WiSafe: A Real-Time System for Intrusion Detection Based on WiFi Signals

Yu Bao
Tsinghua University
baoyu3302@gmail.com

Liang Dong
Tsinghua University
thdongl@163.com

Yue Zheng
Tsinghua University
cczhengy@gmail.com

Ying Liu
LiaoNing Technical University
liuying_dx@lntu.edu.cn

Abstract

Pioneer research works for WiFi-based sensing usually depend on an extremely high sampling rate of over 1000 Hz to collect an abundant number of Channel State Information (CSI) measurements, which depict the environmental changes and human motions. However, these methodologies can be hardly deployed on embedded platforms and cannot be directly commercialized. In this paper, we propose WiSafe, a real-time intrusion detection system that works with a relatively low sampling rate of 15-20 Hz on commercial embedded devices. Both primitive physical features and high-level CSI-based features are adopted as the input of the classification methods. We exploit MLP, TextCNN and Bi-LSTM together for majority voting to improve the detection accuracy and robustness. Experiment results demonstrate that WiSafe achieves an accuracy of 97.8% (AUC=0.9931) that is comparable to those of previous works even with a low sampling rate, and can run in real time, which make it permissible for practical use.

Keywords

WiFi, Channel State Information, Low Sampling, Embedded Device, Real-time Detection

ACM Reference Format:

Yu Bao, Liang Dong, Yue Zheng, and Ying Liu. 2019. WiSafe: A Real-Time System for Intrusion Detection Based on WiFi Signals. In *ACM Turing Celebration Conference - China (ACM TURC 2019) (ACM TURC 2019), May 17–19, 2019, Chengdu, China.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3321408.3321587

1 Introduction

Due to favorable features, such as high transmission rate, appropriate signal coverage area and the convenience of setup, WiFi is promoted widely in people's daily life, and has become necessary infrastructure for work, study and entertainment. Apart from the capability of carrying information for wireless communication, WiFi, as a kind of electromagnetic wave, also exhibits strong potential for being regarded as a new perceptual medium. In the past few years, WiFi has been widely exploited in detecting intrusion ([6, 7, 12, 13, 15, 17, 18]), recognizing activity ([1, 4, 5, 8–10]), and indoor localization ([11, 16]). Moreover, WiFi-based sensing is a device-free approach which doesn't require specialized hardware attached to people. Thus, this kind of device-free detection approach can be used in many scenarios, such as indoor intrusion detection, outdoor perimeter security, child and elder care at home, patient monitoring, etc.

Among the features provided by WiFi, Channel State Information (CSI) presents subcarrier-level measurements of the signal. As a fine-grained feature, CSI keeps relatively stable in the static scenario and is more sensitive to human movements, which makes it feasible for intrusion detection. However, most previous works collect CSI and other data with an extremely high sampling rate, e.g., over 1000 Hz. In addition, they usually require a computer equipped with an Intel 5300 network card. Such methodologies exhibit following problems. Firstly, it will be a very expensive solution for intrusion detection if each monitoring area requires a computer/laptop to be deployed. Thus, the systems will be difficult to be commercialized. Secondly, with data collected at a high sampling rate, the amount of computation will be largely increased, making intrusion detection algorithms infeasible to run directly on embedded platforms. Thirdly, many offline algorithms used in previous works [15] are not easy to run in real time because too much global data information are used. Moreover, previous works only collect data in rather ideal environment and the amount of the collected data in existing works is small.

In this paper, we propose WiSafe, a real-time intrusion detection system that works with a relatively low sampling rate of 15-20 Hz on commercial embedded devices. We firstly transplant the Linux CSI tools[2] to OpenWrt, in order to collect CSI measurements on commercial embedded device. Intrusion detection systems can usually be divided into two parts, feature extraction and the classification algorithm. For feature extraction, we firstly select several primitive physical features reported by the driver, then extract high-level features from raw CSI measurements. The united feature set is more resistant to noises and enough to depict environmental changes even at a low sampling rate. For the classification mechanism, we exploit MLP, TextCNN and Bi-LSTM for majority voting to improve the detection accuracy and robustness.

