Convolutional neural network based evil twin attack detection in WiFi networks

Yinghua Tian[*], *Sheng* Wang, and *Long* Zhang

National Digital Switching System Engineering and Technological Research and Development Center (NDSC), Information Engineering University (IEU),450001 Zhengzhou, China.

Abstract. Evil Twin Attack (ETA) refers to attackers use a device to impersonate a legitimate hotspot. To address the problem of ETAs in the WiFi network, a Convolutional Neural Network (CNN) attack detection method is proposed. The method uses the preamble of the WiFi signal as the feature and uses it to train a CNN based classification model. Next, it uses the trained model to detect the potential ETA device by the inconsistent of the identity it claims and the signal feature. Experiments based on the commercial hardware demonstrate that the proposed method can effectively detect the Evil Twin Attack.

1 Introduction

Wireless Local Area Networks (WLANs) have become an important infrastructure of our modern life. The massively deployed WiFi hotspots in the urban areas are providing access to the Internet for both the public and commercial users. However, the openness of WiFi technology makes it vulnerable to several kinds of cyber-attacks.

One of the most serious threats to WLAN is Evil Twin Attack (ETA). As shown in figure 1, Hackers launch ETA by setting up a rogue access point (AP), which has the same MAC address and service set id (SSID) as the legitimate AP. [1] In this way, hackers create an “Evil Twin” of the legitimate AP, and users can't verify if the currently connected WiFi hotspots is the legitimate AP or the one impersonated by the hackers.

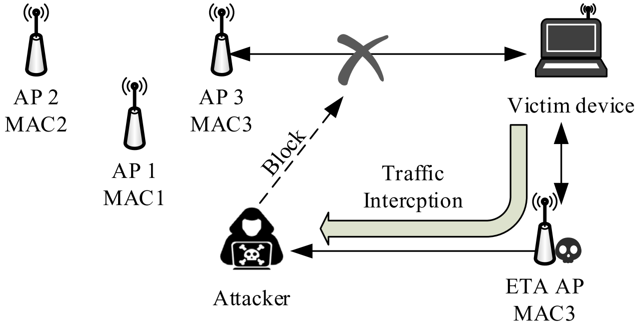


Fig. 1. Illustration of the Evil Twin Attack.

[*] Corresponding author: tian.ying.hua@foxmail.com

Hackers can also introduce the de-auth attack to increase the chance of success in spoofing users. Once the users connect the fake AP, all the network traffic will flow throw the rogue access point controlled by the hackers and leads to a series of high-level attacks to the victims, such as traffic eavesdrop, Man-in-the-Middle Attack (MITM), identity theft, and malware injection. [2-4]

Due to the lack of verification mechanism for hotspots, users can only identify the hotspots by the MAC address and SSID, which are easy to be imitated. Even encrypted networks are vulnerable to ETA. To address these problems faced by traditional security methods, various ETA detection methods are proposed. [5-7] Among them, a major direction is identifying the wireless device and detecting wireless attacks based on the features of raw wireless signals.

The diversity between different hardware of transmitters and the wireless channels brings subtle features to the raw signals. Those signal features are also called as radio frequency fingerprints or RF fingerprints. It is independent of the bit information in the WiFi frame and provides a reliable way to recognize the identities of wireless devices. Depending on the signal features used, these methods can be divided into two main categories: channel-based detection methods and hardware-based detection methods.

The channel feature-based detection methods mainly utilize the variation in the channel environment between illegal and legitimate devices due to their different locations for identification. In [7], a detection method based on channel measurement results is proposed to detect the fake MAC spoofing attacks. Chen et al. of [8] proposed an attack detection method based on the signal strength to identify malicious devices; [9] proposed an attack detection method using the different wireless channel transmission response from the device to the receiver at different locations to detect potential identity forgery attacks.

On the other hand, hardware-based detection methods identify devices by their transmitters' unique hardware features. Commonly used hardware features include carrier frequency offset, instantaneous signal spectrum, IQ imbalance, clock stability, phase shift, signal switching delay, etc. [10] proposed a technique called PARADIS to identify the WiFi devices by its multiple RF fingerprints. Researchers in [11] used the signal spectrum features of different devices to identify the devices in WLAN. [12] proposed a deep learning method to identify WiFi devices by its RF fingerprints in raw signals. In [13], an active detection method is proposed to identify WLAN devices by observing its responses to a series of crafted non-standard 802.11 frames.

In this paper, we propose a signal feature-based detection method for the WiFi ETA by the Convolutional Neural Network (CNN). The aim of our method is using the end-to-end signal feature, which is a combination of the channel and hardware features to identify the hotspots and detect the potential ETA attacks.

