gesture pattern in the CSI waveforms. Our novel pattern extraction algorithm leverages signal processing techniques to extract the patterns from the variations of CSI. We implement VR-Spy using two Commercially Off-The-Shelf (COTS) devices, a transmitter (WAVLINK router), and a receiver (Intel NUC with an IWL 5300 NIC). Finally, VR-Spy achieves a virtual keystroke recognition accuracy of 69.75% in comparison to techniques that assume very advanced adversary models with vision and motion sensors near the victim.

Index Terms: Human-centered computing—Gesture Computing—Virtual Key-logging Attack—Channel State Information

1 Introduction

Virtual reality (VR) technology is experiencing rapid growth, especially in hardware and software technologies. Initially, VR was mainly used for playing video games and watching 3D videos. The evolution of VR technologies has emerged in new applications and objectives. The advancement of VR has now enabled various interactive applications, such as healthcare, military, education, industry prototyping, social networking, etc., in the immersive virtual environment. Although the virtual environment is not fully immersive yet, it is expected to be fully immersive shortly and will be widely used in sophisticated and safety-critical applications. For instance, the US Army has already created an augmented reality-based Synthetic Training Environment (STE) to train its soldiers with millions of artificially intelligent agents [13]. A virtual reality-based driving simulator is typical to observe the users' behavior in different driving conditions [22].

In immersive virtual environments, users have to interact with the virtual world. All of the emerging and future applications of VR involve lots of confidential and safety-critical information. For instance, a VR user has to insert keywords for surfing web pages, passwords to access private accounts, credit card or bank account information for online purchases or shopping, etc. In military training, lots of soldiers' specific information, particular weaknesses or strengths (e.g., acrophobia), are involved in the VR. These pieces of information have specific characteristics while performing in a virtual environment. The performance counters (e.g., power consumption, frequency spectrum, current/voltage consumptions, etc.) of VR have different statistics for different VR activities. Similarly, the VR user's movements associated with the VR activities have distinguishable properties. For instance, the VR users' body movements for different VR activities, such as video games, driving simulations, military training, etc., have unique body movement patterns. So, performance counters or VR users' body movements are potential side-channel information to infer the VR user activities.

The VR side-channel information poses a security threat against confidential and personal information. For instance, in adversarial attacks, any leakage of specific weaknesses or strengths of a particular army member could be critical in military operations. Similarly, personal and financial information theft would create a catastrophic situation for individual life. Several works demonstrate that it is possible to identify the VR user's identity based on biometric data collected from different VR sensors [9, 17]. Fake user authentication through different imaging techniques is also possible in a virtual environment [28]. Consequently, several countermeasures for user authentication threats have been proposed in preexisting literature [10, 18].

In contrast, the side-channel attacks on virtual keystrokes in VR headsets are relatively less explored. Attacks on virtual keystrokes can leverage different side-channel information based on key-logging procedures. In VR, the key-logging process includes a virtual keyboard and a pair of hand controllers or hand trackers. The virtual keystrokes usually involve two steps: target and select keys using hand controllers. Each virtual keystroke is related to a unique hand gesture of the user. The only previous work [16] on virtual keystrokes recognition of VR by Z. Ling et al. proposed vision-based and motion sensors (e.g., accelerometer, gyroscope, and magnetometer, etc.) based virtual keystrokes recognition methods exploiting the VR user activities. Vision-based approaches are fundamentally limited as the vision-based gesture recognition techniques depend on the line-of-sight (LOS) with enough lighting. Moreover, vision sensors breach user privacy, so the user would be reluctant to install the vision sensors. From an adversary's perspective, data acquisition from wearable sensors and built-in sensors is difficult as the attacker has to install malware or hardware trojan on those devices. In comparison with physical keystrokes, virtual keystrokes are out of scope for several attacks, such as acoustic sound [29], electromagnetic emission, finger trace [7], etc., based attacks.

In this paper, we present VR-Spy, a virtual keystrokes recognition method using channel state information (CSI) of WiFi communication. There are two accessible WiFi signal properties from commodity WiFi devices — received signal strength indication (RSSI) and channel state information (CSI). Both RSSI and CSI can be utilized to detect or localize moving objects and/or activities in the range of WiFi signals. Therefore, localization [27] and activities detection [6] using WiFi signals have gained great attention from the research community due to the pervasiveness of WiFi communication in indoor locations. VR-Spy utilizes the CSI of WiFi communication to detect the fine granular hand gestures for virtual keystrokes. Then, using a unique hand gesture for each specific key, VR-Spy recognizes the keystrokes. CSI-based activity detection methods are device-free methods, which means there is no need to install any

*e-mail: abdullah.arafat@knights.ucf.edu
†e-mail: zsguo@ucf.com
‡e-mail: ajawad@ncsu.edu

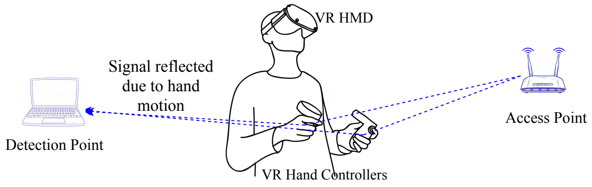

Figure 1: Illustration of VR-Spy

external devices or sensors, unlike vision and sensors-based activity detection methods.

VR-Spy aims to infer keystrokes in the virtual environment of VR. The experimental setup of VR-Spy consists of two Commercially Off-The-Shelf (COTS) devices, an access point, and a detection point to build up a WiFi communication link. Figure 1 illustrates a scenario of VR-Spy where the victim (VR user) is located in between the communication link. The CSI value of the communication link is collected from the detection point. The activities of the VR user in the range of the communication link distort the WiFi signal, and so introduce distinct patterns in CSI values. VR-Spy utilizes the patterns to recognize the activities leveraging the signal processing techniques. Initially, VR-Spy detects the user's hand gestures associated with virtual keystrokes from the CSI stream. Then, it recognizes the keystrokes, comparing the detected gestures with the offline-generated database of unique signatures for each character and digit. VR-Spy is more practical and threatening than the attacks using sensors' information and vision techniques as the later attacks need to install external devices in the victim's environment.

The CSI analysis of virtual keystroke recognition is challenging for several reasons. *First*, unlike the physical keyboard, a virtual keyboard can be easily scaled, and the distance of the keyboard from the user is adjustable based on user preferences. In this paper, we assumed that the virtual keyboard would be fixed in shape and distance from the user whenever the user starts typing on the keyboard. With this assumption, the prior adjustment of the shape and position of the keyboard would only introduce scaling distortion in the gesture patterns. We have used the dynamic time-warping distance metric to mitigate the scaling distortion effect. *Second*, VR user, could quickly move around in the room scale selected by the user at the beginning of the virtual environment setup. So, the user movement introduces macro activities in addition to the hand gesture (micro-activities) for the keystrokes. We proposed a five-step hand gesture detection algorithm in the presence of macro activities. *Third*, the gesture patterns in the CSI stream would change for different users and environments, and hence, the database for the keystrokes would be changed. To address this challenge, we include both temporal and frequency domains of signal components during feature extraction using the time-frequency domain analysis technique, dynamic wavelet transform.

