In the New Era of the Internet of Things Even Your Breathing Rate is Not Private

by

Zihan Chen

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2021

© Zihan Chen 2021

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

This thesis shows a new type of attack that can secretly gather private measurements from the target. Wi-Sneak is a stealthy reconnaissance attack which utilizes the ambient WiFi signal and ubiquitous WiFi-enabled devices without compromising any packets or WiFi networks.

We demonstrated that an adversary can accurately extract people's respiration rate by just sniffing the WiFi network and sending some fake packets no matter outside or inside the house. The deployment of this attack is simple, and anyone with a laptop, a USB WiFi-card and a Micro Controller Unit (MCU) can perform the attack by running the program introduced in this thesis. The attack is tested on various situations and the accuracy has been verified with a solid ground truth. The attack is low-cost, simple to deploy and yet very hard for unprofessional people to detect.

Acknowledgements

I would like to thank Omid Abari and Ali Abedi who made this thesis possible.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The development of IoT devices and WiFi-enabled smart devices are so rapid recently that almost every house is equipped with at least one of them (e.g. smart homes, cell-phones, and smart watches). The existence of these devices has filled the house with overwhelming amount of RF signals. People inside the house are interacting with these signals constantly, and thus these signals carry information about people's locations, movements, and even private measurements like breathing rates. This thesis discovers Wi-Sneak — by sniffing WiFi signals and sending back to back fake packets, an adversary outside the house is able to track the respiration rate of the device user even when the devices are fully secured.

Wi-Sneak can be considered as a reconnaissance attack which is a general knowledge gathering attack usually done by packet sniffing, ping sweeping or port scanning. The attacker can gather some private information from the target. Wi-Sneak can be easily performed by anyone who purchases a WiFi-card and an ESP32 WiFi module which only costs less than a hundred dollars, and it can be very hard for the people in the house to notice. The attacker does not need to decrypt any packet or compromise any device. The only required equipment is a laptop that is connected to the ESP32 WiFi module and the WiFi-card. The reason that makes Wi-Sneak possible lies in the inherent properties of the WiFi signal:

1. People's movement near a WiFi-enabled device will change the signal propagation and then effect the signal strength, and these changes can be easily observed without decipher any device or packet.

2. These signals are able to penetrate walls and floors so that people can receive them at different locations in a house while only one Access Point is needed. However, outside receivers can usually overhear these signals, which makes Wi-Sneak possible.

Wi-Sneak is inspired by an interesting behaviour called *Polite WiFi*, which is discovered by a research [3] that existing WiFi devices send back acknowledgements (ACK) to fake packets received from WiFi devices outside of their network. Following the path of this research, we take a further step to show that this behaviour can be potentially used as an attack to invade privacy. Wi-Sneak does not take any assumption about the WiFi network nor does it need any prior information about the device inside the house. The program will discover the devices to attack, and it will re-pick proper devices if current device is no longer active. The location of the attacker is flexible, it can either be inside or outside the house and the attack distance is around $5 - 8$ m. Moreover, the injected packets are disguised as normal traffic inside the WiFi network, which will not cause any attention for normal people, and this stealthy feature makes it very hard to be noticed.

A prototype of Wi-Sneak is built and evaluated in the thesis, the attacker is a laptop assembled with a USB WiFi card for sending back to back fake packets and a ESP32 WiFi module for receiving acknowledgements. In order to be able to send fake packets, we need to obtain the MAC address of the target device and the SSID of the WiFi Access Point. A common sniffing software like Wireshark can provide all the information we need. After that, the laptop is able to send fake packets to the target device and then extract the breathing rate. However, to ensure the continuity of the data there are two challenges need to be resolved.

How to keep the device awake? Most of the WiFi devices like cell phones and laptops will go to sleep to save energy. During this stage, they will not respond to the any packets and only wake up occasionally to receive beacon frames, and thus the attack cannot take effect. To address this problem, we look into the 802.11 Power Management and find out that the Access Point can monitor all the devices that is in the inactive state and has the ability to wake them up. Therefore, we forge a fake beacon frame and pretend it is from the Access Point to wake up the device we want. This fake beacon frame is sent periodically so that the device is always responsive.

How to process subcarrier information? The ESP32 WiFi module will extract the Channel State Information (CSI) from the acknowledgements sent by the devices. However, the CSI contains 52 subcarriers per packet, and the quality of them varies drastically. Good subcarriers has clear pattern for breathing, while poor ones can barely show the breathing pattern. Since the processing of the acknowledgements need to be real time, an efficient and accurate algorithm is needed to analyze all 52 subcarriers. In this thesis, we propose a soft voting algorithm to efficiently extract useful information from the subcarriers. The algorithm gives each subcarrier weights according to their quality and the final results are based on the weighted voting of all the subcarriers.

In this thesis, Wi-Sneak is evaluated on various aspects including accuracy and effective range. Several experiments are designed to simulate environments including indoor and outdoor scenarios. The results show that the accuracy of the attack can be more than 99%, and the effective range (from target person to target devices) of this attack can be up to 140 cm. In addition, we try to develop some effective defenses against this attack. Although we find a way that may disrupt Wi-Sneak from invading the privacy, it sacrifices a relatively great amount of network throughput.

The rest of the thesis will describe the detailed process of the attack, the challenges being resolved, the evaluations and the defenses. The key contributions of this thesis include:

- We discover a low-cost, stealthy attack which can accurately detect human's respiration rate.
- We propose an algorithm which fully utilizes the information from each subcarrier in Channel State Information (CSI)
- We discover a method which can keep the WiFi-enabled devices awake for packet sending while they are in power-saving mode.
- We propose a possible defense against Wi-Sneak.

For now, Wi-Sneak can only detect well-periodic movements like breathing, and it is not able to identify more elaborate actions such as walking or separate multiple breathing rates from each other. Despite of this, the thesis proposed a practical, low-cost attack that can expose private measurements through ambient WiFi sensing secretly, and the aim of this thesis is to draw more attention into this area so that researches can be done to support the security of people's network.

Chapter 2

Background

Before describing the details of Wi-Sneak, we explain some related background concepts in this chapter.

2.1 WiFi Power Saving mechanism

Wireless communication is very power hungry; therefore, WiFi chipsets spend most of the time in the sleep mode to save power. When a WiFi chipset is in the sleep mode, it cannot send or receive WiFi packets. This creates a technical challenge for Wi-Sneak because it requires a target WiFi device to continuously send packets so that an attacker can analyze the signal to estimate breathing rate. When a WiFi device wants to enter the sleep mode it notifies the WiFi access point so that it buffers any incoming packets for this device. WiFi access points broadcast beacon frames periodically to manage their networks. The beacon frame includes the Traffic Indication Map (TIM) that indicates which devices have buffered packets on the access point. All devices must wake up to receive beacon frames to find out if there are packets waiting for them. When a device has some buffered packets on the access point, it stays awake to receive them. Wi-Sneak takes advantage of the WiFi power saving mechanism to force a target device to stay awake. Details of this mechanism is explained in Chapter 4.2.

