12-12-2022

# Image Based Attack and Protection on Secure-Aware Deep Learning

Peng Wang



Recommended Citation
Wang, Peng, "Image Based Attack and Protection on Secure-Aware Deep Learning." Dissertation, Georgia State University, 2022.
doi: https://doi.org/10.57709/32623819

# Image Based Attack and Protection on Secure-Aware Deep Learning

by

Peng Wang

Under the Direction of Zhipeng Cai, Ph.D. and Wei Li, Ph.D.

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2022

ABSTRACT

In the era of Deep Learning, users are enjoying remarkably based on image-related services from various providers. However, many security issues also arise along with the ubiquitous usage of image-related deep learning. Nowadays, people rely on image-related deep learning in work and business, thus there are more entries for attackers to wreck the image-related deep learning system. Although many works have been published for defending various attack, lots of studies have shown that the defense cannot be perfect. In this thesis, one-pixel attack, a kind of extremely concealed attacking method toward deep learning, is analyzed first. Two novel detection methods are proposed for detecting the one-pixel attack. Considering that image tempering mostly happens in image sharing through an unreliable way, next, this dissertation extends the detection against single attack method to a platform for higher level protection. We propose a novel smart contract-based image sharing system. The system keeps full track of the shared images and any potential alteration to images will be notified to users. From extensive experiment results, it is observed that the system can effectively detect the changes on the image server even in the circumstance that the attacker erases all the traces from the image-sharing server. Finally, we focus on the attack targeting blockchain-enhanced deep learning. Although blockchain-enhanced federated learning can defend against many attack methods that purely crack the deep learning part, it is still vulnerable to combined attack. A novel attack method that combines attacks on PoS blockchain and attacks on federated learning is proposed. The proposed attack method can by pass the protection from blockchain and poison federated learning. Real experiments are performed to evaluate the proposed methods.

INDEX WORDS: Deep Learning, Security, One-pixel Attack, Blockchain, Federated Learning

Image Based Attack and Protection on Secure-Aware Deep Learning

by

Peng Wang

Committee Chair: Zhipeng Cai

Committee: Wei Li

Yingshu Li

Yichen Cheng

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

December 2022

## DEDICATION

This dissertation is dedicated to my mother Qun Xie and my friends for their endless support and love during my Ph.D. years. I cannot finish my Ph.D. without their love and encouragement.

# ACKNOWLEDGMENTS

I would never have been able to finish my dissertation without the guidance of my advisors and committee members, help from my group, and support from my family and my friends.

I would like to show my deepest gratitude to my advisor Dr. Zhipeng Cai and Dr. Wei Li. They provided me with an excellent environment for research, and gave me many opportunities to promote myself. They not only taught and encouraged me in my research but also inspired me to achieve self-actualization.

It is very grateful and a great honor to have Dr. Yingshu Li and Dr. Yichen Cheng in my committee, who gave me great supports for my Ph.D. study and spared time to participate in my defense committee.

Also many thanks go to colleagues in my group and department. Special thanks for my group colleagues Guangxi Lu, Honghui Xu, Kainan Zhang, Dr. Saide Zhu, Dr. Yan Huang, Dr. Yi Liang, Zuobin Xiong, and all roommates Dr. Kiril Kuzmin, Xiulong Yang, and Yixian Chen who helped me study and live in U.S..

Last but not least, it is a pleasure to thank everybody who made the dissertation possible, as well as express my apologies that I could not mention personally one by one.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

People are enjoying a huge convenience from image based Deep Learning services, such as image based text recognition services provided by Google, freshness of fruits and vegetables judging service provided by GL-iNet, face recognition service provide by smart phones, road sign recognition service in many self-driving cars, online image censorship services provided by many social media provider, etc. However, people are suffering from image based attacks while enjoying these convenient services. In this thesis, we propose two protective methods on image based deep-learning system and investigate security issue on blockchain enhanced federated learning.

In the first part of this dissertation, we analysis the mechanism of one-pixel attack [81], which is extremely hard to detect. Two detection algorithms are proposed which are trigger detection and candidate detection. Trigger detection applied a gradient descent based technique and candidate detection utilize a heuristic algorithm to solve the optimization problem. Real experiments are performed to evaluate the proposed methods.

In the second part of this dissertation, we proposed a blockchain and IPFS enhanced image sharing platform to ensure the image transaction and storage are free of alter from attacker. We also designed a pair of compression algorithm and hash algorithm, that guarantees original image and compressed image sharing the same hash. A prototype system is implemented for real data experiment. We present our solution act reliably and effectively under malicious attack.

In the third part of this dissertation, it is shown that the blockchain enhanced federated learning remains vulnerable to a combined attack. First, it is demonstrated that a PoS based blockchain enhanced federated learning system cannot preserve data integrity under long range attack. Second, it is shown that the federated learning is vulnerable to attacker

after the blockchain compromised. The experiment results demonstrate that the long range threatens the blockchain enhanced federated learning. Further more, attacker can achieve federated learning oriented attack such as poison attack.

The rest of this dissertation proposal is organized as follows. Chapter 2 summarizes the related literature. Chapter 3 studies the detection mechanism of one-pixel attack. Chapter 4 expands the protective work to platform oriented. In Chapter 5, we investigate the vulnerability of blockchain enhanced federated learning. Finally, in Chapter 6, we prove future direction and Chapter 7 concludes our work.

# CHAPTER 2

# BACKGROUND

Security of image related deep learning should not very unfamiliar to us since there has been plenty of research focused on secure aware deep learning on images. Although lots of existing studies focus on adversarial attack secure image sharing, it is hard to find existing works exploring the detection and protection against one-pixel attack, trackable image sharing platform and threaten on blockchain enhanced federated learning. In this section, the research progress and related literature are summarized.

## 2.1 Adversarial Attack

In adversarial attack, attackers intend to mislead classifiers by constructing adversarial samples. A. Nguyen *et al.* made efforts on fooling a machine learning algorithm [57] and found that DNNs give high confidence results to random noise, which indicates that universal adversarial perturbation in a single crafted perturbation can cause a misclassification on multiple samples. In [55, 63], back-propagation is used to find gradient information of machine learning models, and gradient descent methods are used to build adversarial samples.

Since it might be hard or impossible to learn the internal information of a DNN model in practice [13], some approaches have been proposed to generate adversarial samples without using the internal characteristics of DNN models [48]. Such approaches are called black box attack [28,56,62]. Particularly, a special type of black box attack is one-pixel attack, in which only one pixel is allowed to be modified. Under one-pixel attack of [70], an algorithm was developed to find a feasible pixel for malicious modification based on differential evolution that has a higher probability of finding an expected solution compared with gradient-based optimization methods. Due to the concealed modification of only one pixel, it becomes more difficult to detect one pixel attack. As mentioned in [70], the one-pixel attack requires

only black box feedbacks that are probability labels without any inner information of target network, like gradients or structure.

### 2.2 Detection of Adversarial Attack

On the other hand, research attentions are also paid to work out detection methods for adversarial attack. Papernot *et al.* provided a comprehensive investigation into the security problems of machine learning and deep learning, in which they established a threat model and presented "no free lunch" theory showing the tradeoff between accuracy and robustness of deep learning models. Inspired from the fact that most of the current datasets are compressed JPG images, some researchers designed a method to defend image adversarial attack using image compression . However, in their proposed method, a large compression may also lead to a large loss of classification accuracy of the attacked images, while a small compression cannot work well against adversarial attack. In [77], Neural Cleanse was developed to detect backdoor attack in neural networks, and some methods were designed to mitigate backdoor attack as well.

### 2.3 Secure Enhanced Image Sharing

Li and Lyu [32] proposed a method to detect deepfake videos using Artificial Intelligence (AI). The proposed method depends on an AI algorithm fighting another AI algorithm. Their technique relies on training convolutional neural networks (CNN) with manipulated and real figures. Testing was carried out using four different CNN networks with varying accuracy results between 84% to 99%. Their results look promising, however, the authors stated many challenges that yet remain to be solved. The presence of glitches in the currently obtained deepfake videos make their method give positive results. Therefore, they reckon that deepfake videos with a high resolution and quality will be hard to detect.

A US-based startup company called has developed a system involving mobile apps for typical users and freelancers for capturing images and saving them to the company's servers [32]. The purpose of saving the images is to preserve their integrity. Hence, any

forgery attempt can be easily discovered by comparing it with the image from the servers. They hope that in the future their technology will be used in collaboration with other social media parties that will verify any uploaded images with the images in the Truepic's servers and any change would therefore, be detected. Truepic also uses blockchain to store metadata of saved images to ensure immutability. This method relies heavily on trusting Truepic with the images and that all the uploaded images are untampered and real. It is not clear how the method works when inserting logos, text tickers, subtitles, or closed captions within the images or video frames.

There are also some works combined distributed storage and blockchain to build a more secure storage system. [21] [30] [76] are works utilizing the IPFS system and Blockchain to build their system. [21] propose a zigzag-based storage model to improve the blockchain enhanced IPFS with blockstorage model. The work improved the performance of BitSwap protocol but considered little about the security of the stored content. Work of [30] mainly target at Internet of Things(IoT). The paper proposed a blockchain based authentication mechanism that can prevent data faking attack in IPFS from malicious users. [76] proposed a method to increase the efficiency of file sharing via IPFS and blockchain. The method not only takes trustworthiness into design consideration, but also tried to solve the problem of proximity awareness during the process of file transfer. [58] designed a IPFS-blockchain based authenticity online publications. The work focuses on online book publication. In the work, the proposed solution is a framework that is extendible and adoptable for other type of digital and media content. [73] proposed a distributed reviewer reputation system. The work discussed that under a distributed context, how the privacy settings for both open peer review and reputation system varies and proposed an approach of supporting both anonymous and accountable reviews. Those works has different coupling method and all of them are designed to store general data and has no optimization for images.