The contributions of our paper are the following:

- We introduced a novel side-channel attack on virtual key-logging in VR based on user activities using CSI of pervasive WiFi signal.
- We proposed a novel five-step hand gesture detection algorithm leveraging the signal processing techniques in the presence of macro activities.
- We developed an end-to-end virtual keystroke recognition model from offline fingerprint generation to online keystroke recognition. We achieved a reasonable keystroke recognition accuracy of 69.75% in comparison to the existing vision and motion sensors-based methods.
- Finally, we exhaustively evaluate our design for different VR users to validate the robustness of the attack model.

2 Background and Motivation

WiFi is one of the most used wireless communication technologies in the wireless local area network (WLAN). The pervasiveness of WiFi communication draws attention from the research community to utilize WiFi to solve various human activity recognition and indoor localization problems. The two most used attributes of WiFi communication are the received signal strength indicator (RSSI) and channel state information (CSI). CSI of the WiFi signal enables one to recognize fine-granular activities.

Channel State Information: CSI in wireless communication represents the known channel properties (e.g., channel gain, channel phase shift) of a communication link. It includes the combined effect of scattering, multi-path fading, and the power decaying of the communication signal that propagates from the transmitter to the receiver. We have built CSI based fingerprint for different VR activities. To collect CSI information, we have implemented RF communication with IEEE 802.11n standards, which support multiple-input multiple-output (MIMO) antennas for the transmitter-receiver pairs. In MIMO, the Orthogonal Frequency Division Multiplexing (OFDM) modulation technique is used to communicate data packets in different sub-frequency bands (narrow bands). In MIMO OFDM communication techniques, the narrow-bands channel (subcarrier) is modeled as $\mathbf{y} = h \times \mathbf{x} + \mathbf{n}$ where $\mathbf{x}$, $\mathbf{y}$ are transmitted and received signal respectively, $\mathbf{n}$ is the channel noise usually modeled as circular symmetric complex normal with $\mathbf{n} \sim c.N(0,S)$, and h is the complex-valued channel frequency response (CFR). The CFR h can be estimated as [21]:

$$\hat{h} \approx \frac{y}{x} \tag{1}$$

CSI of a single sub-carrier is a complex number, $h_i = |h_i| e^{jsin\theta}$ where $|h_i|$ is the channel gain and θ is the channel phase of that sub-carrier. So, the overall CSIs of the communication link are formed as follows:

$$\hat{H} = [\hat{h}_1, \hat{h}_2, \hat{h}_3, \dots, \hat{h}_{N_c}] \tag{2}$$

$\hat{H}$ represents the state of the channel and hence approximate PHY layer CSI over multiple sub-carriers. The dimension of $\hat{H}$ is $N_c \times N_t \times N_r$ matrix where N_c is the number of sub-carriers, N_t is the number of transmitter antennas, and N_r is the number of receiver antennas.

Motivation: In VR, virtual key-logging involves user hand movements in the virtual environment. For each keystroke, the user has to move her hand and wrist uniquely. The movements of hands and wrists have six degrees of freedom with three scaling and three rotational components. The user's hand gestures are potential side-channel information to recognize the keystrokes due to the involvement of hand gestures in the virtual key-logging. Several methods (e.g., vision, wearable sensors, performance counters, etc.) were proposed by the researchers to exploit the hand gesture information to recognize the keystrokes. In this paper, we proposed a novel method to utilize hand gesture-based side-channel information to recognize the keystrokes. We observed that the hand gesture for each keystroke has a unique pattern in the CSI stream. Figure 2 shows two distinct patterns for two different keystrokes in the CSI stream. Although the pattern duration for each keystroke may vary, each pattern has enough distinguishable features to recognize the keystrokes. VR devices usually connect to WiFi to get access to the internet, but the CSI of WiFi communication is entirely orthogonal to the information of VR devices. So, the key-logging side-channel attack exploiting the CSI of the WiFi signal is more practical and easily applicable to these scenarios.

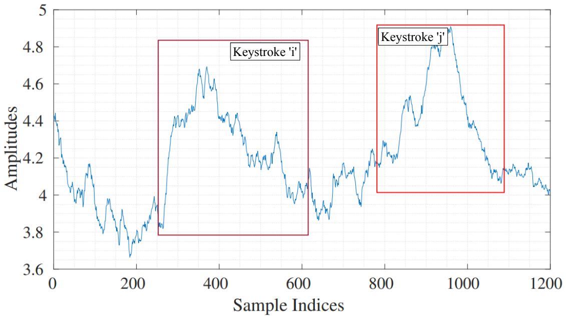

Figure 2: Distinct patterns for different keystrokes

3 Threat Model and Assumptions

In VR-Spy, we consider a WiFi communication link available in the victim's place, and the attacker can access the WiFi access point to get the CSI values. As the victim's activities can be inferred from the CSI values, the attacker does not need to have access to the VR devices. We assume the adversary does **not** install any malware or hardware trojan in the VR device to monitor the behavior of the device's performance counters. Yet the attacker has prior knowledge of the virtual keyboard layout of the VR device—such information is typically publicly available online.

In addition, we make the following assumptions about the VR user to infer the keystrokes:

1. In VR, the size and position of the virtual keyboard are often adjustable—one can change the position and size of the virtual keyboard, depending on his/her preferences. However, it is very unlikely that a user will adjust the position or size of the virtual keyboard while inserting text in a virtual environment. Hence, VR-Spy assumes that the user would not change the keyboard's position or size during the key-logging period in the virtual environment.

2. The virtual key-logging consists of two steps: target and select a keystroke. In the case of targeting a keystroke, almost every VR device uses an optical pointer. For the selection of targeted keystrokes, the hand controllers of VR have either a touchpad or buttons. VR-Spy assumes that after the completion of the target keystroke, the user must select the key using the hand controller's touchpad or buttons. However, VR users can intentionally avoid selection after targeting a key, and those cases are out of the scope of VR-Spy.

4 Framework of VR-Spy

Our virtual keystrokes recognition system, VR-Spy, is a wireless system utilizing the CSI of WiFi communication. Figure 3 shows the workflow of the VR-Spy. The VR-Spy workflow consists of three parts: the CSI data acquisition from commercial network interface cards, processing, and fingerprint formation of the target gestures for virtual keystrokes offline, and the online inference phase to recognize the keystrokes.