2.2 WiFi Channel State Information

The Wi-Sneak attack estimates the breathing rate by analyzing WiFi signals. When we breath, our lungs move only a few centimeters which creates a very small distortion in WiFi signals. Therefore, Wi-Sneak requires an accurate metric to detect and track these small distortions. Coarse grain metrics such as the Received Signal Strength Indicator (RSSI) do not provide enough accuracy to detect breathing rate. This is because RSSI only reports the average received signal power for a packet. On the other hand, the Channel State Information (CSI) reports the amplitude and phase of each OFDM subcarrier. Specifically, for each WiFi packet, CSI provides 52 measurements for amplitude and phase. The channel state information has been shown to be sensitive enough to estimate breathing rate [1, 6]. Previous studies could utilize CSI to estimate breathing rate only in controlled settings in which WiFi devices and the target person are under the control of the experimenter. In contrast, in the Wi-Sneak attack, the transmitter WiFi device and the target person are in a building where the attacker has no access to. This creates new technical challenges that require new solutions as explained in the following chapters.

Chapter 3

Related Work

The attack presented in this thesis is related to existing WiFi sensing techniques. These techniques use WiFi signals to infer some information such as gesture detection to enable a useful application for the user (the good). In addition, Wi-Sneak utilizes some methodologies used in existing attacks against the operating of WiFi networks (the bad). Finally, Wi-Sneak combines some ideas from WiFi sensing and attacks against WiFi networks to create a new privacy attack against users in smart environments (the ugly).

3.1 The Good

WiFi Sensing is a technique that uses ambient WiFi signals to detect events or human activities. The motivation behind WiFi sensing is that we can obtain certain information without dedicated sensors. Instead, WiFi sensing techniques analyzing changes in WiFi signal to infer different types of information. As mentioned earlier in Chapter 2.2 CSI has been shown to be well suited for sensing techniques. For instance, there are applications that can identify the number of people in a closed room and their relative locations[5]. Applications that develop wireless device-free human detection [11, 10] are also implemented to determine the presence of human activities. Other than this, subtle movements like gesture recognition [2] and vital measurement such as respiration rate [1, 6] can also be measured using wireless sensing. These applications are tools that bring convenient to people's lives. However, for this techniques to work WiFi devices should cooperate to enable WiFi sensing. Therefore, how can an attacker perform WiFi sensing when he/she has no access to the target building and the devices inside it?

3.2 The Bad

There are numerous attacks against WiFi networks form denial of service attack for a particular client devices or total disruption of the network [18]. Wi-Sneak utilizes a modified version of these attacks to enable a privacy attack. We now review these techniques.

Beacon Injection: This attack is performed by forging 802.11 beacons and broadcasting them to all devices in a WiFi network. The attacking device pretends to be the actual access point and injects false information in the forged beacons to enable a variety of attacks such as the "evil twin access point" attack. Since beacons are broadcasted to all devices, this will attack all the devices at the same time. The forged beacon frames can also be sent to a particular devices (i.e., unicast) to attack individual devices rather than the entire network.

TIM Forgery: Another attack against WiFi networks is to force client devices not to enter the sleep mode (i.e., WiFi power saving mechanism). Traffic Indication Map (TIM), as mentioned in Chapter 2.1, is used in 802.11 beacon frames. It contains information about whether sleeping devices have buffered packet at Access Point (AP) or not. It is suggested that an adversary can manipulate the Time Indication Map (TIM) inside beacons to change the behavior of WiFi devices [7]. For instance, the adversary can forge the TIM to make a device believe that it has buffered data to receive, so it cannot enter the sleep mode. In contrast, it can also prevent a device from receiving any data by telling it that the AP has no buffered data for it.

3.3 The Ugly

In addition to attacks that sabotage WiFi networks, there are attacks that focus on information gathering, especially about privacy. Wireless signals pass through walls; therefore, an attacker who is outside a building can receive signals coming from inside that building. The attacker can analyze the distortions in WiFi signals, caused by the body of a target person, to infer various types of information from a target building. For example, a recent study shows that by capturing WiFi signals coming out of a private building, it is possible to track user movements inside that building [22]. This is a great threat to the privacy of users in smart environments with numerous WiFi devices.

To make the problem worse, it has been shown that WiFi devices send acknowledgments (ACK) when they receive fake packets coming from outside the their network [3]. This behavior, called *Polite WiFi*, enlightens the possibility of turning any WiFi device into a

secret sensor. This is because the CSI information can be extract from the ACKs send by a victim device to perform WiFi sensing and potentially obtain some sensitive information. Wi-Sneak combines some ideas from WiFi sensing, attacks against WiFi networks, and Polite WiFi to enable a new privacy attack. In this thesis, we show for the first time how an attacker can estimate the respiration rate of target persons inside a building tens of meters away from that building. We also propose potential methodologies to protect against this attack.

Chapter 4

Wi-Sneak

This chapter shows how an attacker can turn WiFi devices into sensors and use them to monitor the breathing rate of people near them. At a high level, the attacker first discovers which WiFi devices exist in an environment. Then, it targets them by sending them back to back packets. The target devices will respond the attacker with ACKs. The attacker then analysis the signal properties of received ACKs and extract the breathing rate of people close to the target devices. In the following we explain each of these steps in more details.

4.1 Discovering WiFi Devices in a Network

To discover which WiFi devices exist in a network, the attacker pretends to be the Access Point (AP) of that network and broadcast fake beacon frames. Note, to do so, the MAC address and the SSID of the network's AP is needed. This can be easily done by sniffing the WiFi traffic using a software such as Wireshark. Once the attacker sends fake beacon frames with the SSID and MAC address of the network's AP, all existing WiFi devices will respond to the attacker. This process enables the attacker to discover the MAC addresses of all WiFi devices in that network. Knowing their MAC addresses enables the attacker to target them individually. Now, the next question is how an attacker can turn a targeted WiFi device to a breathing rate sensor.

4.2 Turning a WiFi Device to a Breathing Rate Sensor

When a person inhales and exhales, her/his chest expands and contracts, respectively. This results in a periodic change in the wireless channel of a WiFi device which is in close proximity of a person. Therefore, if the attacker continuously monitors the signal of the WiFi device, it can discover the breathing rate of the person. However, the issue is that, in practice, WiFi devices are not continuously transmitting. In particular, when a device is is inactive, it barely sends any packet. Therefore their transmission is intermittent and can not be used to sense breathing rate continuously.

To solve this problem, the attacker sends back-to-back packets to the WiFi device, pushing the device to continuously transmitting acknowledgement packets. However, there is still one issue. Most WiFi devices go to sleep to save energy during inactive state such as screen lock. They occasionally wake up to receive the beacon frames, and hence the are mostly in sleep mode. During the sleep mode, the attacker is not able to push them to transmit by sending back-to-back packets. Figure 4.1 show the result of an experiment where the attacker is continuously transmitting fake packets to a WiFi device. In this Figure, we plot the amplitude of CSI over time for the acknowledgement packets received form the WiFi device. As it can be seen, the responses are sparse and discontinued even when the attacker sends back to back packets to the WiFi device.

To keep the target device awake, we found that the attacker can send the target device a forged beacon frame, and claim that it has buffered packets waiting for the target device. Note, this can simply done by alternating bits in TIM as mentioned in Chapter 2.1. This process enables Wi-Sneak to prevent the target device from going to sleep since the target device tries to stay up to receive the buffered packets. Although, sensing fake beacon frames wakes the target device up, sending it very frequently will cause WiFi devices to recognize the attacker's behaviour suspicious and disconnect from it. Therefore, instead of just sending beacon frames back to back, the attacker is continuously transmitting normal packets to a WiFi device and periodically send fake beacon frames to keep it awake. Figure 4.2 shows the result of an experiment where the attacker is continuously transmitting fake packets to a WiFi device and periodically send fake beacon frames. As it can be seen, the target device is continuously awake and responding to fake packets with acknowledgements. Finally, note in this experiment, the device was placed close to a person and therefore a periodic breathing pattern can be seen in the CSI amplitude of the acknowledgment packets responded by the WiFi device. Therefore, Wi-Sneak can easily detect the breathing rate of the person by computing the FFT of this signal.