When implementing image sharing system, one of the unavoidable problem is that how to specifically identify different images. The most popular solution is utilize hashing function, which come with new challenge that how to build a good hashing mechanism or hash

algorithm. [66] proposed a asymmetric encryption based hash function for secret share security modeling. The work makes optimal key selection based on Greay Wolf Optimization(OGWO) and the method shows less computational time and higher extreme entropy with high PSNR when comparing to the previous researches. [60] proposed a hash based image authentication. In the work, a one layer watermark tech is embedded. Furthermore, the paper compounds DWT (Discrete Wavelet Transform) and SVD (Singular Value Decomposition) to generate perceptual hash of images. However, all the above listed works ignored a pretty common application scenario that in an image sharing system an uploaded images will be compressed into different resolution and even a hash function is efficient, to hash large amount of image is still time consuming.

# CHAPTER 3

# DETECTION MECHANISMS OF ONE-PIXEL ATTACK

## 3.1 Introduction

Deep learning is an artificial intelligence technique that follows the structure of human's brain and imitates the neural cells in human brain [40]. Over the past decades, deep learning has made significant progresses in speech recognition, natural language processing [69], computer vision [38], image classification [19], and privacy protection [11, 48, 100]. Especially, with the increase of data volume, traditional machine learning algorithms, such as SVM [75] and NB [41], suffer a performance bottleneck, in which adding more training data cannot really enhance their classification accuracy. Differently, the deep learning classifiers can continue to get improvements if more data is fed. However, it has been revealed that artificial perturbation can make the deep learning models misclassify easily. A number of effective methods have been proposed to produce so-called "adversarial samples" to fool the models [63, 96] and some work focused on fighting against adversarial attack [17, 86, 88]. Among the existing works, one-pixel attack takes an extreme scenario into consideration, where only one pixel of an image is allowed to be modified to mislead the classification models of Deep Neural Network (DNN) such that the perturbed image is classified to another label different from the image's original label [70].

### 3.1.1 One-Pixel Attack

Among the existing works, one-pixel attack is harmful to the performance guarantee of DNN-based information systems. Via modifying only one pixel in an image, the classification of the image may change to an irrelevant label, leading to performance degradation of DNN-based applications/services and even other serious consequences. For examples, in medical image systems, one-pixel attack may make doctor misjudge the disease of patients; and in

auto-driving vehicles, one-pixel attack may cause serious traffic accidents on roads.

More importantly, one-pixel attack is more threatening than other types of adversarial attack as it can be implemented easily and effectively to damage system security. Since one-pixel attack is a type of black box attack, it does not require any additional information of the DNN models. In practice, one-pixel attack only needs the probabilities of different labels instead of the inner information about the target DNN models, such as gradients and hyper-parameters. The effectiveness of one-pixel attack towards DNNs has been validated in [70], where the attack success rate is 31.40% on the original CIFAR-10 image dataset and 16.04% on the Image-Net dataset. Such success rate is large enough to break down an image classification system.

Therefore, to avoid the loss of system performance, detecting one-pixel attack becomes an essential task.

### 3.1.2 Technical Challenges

The following two facts result in the difficulty of examining one-pixel attack in images.

**1) Extremely small modification.** One-pixel attack modifies only one pixel in an image, which is significantly less than other types of adversarial attack. This makes the detection of one-pixel attack very challenging.

**2) Randomness of pixel modification.** For an image, there may be more than one feasible pixel that can cause the change of classification. In [70], one-pixel attack randomly selects one of those feasible pixels to mislead the classifiers. Such randomness makes the detection of the one pixel attack become harder.

### 3.1.3 Contributions

In this chapter, we develop two methods to detect one-pixel attack for images, including trigger detection and candidate detection. In the trigger detection method, based on a concept named "trigger" [77], we use gradient information of the distance between the pixels and target labels to find the pixel that is modified by one-pixel attack. By considering

the property of one-pixel attack, in the candidate detection method, we design a differential evolution-based heuristic algorithm to find a set of candidate victim pixels that may contains the modified pixel. Intensive real-data experiments are well conducted to evaluate the performance of our two detection methods. To sum up, this paper has the following multi-fold contributions.

- To the best of our knowledge, this is the first work to study the detection of one-pixel attack in literature, which can contribute to the defense of one-pixel attack in future research.
- Two novel detection mechanisms are proposed, in which the modified pixels can be identified in two different ways based on our thorough analysis on one-pixel attack.
- The results of real-data experiments validate that our two detection methods can effectively detect one-pixel attack with satisfied detection success rates.

The rest of this paper is organized as follows. In Section Related Works, the existing works on adversarial attacks and detection schemes are briefly summarized. The attack model and the detection model are presented in Section System Models. Our two detection methods are demonstrated in Section Design of Detection Methods. After analyzing the performance of our methods in Section Performance Validation

## 3.2 Related Works

One pixel attack is a special type of adversarial attack and is designed based on differential evolution scheme. Thus, this section summarizes the most related literatures from the following two aspects: adversarial attack and detection of adversarial attack.

### 3.2.1 Adversarial Attack

In adversarial attack, attackers intend to mislead classifiers by constructing adversarial samples. A. Nguyen *et al.* made efforts on fooling a machine learning algorithm [57] and

found that DNNs give high confidence results to random noise, which indicates that universal adversarial perturbation in a single crafted perturbation can cause a misclassification on multiple samples. In [55, 63], back-propagation is used to find gradient information of machine learning models, and gradient descent methods are used to build adversarial samples.

Since it might be hard or impossible to learn the internal information of a DNN model in practice, some approaches have been proposed to generate adversarial samples without using the internal characteristics of DNN models. Such approaches are called black box attack [28, 56, 62]. Particularly, a special type of black box attack is one-pixel attack, in which only one pixel is allowed to be modified. Under one-pixel attack of [70], an algorithm was developed to find a feasible pixel for malicious modification based on differential evolution that has a higher probability of finding an expected solution compared with gradient-based optimization methods. Due to the concealed modification of only one pixel, it becomes more difficult to detect one pixel attack. As mentioned in [70], the one-pixel attack requires only black box feedbacks that are probability labels without any inner information of target network, like gradients or structure.

### 3.2.2 Detection of Adversarial Attack

On the other hand, research attentions are also paid to work out detection methods for adversarial attack. Papernot *et al.* provided a comprehensive investigation into the security problems of machine learning and deep learning, in which they established a threat model and presented "no free lunch" theory showing the tradeoff between accuracy and robustness of deep learning models. Inspired from the fact that most of the current datasets are compressed JPG images, some researchers designed a method to defend image adversarial attack using image compression . However, in their proposed method, a large compression may also lead to a large loss of classification accuracy of the attacked images, while a small compression cannot work well against adversarial attack. In [77], Neural Cleanse was developed to detect backdoor attack in neural networks, and some methods were designed to mitigate backdoor attack as well.

Compared with the existing works, this paper is the first work that focuses on the detection of one-pixel attack. In particular, two novel detection mechanisms are proposed with one using gradient calculation-based method and the other using differential evolution-based method.

## 3.3 System Models

The attack model and detection model in our considered DNN-based information systems are introduced as follows.

### 3.3.1 Attack Model

In this paper, the attack model of [70] is taken into account, in which an adversarial image is generated by modifying only one pixel in the victim image. The purpose of one-pixel attack is to maliciously change the classification result of a victim image from its original label to a target label. As shown in Fig. 3.1, the image is correctly classified as an original label, "Sheep", by a given DNN model. After being modified one pixel, the output label with the highest preference of the model is changed to a target label, "Airplane", leading to a wrong classification result. The attackers perform black box attack only, which means they have the accessibility to the probability labels and can not get the inner information of the network. Also, considering that the attacker aims to make attack as efficiently as possible, it is supposed that all the images in the dataset are altered.

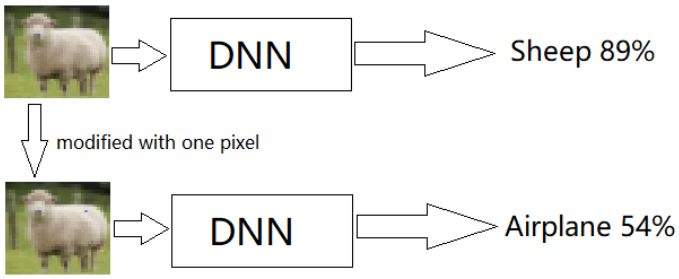


Figure 3.1. Illustration of One-Pixel Attack

### 3.3.2 Detection Model

Suppose that a set of adversarial images, which are modified by the aforementioned one-pixel attack, are given to the system. With these given images, our objective is to distinguish which pixel has been modified by one-pixel attack.

To detect one pixel attack, two novel methods are developed. The first method is "Trigger Detection" to identify the modified pixel, and the second one is "Candidate Detection" to find a set of victim pixels. The "Trigger Detection" model is designed for white-box detection that requires all the network information including inner gradients and network structure. In the trigger detection model, we first propose a new concept named "trigger" for image data and then detect the trigger in a given adversarial image. If the detected trigger is the pixel modified by one-pixel attack, our detection is successful. The "Candidate Detection" is for black-box detection, where only the output probabilities of labels are needed for the detection. In the candidate detection model, we aim to find a set of pixels as the candidate victim pixels. If the selected victim pixels include the pixel modified by one-pixel attack, our detection is successful. The details of our two detection mechanisms are demonstrated in Section Design of Detection Methods.

## 3.4 Design of Detection Methods

Our proposed detection methods are described as follows. For a better presentation, the major notations are summarized in Table 3.1.