2 Methodology

2.1 Overview of our method

The proposed method works as an independent intrusion detection system (IDS), which only monitors the wireless passively and doesn't cost additional communication overhead of the local network. At the training stage, it trains the identification model with the signal features of the legitimate APs. Once the training is completed, it can recognize each of the legitimate APs by their signal features, and other devices would be recognized as "unknown". The method can detect ETAs by comparing the identification results of the wireless APs and the MAC address it claims. If there are ETA AP in the surrounding areas which claim a MAC address of one of the legitimate APs, but the identification result of its signal belongs to “unknown”. Then the method can send out the alarm of ETA and output the potential APs list under attacks.

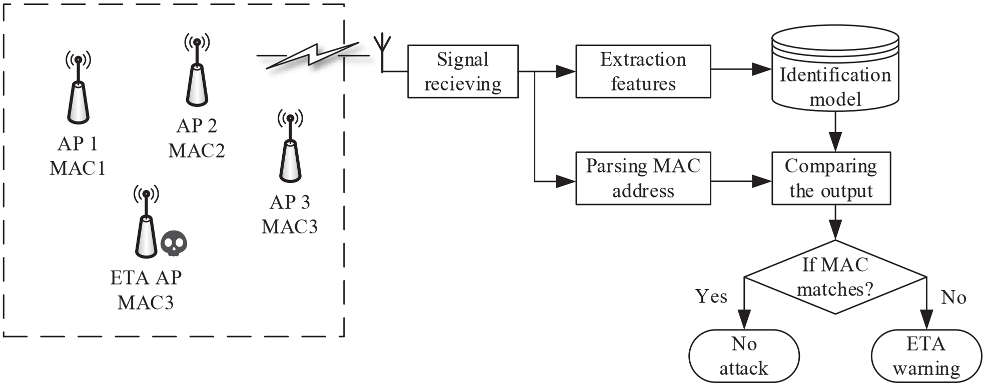


Fig. 2. Overview of the proposed ETA detection method.

The basic assumption of our method is both the hardware features and the channel features of a device keep stable during the process. Considering the WiFi APs are always deployed in the static positions and wouldn’t be moved during the service, our method can be applied to most scenarios.

2.2 Feature extraction

The structure and waveform of the WiFi signal preamble are fixed by the IEEE 802.11 standard and is only affected by the disturbance of different hardware and channels, which makes it a unique signal feature of each WiFi device. The structure of the 802.11 OFDM preamble is shown in figure 3, which consists of a short training sequence and a long training sequence. The short training sequence contains 10 short training symbols (STS) with a duration of $0.8\mu s$; the long training sequence contains 2 long training symbols (LTS) with a duration of $3.2\mu s$. The two training sequences are separated by a guard interval (GI) of $1.6\mu s$. In this paper, we extract both the short training sequence and the training sequence as the signal features.

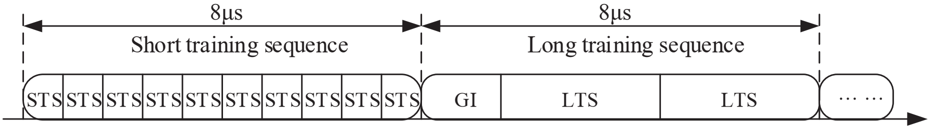


Fig. 3. Structure of the WiFi signal preamble.

The extraction method includes two stages. The first stage is signal detection, which is done by the delay correlation method. [14] The digital sampling $r[n]$ of the short training sequence is obtained by IQ sampling the signal at a sampling rate of 20MHz. As shown in the figure, due to the time domain repetition characteristic of the short synchronous sequence of the WiFi signal, it is possible to detect whether the WiFi signal is received by the normalized delay correlation operation. The expression for the STS detection as (1).

$$\overline{c}_D[n] = \frac{\sum_{k=0}^{D-1} r[n+k]r^*[n+k+D]}{\sum_{k=0}^{D-1} \left| r[n+k] \right|^2} \tag{1}$$

In the equation (1), $\overline{c}_D[n]$ is the output of normalized delay correlation operation and D denotes the recurrence interval of STS, which is 16 at the sampling rate of 20MHz. If the $\overline{c}_D[n]$ over exceeds the threshold, which is usually set at 0.8, it can be determined that a WiFi signal preamble is recieved.