To validate our design, we prototype WiSafe on commercial embedded devices. We collect more than one million CSI measurements in total in both dynamic and static scenarios for overall

ACM TURC 2019, May 17–19, 2019, Chengdu, China

ACM ISBN 978-1-4503-7158-2/19/05…$15.00
https://doi.org/10.1145/3321408.3321587

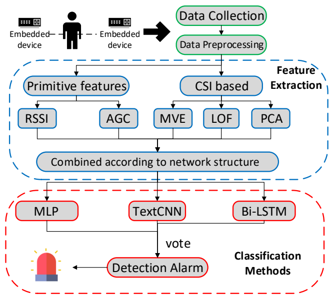


Figure 1: System architecture.

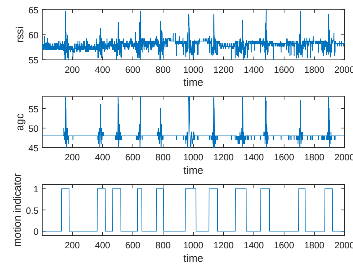


Figure 2: Correlation between features and movements.

evaluation. Experiments results show that WiSafe achieves an accuracy of 97.8% (AUC=0.9931). We also conduct influence factor study to find the optimal configuration of hardware and software, and verify that environment parameters, such as humidity, do pose influence on the detection performance. After productization, this system can be used in a wide range of security scenarios, such as security scenes with a small amount of occlusion and unable to use infrared or video (eg. fences with small amounts of vegetation).

In summary, our main contributions are as follows:

- We are the first effort to transplant the Linux CSI tools to OpenWrt, which runs on many commercial embedded devices.
- We design a novel intrusion detection methodology including feature extraction and the classification algorithm, which can work with a sampling rate of 15-20 Hz and run in real time.
- We present the design and implementation of WiSafe. Experiment results demonstrate that WiSafe can achieve high performance that is comparable to previous works. And we explore the impacts of various factors, such as hardware software and environment factors, on the performance of intrusion detection algorithms, and find the optimal configuration of hardware and software.

2 Related Work

Intrusion detection. Intrusion detection is a primary and fundamental application in wireless sensing, and recent years have witnessed its rapid development, especially based on CSI provided by off-the-shelf WiFi devices. FIMD[13] utilizes the observation that the temporal correlation of CSI amplitude will fall dramatically when there is a moving human to achieve intrusion detection. PADS[6] is similar to FIMD[13], but it is the first effort to utilize CSI phase information as well to determine human presence. Deman[12] proposes a method to further detect whether a static people by fitting CSI sequence to the sinusoidal breathing model to estimate the primary respiratory rate. If the estimated frequency falls within the normal person's respiratory frequency range, then the system assumes that a static person exists. Omni-PHD[18] uses fingerprints and EMD algorithm to detect intrusion in all directions. Despite that these works have achieved high accuracy, their performance cannot be guaranteed with a relatively low sampling rate of 15-20Hz and they can hardly be transplanted to embedded devices for commercial use, which is much different from our work.

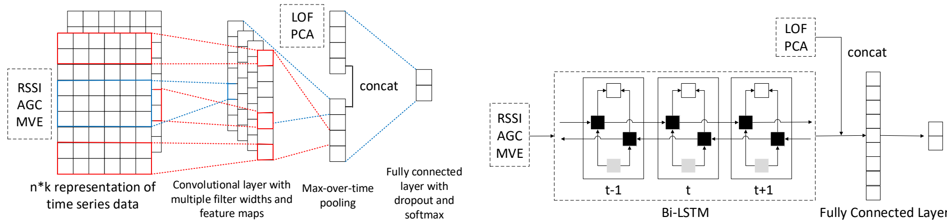
Activity recognition. In addition to intrusion detection, different kinds of sensing applications have been boosted in the past few years. CARM[9] is a pioneer work in the general activity recognition task. It quantifies the correlation between movement speed and CSI dynamics by proposing the CSI-speed model. Then it uses the CSI-activity model to further model the correlation between a specific human activity and movement speeds of different human body parts. The principles of WiKey[1] is similar to CARM, but it mainly targets at keystroke recognition. In addition, Wi-Sleep[5] and [4] use WiFi to monitor the respiration rate during sleep in different methods. The former mainly uses the recurrence plot to predict the respiration rate, while the latter mainly uses the PSD algorithm. WiFall[10] uses WiFi to do falling detection. In order to obtain high recognition accuracy, most of the existing works depend on rigorous requisites, such as significant prior training, rather ideal environment and high sampling rate, which make them infeasible for working in practice. Our paper shows that WiSafe can still work with a satisfying accuracy even with a low sampling rate in practical environment.