In the first part, we used a transmitter and a receiver to build up the WiFi communication link. The transmitter is working as an access point, and the receiver as a detection point. There are two and three antennas in the AP and DP, respectively. VR-Spy collects CSI values from the detection point. CSI values are inherently noisy and hence require the preprocessing of CSI values to remove the noises. We developed a three-step preprocessing module to remove the outliers, high-frequency components, and signal passband noises.

The second part of VR-Spy is the extraction of the gestures and the fingerprint formation using the gestures for each virtual keystroke. In the VR-Spy system, the transmitter continuously sends packets to the receiver. Then, the receiver further samples the received packet to gain fine-granular information about the channel. The CSI matrix for the entire packet sequences becomes $N \times N_c$ matrix for each transmitter-receiver antenna link, where N is the number of data packets, and N_c is the number of subcarriers. We developed a five-step hand gesture detection algorithm from the preprocessed CSI matrix, considering the presence of VR user movements in real scenarios.

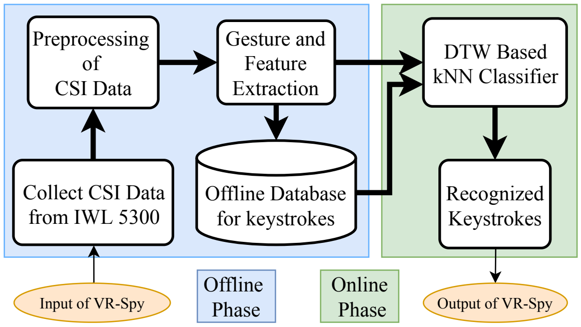
Preprocessing of CSI Data
Gesture and Feature Extraction
DTW Based kNN Classifier
Collect CSI Data from IWL 5300
Offline Database for keystrokes
Recognized Keystrokes
Input of VR-Spy
Offline Phase
Online Phase
Output of VR-Spy

Figure 3: The Workflow of VR-Spy

The final part of VR-Spy is the inference phase. The inference phase recognizes the keystrokes online utilizing the extracted feature vectors of each gesture. In this part, we used a classifier to recognize each gesture to a specific keystroke. The input of the classifier is the feature vectors. Dynamic Wavelet Transform is performed on the feature vectors before feeding to the classifier to reduce the feature vector's dimensions. Instead of Euclidean distance, we used dynamic time warping to measure the distance between the classifier's features and the fingerprints.

4.1 Preprocessing of CSI Stream

The CSI streams from the network interface cards are inherently noisy, and noise levels for different subcarriers are also varied significantly [26]. Noises in the CSI stream can be categorized into three classes. The *first class* of noise is the outlier in the CSI stream, which is introduced by the abrupt adaptation of transmission signal power, transmission rate, etc. It is essential to remove outliers before using any filter operation due to the filers' unexpected behavior in the region of interest. The *second class* of noise is the out-of-band frequency noises. In VR-Spy, the frequency band of the hand gesture lies at the lower end of the frequency spectrum due to the slower motion of the hands required for virtual keystrokes. So, for the VR-Spy, the high-frequency components are the noisy signal in the CSI stream, and hence, VR-Spy employs a low-pass filter to remove the high-frequency components. The *third class* of noise in the CSI stream is the noise involving the signal's passband. Signal smoothing techniques are most suitable to mitigate the passband noises. VR-Spy utilizes the weighted moving average filter to reduce the effect of passband noises.

4.1.1 Outliers Removal

Outliers are one of the dominant noise components in the CSI streams. The source of outliers in the CSI streams is the transient adaptation of transmission power, transmission rate, etc., of the transmitted signal. Hampel outliers' identifier is a well-known outliers removal filter and removes only outliers rather than any desired signal components [20]. Hence, we applied the Hampel identifier filter to remove the outliers from the CSI stream. The working principle of the Hampel identifier filter is similar to the local median filter. Anything out of $[\mu - \gamma\sigma, \mu + \gamma\sigma]$ range is considered as outlier where μ is the median of the local data points, σ is the deviation from the median, and γ is the adjustment factor of the data windows

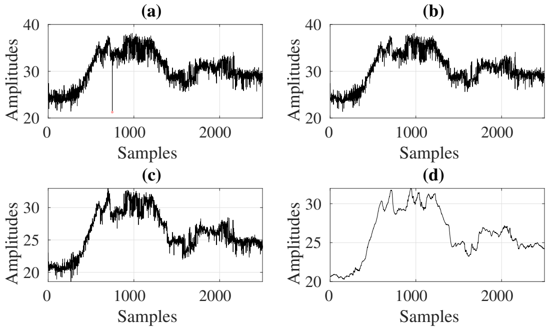

Figure 4: Preprocessed CSI stream: (a) A raw CSI stream, (b) CSI stream after removing the outliers, (c) CSI stream after applying low-pass filter, and (d) CSI stream after applying the weighted average filter. The CSI stream is taken from OFDM subcarrier 27.

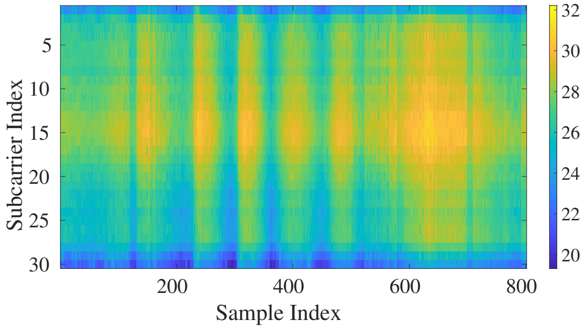

Figure 5: CSI stream patterns for different gestures in subcarriers

(typically $\gamma = 3$). Figure 4(a) shows a raw CSI stream with outliers, and Figure 4(b) plots the CSI stream after the removal of outliers.

4.1.2 Low-Pass Filter

The desired frequency components of the CSI stream lie at the low end of the frequency spectrum due to the hand and wrist motion associated with virtual keystrokes, and the undesired CSI stream lies at the high end of the frequency spectrum. The low-pass filter's desired properties are the flat frequency responses in both the passband and stopband of the filter to avoid any attenuation in the passband signal amplitude and complete rejection of the stopband signal components. *Butterworth* low-pass filter is the most suitable candidate to meet the requirements. In VR-Spy, we implement the Butterworth loss-pass filter with a cutoff frequency, $w_c = \frac{2\pi f}{F_s}$, where f is the frequency of the desired hand gesture and F_s is the sampling frequency of the CSI stream. Figure 4(c) plots the resultant CSI stream from the Butterworth low-pass filter applied on the CSI stream without outliers.