### 3.4.1 Trigger Detection Method

**Main Steps** Formally, the trigger of an image is defined to be the pixel that has the greatest impact on the model classification. Thus, any image, which has a properly modified trigger, should have a higher confidence on a target label and will be likely to be classified as the target label regardless of other unchanged image features. In other words, the classification result would be changed to a target label if the trigger is modified properly

Table 3.1. Notations

| Notations | Meaning |
|---|---|
| $\mathbb{L}$ | output labels in a DNN model |
| $L_{in}$ | the original label |
| $L_t$ | the target label of one-pixel attack |
| $A(\cdot)$ | the function that applies a trigger to an image |
| $x$ | the original image |
| $x'$ | the modified image |
| $x_{i,j,c}$ | a pixel of $x$ and $x'_{i,j,c}$ represent a pixel of $x'$<br>$i$ and $j$ are the $x$ and $y$ coordinates of the pixel, respectively,<br>and $c$ is the color channel |
| $\Delta$ | a trigger of an image |
| $Loss(\cdot)$ | the loss function measuring classification error |
| $x_{next}$ | an element of the candidate solution |
| $T_{max}$ | maximum iteration numbers |

using DNNs' properties.

Let $\mathbb{L}$ represent the set of output labels in a DNN model. For any adversarial image, we assume its original label is $L_{in} \in \mathbb{L}$ and its target label is $L_t \in \mathbb{L}$, where $in \neq t$. Under one-pixel attack model, the modification on the trigger pixel can transform all inputs of $L_{in}$ to be classified as $L_t$.

Motivated by the above observations, one-pixel attack can be detected via identifying the triggers of adversarial images according to the following steps.

- **Step 1.** For a given label, we treat it as a potential target label in one-pixel attack. We use an optimization-based scheme to find the trigger that can misclassify all samples from other labels into the target label.

- **Step 2.** We repeat Step 1 for each output label in the DNN model and obtain $N$ potential triggers, in which $N = |\mathbb{L}|$ is the number of labels of the DNN model.

- **Step 3.** After calculating $N$ potential triggers, we measure the size of each trigger. The size of a trigger is defined to be the modified RGB value of the trigger pixel. Then, the outlier detection algorithm of [5] is adopted to detect whether the perturbation of each potential trigger is significantly smaller than others'. A significant outlier is likely to indicate a real trigger, and the label matching the real trigger is the target label in one-pixel attack.

**Trigger Identification** The details of our optimization-based scheme for identifying trigger are addressed below.

Suppose $\boldsymbol{X}$ is a set of clean images without modification. A generic form of injecting trigger for any original image, $\boldsymbol{x} \in \boldsymbol{X}$, is given in Eq. (3.1).

$$A(\boldsymbol{x}, m, \boldsymbol{\Delta}) = \boldsymbol{x}', \tag{3.1}$$

where $A(\cdot)$ represents the function that applies a trigger to $\boldsymbol{x}$. Correspondingly, the modified image is denoted as $\boldsymbol{x}'$. Let $\boldsymbol{x}_{i,j,c}$ represent a pixel of $\boldsymbol{x}$ and $\boldsymbol{x}'_{i,j,c}$ represent a pixel of $\boldsymbol{x}'$, where $i$ and $j$ are the $x$ and $y$ coordinates of the pixel, respectively, and $c$ is the color channel. The relationship between $\boldsymbol{x}$ and $\boldsymbol{x}'$ can be mathematically expressed by Eq. (3.2).

$$\boldsymbol{x}'_{i,j,c} = (1 - m) \cdot \boldsymbol{x}_{i,j,c} + m \cdot \boldsymbol{\Delta}_{i,j,c}, \tag{3.2}$$

in which $\boldsymbol{\Delta}$ represents a trigger of $\boldsymbol{x}$, and $m \in [0, 1]$ describes how much $\boldsymbol{\Delta}$ can overwrite the original image. Particularly, when $m = 1$, the pixel of trigger completely overwrites the original color; and when $m = 0$, the original color is not modified at all. The original image $\boldsymbol{x}$ is classified to the original label $L_{in}$, and the modified image $\boldsymbol{x}'$ is classified to the target label $L_t$.

Then, given the target label $L_t$, the problem of finding a trigger can be formulated as a multi-objective optimization problem, *i.e.*,

$$\min_{m,\boldsymbol{\Delta}} Loss(L_t, f(A(\boldsymbol{x}, m, \boldsymbol{\Delta}))) + \boldsymbol{m}, \forall \boldsymbol{x} \in \boldsymbol{X}. \tag{3.3}$$

In Eq. (3.3), $Loss(\cdot)$ is the loss function measuring classification error that is computed by cross entropy, and $f(\cdot)$ is the prediction function of the DNN model.

In this paper, $L1$ norm of $m$ is adopted to measure the magnitude of the trigger. By solving the above optimization problem, we get the trigger, $\boldsymbol{\Delta}$, for each target label and its $L1$ norm. Next, in Step 3, we identify the triggers that show up as outliers with smaller $L1$

norm by utilizing the outlier detection algorithm of [5].

### 3.4.2 Candidate Detection Method

Notably, to generate adversarial images, one-pixel attack of [70] uses differential evolution algorithm to randomly select a pixel that can lead misclassification on an image. Mathematically speaking, for the problem of image generation, the selected pixel is a feasible solution and may not be an optimal solution. Therefore, the obtained trigger pixel might not be actually modified in one-pixel attack, resulting in failed detection. In order to improve the attack detection success rate, the goal of our candidate detection method is to find a set of victim pixels (*i.e.*, a set of feasible solutions), each of which satisfies the requirement of adversarial image generation in one-pixel attack.

**Problem Formulation** Without loss of generality, we assume an input image can be represented by a vector in which each scalar element represents one pixel. In a DNN model, $f(\cdot)$ receives an image as input and gives confidence of $N$ labels. Accordingly, the probability of $\boldsymbol{x}$ being classified to a label $L \in \mathbb{L}$ is $f(\boldsymbol{x})[L]$. For an original image $\boldsymbol{x}$, an additive adversarial perturbation is represented by a vector $\boldsymbol{v}$. The modification degree is measured by the length of $\boldsymbol{v}$, and the allowable maximum modification is 1 in one-pixel attack model. The problem of generating adversarial images using one-pixel attack is formulated as follows.

$$\begin{aligned} &\max_{\boldsymbol{\Delta}} f(\boldsymbol{x}+\boldsymbol{\Delta})[L] \\ &\text{s.t.}||\boldsymbol{\Delta}|| \leq 1. \end{aligned} \tag{3.4}$$

**Differential Evolution-based Heuristic Algorithm** To obtain the solution to Eq. (3.4), a heuristic algorithm is designed based on differential evolution (DE), which brings the following benefits.

- DE gives a higher probability of finding global optimal solutions as well as a lower probability of getting "trapped" in the local solutions compared with gradient descent

and greedy searching algorithms.

- DE requires less information from the optimization objectives. DE does not require gradient information from the dataset, which means it even does not require the problem to be differentiable. Under the extremely strict constraint of modifying only one pixel in an image, the problem is not differentiable and can be effectively resolved by DE.

- To detect one-pixel attack, we only need to know whether the confidence changes after modifying a pixel, which can be formulated and solved in a simple way using DE.

In this paper, we encode the perturbation into an array (*i.e.*, a candidate solution) which is optimized (evolved) by differential evolution. One candidate solution contains a fixed number of perturbations, and each perturbation is a tuple holding five elements including $x - y$ coordinates and RGB value of the perturbation, where one perturbation modifies one pixel. The DE algorithm is performed iteratively and is terminated when one of the two conditions is satisfied: (i) the maximum number of iteration $T_{max}$ is reached; or (ii) the probability of being classified to the target label exceeds a threshold $p_{th}$. Let $N_{ini}$ be the initial number of candidate solutions (population) and $N_c$ be the number of candidate solutions (*i.e.*, children) produced in each iteration. At the $(g+1)$-th iteration, $N_c$ candidate solutions are produced from the $g$-th iteration via the following DE formula.

$$x_{next}(g+1) = x_{r_1}(g) + F(x_{r_2}(g) - x_{r_3}(g)), \tag{3.5}$$

where $x_{next}$ is an element of the candidate solution, $r_1$, $r_2$, $r_3$ are random values with $r_1 \neq r_2 \neq r_3$, and $F$ is the scale parameter. After being generated, each candidate solution competes with their parents according to the index of population, and the winners survive in the next iteration. When the algorithm terminates, $C$ candidates are output.

The pseudocode of our algorithm is shown in Algorithm 1. For each image, the above algorithm will go through all the $N$ labels, *i.e.*, the "for loop" in lines 4-19 of Algorithm 1.

**Algorithm 1** Algorithm of Candidate Detection Method

**Input:** An adversarial image generated by one-pixel attack, a DNN classifier, and a label set $\mathbb{L}$ with $|\mathbb{L}| = N$
**Output:** A set of pixels containing $C$ candidate victim pixels

```
1: Initialize model with the target DNN model
2: Randomly chose N_ini pixels from the image, and for each point, randomly set a color as parent pixel set.
3: Calculate the confidence of the image on all N labels
4: for all each label L ∈ 𝕃 do
5:     while True do
6:         Calculate the change of the confidence on L when modifying with parent pixels
7:         Generate offspring pixels based on Eq. (5)
8:         Calculate the change of the confidence on L when modifying with offspring pixels
9:         Select N_c pixels with the highest confidence changed as new parent pixels.
10:        if All top C confidences changed on targets are larger than p_th then
11:            Save the top C pixels as candidate victim pixels
12:            Set AttackSucc = True
13:            BREAK while loop
14:        end if
15:        if while loop is over T_max times then
16:            BREAK while loop
17:        end if
18:    end while
19: end for
20: if AttackSucc is True then
21:    return C candidate victim pixels
22: else
23:    return Fail to find candidates
24: end if
```

Algorithm 1. Algorithm of Candidate Detection Method

For each label, the candidate selection process will run up to $T_{max}$ iterations, *i.e.*, the "while loop" in lines 5-18 of Algorithm 1. Each iteration costs a constant time to generate $N_c$ children and pick $N_c$ winners. As a result, the time complexity of Algorithm 1 is $O(N \cdot T_{max})$.

## 3.5 Performance Validation

In this section, extensive real-data experiments are conducted to evaluate the performance of our two detection methods.