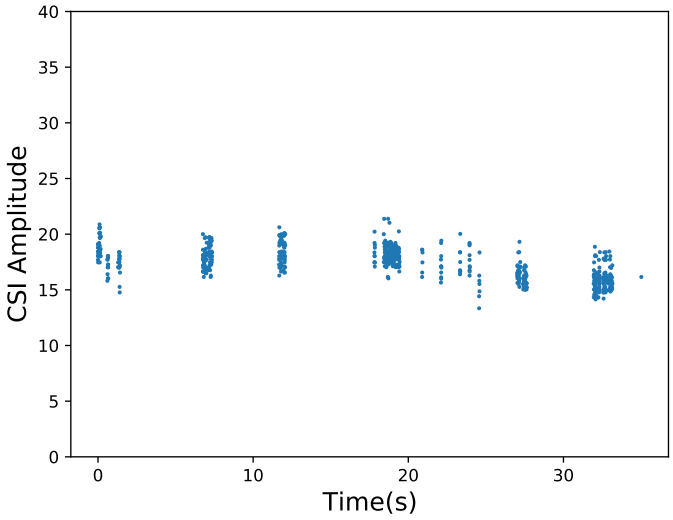

Figure 4.1: Disjoint data flow

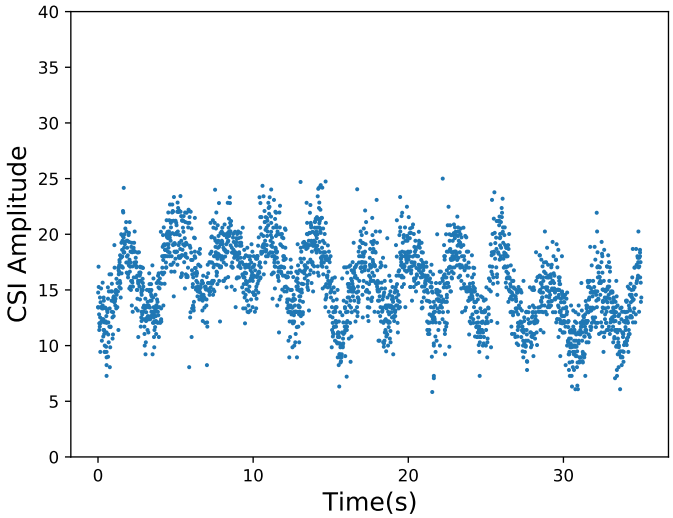

Figure 4.2: Continuous data flow

Figure 4.3: Comparison of data flow with and without the fake beacon frames

4.3 Choosing a Target Device

So far we have explained how Wi-Sneak can turn a WiFi device to a breathing sensor. The next question is that which WiFi device in a network would be a good candidate for the attack. Although in theory Wi-Sneak can make any WiFi device into a sensor, not all devices are good candidates for measuring breathing rate. For instance, people do not stay close to devices like TV or fridge for a long time, and thus their WiFi devices are not suitable candidates for Wi-Sneak. Devices such as laptops and mobile phones are better options for measuring subtle movements since people are likely to spend a good amount of time near them. To pick the target device, Wi-Sneak first sends fake packets to all devices continuously. As mentioned before, all WiFi devices respond to these fake packets with acknowledgement packets. Wi-Sneak then analysis the the responses of each WiFi device, looking for breathing patterns. Note, as we will show in the following section, there is no complex computations during analysing the signal. Therefore, the data from all WiFi devices can be processed on a single processing unit such as a typical laptop. Once Wi-Sneak discovers the signal of which WiFi devices includes breathing rate patterns, it stops monitoring the other devices and continues sending fake packets only to those specific WiFi devices. In the following section, we explain how Wi-Sneak extracts breathing rate from the responses of a WiFi device using non-uniform FFT.

4.4 Extracting Breathing Rate

So far we have explained how an Wi-Sneak can discover the MAC addresses of WiFi devices in a network and push them to consciously transmit packets. However, if Wi-Sneak sends too many packets per second, it will be always occupying the channel. This will result in a significant drop in the throughput of the WiFi network. In this section, we investigate what is the minimum number packets required to extract the breathing rate without impacting the network performance.

An adult's normal breathing rate is around 12-20 times per minute (i.e. 0.2-0.33 per second). One can imagine that based on the Nyquist rate, sampling rate of a few times per second (i.e. sending a few packets per second) is more than enough to detect the breathing rate. However, based on our experiments, we found that we require at least 100 samples per second to detect breathing rate. This is mainly due to the fact that the collected data is not uniformly spaced in time. In particular, when Wi-Sneak tries to send back-to-back packets to the target device, there is a random delay due to channel access protocol and also operating system which cause the sample measurements (i.e. ACKs received from the

target) to be non-uniformly spaced in time. Therefore, Wi-Sneak requires to send packets to the target device at much higher rate (i.e. at least 100 packets per second). This is not practical for the attacker. Sending a packet and receiving an ACK takes at least a millisecond. Therefore, when Wi-Sneak needs to monitor multiple target devices, sending 100 packets per second to each device is impossible and significantly impact the throughput of the network which makes the attack detectable. In the following, we explain how an attacker can solve this problem by capturing less number of samples and then performing a non-uniform FFT.

Algorithm 1: Non-uniform FFT

Data: Two lists t, x of length n and a number d
Result: The non-uniform FFT results
for $i \leftarrow 1$ **to** $n-1$ **do**
  $interval \leftarrow t[i] - t[i-1]$;
  if $interval > d$ **then**
    $count \leftarrow \lfloor interval/d \rfloor$;
    Interpolation(t, x, $t[i]$, $t[i-1]$, $count$);
  end
end
return FFT(t, x)

Non-uniform FFT: Let $t_0, t_1, ..., t_n$ be the timestamp of all the data points. The attacker first finds the minimum time gap between these data points d where

$$d = \min(t_i - t_{i-1}) \quad i = 1, 2, ..., n.$$

Then for any interval between two consecutive data points such that $t_i - t_{i-1} > d$, interpolated points will be added between them. As shown in Algorithm 1, once the interval that needs interpolations is found, the number of points needed is recorded into *count*. Then the function **Interpolation**$(x, t, start, end, n)$ will modifies the lists x and t adding n interpolated points between *start* and *end*. Specifically, the linear interpolation is used in this algorithm. Finally, to reduce the noise, the attacker applies a low pass filter before feeding the data to the FFT. Figure 4.5 and 4.6 show the amplitude of CSI before and after interpolation, respectively, when the attacker sends 30 packets per second to a WiFi device which is close to a person. Each Figure shows both the original data (in blue) and the filtered data (in orange). These figures show that both interpolation and filtering significantly help in extracting the breathing pattern. To better evaluate this, we compare the output of the FFT for these signals when a standard FFT or a non-uniform FFT is

used. Figure 4.4 shows the result of this comparison. The output of the standard FFT does not have any clear peak, making it impossible to detect the breathing rate. On the other hand, the output of the non-uniform FFT has a clear peak around 0.3 Hz, which is presenting the actual breathing rate of 18 breaths per minute.