4.1.3 Weighted Moving Average

After applying the low-pass filter, some noises are still present in the passband of the CSI stream, Figure 4(c). A smoothing filter needs to be applied to remove the passband noise. We adopt a weighted moving average filter with m previous data points on the low-pass filtered signal:

$$h_{t_i} = \frac{[m.h_{t_i} + (m-1).h_{t_i-1} + \cdots + h_{t_1-m+1}]}{m + (m-1) + \cdots + 1} \tag{3}$$

In Equation 3, we used linearly decreasing weights for the previous data points to emphasize the present signal value. Figure 4(d) plots the resultant weighted moving averaged signal of the low-pass filtered CSI stream.

4.2 Virtual Keystrokes Extraction

The preprocessing step removes all unwanted signal components from the CSI waveforms. Here, we discuss the hand gestures extraction from the waveforms in detail. To extract the hand gestures from CSI streams, we need to find the starting and ending points in the waveforms. The gestures in the CSI waveforms are distributed in all sub-carriers. Besides, the VR users could have other activities along with the hand gestures associated with virtual keystrokes. Hence, we need to develop an efficient and robust algorithm to extract the gestures in the presence of other activities. In VR-Spy, we implement the following five-step algorithm to extract virtual keystrokes from the CSI stream.

PCA of Normalized CSI Stream. The preprocessed CSI stream from Section 4.1 has thirty sub-carriers for each transmitter-receiver pair link. The gesture for a virtual keystroke spreads in all of the sub-carriers. Figure 5 plots the heat map of keystroke gestures where the variation of amplitudes over the sub-carriers are distinguishable. However, the CSI stream with all thirty sub-carriers has a high computational cost due to the high volume of data. As the sub-carriers highly correlate them, the waveform matrix can map into a lower dimension space without compromising important information. To reduce the CSI matrix dimension without compromising vital information, we employed principal component analysis (PCA) based dimension reduction on the CSI matrix. Before applying PCA, we normalized the CSI stream with zero mean, and unit variance as PCA is not scale-invariant [5]. We observed that the first four principal components represent the majority (e.g., at least 90% for any CSI matrix) of information. In PCA, the principal components are arranged in descending order of variance. As the variance of macro activities is higher than the variance of micro activities, we exclude the first principal component, representing the maximum variance. Besides, the first principal component also includes most of the system noises. Hence, VR-Spy utilizes only three principal components (second to fourth components) for the subsequent steps.

MAD of Sliding Windows. Instead of the sample-by-sample comparison of the CSI waveform, we implement a sliding window technique to analyze samples in the granularity of window length. The advantage of the window technique is that the comparison of the samples has a larger memory bandwidth than the instantaneous comparison. VR-Spy calculates the mean absolute deviation (MAD) for the windows. To reduce the impact of extreme points, VR-Spy implements MAD instead of the variance, which squares the extreme deviations. The equation 4 calculates the MAD of the waveforms.

$$\begin{aligned} H_p^{MAD}(i,j) = \sum_{k=w(i-1)+1}^{i\times w} |H_p(k,j) - \\ \frac{1}{w} \times \sum_{k=w(i-1)+1}^{i\times w} H_p(k,j)| \end{aligned} \tag{4}$$

where $i = 1,2,3,\ldots,\lfloor \frac{N}{w} \rfloor$, and $j = 1,2,3$. $H_p \in \mathbb{R}^{N\times 3}$ is the principal components of the CSI matrix, w is the length of the window. After the computation of H_p^{MAD}, VR-Spy takes a single waveform for each transceiver link by adding the principal components, $H^{MAD}(i) = \sum_{j=1}^{3} H_p^{MAD}(i,j)$.

Weighted Average of Windowed Waveform. In this step, VR-Spy applied a weighted moving average filter (Equation 3) on the waveform generated in the `step 2` to smooth the waveform.

Detection of Terminal Points. We observed that keystroke gestures form increasing and decreasing patterns in the terminal points. So, we find out the peaks in the waveform generated from `step 3` and choose an empirical threshold value (e.g., in our implementation, we choose 0.05 as the threshold value) for the peak prominence. Any peaks less than the threshold of the peak prominence are discarded. Then, we select the starting and ending points, assuming that peaks are the middle points of the gesture pattern. In practice, peaks are not always in the middle. So, we add guard samples on both sides of the pattern to avoid missing any gesture segment from the pattern. We empirically choose the gesture pattern bandwidth and the number of guard samples.

Gesture Extraction. In this step, we convert the extracted terminal points of the gesture pattern to the original coordinates of the CSI stream. Then, the gesture pattern from the second, third, and fourth principal components is extracted using the terminal points. We use the average of three patterns from three PCA waveforms for the gesture patterns.

In each transceiver link, we extract a single waveform for each keystroke. In VR-Spy, we extract the gesture pattern from all communication links between the transmitter and receiver. Then, the waveforms for a keystroke are padded together to form a single waveform.

4.3 Feature Extraction

To classify the extracted gesture patterns into keystrokes, we need to extract features from the patterns. The patterns for different hand gestures are closely related, and the statistical properties, such as energy, moments, mean, RMS values, etc., have minor variations. So the statistical features are not sufficient enough to classify the patterns into keystrokes. To classify the patterns successfully, we need to consider the patterns' whole shape as the features. Due to the length of patterns, all patterns are computationally inefficient in training the classifier. We employed a dynamic wavelet transform (DWT) on the waveform of patterns to get both temporal and spatial information. And then, we choose the scaling and detail coefficients of DWT as the features.

Discrete wavelet transform of a waveform $x[n]$ is defined based on approximation coefficients, $\lambda_\phi[j_0,k]$, and detail coefficients, $\gamma_\psi[j,k]$, as follows [12]:

$$\lambda_\phi[j_0,k] = \frac{1}{\sqrt{L}} \sum_n x[n]\phi_{j_0,k}[n] \tag{5}$$

$$\gamma_\psi[j,k] = \frac{1}{\sqrt{L}} \sum_n x[n]\psi_{j,k}[n], \quad \text{for } j > j_0 \tag{6}$$

and the inverse DWT is given by:

$$\begin{aligned} x[n] = \frac{1}{\sqrt{L}} \sum_k \lambda_\phi[j_0,k]\phi_{j_0,k}[n] + \\ \frac{1}{\sqrt{L}} \sum_{j=j_0}^{J} \sum_k \gamma_\psi[j,k]\psi_{j,k}[n] \end{aligned} \tag{7}$$

where $n = 1,2,3,\ldots,L-1$, $j = 1,2,3,\ldots,J-1$, and $k = 0,1,2,\ldots,2^j-1$, and L represents the length of the waveform, $x[n]$. The basis functions $\phi_{j,k}[n]$ and $\psi_{j,k}[n]$ are defined as:

$$\phi_{j,k}[n] = 2^{j/2}\phi[2^j n - k] \tag{8}$$

$$\psi_{j,k}[n] = 2^{j/2}\psi[2^j n - k] \tag{9}$$

where $\phi_{j,k}$ is the scaling function and $\psi_{j,k}$ is wavelet function or mother wavelet. In VR-Spy, we use `Daubechies D4` [23] as the scaling function and wavelet function in our implementation.