### 3.5.1 Experiment Settings

Our experiments adopt CIFAR-10 as the dataset and VGG-16 as the DNN model. Table 3.2 shows the structure of VGG-16 network which is the same as the network used in one-pixel attack. After training, we get the model with the accuracy as shown in Table 3.3.

Table 3.2. Network Structure of VGG-16

| |
|---|
| Conv2d layer (kernel=3, stride=1, depth=64) |
| Conv2d layer (kernel=3, stride=1, depth=64) |
| max pooling layer(kernel=2, stride=2) |
| Conv2d layer (kernel=3, stride=1, depth=128) |
| Conv2d layer (kernel=3, stride=1, depth=128) |
| max pooling layer(kernel=2, stride=2) |
| Conv2d layer (kernel=3, stride=1, depth=256) |
| Conv2d layer (kernel=3, stride=1, depth=256) |
| Conv2d layer (kernel=3, stride=1, depth=256) |
| max pooling layer(kernel=2, stride=2) |
| Conv2d layer (kernel=3, stride=1, depth=512) |
| Conv2d layer (kernel=3, stride=1, depth=512) |
| Conv2d layer (kernel=3, stride=1, depth=512) |
| max pooling layer(kernel=2, stride=2) |
| Conv2d layer (kernel=3, stride=1, depth=512) |
| Conv2d layer (kernel=3, stride=1, depth=512) |
| Conv2d layer (kernel=3, stride=1, depth=512) |
| max pooling layer(kernel=2, stride=2) |
| flatten layer |
| fully connected(size=2048) |
| fully connected(size=2048) |
| softmax classifier |

Then, we adopt the first 8000 succeed attacks as the dataset for our experiment.

Table 3.3. Classification Accuracy of VGG-16

| **Model** | **Accuracy on Test Set** | **Claimed Accuracy on Test Set** |
|---|---|---|
| VGG-16 | 93.4% | 94% |

To measure the performance of one-pixel attack, we calculate the classification accuracy of VGG-16 model on the test images. Also, we calculate the success rate of launching one-pixel attack, in which "airplane" is set as the target label. The success of one-pixel attack means after being modified a pixel, an image whose original classified label is not airplane is misclassified to airplane by the DNN network. Moreover, to reduce the influence caused by the randomness in the differential evolution algorithm of one-pixel attack, the classification process is repeated 5 times, and the average accuracies and their variances are presented in Table 3.4.

Table 3.4. Success Rate of One-Pixel Attack

| **Model** | **Accuracy on Test Set** | **Success Rate of One-Pixel Attack** |
|---|---|---|
| VGG-16 | 93.4% ± 1.65% | 17.3% ± 3.61% |

Specifically, one-pixel attack is launched towards 60,000 testing images and succeeds in making 8655 non-airplane images get classified to airplane. In the experiments, we pick 8000 such adversarial images to evaluate our detection method.

### 3.5.2 Performance Metrics

The performance metrics are introduced as follows.

- **Label Confidence.** The Label confidence is the confidence of different labels given by the DNN model to an image. For a label, higher confidence means higher probability that the image is classified to the label.

- **Detection Success Rate.** The definition of successful attack detection is different

in our two detection methods. In trigger detection, our detection is successful if the detected trigger is the pixel modified by one-pixel attack; while, in the candidate detection model, our detection is successful if the pixel modified by one-pixel attack is included in the set of selected victim pixels. For both two detection methods, the detection success rate is defined as the ratio of the number of successful detection to the number of adversarial images.

### 3.5.3 Performance of Trigger Detection

Since L1 norm has a better feature selection performance and interpretability [83], our trigger detection method uses $L1$ norm to measure the distance between the pixel and a label. If the $L1$ norm of a pixel is obviously different from others, we can consider the pixel is infected. Fig. 3.2 shows the L1 norm of all pixels of an infected image.

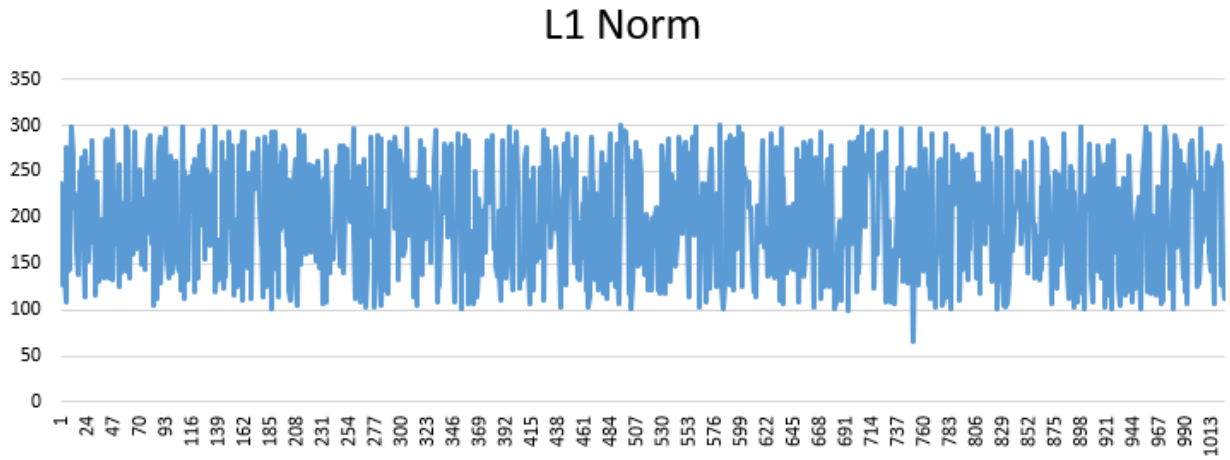


Figure 3.2. L1 Norm of an Infected Image to Label Airplane

From Fig. 3.2, one can find that the $L1$ norm of the 751st pixel is obviously different from others. With verification, we know that the 751st pixel is the pixel modified in one-pixel attack. Also, to understand how the affected target label is related to the modified pixel, we calculate the $L1$ norm of the infected pixel to different labels. In Fig. 3.3, we can find the $L1$ norm to airplane is lower than that to the other labels. Thus, our approach can also

determine which label is target label.

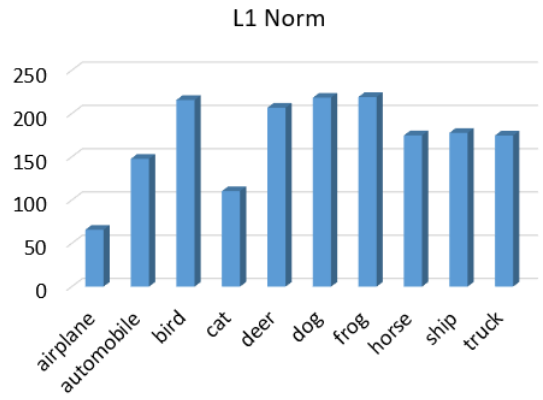


Figure 3.3. L1 Norm of an Infected Image to Airplane Label

The average detection success rate of our trigger detection method is 9.1%.

### 3.5.4 Performance of Candidate Detection

In our candidate detection method, the initial number of candidate solutions and the number of produced candidate solutions are set to be 400, *i.e.*, $N_{ini} = N_c = 400$, the maximum number of iteration is $T_{max} = 100$, the scale parameter is $F = 0.5$ and the threshold for probability of being classified to the target label is $p_{th} = 90\%$. To eliminate the influence of random variables in our Alg. 1, we run the experiment 5 times with the fixed parameter settings and present the results in Table 3.5. Moreover, to investigate the impact of the size of candidate set, we also compare the detection success rates when $C$ is set to be different values.

Table 3.5. Detection Success Rate of Candidate Detection

| **Experiment** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| Success Rate (%) | 20.4 | 24.3 | 30.1 | 21.9 | 26.7 |

As shown in Fig. 3.4, when $C = 1$, the success detection rate is 5.4% smaller than the success detection rate of our trigger detection method. Particularly, with $C = 1$, both the

trigger and the candidate detection methods output one detected pixel but differ in their pixel selection schemes: (i) in out trigger detection, the trigger pixel is an optimal solution of trigger identification problem as well as has a smallest value of $L1$ form; while (ii) in our candidate detection, the only one candidate victim pixel is randomly selected by differential evolution-based heuristic algorithm. From the definition of trigger pixel in this paper, the probability of trigger pixel being modified in one-pixel attack is lager than that of other pixels. Therefore, the trigger detection method outperforms the candidate detection method when $C = 1$, confirming that the idea of finding trigger pixel to detect one-pixel attack is solid.

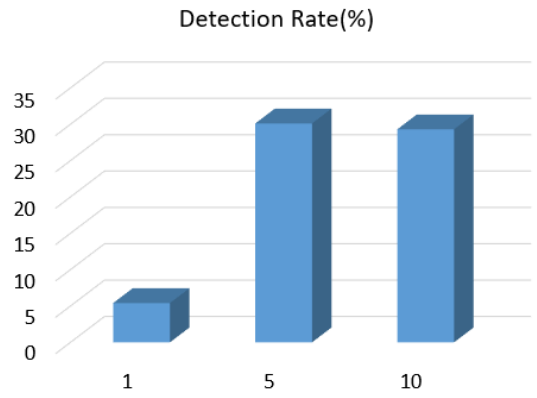


Figure 3.4. Impacts of the Number of Output Candidates

When $C$ is increased from 1 to 5, the detection success rate grows from 5.4% to 30.1%. The main reason is that with a larger number of output candidate pixels, more possible modified pixels can be examined, and the probability of finding the actual modified pixel is increased.

However, when $C$ is increased from 5 to 10, the detection success rate nearly remains the same (in Fig. 3.4, the subtle difference of the detection success rate results from the randomness of DE algorithm). This illustrates that only increasing the number of candidate pixels may not always enhance the detection success rate. In a summary, for the detection success rate, the marginal benefit of enlarging the number of candidate pixels is diminishing,

and thus setting an appropriate value to $C$ can help effectively and efficiently detect one-pixel attack (*e.g.* $C = 5$ in our experiments).