3 Methodology

In this section, we elaborate the principles and steps of the methodology, which can be divided into two parts: feature selection and classification algorithm. Compared with the previous work on motion detection, we newly add the physical feature, Automatic Gain Control (AGC), extracted from the driver, in addition to the traditional feature Received Signal Strength Indicator (RSSI). Considering that noise usually exists in the environment, we complement three additional features by computing Moving Variance Energy (MVE) and Local Outlier Factor (LOF), and perform Principal Component Analysis (PCA) with the collected CSI measurements. In terms of classification methods, we use a traditional classifier MLP, TextCNN that can combine regional context and Bi-LSTM that can fully analyze the timing relationship, and finally get a more accurate classification result by a voting mechanism. Fig. 1 shows the main architecture of the methodology, illustrating the specific calculation process of the extracted features in the neural networks.

3.1 Primitive Feature Selection

In order to get the first component of the final feature vector, we go through all the original information that can be extracted from the driver, such as RSSI, AGC and data rate. After normalizing each raw feature, we calculate the correlation coefficient with the motion indicator which is the ground truth of motions, and select the features with larger correlation coefficients as the first component of the final feature vector. Other features are treated as candidate features because they may have correlations other than linear correlation, and are selectively added and compared



(a) Structure of TextCNN. **(b) Structure of Bi-LSTM.**

Figure 3: Network structure.

in subsequent experiments. As shown in Fig. 2, the peak of RSSI and AGC are closely related to human movements. And AGC has a higher degree of correlation with motion indicator than RSSI. Thus, we finally choose RSSI and AGC as the first component of the feature vector. After the experiment in Sec. 4.2, we will get final primitive features according to the AUC.

3.2 CSI Feature Extraction

CSI is closely related to motion detection, so we extract various features from the CSI matrix from different aspects, and strive to retain more information that reflects human motion.

Moving Variance Energy(MVE). MVE reflects the degree of fluctuation of the data in a period. It is a coarse-grained statistical feature. When someone passes through the detection area, the fluctuation of CSI data will become larger, so the MVE value will also become larger.

Let the CSI matrix collected from the P subcarriers be: $C = [C_1, C_2, ..., C_P]^T$, among them, $C_i = [c_i(1), ..., c_i(T)]$, it represents the amplitudes of the T CSI collected from the i^{th} subcarrier. Then the moving variance of the P subcarriers is: $V = [V_1, V_2, ..., V_P]^T$, among them, $V_i = [v_i(1), ..., v_i(T)]$, it denotes the moving variance calculated from C_i. N is the number of samples in a sliding window. Then the moving variance energy (MVE) of the P subcarriers is:

$$MVE = \frac{1}{NP} \sum_{p=1}^{P} \sum_{n=1}^{N} \left| v_p(n) \right|^2 \tag{1}$$

Local Outlier Factor(LOF). When there is a small amount of noise in the environment, there will be some abnormal points in the CSI sequences, which cause the increase of MVE. To alleviate false alarms, we propose LOF, by which we can get the deviation degree of each feature vector compared to all the other feature vectors in the same feature space. The higher the deviation degree of a CSI vector, the greater its LOF value. In a time window, we calculate the LOF of each sample and use the tournament algorithm to select the largest m LOF values as the features calculated in this step. When there is a small amount of noise in a static environment, we will get a few large LOF values, and the remaining LOF values are generally small. However, when someone passes the detection area, the distribution of LOF values for each sample is relatively even. The calculation process of LOF is as follows:

The CSI vector obtained by each sampling can be regarded as a point in a feature space. Suppose the distance between point p and point o in the feature space is $d(p, o)$, the k-distance of point p $d_k(p)$ is defined as the k^{th} nearest distance from p (excluding point p). The k-distance neighborhood of point p $N_k(p)$ is the set of points within the k-distance (including the points on $d_k(p)$). Then the k^{th} reachable distance from point o to point p is defined as: $reach\_dis_k(p, o) = max\{d_k(o), d(p, o)\}$, thus, the local reachability density of point p is:

$$lrd_k(p) = 1/(\frac{\sum_{o \in N_k(p)} reach\_dis_k(p, o)}{|N_k(p)|}) \tag{2}$$

Finally, LOF is defined as:

$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} lrd_k(o)}{|N_k(p)|} / lrd_k(p) \tag{3}$$

Principal Components Analysis. When the noise in the environment is relatively large, there will be a case where the fluctuation of the whole CSI data is also large. In this case, neither MVE nor LOF can make an accurate judgment. Therefore, we propose a PCA-based denoising and feature extraction process with reference to CARM[9] in this paper. The purpose of this step is to reduce the noise and to leave as much movement information as possible.