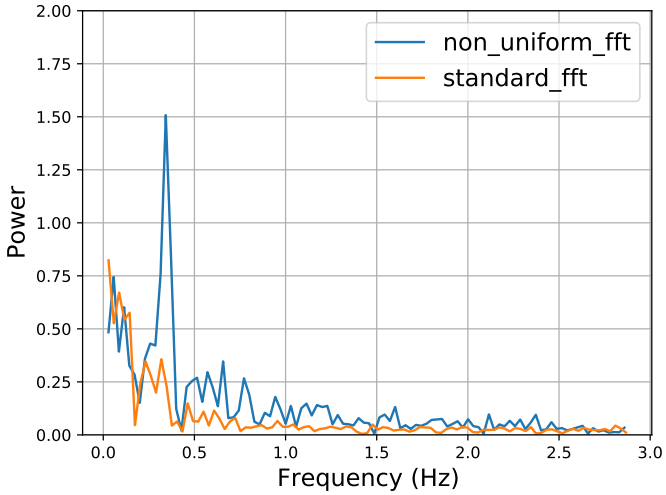

Figure 4.4: FFT results comparison

Voting Algorithm: Since each WiFi packet has 52 sub-carries, Wi-Sneak can compute the breathing rate by taking FFT over CSI of any sub-carriers. However, we found that some subcarries are more robust than others in detecting the breathing rate, depending on the environment. Therefore, instead of computing the breathing rate using only one of the subcarriers, Wi-Sneak takes FFT over CSI of all subcarriers and uses a soft voting mechanism to calculate the breathing rate. In particular, each subcarrier gives a weighted vote to a breathing rate. We then calculate which breathing rate has the highest number of votes. In the following, we explain the attacker voting mechanism.

Let's assume P_i is the power of each bin in the FFT output, where $i = 1, 2, ..., n$. We first find the power of peak ($P_{peak} = max(P_i)$), and then calculate the average power of other bins ($P_{ave} = \frac{\sum_{i \neq peak} P_i}{n-1}$). We then calculate the ratio of these two value ($\frac{P_{peak}}{P_{ave}}$) which defines the quality or SNR of the FFT peak for that sub-carrier. Finally, we use these values as a weight for the subcarrier vote. However, instead of just directly use these values, we use $w = e^{\frac{P_{peak}}{P_{ave}}}$ as the weights. This guaranties that subcarriers with higher SNR has significantly more votes than the rest of subcarriers. For example, even if there

is only one sub-carrier which shows the breathing pattern, it has higher weight than the summation of other votes.

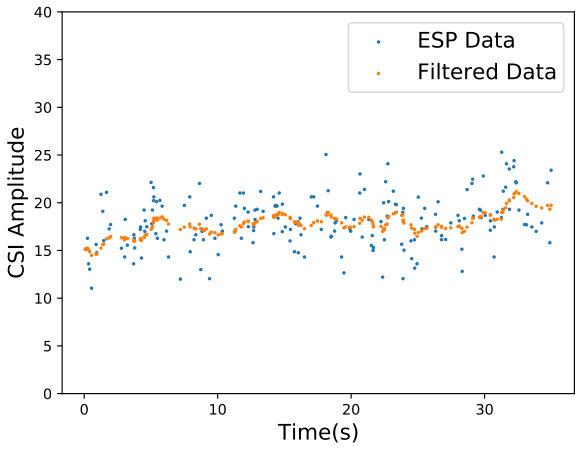

Figure 4.5: Data before interpolation

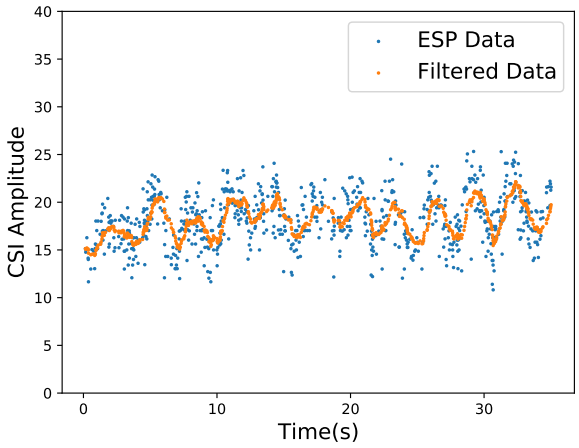

Figure 4.6: Data after interpolation

Figure 4.7: Comparison of the original data and interpolated data

Chapter 5

Evaluation & Analysis

To evaluate the feasibility of Wi-Sneak, we conduct a set of experiments in different scenarios. In the following, we first describe our attack scenarios and experiment setup. We then present evaluation results on accuracy and effective range of Wi-Sneak.

We evaluate Wi-Sneak in two different scenarios: Indoor and Outdoor. In the indoor scenario, the attacker and the target are placed in the same building but at different floors. The height of one floor in the building is around 2.8 m. In the outdoor scenario, the attacker is outside of the target's house. For each scenario, we evaluate the attack for different target and attacker locations, as shown in Figure 5.1. For example, in indoor experiments, the attacker is placed at location C in the basement of the house, while the target is tested at both location A and B in the first floor. In outdoor experiments, the attacker is placed at location D in the backyard, which is around 3 m away from the outside wall of the house. The target device is tested at location A, B and C. In all experiments, the target WiFi device is placed 0.5 m away from the person's body.

5.1 Attacker Hardware

The attacker uses a Linksys AE6000 WiFi-card and a ESP32 WiFi module as an attacker device. Both devices are connected to a ThinkPad laptop through USB. The Linksys AE6000 is used to send fake packets and the ESP32 WiFi module is used to received acknowledgements. Although, we use two different devices for sending and receiving, one can simply use a ESP32 WiFi module for both purposes.

As for the target device, we use a One Plus 8T cell phone without any software or hardware modification. It worth mentioning that any WiFi device can be a target for Wi-Sneak.

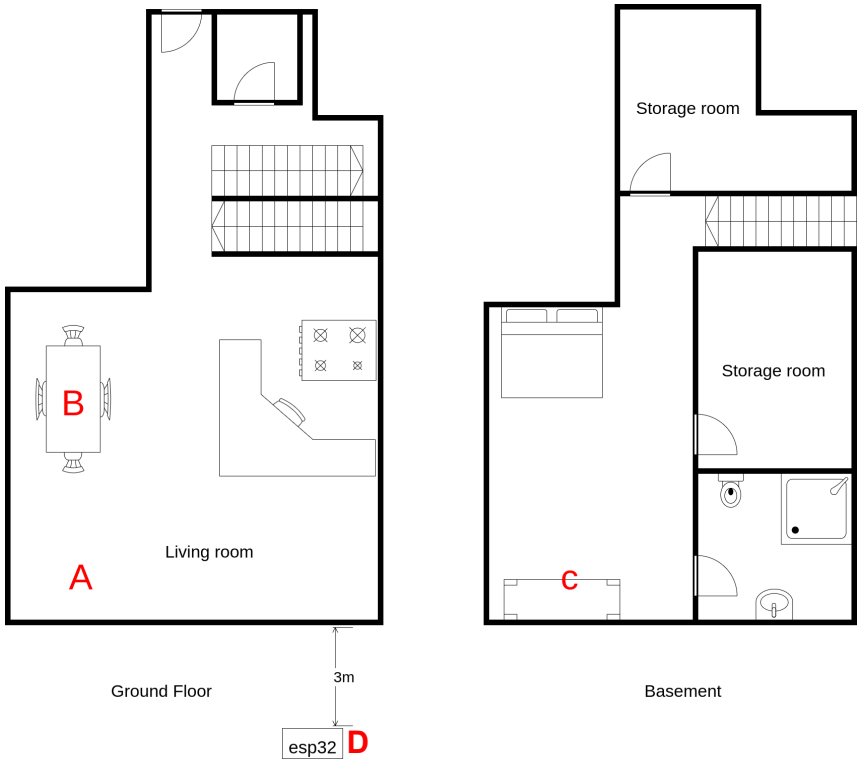

Figure 5.1: Floor Plan

5.2 Attacker Software

We implemented the CSI collecting script on the ESP32 WiFi module, and the algorithm mentioned in Chapter 4.4 on the laptop. The collected CSI data is fed to the algorithm and produce the breathing rate estimation in real-time.