# CHAPTER 4

# BLOCKCHAIN BASED TRACKABLE SECURE IMAGE SHARING SYSTEM

## 4.1 Introduction

In the recent years, the rise of deep learning have led the way to the production of adversarial image [1].

Image, as one of the most used media format, takes important roles in our daily life. Especially with the rise of deep learning, the application combined with image are showing incredible opportunities. In hospital, the doctor may use patients' medical image to make diagnosis. Auto driving system may use image from captured street view to assist driving. Most of the learning related image application requires large amount of training images and image owners share their image with each other becomes the solution for dealing with lack on training images [79]. However, it cannot be ignored that with such convenient environment, the image data may be susceptible to attacks, tampering, and misplacement during the transmission or storage process. For example, in Deep Neural Networks (DNNs) in medical application, the medical images like X-ray images for judging diabetic retinopathy, Dermoscopy are easily to be modified and fool the trained DNNs model [54] [16] [102]. Coincidentally, auto-driving are also faced with the same problem. Liu *et al.* [50] illustrated that some of the existed model are vulnerable for attackers, who only have access to datasets, which are images. And such kind of attack may lead to severe consequences, like murdering human lives and make it looks like an accident.

Currently there are already some methods of defencing image oriented attacks. Zhou *et al.* [104] proposed GAN based anomal image detection. Hu *et al.* [36] proposed utilizing the vulnerability density to design a signature characteristic to detect adversarial attacks. Yao *et al.* [64] disclosed that CapsNets' reconstructive attack and the perturbations can

cause the image to appear not much difference against the target class hence become non-adversarial [99].

However, the existed methods are mainly concentrate on the defencing the attack on image itself [27] [74] [72]. In this paper, we novelly propose a method that not against the original image. Instead, we propose make sure the image sharing path from resource to client. The traditional solutions are mainly proving the authenticity of networks and the security of sharing server and the integrity are guaranteed by server security. However, lots of attacks are happened in mid-man attack [59] [42] and even some methods that can protect the authentication of server, the integrity may also in danger [33]. Thus we propose utilize blockchain and decentralized file system to build a trackable secure image sharing system.

Nowadays, cloud storage and cloud services has taken an important role in image sharing , which leads to the increasing of vulnerability of image sharing process. Thus to protect the image sharing integrity, security measurement have to be taken in each link [35] [9].

The existed image sharing system structure are mainly based on centralized server design, which may fall down from single failure. In order to complement the vulnerability caused by centralized structure, we propose use de-centralized file system, which is Inter Planetary File System (IPFS) [6]. However, when applying the smart contract and IPFS to our system, sever challenge may arise.

Also, with the spreading of image based learning application, more and more attacks take aim at shared image like poisoning training image [29]. The essential feature of such kind of attack is that attackers need to be granted to image and modify the training image without being revealed. Currently there are some works are about defending such kind of attack [97]. However, most of them are purely focus on detecting attack on image and few of those works take the protection of image sharing approach into consideration [78] [101].

Under the concerning of the shared image security, a new image sharing protection method is urgent needed. Meanwhile, blockchain provides a novel approach to protection the integrity of shared image and shows the possibility of implementing a trackable sharing system.

Current works of validating integrity are conducted by a centralized server, which may get crushed down from single point of failure. In this situation, blockchain provides an de-centralized method to implement the authentication of resources [8] [47].

In addition to blockchain, we utilize smart contract when building our system. Smart contract is a special account with associated code and data. Different from normal accounts, in smart contract it provides several functions or application binary interfaces(ABI) for interacting. Through communicating via ABI, users can store some message or data through creating transactions with smart contracts. Then we have a perfect solution to store the image information with a dependable approach.

Thus, in this paper, we propose a secure image sharing system which ensures the canal of sharing image from image owner to image users. In our designed system, all the unauthorized modification on image will come to light and, forasmuch, all the image shared from our designed system are benign and won't bring any concern about polluted image set problem.

At the present time, there's already some considering the image sharing system, one of the main problems is that different from other datatype, the image usually get compressed and then shared to other users. Thus on the sharing server, external space is required to store the compressed images. Also when applying smart contract to record the image information, However, the compressed image may give the Ethereum higher pressure and calculation requirements [23].

To deal with the situation, we propose a new hashing mechanism that gives the same hash of both original image and the compressed image. Via utilizing the proposed hashing mechanism, we can highly reduce the workload of Ethereum. When implementing the smart contract, we also take some of the Ethereum security problem into consideration and as a result, our smart contract is more resistant to some attacks.

In this paper, we propose a blockchain-based solution and de-centralized framework for the proof of authenticity of digital assets of images. Our solution supports tracking the publish information and uploader of images. Our solution are optimized against image storing process in the de-centralized platform. To sum up, this paper has the following

multi-fold contributions.

- To the best of our knowledge, this is the first work to study the image oriented IPFS-Blockchain based sharing system and has special optimization for image storing and sharing. Our study can contribute to the storing and sharing image in the future research.

- A pair of novel compression and hashing function are proposed, in which can significantly reduce the storage load in verifying the stored images.

- We designed and implemented the prototype of smart contract based trackable secure image sharing system. In our implementation, the secure of system are taken with special attention and showed practically meaningful performance.

The rest of this paper is organized as follows. In Section 4.3, the existing works on adversarial attacks and detection schemes are briefly summarized. The attack model and the detection model are presented in Section 4.4. Our two detection methods are demonstrated in Section 3.4. After analyzing the performance of our methods in Section 4.8

## 4.2 Preliminaries

In this section we introduces the concepts of blockchain and InterPlanetary File System and some formal definitions of image compression algorithm.

### 4.2.1 Blockchain and InterPlanetary File System (IPFS)

**Blockchain** The Blockchian is a chain of blocks that contain the hash of the previous block, transaction information. Block chain originates from a bitcoin network as an append-only distributed and decentralized ledger to record peer to peer transaction permanently and immutably.

**Ethereum and Smart Contract** Ethereum is an open-source blockchain and featuring smart contract functionality. Ethereum is the native cryptocurrency of the platform

and the second largest after Bitcon.

One of a special type of Ethereum account is Smart Contract. User accounts can interact with a smart contract by submitting transactions and execute the function defined by smart contract. The function is predefined before the establishing of Ethereum and cannot be modified after compile. Via smart contract, the Ethereum enables the capability's of storing data on chain.

**IPFS** The IPFS is a distributed peer-to-peer file system that enables distributed computing nodes. It works by connecting all devices on the network to the same file structure. Every time when the users trying to fetch a file, the user need to use content address to identify the content by what's in it rather than by where it's located. Also, to find where the content hosted, IPFS uses a distributed hash table, which is one where the table is split across all the peers in a distributed network [6]. In our implementation, we use IPFS storing the shared image that provides hashes of data locations as only access credentials.

### 4.2.2 Image Compression Algorithm

Image compression addresses the problem of reducing the amount of information required to represent a digital image. It is a process intended to yield a compact representation of an image, thereby reducing the image storage transmission requirements.Every image will have redundant data. Redundancy means the duplication of data in the image. Either it may be repeating pixel across the image or pattern, which is repeated more frequently in the image. The image compression occurs by taking benefit of redundant information of in the image. Reduction of redundancy provides helps to achieve a saving of storage space of an image. However the current image compression has a problem that the compression will change the file attributes, which make the hashes of original image file and compressed image file different, even if the images share totally the same visual characteristics with its compressed version. Thus, in our implementation we design a new image compress algorithm matching our proposed image hash mechanism.

### 4.2.3 Image Sharing System

On the other hand, research attentions are also mostly paid to work out on the image sharing in social media area. [95] [84] [93] proposed secret image sharing via various approaches. The methods focus on protecting secrets, which includes user privacy and some sensitive information that can be derived from image content. Meanwhile, under the context of wildly used learning scenario, the image sharing system requires more guarantee on security, which is manifested by the integrity, availability and trackability of shared image, rather than the privacy protection of images [71]. Therefore, we propose a secure image sharing system. The system is able to guarantee integrity and trackability of the shared images.

## 4.3 Related Works

In this section, we review and discuss related work found in the literature on authenticity and the originality of image sharing content.

Li and Lyu [32] proposed a method to detect deepfake videos using Artificial Intelligence (AI). The proposed method depends on an AI algorithm fighting another AI algorithm. Their technique relies on training convolutional neural networks (CNN) with manipulated and real figures. Testing was carried out using four different CNN networks with varying accuracy results between 84% to 99%. Their results look promising, however, the authors stated many challenges that yet remain to be solved. The presence of glitches in the currently obtained deepfake videos make their method give positive results. Therefore, they reckon that deepfake videos with a high resolution and quality will be hard to detect.

A US-based startup company called has developed a system involving mobile apps for typical users and freelancers for capturing images and saving them to the company's servers [32]. The purpose of saving the images is to preserve their integrity. Hence, any forgery attempt can be easily discovered by comparing it with the image from the servers. They hope that in the future their technology will be used in collaboration with other social media parties that will verify any uploaded images with the images in the Truepic's servers

and any change would therefore, be detected. Truepic also uses blockchain to store metadata of saved images to ensure immutability. This method relies heavily on trusting Truepic with the images and that all the uploaded images are untampered and real. It is not clear how the method works when inserting logos, text tickers, subtitles, or closed captions within the images or video frames.

There are also some works combined distributed storage and blockchain to build a more secure storage system [43]. [21] [30] [76] are works utilizing the IPFS system and Blockchain to build their system. [21] propose a zigzag-based storage model to improve the blockchain enhanced IPFS with blockstorage model. The work improved the performance of BitSwap protocol but considered little about the security of the stored content. Work of [30] mainly target at Internet of Things(IoT). The paper proposed a blockchain based authentication mechanism that can prevent data faking attack in IPFS from malicious users. [76] proposed a method to increase the efficiency of file sharing via IPFS and blockchain. The method not only takes trustworthiness into design consideration, but also tried to solve the problem of proximity awareness during the process of file transfer. [58] designed a IPFS-blockchain based authenticity online publications. The work focuses on online book publication. In the work, the proposed solution is a framework that is extendible and adoptable for other type of digital and media content. [73] proposed a distributed reviewer reputation system. The work discussed that under a distributed context, how the privacy settings for both open peer review and reputation system varies and proposed an approach of supporting both anonymous and accountable reviews. Those works has different coupling method and all of them are designed to store general data and has no optimization for images [22].