Due to space constraints, we directly use the conclusions derived from CARM: The noise in the environment affects every subcarrier in the CSI, and due to the high correlation, noise is extracted in the first principal component along with the movement signal. However, the first principal component contains only some of the movement-related signals, and the rest is contained in the remaining principal components. Therefore, we discard the first principal component and use the second principal component to get more movement-related signals with less noise.

The above three features have their own advantages and disadvantages, some are sensitive to motion but cannot eliminate noise interference, and some can distinguish noise or even remove some noise. We can combine the features of different aspects to make more accurate judgments.

3.3 Classification Methods

As shown in Fig. 1, all the features above are combined in different ways for different network structures. We use the following classifiers: MLP, TextCNN, Bi-LSTM, and use the voting mechanism to get the final result.

MLP. MLP is a forward-structured artificial neural network that maps an input vector into an output vector. It is a general-purpose function approximation that can be used to fit complex functions and can also be used to solve classification problems. In this application, the normalized data of different features at different time are arranged into a one-dimensional vector as the input of the MLP, and the MLP adopts two hidden layers, each of which contains 500 hidden nodes. Furthermore, dropout layers and L2 regularization are also adopted in this network.

TextCNN. TextCNN[3] is an algorithm for classifying text sequences using convolutional neural networks. Similar to the convolution operation in image classification, the convolutional layer in TextCNN contains many convolution kernels, with which we can extract features of multiple different dimensions from the data in a same local region. Therefore, this algorithm can fully synthesize the local context information to do the final classification. In the end, we used TextCNN with the embedding layer removed as one of our classifiers, and its main structure is shown in Fig. 3a, *n* denotes a fixed number of samples, and k represents the dimension of the feature vector input to the TextCNN.

Bi-LSTM. LSTM is a neural network that processes time series information and can remember the information a period of time before. Bi-LSTM is a bidirectional recurrent neural network formed on the basis of LSTM, and is split into two directions, one for positive time direction (forward states), and another for negative time direction (backward states). Compared to the basic LSTM, it can synthesize the global information better, so that we can achieve a better classification result. The intrusion process is a time series process, so we consider using Bi-LSTM as another classifier. In this application, the input unit dimension of the network is the same as the data dimension of a single sample, the network has 1 hidden layer, which contains 200 hidden units. Its main structure is shown in Fig. 3b. The gray blocks refer to inputs at one point, white blocks are outputs, and the black blocks denote basic LSTM cells. After passing through the Bi-LSTM, the data is concatenated with other features and transferred into a fully connected layer for final classification.

4 Experiments and Evaluation

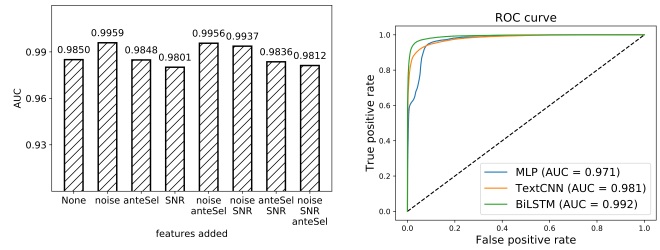
4.1 Experiment Setup

The software and hardware configuration used in the process of data collection is the optimal configuration selected after considering various factors. We firstly introduce the overall performance under optimal configuration in Sec. 4.2. And the influence factor study will be further described in detail in Sec. 4.3. The experiment setup is as follows:

At the hardware level, we use the mt7628 chip as our hardware platform, which is an embedded device with two transmit antennas. The cost price of a single node is only about 150 RMB, much cheaper than a laptop. Then a RF test instrument named IQ2010 is used to find Intel 5300 network cards with better RF parameters as the receivers. In addition, we remove long antenna SMA cable and customize impedance matching antenna for the circuit board. Antennas are roughly aligned in the vertical direction and the height from the ground is about 1.2m. The distance between transmitter and receiver is 25m.