Upon running the program, the data packets constantly flows into Wi-Sneak. To process these data in real-time, a sliding window (buffer) is used. The size of the window is 30 s and the sliding step is 1 s. As shown in Figure 5.2, the window is a queue of data points, and it updates every second by including 1 second new data points to its head and

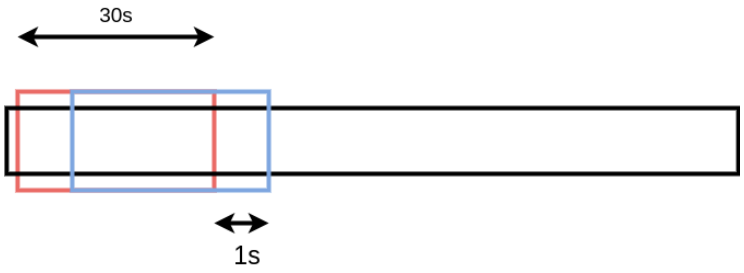

Figure 5.2: Real-time analysis with sliding window

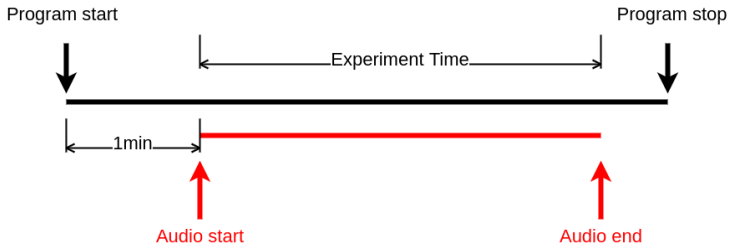

Figure 5.3: Synchronization

removing 1 second old data points from its tail. Wi-Sneak runs the analysis algorithm on the data points in the window whenever it is updated. The window slides once per second. Hence, Wi-Sneak reports an estimation of breathing rate every second. Note there is a 30 s starting delay since the data need to fill the sliding window first.

5.3 Ground Truth

In order to evaluate the accuracy of the attack in estimating the breathing rate, we need to measure the ground truth breathing rate. To do so, we use a similar a method as introduced in [9]. In particular, we place a microphone near the target person's nose to record the sound of breathing. We then take an FFT of the sound signal to estimate the breathing rate accurately. However, as shown in Figure 5.4, each peak of the sound signal is very noisy. Therefore, an envelope function is applied before taking FFT such that these noises are removed.

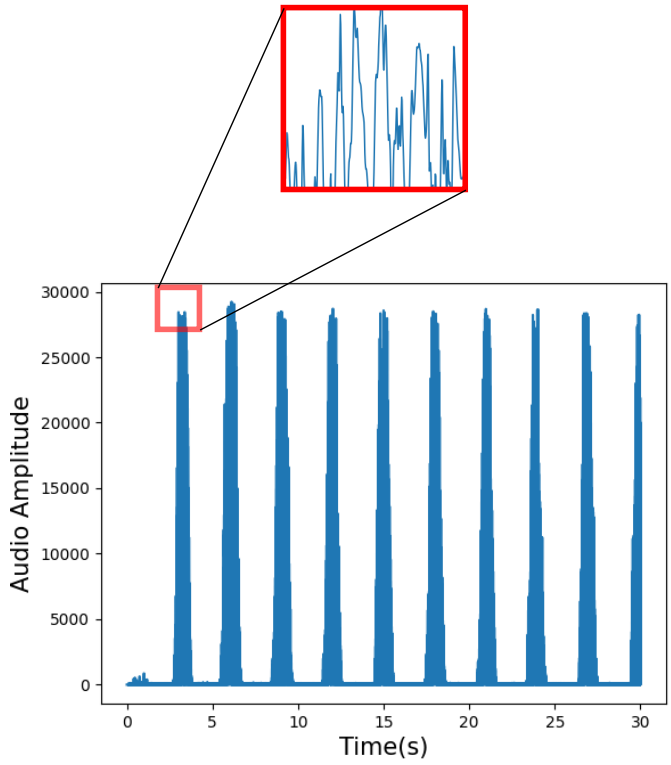

Figure 5.4: Audio Signal

5.4 Attack Scenarios

Similar to processing the WiFi data, we also apply the sliding window method to the ground truth data. Thus, the results from both ground truth and WiFi signal can be matched with each other. In addition, the ground truth and the attack results need to be synchronised on the time domain. To do so, once the program starts, we start a timer which serves as a coordinator. As the timer reaches one minute, the ground truth recorder starts, and this marks the start of experiment as shown in Figure 5.3. The one minute synchronizing time also covers the 30s starting delay of the estimation processing, so that the window has already buffered enough data for analyzing.

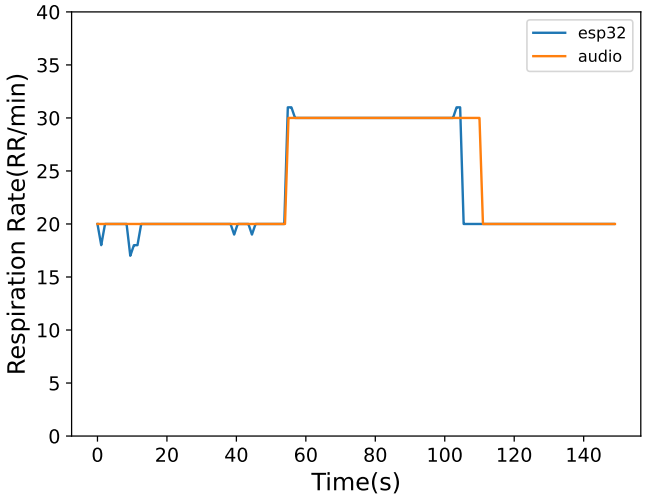

Figure 5.5: A controlled experiments with a changing breathing rate

5.5 Sensitivity and Accuracy

We first evaluate the sensitivity of Wi-Sneak in detecting breathing rate by performing a controlled experiment where a person change its breathing rate. The attacker is outside the house at location D and the target person is at location A. This experiment lasts three minutes and the target person breathes slow in the first minute, fast in the following minute and back to slow again in the last minute. As shown in Figure 5.5, Wi-Sneak can accurately capture the changes of breathing rates. Note that each data point in this figure is an FFT estimation over a 30 s window for both the ground truth and the estimation. This controlled experiment shows that although people's breathing do not usually change, Wi-Sneak is sensitive enough to capture the change of breathing rate.