When implementing image sharing system, one of the unavoidable problem is that how to specifically identify different images. The most popular solution is utilize hashing function, which come with new challenge that how to build a good hashing mechanism or hash algorithm. [66] proposed a asymmetric encryption based hash function for secret share security modeling. The work makes optimal key selection based on Greay Wolf Optimization(OGWO) and the method shows less computational time and higher extreme entropy

with high PSNR when comparing to the previous researches. [60] proposed a hash based image authentication. In the work, a one layer watermark tech is embedded. Furthermore, the paper compounds DWT (Discrete Wavelet Transform) and SVD (Singular Value Decomposition) to generate perceptual hash of images. However, all the above listed works ignored a pretty common application scenario that in an image sharing system an uploaded images will be compressed into different resolution and even a hash function is efficient, to hash large amount of image is still time consuming.

## 4.4 Attack Models

In this section, we will give our attack model and our system defence model. The attack model shows the view of attackers and the resources that is available to attackers.

In this paper, the attack model and "hijack" are taken in to account, in which the images are modified without authorizing. The purpose of attackers is to maliciously modify the image stored in the server and will make efforts to hide the modification actions.

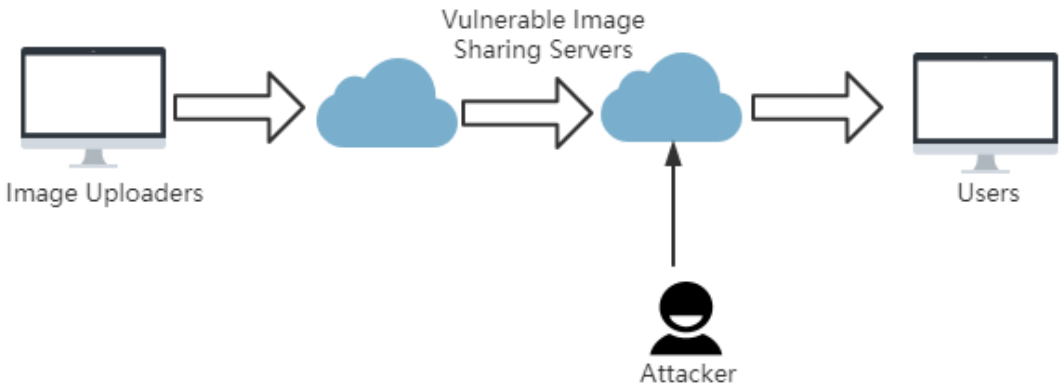


Figure 4.1. Illustration of Attack Model

Our model has two kind of authorized user groups. One is Image Owner, also considered ad Image Uploader. The image owner holds the privilege of uploading image and replace the uploaded images with new images through an authorized way. Another user group is Image Users. The image users are the clients that utilize the shared image to do research

or training model etc. A image user is able to access the shared image and require verifying the image source, but is not able to modify the shared image legally.

In the real world, it is impossible to fully protect a system from attackers. Especially the The final purpose of attackers is to modify the image stored in the image sharing server without being revealed under any manner. Then the attackers will try to erase all the access trace and we assume the attackers have the ability to modify the file modification log, system login log and other log file. Then the owner of image sharing system, the image uploader and the image user will not be able to know whether the images are modified or not purely through the information from the image sharing server. Under this situation, the only way to protect the user from affect by the modified image is to build a separate secure system and ensure it can provide integrity proof for the image stored in the image sharing server.

In the above attack model, the attackers can break the integrity of the shared image that stored in the server. Our aim is to detection any unauthorized modification on the shared image and track the sources of all the shared image to ensure the image users are always able to fetch "clean" image or get warned when fetching altered images.

Also, when applying blockchain to our system, the blockchain may also bring the vulnerability of blockchain to our system. Thus our design should take the vulnerability of blockchain into consideration. In this paper, we take re-entry attack into account. Re-entry attack is that in a blockchain network, a corrupted player return to the scheme using a new identities and then the attacker can control the untrusted contract that is able to recursive call back the original function. If the vulnerable blockchain is controled by attacker, the attacker can recursively call the withdraw function to drain the contract, which will deny the smart contract in blockchain.

## 4.5 Design of a Secure Image Sharing System

In this section, we will first introduce the model of our proposed system. The system defence model illustrates that how our system protect the integrity of shared images and how our system improves the performance while utilizing smart contract to record image

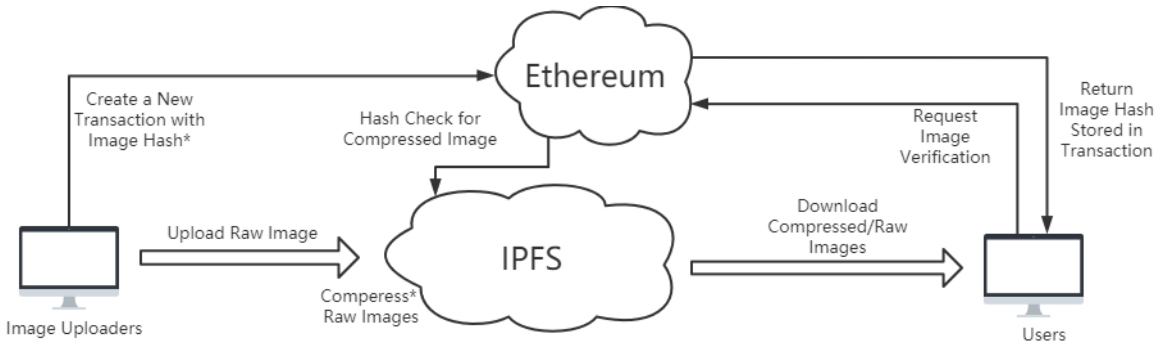


Figure 4.2. Structure of the Proposed Image Sharing System

information. Then our proposed system will be shown in detail, of which the structure mainly consists of two parts, IPFS storage part and blockchain Ethereum part. The structure of your proposed system is shown in the Fig 4.2.

### 4.5.1 Model of the Proposed System

In our designed system, we suppose that the image sharing server is vulnerable to unauthorized attackers and the attackers are able to utilize the vulnerability to modify the images that stored in the image sharing sever without leaving any trace. A successful defence not requires preventing the attacker from altering the image. Given an image sharing server, which is vulnerable to unauthorized attackers, and some images are stored in the server, of which some have been maliciously modified. The defender need to identify all the malicious modified images and when image user trying to fetch the modified image, the defender system should give warning or decline the fetching request to protect the image user from suffering the malicious modified images.

To protect the integrity of the stored images, we propose a smart contract based secure image sharing system. In our system, we firstly propose a secure image hashing algorithm mechanism. The algorithm is to compress and hash the image following the same manner and the compressed images and original images will share the same hash. In our detection model, we aim to find all the images stored in the server that get maliciously modified. If all the modified image are distinguished by our system, our defence is successful. The detail

of our system is demonstrated in next section.

### 4.5.2 Design of the Proposed System

In our proposed system, there are mainly two kinds of user group. One is image uploader group, who provides the image and uploads shared image to image sharing system, and the other one is image user group, who utilizes the shared image to do research, develop application and so on. We consider those two group of user are honest and not curious or evil to the shared image and image sharing system. Also, the security of users' authorization are reliable. We assume all the attack and potential vulnerability utilization are from unauthorized access to image sharing system.

Under the above precondition, our designed system firstly is supposed to be tough enough to fight against some illegal entering that may from potential attackers and perturbation from other aspects. Also, our system should at the same time provide the ability of revealing all the illegally modified image, which may require a separate system to guarantee the integrity of images that stored in the image sharing system that is open to public access. The separate system should be a bounded system that only provide necessary port to the image sharing system and no public access port.

In our design, all uploaders are considered benign and won't perform any attacking act to our system. When uploading, the uploader will upload raw image to our system. Then our system will calculate the hash for uploaded image and automatically create a new transactions with the smart contract account. Then the system will compress the raw image and verify the hash of the compressed images and raw images. Then the raw image and compressed images will be stored in IPFS. When retrieving image, the system will access the stored image through address value that stored in the server data base.

Image users only download images from our system. The image users may require verifying the downloaded image either or not. When requesting verification, the system will call the smart contract built-in function to give the stored transaction information, which is the hash of shared image and uploaders. The system will compare the retrieved information

and the calculated hash. If the record matches the retrieved information, then the system will give the image user requested image. If not, then the system will show image users a warning and decline the image request.

### 4.6 A Secure Image Hashing Algorithm

As we illustrated in the section 3.2, the current image hashing methods are not sufficient on image sharing system, which may require hash calculation on the images that are from the same original image by compression. Whats more, in our proposed system, we embedded blockchain to ensure the track-ability of the shared images and each image, including compressed image, may result in a new transaction in the private Ethereum, which is a waste for the computation capacity. Thus new compression-hash methods is urgent needed for our proposed image sharing system to improve performance.

In considering the features that the image sharing may have problems that the servers may pre-compress image and provide different resolution for users to download, we design a pair of algorithms, which consists of image compression part and image compress hash part. the first part gives an algorithm of lossless compression and the second part gives a hash method.

If the image is compressed following our proposed compress algorithm, then the compressed images and original image will share the same hash when applying our proposed hashing algorithm.

To match the compression algorithm, we also need a hashing mechanism. Here we propose a Image Compression Hash(IC hash) for further image verification. The design of hash calculation function is shown in the Algorithm 3.