At the driver software level, we firstly transplant the Linux CSI tool[2] which works with 3.5.7 Linux kernel to OpenWrt. Furthermore, we disable one transmitting antenna of the transmitter at the driver level and fix the data rate to 8451.

Using these experiment configurations, we collect more than 1 million samples as dataset with which we develop and evaluate the algorithm.



(a) AUC of different feature selections.

(b) AUC of different classifiers.

Figure 4: Comparison between features and classifiers.

4.2 Overall Performance

Evaluation metric. Since the static situation is far more than the dynamic situation, there is a problem of imbalance in the collected data, so that the evaluation metric such as accuracy cannot effectively describe the classification effect of the classifier for the less data volume. We finally adopted the AUC (Area Under Curve) of the ROC curve as the criterion for the final model. In order to compare different feature selection methods and network structures on the same basis, the subsequent evaluation process is performed on the collected data sets, although the above algorithm can be easily rewritten as a real-time version.

Impacts of feature selection. As shown in Fig. 2, RSSI and AGC are closely related to motion detection. So we choose RSSI, AGC, MVE, the second principal component, LOF as features. Besides, movements may also have influence on other raw features reported by the driver, such as noise, antennaSel, SNR. Thus, we try different combinations of features to explore the impact of different combinations on the final test results.The x-axis in Fig. 4a represents additional combinations of features in addition to the selected features described in Section 3. None means that no new features are added. We use TextCNN as the classifier here. As shown in Fig. 4a, with the feature 'noise' added, AUC will improve a little bit. Thus, we finally choose the features, RSSI, AGC, noise, MVE, LOF and PCA, as the input of the classification methods.

Impacts of classifiers. Different classifiers have different characteristics. Fig. 4b is a comparison under the same feature selection (using RSSI, AGC, MVE, LOF, and PCA). As shown in Fig. 4b, as a single classifier, Bi-LSTM performs better than TextCNN and MLP. In addition, from the perspective of model convergence speed, both Bi-LSTM and TextCNN can gradually converge in about 10 epochs, while MLP requires dozens of epochs. With voting mechanism used, the AUC of the final classifier is 0.9931, and the accuracy on the dataset is 97.8%.

4.3 Influence Factor Study

In this section, we study the influence of hardware, software and environment on the detection performance. The goal of the following experiments is to get the optimal configuration of the system, so they are done before the data collection. The classification method used in the following experiments is a coarse-grained algorithm based on MVE.

We let a person walks through the detection area for 15 times, with the routes distributed evenly within the detection area. During

Table 1: Impacts of hardware factors.

Factor	Value	Number of trials	Pass rate
Tx RF parameter	High quality	56	73.2%
	Low quality	49	53.1%
NIC RF parameter	High quality	194	72.6%
	Low quality	158	68%
Antenna selection	Directional antenna	39	25.6%
	PCB antenna	47	42.6%
Antenna arrangement	Vertical misalignment	99	52.5%
Height of antenna	1.2m	42	69.1%
	0.6m	32	43.8%
Electromagnetic shield	Exists	53	69.8%
	Not exists	59	74.6%

Table 2: Impacts of software factors.

Factor	Value	Number of trials	Pass rate
Sample rate	10Hz	93	78.5%
	20Hz	69	81.2%
Transmitter power	100 mW	78	84.6%
	1 mW	75	89.3%
CSI Tools	Atheros CSI Tools	45	26.7%
Ntx	1	73	74.0%
	2	62	61.3%
Data rate	263	43	20.9%
	8451	257	71.9%

Table 3: Impacts of environment factors.

Factor	Value	Number of trials	Pass rate
Humidity	10%-18%	257	71.9%
	31%	23	91.3%
	51%	39	82.1%

this period, if there are more than 2 false positives or false negatives, we record the result as failed, otherwise passed. Under such statistical criteria, the following results are obtained.

From Table 1, we can conclude that, using transmitters and receivers with higher quality RF parameters is very helpful for improving the detection accuracy. The selection and position arrangement of antennas have an important impact on the accuracy of detections. Electromagnetic shield has less impact on the final detection effect.

From Table 2, we can get the following conclusions. Compared to using both transmitting antennas, we can achieve a better result by disabling one transmitting antenna. The selection of data rate has a great impact on the detection results. At low sampling rates, a small increase in the sampling rate has almost no effect on the effectiveness of the detection algorithm. Transmit power has less influence on the final detection effect. And the result of system based on CSI tools using Intel 5300 is much better than the ones based on Atheros CSI tools[14].