In the next stage, we use the cross-correlation operation with 2 LTS as a local reference signal to search the starting point of the preamble.

$$\hat{\delta}_{LTS}=\text{argmax}\left|\sum_{k=0}^{L-1} r[n+k]s^{*}_{LTS}[n+k]\right|^{2} \tag{2}$$

In the expression (2), $\text{argmax}\left|\bullet\right|$ denotes finding the index of the peak location and $\hat{\delta}_{LTS}$ is the starting point of the first LTS. As shown in Equation 3, then we can extract the short training sequence and long training sequence (including the GI) from the raw signal.

$$\begin{aligned} r_{LTS} &= r[\hat{\delta}_{LTS}\text{-}32:\hat{\delta}_{LTS}+128] \\ r_{STS} &= r[\hat{\delta}_{LTS}-192:\hat{\delta}_{LTS}\text{-}32] \end{aligned} \tag{3}$$

2.3 Architecture of the detection network

Once we have extracted the preamble of the WiFi signal as the feature, we use a Convolutional Neural Network to identify each device. Our proposed architecture is inspired from [12], but with 2 pipelines handling the short training sequence and the long training sequence simultaneously. The structure of CNN used is shown in figure 4. It's inputted with the WiFi signal preamble and output the classification result of the device.

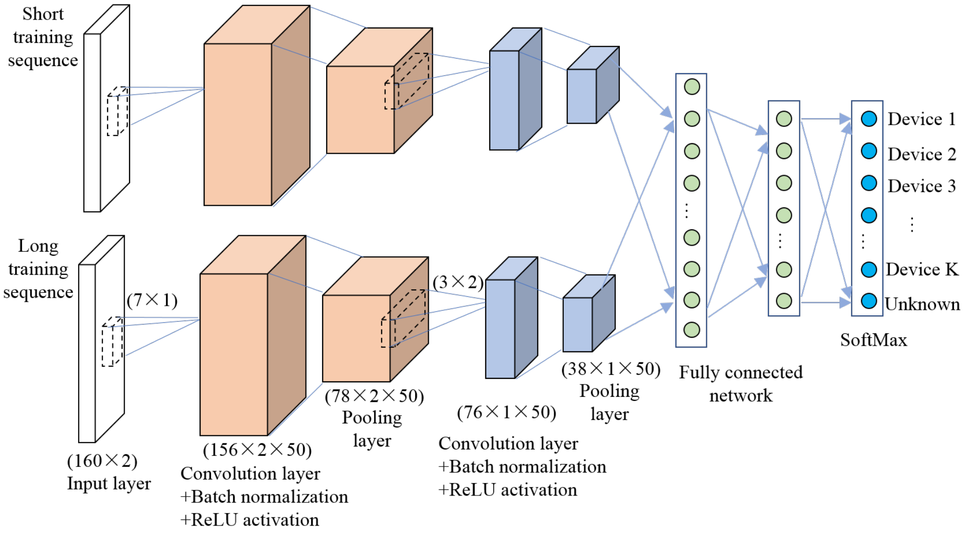


Fig. 4. Proposed convolutional neural network architecture.

The proposed CNN architecture contains two pipelines, handling the short training sequence and the long training sequence separately. Each pipeline consists of 2 convolution layers. The output of the pipelines is fused by a 2 layer fully connected network. The input of the CNN is the extracted preamble of the WiFi signal. The short training sequence and the long training sequence are feed into two separate pipelines. At the input layer of each pipeline, we expand the signal sample of 160×1 complex numbers into a 160×2 matrix of real numbers. The first convolution layer consists of 50 kernels, the size of which is 7×1, then it is followed with the batch normalization and the Rectified Linear Unit (ReLU) function. The output of ReLU is fed into a max-pooling layer, which downsampling the size of feature maps to $78\times2\times50$. The second convolution layer has a similar structure to the first convolution layer, but with a kernel size of 3×2, and the output size is $38\times1\times50$.

The convolution layers' output is then fed into a 2 layer fully connected network. The first layer has 256 neurons and the second layer has 80 neurons.

Then follows two separate channels for the short training sequence and the long training sequence and fuses them by a fully connected layer at the last stage. Finally, a softmax layer is used to output the classification result, which is the probability distributions of a list of device identities. We choose the max probability output as the final prediction result.

3 Experiment

3.1 Experimental setup

To evaluate the effectiveness of the proposed method, we build an ETA scenario and implement a prototype attack detection system based on the software-defined radio (SDR) platform in the real-world environment. The test scenario contains 8 legitimate APs and 8 client devices. All the devices are placed randomly in a room of $11\text{m}\times12\text{m}$. We use several routers, RaspberryPi 3b embedded devices, and a laptop with wireless NIC to simulate the public wireless environment and use a RaspberryPi 3b as the attack device to launch the fake wireless hotspot ETA. The goal of this attack is to convince the target device to connect to the Evil Twin device by constantly sending forged beacon frames with the same address as a legitimate hotspot.