For each input image metadata $M$ with size of N, in Algorithm 1, all three color channel will be iterated. Then in the for loop, we iterate all the value once. The time complexity of Algorithm 1 is $O(N)$. For each input image metadata $M$ with size of N, in Algorithm 3, the hash function will always pick 64 pixels to do a non-loop calculation. As a result the time complexity of Algorithm 3 is constant $O(1)$.

**Algorithm 2** Lossless Image Compression

**Input:** The metadata $M$ of an image An image

**Output:** A compressed image metadata $C\_M$ of the original image $M$

1: Set an empty 2-dim array $C\_M$.
2: **for all** each channel $c \in \mathbb{C}$ **do**
3: Pick the first value in channel $c$
4: Append the picked value to the array $C\_M[c]$
5: Count the number of subsequent occurrences of the picked value
6: Append the count to $C\_M[c]$
7: **if** Reach the end of channel $c$ **then**
8: Continue for next For Loop
9: **else**
10: Pick the next value and **GOTO** step 4
11: **end if**
12: **end for**

**Algorithm 3** Image Compression Hash(IC hash)

**Input:** The metadata $M$ of an image, where the size of $M$ is larger than $64 \times 64$.

**Output:** A $64 - bits$ number $H$, which is the hash of the input image $M$, a secure hash function $f$

1: Uses bilinear interpolation to reduce the size of input $M$ to $8 \times 8$ and save the reduced array in an array $H\_M$
2: Convert the value of RGB channel in $H\_M$ to a gray value
3: Calculate the average gray value $avg\_g$ of $H\_M$
4: Use $avg\_g$ as threshold to binarize $H\_M$
5: Duplicate 64 binary value $H\_M$ 8 times to get a 512-bit length value $H\_M'$
6: Input $H\_M'$ into a secure hash function $f$ and store the result in $H$

## 4.7 Implementation of Secure Image Sharing System

In our design, we choose a private Ethereum to store the identification of images that stored in the image sharing system.

### 4.7.1 IPFS Storage Part

In our system we propose use a decentralized storage to store the shared images. Via distributing data storage, we provide better protection to the data security. To implement the storage system, we utilized IPFS.

IPFS system is a peer to peer storage system. It consists of multiple nodes and all nodes are not privileged. In IPFS, nodes store images in local storage and nodes are connected to each other. Nodes can transfer images between each other and has a special designed routing system, which enables the nodes in the system finding other peers' network address and who can serve the particular requirements.

Considering the feature of image sharing system that users may requires different resolution version of the shared images, we designed two algorithm that can significantly reduce the hash calculation of the images. We provide image compression and a matched hashing algorithm. For all the shared image, if it is compressed by our proposed compression algorithm, then all the compressed image and the original image will get the same hash value that calculated from our proposed hash mechanism. The IPFS part is for storing the original images and the compressed images.

### 4.7.2 Ethereum

In the Ethereum part, we designed our new smart contract, to store the image information (IC hash) in a more secure way. In the Ethereum part, the smart contract will accept transactions and store the submitted hashing information of uploaded images and when verification requests come, the smart contract will call the contract function to verify the image information.

Table 4.1. Reentrancy attack

| | address.send() | address.transfer() | address.c |
|---|---|---|---|
| Gas limit (traditional) | 2300 | 2300 | all/s |
| Gas limit (our system) | 400 | 400 | -(not |
| Behavior in error (traditional) | Return false | Throw exception & revoke to previous state | Retu |
| Behavior in error (our system) | Return false | Throw exception & keep status | -(not |

## 4.8 Performance Validation

In this section, extensive real-data experiments are conducted to evaluate the performance of our proposed system and traditional system.

### 4.8.1 Experiment Settings

Our experiments adopt CIFAR-10 and Food Images (Food-101) as the testing data sets . The prototype of our proposed system is built based on docker. We also adopted Pubma, which is a docker plugins based on chaos monkey [37], to simulate the network environment including latency and packet loss.

### 4.8.2 Improvement of the Proposed Hashing Algorithm

This experiment is designed to compare the traditional algorithm and our proposed algorithm when applying to the image sharing system. The traditional algorithm is a deep supervised hashing [49]. To measure the performance of the two algorithms, we apply both two hashing algorithms to our system. Also, in our experiment we use gas cost to measure the performance of two algorithms. For the traditional algorithm, each compressed image is hashed individually, and all the hash is recorded using our designed smart contract. Meanwhile, the proposed algorithm only needs to record original hash once, which is also suitable

to all the compressed images.

We firstly upload 8000 large images from image set Food-101 [7] to our image sharing system and each image will be compressed into 3 different sizes. Then image user will randomly fetch 2000 compressed images from our system. During the experiment we will monitor all the gas cost in the Ethereum functions.

The gas cost when applying our proposed algorithm is list as the Table 4.2.

Table 4.2. Gas Cost When Applying the Proposed Algorithm and the Traditional Algorithm

| Applied Algorithm | Functions | Function Name | Transaction gas | Execution Gas |
|---|---|---|---|---|
| New algorithm | Uploader | RequestCnn | 63250 | 37802 |
| New algorithm | Mid-agent | GrantPermission | 153427 | 124649 |
| New algorithm | Mid-agent | SetProvenance | 109432 | 84375 |
| New algorithm | Mid-agent | RequestPermission | 127723 | 103321 |
| New algorithm | Image User | RequestCnn | 36340 | 18621 |
| Traditional algorithm | Uploader | RequestCnn | 216480 | 102535 |
| Traditional algorithm | Mid-agent | GrantPermission | 401863 | 352706 |
| Traditional algorithm | Mid-agent | SetProvenance | 271830 | 251864 |
| Traditional algorithm | Mid-agent | RequestPermission | 386452 | 291861 |
| Traditional algorithm | Image User | RequestCnn | 95248 | 53428 |

### 4.8.3 Data integrity of the proposed system

In this experiment, we will illustrate that our system is able to protect the integrity of images uploaded into our system. At first, we will upload a large number of images to our system. Here we upload full cifar-10 image set, which consists of 60000 colour images, to our system.

Table 4.3. Downloading Result in Two Systems after Attacks

| System | System responds | Event Count | Percentage in all Cases(%) |
|---|---|---|---|
| Proposed system | File not found | 133 | 6.65 |
| Proposed system | Reporting file changed | 1867 | 93.35 |
| Proposed system | Fetching modified images | 0 | 0 |
| Common system | File not found | 536 | 26.8 |
| Common system | Reporting file changed | 0 | 0 |
| Common system | Fetching modified images | 1464 | 73.2 |

Then we will simulate attack actions from an attacker who's already obtained a high privilege of image servers and is able to alert images stored in the system. The attack actions include modifying multiple images stored in the system.

After the attack actions, we will try to random download altered image as an image user. Then check how much percentage of the altered images are detected has been altered or untouched.

To make a comparison, we also apply the same attack to a widely used open source image sharing platform "PINRY" and test how much percent alerted image will be detected.

In addition, our system supports that the uploader can upload new images and overwrite the original images. Under such situation, no matter whether the original image is still benign or has been malicious modified, the new uploaded image can be consider as "fix" corrupt files. To evaluate how the funcion works, we design an experiment to test the system. In the experiment, we firstly execute attack on pre-uploaded images and then let image users try to retrieve all the images stored in the system. Then the image user will record all the reported abnormal images and then ask the image owners to re-upload the reported images. Then the image users will try to retrieve the image again. The result is shown in the table 4.4.

Table 4.4. Downloading Result after Re-uploading

| System responds | Event Count | Percentage(%) |
|---|---|---|
| File not found(after attack) | 106 | 5.3 |
| File not found(after re-uploading) | 106 | 5.3 |
| Reporting file changed(after attack) | 394 | 19.7 |
| Reporting file changed(after re-uploading) | 0 | 0 |
| Fetched clean images(after attack) | 1500 | 75 |
| Fetched clean images(after re-uploading) | 1894 | 94.7 |

### 4.8.4 Evaluation on false alert

As aforementioned, our proposed method is efficient to reveal the possible modified image that stored in the image shring system. However, sometimes the high accuracy on detection may come with a volume of false alerts, which is mis-classifying the benign image as the unauthorized modified image.

To evaluate the performance on resisting false alert, we execute the same attack which is launched in "Data integrity" experiment and then replace some of the modified image back to the original images. Then we check all the image when fetching from the image user side. From our defined attack model and defence model, the "modified back" images should be considered as benign images and only the "really" modified images are considered as file changed. the result is shown in the table 4.5

Table 4.5. False Alert on Attacks

| | Count |
|---|---|
| Uploaded image | 536 |
| Reporting file changed | 0 |
| Fetching modified images | 1464 |

### 4.8.5 Blockchain Security

In our designed system, we applied an Ethereum to record the image hash. This experiment we will show that our local Ethereum is resistant to most of the existed smart contract.

In this part, we will apply several attacks to our system.

**Reentrancy attack** Reentrancy attack is a kind of attack that try to steal balance from a wallet. The mechanism is that in all the smart contract, there's a built-in function "fallback".

Normally, there are three ways to transfer between wallets and smart contracts, which are "send()", "Transfer()" and "call.value()". When faced with some error during the transferring process, "send()" and "call.value()" will return false, while transfer() will throw an exception and revert state to what it was before the function call. Then attackers may use the built-in function "fallback" and re-call the transfer() function again, which may lead to duplicated transaction.

In our system, our smart contract checks the balance change first and then call the fallback function. On the other hand, we used the private Ethereum, which means we can set a lower gas limit and make the reentrancy attack harder to implement. The following of our modified smart contract.

# CHAPTER 5

# ATTACK ON BLOCKCHAIN BASED FEDERATED LEARNING

## 5.1 Introduction

The technological advancement in information and communication technology and their incorporation in the Internet of Things (IoT) enable almost every aspect of our lives from industry to home smarter [2] [24] [25]. Due to the functional properties of IoT, the application of IoT technology can be found in every field ranges from home automation to industrial applications [98]. It is predicted that the number of IoT devices that are going to be deployed for application purposes will be around 125 billion by the end of 2030 [18]. These devices interact with the physical process and surrounding environments [45], which leads to the generation of massive information data. The data generated will be stored at the central server, processed, and will be used for various applications [105] [92].