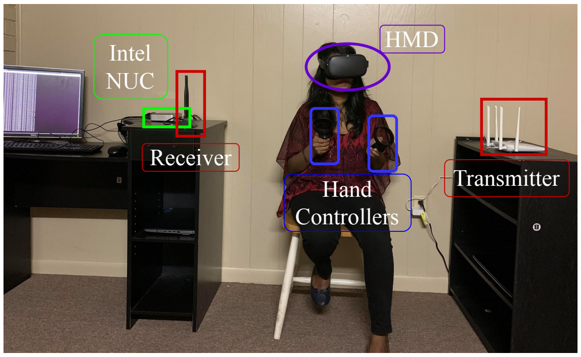

Figure 6: Experimental setup of VR-Spy

4.4 Classification

After extracting the DWT-based features of gesture patterns, we build a classifier based on the features. We choose an ensemble of k-nearest neighbor (kNN) classifiers using features of the patterns. We used dynamic time warping distance metric for the classifiers. The reason behind using dynamic time warping is that the patterns for different gestures and different users often vary. Dynamic time warping can measure the optimal distance between two distorted waveforms and hence mitigate the duration and environment variation-related distortion.

Dynamic time warping estimates minimum distance alignment between any two-time series by dynamic programming. In contrast to Euclidean distance, dynamic time warping provides the distance of two time-dependent waveforms, $X = (x_1, x_2, \ldots, x_n)$ of length $n \in \mathbb{N}$ and $Y = (y_1, y_2, \ldots, y_m)$ of length $m \in \mathbb{N}$ and allows non-linear mapping of one waveform to another. Dynamic time warping distance is the Euclidean distance of the optimal warping path between two waveforms calculated under the resilient of lengths and shifts [19].

5 Implementation & Evaluation

5.1 Hardware Setup

VR-Spy consists of two commercially off-the-shelf (COTS) hardware devices in wireless local area networks (WLAN). We used an access point and a detection point to form the WiFi communication link in the experimental environment. We have employed WAVLINK dual-band WiFi router [4] with two transmitting antennas as an access point (transmitter), and Intel NUC [1] with Intel WiFi link 5300 network interface card [2] with three receiving antennas as a detection point (receiver). The communication link is operating in the IEEE standard of 802.11n with a channel width of 20MHz at channel 157. WiFi communication has two operating bands—2.4G and 5G band. In our experiment, we used WiFi with a 5G band instead of the 2.4G band to avoid interference from other ISM band communication links, such as Bluetooth communication. To collect CSI values from the Intel 5300 network interface card, we used a modified driver developed by Halperin et al. [11]. Linux 802.11n CSI tool reports $N_c = 30$ sub-carriers for each transmitter-receiver pair. The access point (WAVLINK-N router) and detection point (Intel 5300 NIC) have $N_t = 2$ antennas and $N_r = 3$ antennas, respectively. Hence, the dimension of the overall CSI matrix, $\hat{H}$ (Equation 2) in our experiment is $30 \times 2 \times 3$. In the CSI matrix, there are six transmitter-receiver pairs; therefore, there are six different CSI streams.

For the experiment of virtual keystrokes in a virtual environment, we utilized the Oculus Quest VR headset [3], which is the latest VR headset with advanced features. Figure 6 shows the experimental setup of VR-Spy where the VR user (Victim) is placed in between the direct link of the transmitter and receiver of the WiFi communication

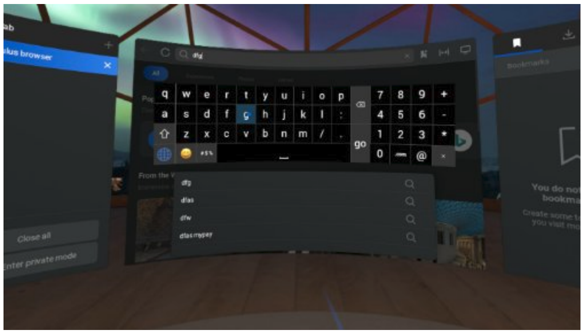

Figure 7: Layout of virtual keyboard in VR environments (Screenshot taken from the Oculus Quest VE)

link. The distance between the transmitter and the receiver is 50 inches.

Oculus Quest is an all-in-one VR, which means it includes all the necessary hardware in the headset. Unlike other VR devices, Oculus Quest does not need to connect to any external devices to run the applications in a virtual environment. Oculus Quest has a head-mounted display (HMD) and two-hand controllers. In the headset, there are four cameras in four corners to calibrate the real world coordinate with virtual worlds, and also these cameras are used to avoid any hazardous collision with the real-world when the user is immersed in the virtual world. The hand controllers include several buttons instead of a touchpad. The hand controllers are used to target and select virtual objects in the virtual environments using an optical pointer and buttons. Both the hand controllers and the HMD are working as input devices in VR. These input devices help the VR user to interact with the virtual world. Oculus Quest is a WiFi-compatible device, and hand controllers are connected to the headset through an RF communication (Bluetooth) link. Besides, the headset can connect to the external device through both Bluetooth and USB connectors. Hence, the headset and the externally connected device can install/update the software and firmware of VR.

Virtual Keyboard: VR includes a virtual keyboard for the text input in virtual environments. The user can write anything with the help of the virtual keyboard and the hand controllers. A screenshot of Oculus Quest's virtual keyboard layout is shown in Figure 7. Like a regular keyboard, there are 26 letters, 10 digits, space, and enter keys on the keyboard. The user writes in virtual environments using an optical pointer (e.g., one of the hand controllers' buttons generates an optical pointer) to target a character and then select the character using the hand controllers' press buttons. So, for each keystroke, the user has to move his/her hand with a unique hand gesture. VR-Spy aims to detect hand gestures and, hence, recognize virtual keystrokes. Compared with a physical keyboard, the usual user's keystroke rate would be less in the case of the virtual keystroke. So, the duration for each gesture will be lager than the gesture duration of physical keystrokes.

Data Collection: We have collected the experimental data in two different locations—lab and home. In the experimental setup, we set up WiFi communication links between the WiFi router and the attacker device without an internet connection in the WiFi router (in other words, we avoid data communication). To minimize external interferences, we did not allow any moving objects but VR users in line-of-sight of communication links. The experiments are conducted for ten participants. The voluntary participant's ages range from 25 years to 32 years, including variant sex, heights, weights, and race.