Since water can interfere with the transmission of electromagnetic waves, humidity has an apparent effect on the detection result as Table 3 shows. However, because the system needs to work in different environments, what we need is to ensure the diversity of data sources when collecting data, which include different environments, and then improve the environmental adaptability of the system through model training.

5 Conclusion

In this study, we design and implement a WiFi-based intrusion detection system that works on embedded devices. We have collected over one million CSI measurements and conduct extensive experiments to verify the performance of WiSafe and explore how system configuration affects. In the future, with more NIC vendors providing the interface for reading CSI data, we can even remotely update the router's firmware, making the router more than just communication infrastructure, but also a smart sensor.

References

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

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

[3] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[4] Jian Liu, Yan Wang, Yingying Chen, Jie Yang, Xu Chen, and Jerry Cheng. 2015. Tracking vital signs during sleep leveraging off-the-shelf wifi. In *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 267–276.

[5] Xuefeng Liu, Jiannong Cao, Shaojie Tang, and Jiaqi Wen. 2014. Wi-Sleep: Contactless sleep monitoring via WiFi signals. In *Real-Time Systems Symposium (RTSS), 2014 IEEE*. IEEE, 346–355.

[6] Kun Qian, Chenshu Wu, Zheng Yang, Yunhao Liu, and Zimu Zhou. 2014. PADS: Passive detection of moving targets with dynamic speed using PHY layer information. In *Parallel and Distributed Systems (ICPADS), 2014 20th IEEE International Conference on*. IEEE, 1–8.

[7] Kun Qian, Chenshu Wu, Yi Zhang, Guidong Zhang, Zheng Yang, and Yunhao Liu. 2018. Widar2. 0: Passive human tracking with a single wi-fi link. *Procs. of ACM MobiSys* (2018).

[8] Kun Qian, Chenshu Wu, Zimu Zhou, Yue Zheng, Zheng Yang, and Yunhao Liu. 2017. Inferring motion direction using commodity wi-fi for interactive exergames. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 1961–1972.

[9] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st annual international conference on mobile computing and networking*. ACM, 65–76.

[10] Yuxi Wang, Kaishun Wu, and Lionel M Ni. 2017. Wifall: Device-free fall detection by wireless networks. *IEEE Transactions on Mobile Computing* 16, 2 (2017), 581–594.

[11] Chenshu Wu, Zheng Yang, and Chaowei Xiao. 2018. Automatic radio map adaptation for indoor localization using smartphones. *IEEE Transactions on Mobile Computing* 17, 3 (2018), 517–528.

[12] Chenshu Wu, Zheng Yang, Zimu Zhou, Xuefeng Liu, Yunhao Liu, and Jiannong Cao. 2015. Non-invasive detection of moving and stationary human with wifi. *IEEE Journal on Selected Areas in Communications* 33, 11 (2015), 2329–2342.

[13] Jiang Xiao, Kaishun Wu, Youwen Yi, Lu Wang, and Lionel M Ni. 2012. Fimd: Fine-grained device-free motion detection. In *Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on*. IEEE, 229–235.

[14] Yaxiong Xie, Zhenjiang Li, and Mo Li. 2018. Precise power delay profiling with commodity Wi-Fi. *IEEE Transactions on Mobile Computing* (2018).

[15] Tong Xin, Bin Guo, Zhu Wang, Mingyang Li, Zhiwen Yu, and Xingshe Zhou. 2016. Freesense: Indoor human identification with Wi-Fi signals. In *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–7.

[16] Zheng Yang, Zimu Zhou, and Yunhao Liu. 2013. From RSSI to CSI: Indoor localization via channel response. *ACM Computing Surveys (CSUR)* 46, 2 (2013), 25.

[17] Zimu Zhou, Zheng Yang, Chenshu Wu, Longfei Shangguan, and Yunhao Liu. 2013. Towards omnidirectional passive human detection. In *INFOCOM, 2013 Proceedings IEEE*. IEEE, 3057–3065.

[18] Zimu Zhou, Zheng Yang, Chenshu Wu, Longfei Shangguan, and Yunhao Liu. 2014. Omnidirectional coverage for device-free passive human detection. *IEEE Transactions on Parallel and Distributed Systems* 25, 7 (2014), 1819–1829.