In this experiment, we build a prototype ETA detection system based on SDR. The system uses a USRP (Universal Software Radio Peripheral) B210 SDR device to collect the wireless signal. The WiFi signal is captured at a sample rate of 20MHz and stored in the complex float32 format. Figure 5 shows a frame of captured WiFi signal and the interim outputs of our extraction method. The second curve in figure 5 is the delay correlation output, which triggering the extraction process. The third curve is the cross-correlation with LTS, the peak indicates the starting position of the first long training symbol.

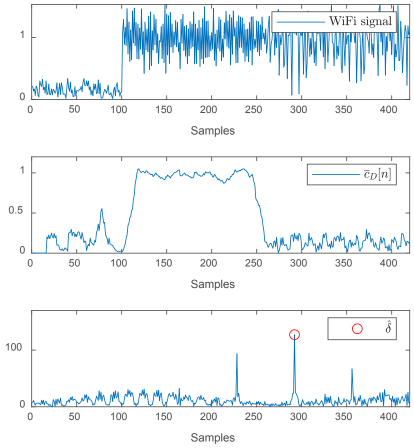


Fig. 5. Structure of the WiFi signal preamble.

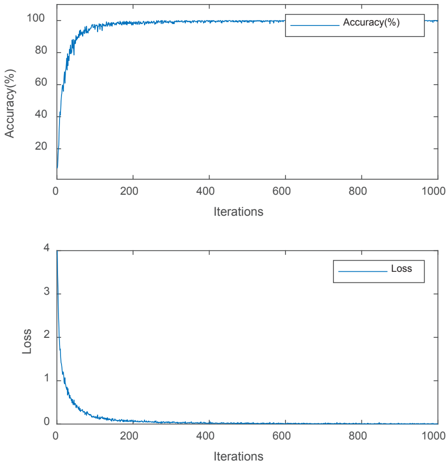


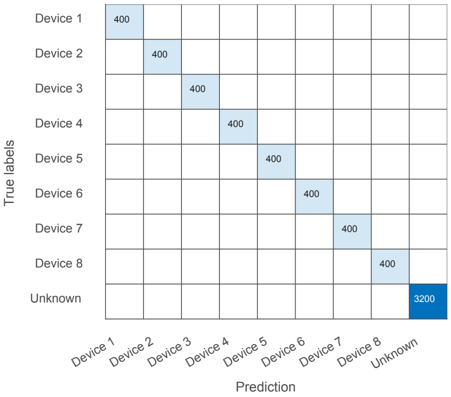
Fig. 6. Training progress of the neural network.

3.2 Training the network

The test scenario contains 8 legitimate APs and 8 client devices. For each device, 4000 frames of WiFi signals are collected for training. The legitimate APs' samples are labeled by their MAC address and the samples of client devices are marked as "unknown". Then we use the labeled dataset to train the network. The network is trained with the ADAM optimizer with a learning rate of 0.0001. A dropout rate of 50% is set for the full connected layers to counter the overfitting problem. The whole CNN is implemented on Matlab 2020a with the deep learning toolbox. The training progress is shown in figure 6.

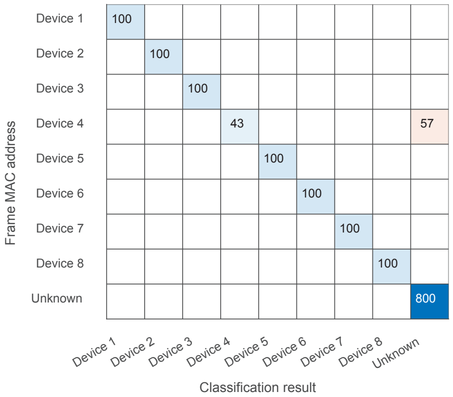
3.3 Detection test

First, we test our model's ability in identifying the WiFi devices. For each device in the test scenario, 400 WiFi frames are collected. All the frames are sent into the trained network and plot the confusion matrix for the test frames. As the confusion matrix is shown in figure 7, all the frames are classified correctly in our test scenario.



	Device 1	Device 2	Device 3	Device 4	Device 5	Device 6	Device 7	Device 8	Unknown
Device 1	400								
Device 2		400							
Device 3			400						
Device 4				400					
Device 5					400				
Device 6						400			
Device 7							400		
Device 8								400	
Unknown									3200

Fig. 7. Confusion matrix for device identification.