5.2 Results

To evaluate VR-Spy, we design text strings from `26 letters` and `ten digits`. We select the text strings so that no characters are repeatable in a string, and the overall number of appearances of each character in the strings is equal. We randomly shuffle all 36 characters and divide the whole string into six texts with a length of six characters. For each experiment, we repeat the text generation procedure three times. We did ten experiments for ten different users following similar procedures. So, the dataset for evaluation of VR-Spy consists of ten different users' data samples, and each character in the dataset appears thirty times. All the experimental data was collected from a controlled experimental environment with the least possible interference. VR-Spy trains a kNN classifier with ten-fold cross-validation. So, the entire dataset is split into ten partitions, and in each iteration, one partition is considered as a test dataset. The union of the remaining partitions is considered as a training dataset.

Performance Metric: The keystroke recognition involves two significant steps: the gesture extraction of each keystroke and then classifying the gesture into a keystroke. The following equation defines the gesture pattern extraction accuracy.

$$Accuracy_{Gesture} = 1 - \frac{|G_{present} - G_{extracted}|}{G_{present}} \tag{10}$$

where $G_{present}$ and $G_{extracted}$ represent the number of the gesture patterns present in the CSI waveform and the number of the extracted gesture patterns from the waveform, respectively. The classification accuracy is defined as:

$$Accuracy_{Classification} = \frac{N_{correct}}{N_{total}} \tag{11}$$

where N_{total} and $N_{correct}$ represent the total predictions and total correct predictions respectively. The overall keystroke recognition accuracy is the resultant of these two dependent steps. We define the keystroke recognition accuracy as: $Accuracy_{Recognition} = Accuracy_{Gesture} \times Accuracy_{Classification}$.

Tuning Parameters: Here, we mentioned the major hyperparameters associated with the design of VR-Spy.

- **Sampling Frequency:** The sampling frequency of the CSI stream plays a vital role in terms of the resolution of gesture patterns in the CSI stream. We select sampling frequency, $F_s = 2000$ samples/second following [26].

- **Cut-off Frequency of Low-pass Filter:** The frequency components of the CSI stream for keystroke gestures lie below 100 Hz. Hence, we choose the cut-off frequency of the low-pass filter as $w_c = \frac{2\pi f}{F_s} = \frac{2\pi \times 100}{2000} = 0.31$, where $f = 100Hz$ is the frequency of the desired gesture and $F_s = 2000$ samples/second is the sampling frequency of the CSI stream.

- **Length of Weighted Moving Average Filter:** We heuristically choose the weighted moving average filter length. We find the weighted average filter's desired performance for a length of $m = 30$ in our implementation.

5.2.1 Key-logging Detection

Before starting to execute VR-Spy to recognize virtual keystrokes, we have to detect whether the VR user is writing something in the virtual environment or not. Wang et al. [26] showed that the frequency distributions of CSI waveforms for various activities, such as walking, running, sitting, and no actions, differ significantly. Similarly, we also investigate the frequency distribution of CSI waveforms for various virtual activities. We consider three VR activities to simplify the experimental design: watching videos, playing video games, and key-logging. The first two VR activities, playing games and watching 3D or 360° videos in the virtual environment, are popular and frequently used VR applications. The user behavior during these two VR activities varies depending on the content of

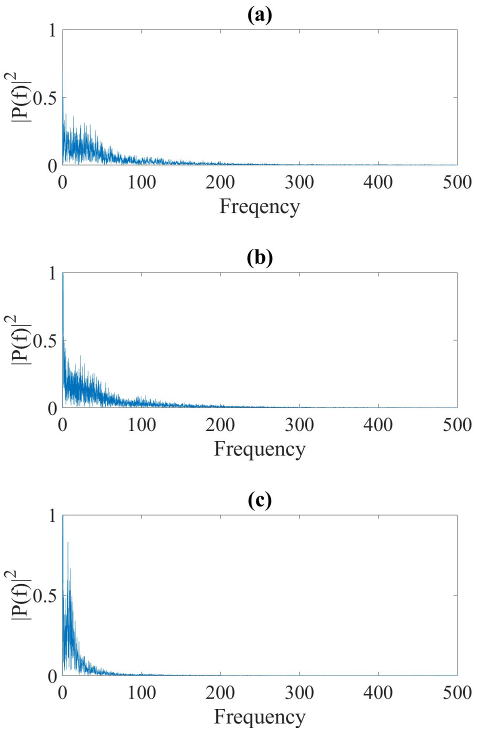

Figure 8: PSD of second principal components of CSI matrix for (a) key-logging, (b) gaming, and (c) 3D video activities

videos or games. The third activity, virtual key-logging, is not a specific VR application; instead, it may occur with any applications that require searching or inserting any text. Apparently, the least user movements would be associated with watching videos, and the highest activities with gaming.

We analyze the second principal component of the preprocessed CSI matrix for consistency with the VR-Spy framework instead of analyzing the raw CSI matrix. We estimate the fast Fourier transform (FFT) based power spectrum density (PSD) for all three VR activities. Figure 8 shows the PSD of VR activities. The PSD of games and key-logging activities is similar but has a different energy level. Table 1 shows the energy of the CSI stream for VR activities. The energy is calculated from the PSD of the waveforms, $\sum_f PSD(f)$. We find that the CSI stream's energy for key-logging activities is between 90 and 95.

5.2.2 Individual Keystroke Recognition Accuracy

We evaluate the performance of VR-Spy for each character. VR-Spy fused all user data into a single dataset to estimate individual character recognition accuracy. Then, we trained the classifier for 36 characters. Figure 9 plots the accuracy of individual keystroke recognition accuracy. The minimum accuracy is 59.8% for the keystroke '5', and the maximum accuracy is 76.18% for the keystroke '9'. The average accuracy of the keystrokes is 69.75%. The average accuracy of virtual keystroke recognition is reasonable with the vision and motion sensors-based keystroke recognition methods.

Table 1: Energy of CSI stream for different activities

Activity	Energy
Key-logging	91.40
Games	133.71
Videos	80.68

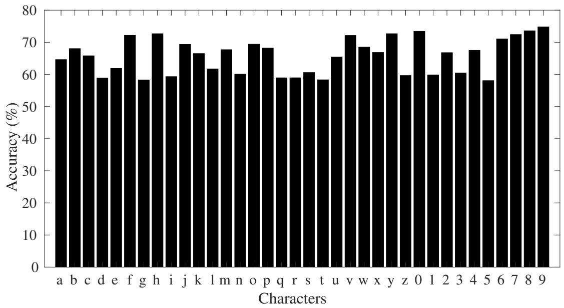

Figure 9: Individual keystroke recognition accuracy averaged over all users

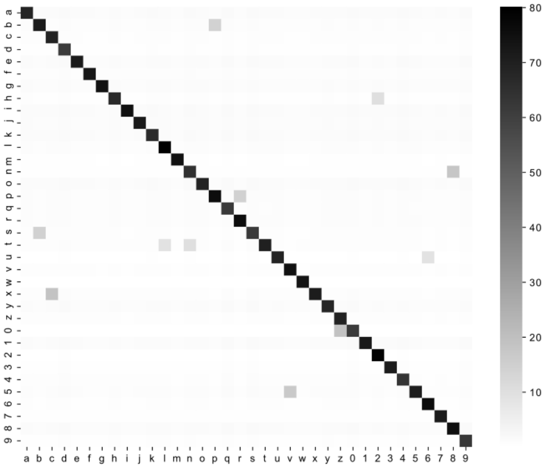

Figure 10: Color map of a confusion matrix for individual character recognition averaged over all users

Figure 10 displays the color map of the confusion matrix for the individual keystrokes. The color map shows a prominent diagonal, and the rest of the points have approximately uniformly distributed small weights. So, the model does not predict false positives for any biased character. The confusion matrix is also averaged over all users' data.