Next we run an experiment to examine Wi-Sneak's capability in detecting whether there is a target close to the WiFi device or not. For this experiment, the person stays around the device for 30 s, then walks away from the device, and then comes back to the device. Note, in our algorithm, when there is no majority vote during the voting phase, we return -1. Figure 5.6 shows the result of this experiment. It is clear that Wi-Sneak detects whether there is a person close to the WiFi device or not.

Finally, we evaluate the accuracy of Wi-Sneak in estimating the breathing rate for both indoor and outdoor scenarios as explained in Chapter 5.4. For each scenario, we evaluate

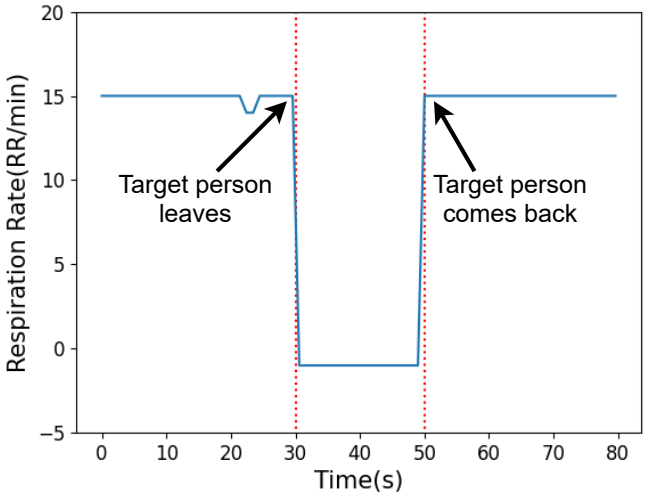

Figure 5.6: An experiment where the target person leaves the device and then comes back

Target Location	Attacker Location	Method	12	15	20	30
A	C	Estimated Result	11.97	15.03	20.01	29.94
		Ground Truth	12.05	14.91	20.06	30.17
B	C	Estimated Result	11.95	15.00	19.97	29.94
		Ground Truth	11.98	15.07	20.01	30.06
A	D	Estimated Result	11.96	15.00	19.98	29.96
		Ground Truth	11.97	15.12	20.05	30.05
B	D	Estimated Result	11.99	15.00	20.00	29.97
		Ground Truth	12.08	15.04	20.11	30.05
C	D	Estimated Result	12.00	14.94	19.98	29.95
		Ground Truth	12.04	15.01	20.08	30.16

Table 5.1: Results comparison of different respiration rate across all experiments

Wi-Sneak's accuracy when the target's breathing rate is 12, 15, 20 and 30 per minute. Note, normal respiration rate for an adult is 12-20 per minute while resting, and higher when exercising. To make sure the person's breathing rate is close to these numbers, we place a timer in front of the target, where they can adjust their breathing rate accordingly. We run each experiment for two minutes. During this time, we collect the estimated breathing rate from both audio (used for ground truth) and Wi-Sneak. Table 5.1 shows

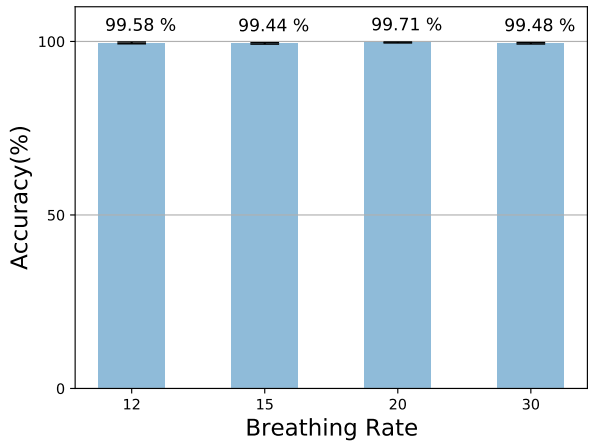

Figure 5.7: Accuracy across all experiments

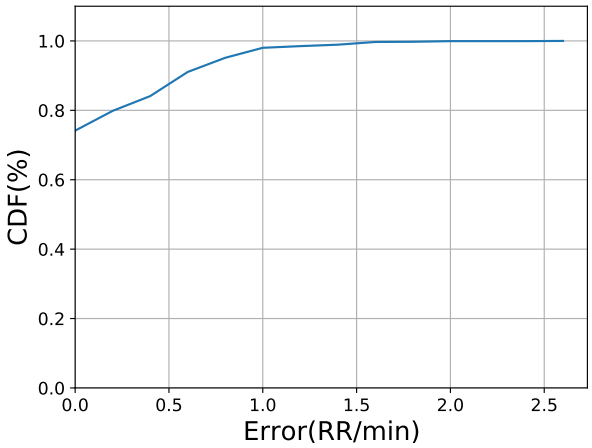

Figure 5.8: Experiment CDF over 2400 estimations

he result of these experiments. The result shows that Wi-Sneak accurately detects the breathing rate of the target person in different scenarios and for various breathing rate. To quantify the accuracy of Wi-Sneakin estimating the breathing rate, we also plot the average accuracy of Wi-Sneakin estimating breathing rate for all experiments in Figure 5.7. The accuracy is calculated as the ratio of estimated breathing rate by Wi-Sneak over the ground truth breathing rate. The figure shows that Wi-Sneak's accuracy is over 99% in all scenarios. Finally, Figure 5.8 plots the Cumulative Distribution Function (CDF) of the error in detecting breathing rate for over 2400 measurement. The CDF is generated based on the estimated breathing rate reported every second by Wi-Sneak. Therefore, each 2 minutes experiment has generated 120 estimation. The figure shows that 78% of the estimated results have no error. The figure also shows that 99% of measurements have less than one breath per minute error which is negligible.

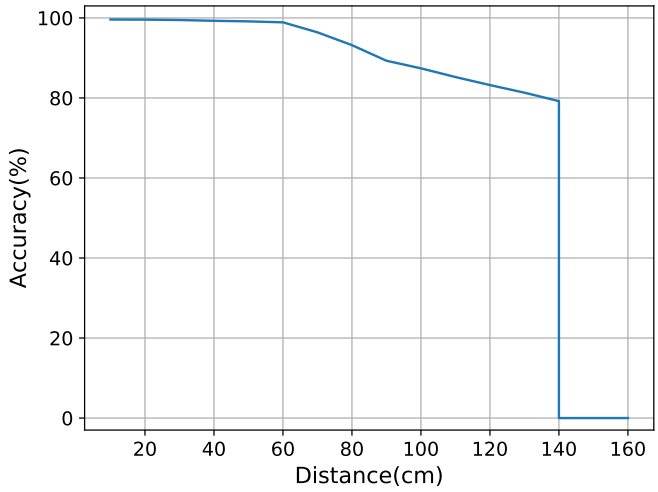

Figure 5.9: Target person distances vs Accuracy

5.6 Effective Range

So far, the target device is placed 0.5 m away from the person's body in our experiments. Here, we evaluate Wi-Sneak performance for different distances between the target device and the target person. In particular, we are interested to find out what the maximum distance between the target device and the person can be while Wi-Sneak still detects the

person's breathing rate. To do so, we place the attacker device and the target device in two different rooms with a wall in between. The distance between the attacker and the target device is around 7 m. We then run experiments when the person changes distances from the target device. In each experiment, we measure the breathing rate for two minutes and calculated the average breathing rate over this time. Finally, we compare the estimated breathing rate to the ground truth and calculate the accuracy as mentioned before. Figure 5.9 shows the result of this experiment. The figure shows that Wi-Sneak's accuracy is over 99% when the distance between the target device and the person is less than 60 cm. Note, in reality, people have their laptop or cellphone very close to themselves most time. The figure also shows that the accuracy drops as we increase the distance. However, even when the device is at 1.4 m from the person's body, the attack can still estimate the breathing rate with 80% accuracy (i.e. estimating 12 when the person's breathing rate was 15).