	Device 1	Device 2	Device 3	Device 4	Device 5	Device 6	Device 7	Device 8	Unknown
Device 1	100								
Device 2		100							
Device 3			100						
Device 4				43					57
Device 5					100				
Device 6						100			
Device 7							100		
Device 8								100	
Unknown									800

Fig. 8. Confusion matrix for ETA detection.

Next, we introduce an ETA device, which impersonates one of the legitimate APs by replicating its MAC address. Like the first test, we collect 100 frames for each device. Then we plot the confusion matrix by the classification result of our CNN and the MAC address of each frame. As shown in figure 8, there are 57 WiFi frames claims an address of Device 4, but are classified as Unknown by the classification model. The MAC addresses in the WiFi frames are inconsistent with their signal features, which means those frames are emitted by the ETA device.

4 Conclusion

In this paper, we propose an Evil Twin Attack detection method which first captures the WiFi signal, then extracts the short training sequence and the long training sequence as the feature of each WiFi hotspot, and finally trains a Convolutional Neural Network to verify the identity of each hotspot. The experimental results show that the proposed method can identify the hotspot with pre-knowledge and detect the Evil Twin Attack effectively. The limitation of our method is that it leverages both the hardware features and the channels feature to recognize the device, which theoretically makes it only can be applied in identifying the statically deployed hotspots. And the impact of noise on the performance isn't evaluated due to the lack of a controlled test environment. In the further study, we will evaluate the performance of our method in different conditions and develop its ability in the channel variations environment.

References

1. Yimin Song, Chao Yang, Guofei Gu. Who is peeping at your passwords at Starbucks?—To catch an evil twin access point. *IEEE International Conference on Dependable Systems & Networks*, 323 (IEEE, 2010)
2. Diogo Mónica, Carlos Ribeiro. Wifihop-mitigating the evil twin attack through multi-hop detection. *European Symposium on Research in Computer Security*, 21 (Springer, 2011)
3. Tuomas Tenkanen, Heli Kallio, Janne Poikolainen. Security Challenges of IoT-Based Smart Home Appliances. *Cyber Security: Power and Technology*, 271 (Springer, 2018)
4. Vineeta Jain, Vijay Laxmi, Manoj Singh Gaur. ETGuard: Detecting D2D Attacks using Wireless Evil Twins. *Computers & Security*, **83** 389 (2019)
5. Omar Nakhila, Cliff Zou. User-side wi-fi evil twin attack detection using random wireless channel monitoring. *In 2016 IEEE Military Communications Conference*, 1243 (IEEE, 2016)
6. EnChun Kuo, MingSang Chang, DaYu Kao. User-side evil twin attack detection using time-delay statistics of TCP connection termination. I*n 2018 20th International Conference on Advanced Communication Technology*, 211 (IEEE, 2018)
7. Jiang Peng, Hongyi Wu, Cong Wang, Chunsheng Xin. Virtual MAC spoofing detection through deep learning. *In 2018 IEEE International Conference on Communications*, 1 (IEEE, 2018)
8. Yingying Chen, Wade Trappe, Richard P. Martin. Detecting and localizing wireless spoofing attacks. *In 2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, 193 (IEEE, 2007)
9. Liang Xiao, Larry J. Greenstein, Narayan B. Mandayam, Wade Trappe. Channel-based spoofing detection in frequency-selective Rayleigh channels. *IEEE Transactions on Wireless Communications* **8** 5948 (IEEE, 2009)
10. Vladimir Brik, Suman Banerjee, Marco Gruteser, Sangho Oh. Wireless device identification with radiometric signatures. *In Proceedings of the 14th ACM international conference on Mobile computing and networking*, 116 (ACM,2008)
11. William C Suski II, Michael A. Temple, Michael J. Mendenhall, Robert F. Mills. Using spectral fingerprints to improve wireless network security. *In Proceedings of the 2008 IEEE Global Telecommunications Conference*, 1 (IEEE, 2008)
12. Kunal Sankhe, Mauro Belgiovine, Fan Zhou, Shamnaz Riyaz, Stratis Ioannidis, Kaushik Chowdhury. ORACLE: Optimized radio classification through convolutional neural networks. In 2019 IEEE Conference on Computer Communications, 370 (IEEE, 2019)
13. Sergey Bratus, Cory Cornelius, David Kotz, and Daniel Peebles. Active behavioral fingerprinting of wireless devices. *In Proceedings of the first ACM conference on Wireless network security*, 56-61 (ACM, 2008)
14. Chia-Horng Liu. On the design of OFDM signal detection algorithms for hardware implementation. *IEEE Global Telecommunications Conference*, **2**, 596 (IEEE, 2003)