5.2.3 Individual User's Accuracy

To evaluate the robustness of VR-Spy for different users, we experiment with VR-Spy for ten different users. We separately trained VR-Spy for all ten users. The performance differs for the different users as the gesture patterns are often varied in duration and motion for different users. Figure 11 plots the average keystroke detection accuracy for ten different users. The variance of the accuracy among the users is 19.17.

6 Related Works

Attacks at VR. Recently, VR has become a target of attackers from various perspectives. Researchers explore the possible attack surface to find out the vulnerabilities associated with VR technologies. Several researchers address security concerns of authentication methods [9, 10, 17, 18, 28]. George et al. [10] investigate the security of authentication methods in HMD-based VR systems. The authors find out that the PIN and pattern-based authentication methods in VR can be more resistant to the attacker than the methods applied to the smartphone in the physical world by evaluating the resistance to shoulder surfing attacks via human eyes. In *Oculock*, Luo et al. [17] proposed an authentication system utilizing the human visual system

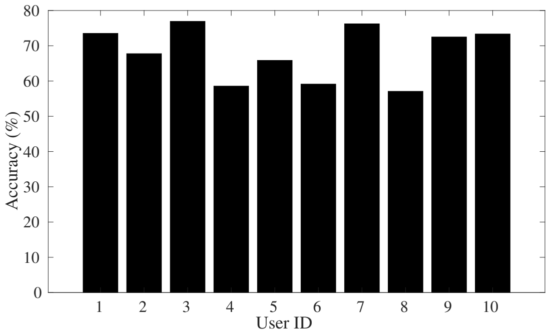

Figure 11: Keystrokes recognition accuracy for individual users averaged over all characters

rather than the PIN or pattern-based methods. The paper explained that human visual system-based authentication methods are more suitable in VR with HMD than smartphones and PCs. However, there are limited preexisting works on the security of virtual key logging in VR. In the only prior work, Ling et al. [16] presented computer vision-based and motion sensor-based models to exploit the side channel information of users' activities to infer passwords for a VR device. The models are based on the Samsung Gear VR headset. There are two ways to target and select keys in the virtual keyboard: a) the controller touchpad is used for both targeting and selecting the keys, b) the headset is used for targeting, and the controller touchpad is used for selecting the keys. In the computer vision-based attack model, a stereo camera is used to record the VR user's activities, especially the headset and controller touchpad motions. The input text is then inferred from the target and selection actions achieved by the headset and the controller. In the motion sensor-based attack model, a malware or hardware trojan is installed in the victim's device to obtain motion sensor readings. Then, the input text is inferred using pre-computed rotation angles on the assumption that the size and position of the virtual keyboard are always fixed.

Gesture computing using WiFi. A wide range of human-computer interaction and mobile computing problems have been extensively studied using the WiFi signal over the last decade. In [6,15], physical keystrokes and mobile phone password recognition methods were developed, recognizing fine-granular finger gestures. Compared with the physical keystrokes recognition attack in [6], in VR-Spy, the user can adjust the size and position of the virtual keyboard before starting key-logging—such uncertainty makes the attack on virtual keystrokes much harder than attacking a user with a physical fixed-position-and-size keyboard. Fu et al. [8] developed an air handwriting recognition method for VR devices leveraging the CSI of WiFi communication. Li et al. [14] proposed an American Sign Language (ASL) detection method to communicate with smart devices for physically challenged people. Moreover, researchers explore lots of other sophisticated fine-granular information-based problems such as speech recognition [24], gait recognition [25], etc., utilizing CSI. However, to the best of our knowledge, there is no prior work on the virtual key-logging attack in VR headsets leveraging the CSI signal variation.

7 Discussion

In this section, we will discuss the critical hyper-parameters and limitations of VR-Spy and point out potential future research directions.

Keyboard Layout. The gesture patterns for different keystrokes are related to the virtual keyboard layout. In VR-Spy, we used the default keyboard layout of `Oculus Quest`, the most common virtual and physical keyboard layout. It is possible to design a customized keyboard layout, and in those cases, VR-Spy cannot be implemented directly.

Antennas. The CSI signal strength inversely depends on the distance between transmitter and receiver antennas. In the evaluation of VR-Spy, we set the distance between transceivers to 50 inches ($4.167 feet$). The performance will vary if the distance between the antennas is changed. The types of antennas, such as omnidirectional and directional antennas with different gains, directly affect the CSI signal strength. We installed omnidirectional antennas to simulate the real environment (mostly used WiFi antenna) in our experimental setup. The performance of VR-Spy might increase with high-gain directional antennas.

Frequency Band. WiFi communication technology has two operating frequency bands, 2.4G and 5G. There are many other ISM band communication technologies in the 2.4G band, so the WiFi signal at 2.4G has lots of interference. The signal strength of the 5G band is also better than the 2.4G band.

Sampling Rate of CSI. The sampling rate of the CSI stream affects the gesture resolution of keystrokes, e.g., how fast a user can move his/her hand to input a key. Besides, the minimum time gap between two consecutive keystrokes is also related to the sampling rate of CSI. So, the time resolution of keystrokes will increase with the higher CSI sampling rate. A higher sampling rate will increase the virtual keystroke detection and recognition accuracy from the CSI stream. Following the discussion of the sampling rate of CSI for fine-granular activity detection in [26], we choose 2000 samples per second as the sampling rate of the CSI stream.

Dataset. The ways of offline database formation have an effect on the adaptability of the attack model in robust attack scenarios. We chose all the strings with similar lengths to ease the complexity of the database development in the VR-Spy experiment setup. Although our database is working well enough in the inference phase with variable string lengths, the attack model would perform better for a database with different string lengths. Besides, the size of the database has an impact on the performance of the model. A dataset with large samples increases the accuracy of the model. In our case, we collect thirty samples per character in the database.

Adaptability. Although VR-Spy targets VR devices, the workflow of VR-Spy is easily adaptable to any virtual keystroke recognition, including augmented reality (AR) devices.