Chapter 6

Is It Possible to Stop This Attack?

In the previous chapters, we showed how effective Wi-Sneak is in detecting the breathing rate of a target person. A natural question is whether it is possible to stop an attacker from monitoring a person's breathing rate. As explained before, Wi-Sneak relies on the CSI changes of WiFi signal to estimate the breathing rate. Therefore, one possible solution to stop such an attack is to artificially create similar changes to CSI. For example, if the target WiFi device periodically change its transmission power, it might be possible to prevent the attacker from estimating breathing rate of a person from CSI changes.

To verify the effectiveness of such technique, we perform an experiment. In this experiment, we periodically change the transmission power of the target device between 10 dBm and 18 dBm every 1 second, while Wi-Sneak tries to estimate breathing rate. Figure 6.1 shows the result of this experiment. The periodic pattern can clearly be seen in the CSI amplitude of the WiFi packets measured by the attacker. In this experiment, Wi-Sneak reported breathing rate of 30 breath per minute which is the frequency of change in the transmission power. Although, this result shows the effectiveness of our approach in preventing Wi-Sneak from monitoring breathing rate, changing the transmission power by as much as 8 dB every second can significantly impact on the throughput of the WiFi device. Next, we try to find what the minimum required change in transmission power is to disrupt Wi-Sneak.

Similar to previous experiment, we run a set of experiments where the target device is changing its transmission power. However, we try different transmission power changes at different intervals. In all experiments, there is a person next to the target device and the attacker tries to estimate their breathing rate. Figure 6.2 shows the result of this experiment. In particular, the figure shows if the attacker was able to successfully detect

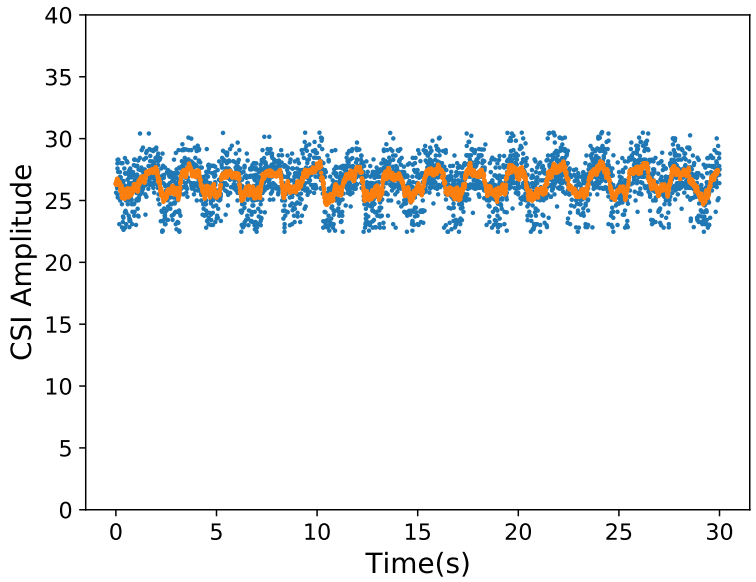

Figure 6.1: Changing Tx power periodically when no person is near target device

the person's breathing rate for different change in the transmission power at different change intervals. The shaded area on the plot shows the area where the attacker was not able to detect the true breathing rate of the target person, which means the defense is successful. This results shows that if we lower the interval (i.e. increase the frequency of change), the required change in the transmission power to prevent the attacker decreases. However, note, lowering the intervals bellow 0.5 s (i.e. 60 times per minute) will make the defence ineffective. This is due to the fact that adult's breathing rate is in the range of 12-20 breaths per minute and baby's breathing rate is in the range of 40-60 breaths per minute [14]. Therefore, the attacker can easily filter out any changes which are above 60 times per minute. Hence, the optimal way to prevent the attacker from estimating breathing rate is to make the target's device transmission power to change by 3 dB every 0.5 s.

Now, the next question is whether such a change in the transmission power impacts on the throughput of the WiFi device. To examine this, we run another experiment.

We setup two laptops. One is a server acting as an Access Point (AP), and the other one is a client acting as a target device. The server hosts a WiFi network and the target device connects to it. The client sends UDP packets to the server while we use iperf comment to

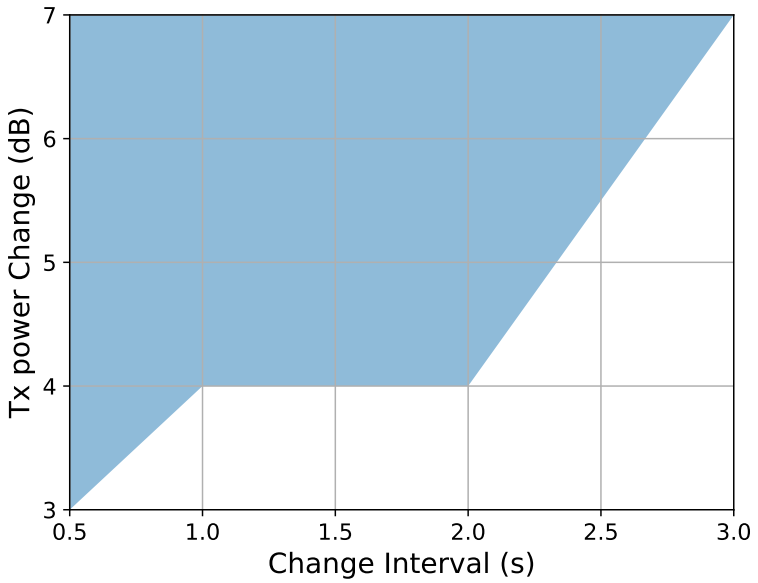

Figure 6.2: Results of different combinations that can disrupt attack

monitor the throughput over 10 minutes. We set the target bandwidth to 20 Mbps and perform this experiment in three different scenarios: (1) the Tx power is stable at 12dBm, (2) the Tx power is stable at 15dBm, and (3) the Tx power oscillates between 12dBm and 15dBm every 0.5 s.