8 Conclusion

In this paper, we presented a novel human-activity-based side-channel attack, VR-Spy, at virtual reality devices to infer the text inputs. VR-Spy exploits the variations in channel state information (CSI) of pervasive indoor WiFi signals to detect and recognize the micro activities associated with virtual keystrokes. The attack consists of a novel five-step hand gesture detection algorithm to extract the virtual key-logging gestures from the variations of the CSI stream. VR-Spy leverages signal processing techniques to extract the gesture patterns from the CSI waveforms and machine learning algorithms to recognize the keystrokes from gesture patterns. Implementation of VR-Spy consists of two commercially-off-the-self (COTS) devices, a transmitter (WAVLINK Router), and a 50-inch apart receiver (IWL 5300 NIC) to build the WiFi communication link. The victim, the VR user, is placed in between the transmitter and receiver. VR-Spy has obtained an average virtual keystroke recognition accuracy of 69.75%. In future work, we will consider a more relaxed threat model allowing moving objects with the victim and will investigate the attack in unknown keyboard layouts.

Acknowledgments

Most of this work was completed and funded when Abdullah was working as a PhD student under the supervision of Amro Awad at UCF. Amro Awad is currently with the ECE Department at NC State University. This work is also partially supported by NSF grant CNS-1850851.

References

[1] Intel® nuc kit d54250wyk. https://ark.intel.com/content/www/us/en/ark/products/76977/intel-nuc-kit-d54250wyk.html, 2020.

[2] Intel® ultimate n wi-fi link 5300. https://www.intel.com/content/www/us/en/products/docs/wireless-products/ultimate-n-wifi-link-5300-brief.html, 2020.

[3] Oculus quest. https://www.oculus.com/quest/, 2020.

[4] Wavlink-ac. https://www.wavlink.com/en_us/product/WL-WN579G3.html, 2020.

[5] H. Abdi and L. J. Williams. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459, 2010.

[6] K. Ali, A. X. Liu, W. Wang, and M. Shahzad. Keystroke recognition using wifi signals. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, p. 90–102. Association for Computing Machinery, New York, NY, USA, 2015. doi: 10.1145/2789168.2790109

[7] S. P. Banerjee and D. L. Woodard. Biometric authentication and identification using keystroke dynamics: A survey. *Journal of Pattern Recognition Research*, 7(1):116–139, 2012.

[8] Z. Fu, J. Xu, Z. Zhu, A. X. Liu, and X. Sun. Writing in the air with wifi signals for virtual reality devices. *IEEE Transactions on Mobile Computing*, 18(2):473–484, 2019.

[9] M. Funk, K. Marky, I. Mizutani, M. Kritzler, S. Mayer, and F. Michahelles. Lookunlock: Using spatial-targets for user-authentication on hmds. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–6, 2019.

[10] C. George, M. Khamis, E. von Zezschwitz, M. Burger, H. Schmidt, F. Alt, and H. Hussmann. Seamless and secure vr: Adapting and evaluating established authentication systems for virtual reality. NDSS, 2017.

[11] D. Halperin, W. Hu, A. Sheth, and D. Wetherall. Tool release: Gathering 802.11 n traces with channel state information. *ACM SIGCOMM Computer Communication Review*, 41(1):53–53, 2011.

[12] N. Kehtarnavaz. Chapter 7 - frequency domain processing. In N. Kehtarnavaz, ed., *Digital Signal Processing System Design (Second Edition)*, pp. 175 – 196. Academic Press, Burlington, second edition ed., 2008. doi: 10.1016/B978-0-12-374490-6.00007-6

[13] J. Lacdan. Army testing synthetic training environment platforms. https://www.army.mil/article/222722/army_testing_synthetic_training_environment_platforms/, 2019.

[14] H. Li, W. Yang, J. Wang, Y. Xu, and L. Huang. Wifinger: talk to your smart devices with finger-grained gesture. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 250–261, 2016.

[15] M. Li, Y. Meng, J. Liu, H. Zhu, X. Liang, Y. Liu, and N. Ruan. 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*, pp. 1068–1079, 2016.

[16] Z. Ling, Z. Li, C. Chen, J. Luo, W. Yu, and X. Fu. I know what you enter on gear vr. In *2019 IEEE Conference on Communications and Network Security (CNS)*, pp. 241–249, June 2019. doi: 10.1109/CNS.2019.8802674

[17] S. Luo, A. Nguyen, C. Song, F. Lin, W. Xu, and Z. Yan. Oculock: Exploring human visual system for authentication in virtual reality head-mounted display. NDSS, 2020.

[18] F. Mathis, H. I. Fawaz, and M. Khamis. Knowledge-driven biometric authentication in virtual reality. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–10, 2020.

[19] M. Müller. Dynamic time warping. *Information retrieval for music and motion*, pp. 69–84, 2007.

[20] R. K. Pearson, Y. Neuvo, J. Astola, and M. Gabbouj. Generalized hampel filters. *EURASIP Journal on Advances in Signal Processing*, 2016(1):1–18, 2016.

[21] T. F. Sanam and H. Godrich. Fuseloc: A cca based information fusion for indoor localization using csi phase and amplitude of wifi signals. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7565–7569, May 2019. doi: 10.1109/ICASSP.2019.8683316

[22] R. Venkatakrishnan, M. Volonte, A. Bhargava, H. Solini, R. Venkatakrishnan, A. C. Robb, S. V. Babu, K. M. Lucaites, and C. Pagano. Towards an immersive driving simulator to study factors related to cybersickness. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 1201–1202. IEEE, 2019.

[23] C. Vonesch, T. Blu, and M. Unser. Generalized daubechies wavelet families. *IEEE Transactions on Signal Processing*, 55(9):4415–4429, 2007.

[24] G. Wang, Y. Zou, Z. Zhou, K. Wu, and L. M. Ni. We can hear you with wi-fi! *IEEE Transactions on Mobile Computing*, 15(11):2907–2920, 2016.

[25] W. Wang, A. X. Liu, and M. Shahzad. Gait recognition using wifi signals. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 363–373, 2016.

[26] W. Wang, A. X. Liu, M. Shahzad, K. Ling, and S. Lu. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st annual international conference on mobile computing and networking*, pp. 65–76, 2015.

[27] K. Wu, J. Xiao, Y. Yi, D. Chen, X. Luo, and L. M. Ni. Csi-based indoor localization. *IEEE Transactions on Parallel and Distributed Systems*, 24(7):1300–1309, 2012.

[28] Y. Xu, T. Price, J.-M. Frahm, and F. Monrose. Virtual u: Defeating face liveness detection by building virtual models from your public photos. In *25th USENIX Security Symposium (USENIX Security 16)*, pp. 497–512. USENIX Association, Austin, TX, Aug. 2016.

[29] L. Zhuang, F. Zhou, and J. D. Tygar. Keyboard acoustic emanations revisited. *ACM Transactions on Information and System Security (TISSEC)*, 13(1):1–26, 2009.