Figure 6.3 shows the CDF of throughput for all three scenarios. The figure shows that the WiFi device archives the same throughput in all scenario. In particular, oscillating the transmission power does not impact on the throughput of the device. Note, in this experiment, we set the target bandwidth to 20 Mbps. Although this is more than what most WiFi devices uses, there are applications where the device require higher bandwidth. Therefore, next we evaluate if oscillating the transmission power impact on the throughput of the WiFi device when the device transmit at the maximum data rate (i.e. saturating the WiFi channel). Figure 6.4 shows the the CDF of throughput when the WiFi device saturates the channel for all three scenarios. As expected, lowering the transmission power from 15 dBm to 12 dBm reduces the throughput. However, the results show that when the transmission power oscillate between 12 and 15 dBm, the throughput is even lower than the case where the transmission power is kept at 12 dBm. The reason for this is that WiFi protocol uses a rate adaption algorithm, where the data rate is adapted according

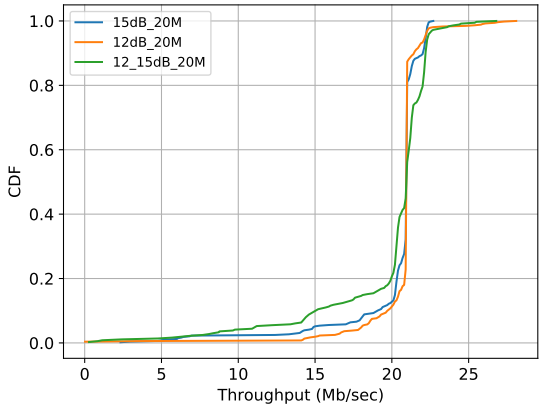

Figure 6.3: CDF for throughput under different Tx power with 20 M bandwidth

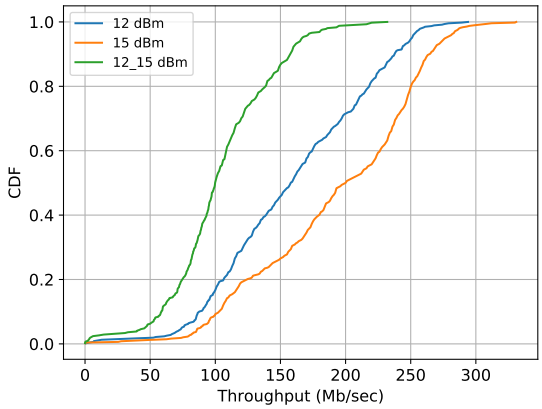

Figure 6.4: CDF for throughput under different Tx power

to the signal strength. High data rates are adopted when the signal is strong, while low data rates are adopted during weak signals. Therefore, when we change the transmission power back and forth, the algorithm tries to adopt. However, since there is a delay, these repeated changes causes packet drops which result in reduction in the throughput.

In summary, although our idea of oscillating the transmission power prevents an attacker from monitoring the breathing rate, it also impacts on the maximum throughput the WiFi device can achieve. However, the impact will be negligible for the WiFi devices that do not require more than 20Mbps throughput.

Chapter 7

Conclusion & Future Works

This thesis has explored Wi-Sneak , a way of utilizing the ambient WiFi signals to perform a stealthy reconnaissance attack on people's private data. The evaluations on various aspects has proven that Wi-Sneak can retrieve target's respiration rate accurately under various scenarios, and the effective range can be up to 1.4 m. Moreover, the attack is also effective when the WiFi devices are inactive during the sleep mode. Wi-Sneak does not cause any suspicious behaviour in the WiFi network, so people without special training can hardly detect it. The technique can be further explored to detect more detailed movements or actions. Since the signal of the CSI is very sensitive to fine movements, applying machine learning on the signal analysis may be able to extract even more information other than breathing rate. Knowing the possibility of this stealthy reconnaissance attack, this thesis proposed a limited defense method. Although it sacrifices some network throughput, we hope it raises people's safety awareness on WiFi network and encourages more studies and researches in this area to make the WiFi network more secure.

References

[1] Heba Abdelnasser, Khaled A Harras, and Moustafa Youssef. Ubibreathe: A ubiquitous non-invasive wifi-based breathing estimator. In *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 277–286, 2015.

[2] Heba Abdelnasser, Moustafa Youssef, and Khaled A Harras. Wigest: A ubiquitous wifi-based gesture recognition system. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 1472–1480. IEEE, 2015.

[3] Ali Abedi and Omid Abari. Wifi says" hi!" back to strangers! In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*, pages 132–138, 2020.

[4] Fadel Adib, Zachary Kabelac, and Dina Katabi. Multi-person localization via {RF} body reflections. In *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*, pages 279–292, 2015.

[5] Fadel Adib and Dina Katabi. See through walls with wifi! In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, pages 75–86, 2013.

[6] Fadel Adib, Hongzi Mao, Zachary Kabelac, Dina Katabi, and Robert C Miller. Smart homes that monitor breathing and heart rate. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, pages 837–846, 2015.

[7] John Bellardo and Stefan Savage. 802.11 denial-of-service attacks: Real vulnerabilities and practical solutions. In *USENIX security symposium*, volume 12, pages 2–2. Washington DC, 2003.

[8] Apurv Bhartia, Yi-Chao Chen, Swati Rallapalli, and Lili Qiu. Harnessing frequency diversity in wi-fi networks. In *Proceedings of the 17th annual international conference on Mobile computing and networking*, pages 253–264, 2011.

[9] Eliran Dafna, Ariel Tarasiuk, and Yaniv Zigel. Sleep-wake evaluation from whole-night non-contact audio recordings of breathing sounds. *PloS one*, 10(2):e0117382, 2015.

[10] Liangyi Gong, Wu Yang, Dapeng Man, Guozhong Dong, Miao Yu, and Jiguang Lv. Wifi-based real-time calibration-free passive human motion detection. *Sensors*, 15(12):32213–32229, 2015.

[11] Liangyi Gong, Wu Yang, Zimu Zhou, Dapeng Man, Haibin Cai, Xiancun Zhou, and Zheng Yang. An adaptive wireless passive human detection via fine-grained physical layer information. *Ad Hoc Networks*, 38:38–50, 2016.

[12] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1994.

[13] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. Tool release: Gathering 802.11 n traces with channel state information. *ACM SIGCOMM Computer Communication Review*, 41(1):53–53, 2011.

[14] Stanford Children's Health. Breathing problems. https://www.stanfordchildrens.org/en/topic/default?id=breathing-problems-90-P02666.

[15] Donald Knuth. *The TeXbook*. Addison-Wesley, Reading, Massachusetts, 1986.

[16] Bastian Könings, Florian Schaub, Frank Kargl, and Stefan Dietzel. Channel switch and quiet attack: New dos attacks exploiting the 802.11 standard. In *2009 IEEE 34th Conference on Local Computer Networks*, pages 14–21. IEEE, 2009.

[17] Piers O'hanlon, Ravishankar Borgaonkar, and Lucca Hirschi. Mobile subscriber wifi privacy. In *2017 IEEE Security and Privacy Workshops (SPW)*, pages 169–178. IEEE, 2017.

[18] Mathy Vanhoef, Prasant Adhikari, and Christina Pöpper. Protecting wi-fi beacons from outsider forgeries. In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 155–160, 2020.

[19] Mathy Vanhoef and Frank Piessens. Advanced wi-fi attacks using commodity hardware. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 256–265, 2014.

[20] Ju Wang, Hongbo Jiang, Jie Xiong, Kyle Jamieson, Xiaojiang Chen, Dingyi Fang, and Binbin Xie. Lifs: low human-effort, device-free localization with fine-grained

subcarrier information. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 243–256, 2016.

[21] Siamak Yousefi, Hirokazu Narui, Sankalp Dayal, Stefano Ermon, and Shahrokh Valaee. A survey on behavior recognition using wifi channel state information. *IEEE Communications Magazine*, 55(10):98–104, 2017.

[22] Yanzi Zhu, Zhujun Xiao, Yuxin Chen, Zhijing Li, Max Liu, Ben Y Zhao, and Haitao Zheng. Et tu alexa? when commodity wifi devices turn into adversarial motion sensors. *arXiv preprint arXiv:1810.10109*, 2018.