However, directly exchanging data among devices may cause serious risks in privacy leakage and information hijacking [52] [87] [91] [89]. To reduce this risk, federated learning (FL) is proposed, which is a new ML framework that trains an AI model across multiple distributed devices holding local datasets. In details, FL allows to train machine learning models locally at distributed clients and the clients share the parameters of the locally trained models to a central server [68] (i.e., the aggregator) where a global model is aggregated [44]. Therefore, the clients under the FL framework have the capability to cooperatively learn a global model without exchanging their data directly [61] [103]. Moreover, FL has been applied to real-world applications, including health care and autonomous driving [17] [80] [20] [86].

Although FL shows its effectiveness in preserving privacy [88] [10], it still has several limitations. First, in the FL process, the single centralized aggregator is assumed to be trustworthy and it shall make fair decisions in terms of the user selection and aggregation. However, this assumption is not always satisfied, especially in the real-world practise. This is

because a biased aggregator can intentionally emerge prejudice to a few selected clients, then damaging the learning performance [53]. Second, the target of FL is restricted to applications orchestrated by the centralized aggregator. As a result, the resiliency of an aggregator depends on the robustness of the central server, and a failure in the aggregator could collapse the entire FL network. Then, although local data is not explicitly shared in the original format, it is still possible for adversaries to reconstruct the raw data approximately [34], especially in the aggregation process. In particular, privacy leakage may happen during model aggregating by outsider attacks. Lastly, the existing design is vulnerable to the malicious clients that might upload poisonous models to attack the FL network [46]. At the time, Blockchain come to researchers view.

In blockchain technology, privacy is preserved using a secure encryption algorithm. Individual nodes that are included in blockchain network has their own private and public keys [106]. In most cases, the information is encrypted using the end-user public key and the private key is used to decrypt the block of information received from the sender [15]. The node that has a specific key matching pair will be able to decrypt the message and get its content [26] [12]. This encryption process ensures that the adversary will not get any confidential information from the blockchain network. As a secure technology, blockchain has the capability to tolerate single point failure with distributed consensus, and it can further implement incentive mechanisms to encourage participants to effectively contribute to the system.

Through blockchain-enabled IoT system, privacy can be achieved, however, the management of the huge data for an intelligent IoT application while keeping privacy, is also a very challenging task [14]. In the FL mechanism, a local model is trained on the data stored on end-devices and then the model parameters are shared with the central server for global model updates. Then in FL, only model parameters are shared which ensures privacy of the data, which provides a provide related solution.

In the case of blockchain-enabled IoT networks, the public keys are used to identify the IoT devices, so in case if an intruder gets information about the public key then the

malicious user might get the user's private information [107] [11]. For the area of a secure blockchain network, the researchers are trying to develop effective privacy preservation techniques. Some of the techniques so far introduced are anonymization, smart contracts, mixing, and differential privacy [26] [90].

While blockchains are considered very secure and privacy-safe by their very nature, this is not at all an accurate sentiment, as there are several attacks that may exploit them in various ways. Some of these attacks are somewhat technical and may only apply to specific blockchains. One should consider that blockchains are a technology which may be around for a decade, yet the bulk of the people who are using them have been only recently involved with them and in many cases treat them as a "blackbox." Therefore, we argue that there is a need to shed light on various security aspects of blockchains, analyse the threats they are exposed to, the attacks that can be performed, their expected impact as well as possible countermeasures

At the present time, there's already some application combines the FL and Blockchain. However as discussed before. The credibility of such kind of application still needs further verification. Thus in this paper, we propose a new attack targeting on Blockchain based federated learning.

In this paper, we propose a new attack targeting on the Blockchain embedded FL. The attack consist of two parts.

- The first part will exploit the vulnerability of blockchain and use it as the first attack array to break the integrity of the targeted system and grant the user level privilege which enables the attacker to illegally participate in federal learning.

- After the attacker takes part in the federal learning model training, the attacker will try to poison the federated learning. In our designed attack, the attacker will try to backdoor the federated learning, which which improves persistence and is able to evade anomaly detection.

The rest of this paper is organized as follows. In Section 5.2, we will introduce the pre-

liminaries. In Section 5.3 the existing works on adversarial attacks and detection schemes are briefly summarized. The attack model and the attack mechanism are presented in Section 5.4 and Section 5.5.

## 5.2 Preliminaries

In this section we introduces the concepts of blockchain and InterPlanetary File System and some formal definitions of image compression algorithm.

### 5.2.1 Proof of Stake

The Blockchian is a chain of blocks that contain the hash of the previous block, transaction information. Block chain originates from a bitcoin network as an append-only distributed and decentralized ledger to record peer to peer transaction permanently and immutably.

In blockchain, we have two total different ways to validate the blocks. One is based on the concept of Proof of Work (PoW) and another type is in Proof of Stake (PoS) protocols. In traditional blockchains, e.g., Bitcoin, users compete with each other in solving difficult cryptographic/mathematical problems which are easy to verify. This process is called "mining," and the winner gets new coins as a reward for her services. These blockchains are therefore based on PoW. While in PoS, the users that validate transactions are chosen based on their wealth (stake). Therefore, the coins are generated in the initialisation of the blockchain and to motivate validators; they get as a reward a share of each transaction they validate (transaction fees). In this regard, users are considered to be trustworthy since they "stake" a part of their property in block validation.

### 5.2.2 Long Range Attack

One of the greatest threats against PoS is called Long-Range attacks. In a Long-Range attack, the adversary creates a branch on the original blockchain which may contain different transactions and blocks and overtakes the main chain. This branch is also referred to in the

literature as Alternative History or History Revision attack. Via alternative history, the posterior participant may have the chance of play an important role when validating new blocks.

### 5.2.3 Federated Learning

Federated learning decentralizes deep learning by removing the need to pool data into a single location. Instead, the model is trained in multiple iterations at different sites. For example, say three actors decide to team up and build a model to solve a machine learning task. If they chose to work with a client-server federated approach, a centralized server would maintain the global deep neural network and each participating actor would be given a copy to train on their own dataset [31]. Once the model had been trained locally for a couple of iterations, the participants would send their updated version of the model back to the centralized server and keep their dataset within their own secure infrastructure. The central server would then aggregate the contributions from all of the participants. The updated parameters would then be shared with the participating actors, so that they could continue local training. The general principle consists in training local models on local data samples and exchanging parameters (e.g. the weights and biases of a deep neural network) between these local actor at some frequency to generate a global model shared by all actors [94].

## 5.3 Related Works

In this section, we review and discuss related work found in the literature on authenticity and the originality of Blockchain based Federated Learning and corresponding attacks.

In [39], a blockchained FL architecture was developed to verify the uploaded parameters and it investigated the related system performances, such as the learning delay and the block generation rate. Moreover, work [51] proposed a privacy-aware architecture that uses blockchain to enhance security when sharing parameters of machine learning models with other clients. In addition, the authors in [4] proposed a high-level but complicated framework by enabling encryption during model transmission and providing incentives from

participants, and the work [67] further applied this framework in the defensive military network. With the advanced features of blockchain such as tamper-proof, anonymity and traceability, an immutable audit trail of ML models can be created for greater trustworthiness in tracking and proving provenance [82].

In the scene of image recognition, the attacker modifies the Chaines in the training set, implants a special backdoor trigger and modifies the corresponding label, then uses the trigger Chaines for training, so that the trained neural network will classify the Chaines with the trigger as the specific label. In this way, the attack of implanting a backdoor into the neural network is called a backdoor attack [29]. For example, in the task of handwritten digit recognition, the attacker implanted a white square trigger with a side length of 4 in the upper left corner of many Chaines, and changed their label to 1. When the finally trained neural network encounters an image with the same trigger, the classification result will be wrong, but the neural network will still maintain a high accuracy rate (97%) for clean samples. Backdoor attacks on federated learning mainly include model replacement attack [3] and distributed backdoor attack [85]. Model replacement attack only requires the attacker to be selected once in the whole federated learning training process to achieve an efficient backdoor attack. The distributed backdoor attack is a combination of attacks, so the backdoor implanted is stealthier and not easily detected.

Model replacement attack [3] is a backdoor attack against federated learning. The attacker takes full advantage of federated learning to perform only a single attack to achieve the effect of replacing the global model with the local malicious model. Specifically, the model replacement attack occurs when the federated learning training is close to converge.

The distributed backdoor attack is a new type of attack proposed in [85]. Different from the previous centralized backdoor attack, distributed backdoor attack has multiple attackers who can cooperate to implant the backdoor. Specifically, this method decomposes the single trigger, called the global trigger, used in the traditional centralized attack to get a series of local triggers, and inject these local triggers into different attacker training sets. Then each attacker uses the allocated local trigger to attack. For example, the complete trigger is a

white square with a side length of 4, then it can be decomposed into 4 small triggers with a side length of 1. These 4 small local triggers are assigned to 4 attackers to carry out backdoor attacks. When all four attackers complete their attacks and submit model updates to the central server, the original white square backdoor with a side length of 4 has been implanted into the final model. In general, these local triggers are relatively small, so the anomaly detection algorithm is not easy to detect the attack. Thus, the distributed backdoor attack is stealthier than the centralized backdoor attack.

## 5.4 Threat Model

In our assumption, the blockchain provides the data access for federated learning. All the data that uploaded to aggregation server are recorded by blockchain and all the client in blockchain at the beginning are considered as benign. Attackers (1) have limited access to blockchain at the beginning. (2) have enough local server that are fully under the control of attackers. (3) do not control the aggregation server. Our attack has 2 objectives.

- Success on each one will bring different level of thread to the system. Security vulnerabilities is dangerous even if it cannot be exploited every single time. The first object is to grant illegal authentication to participant in the federated learning. The attackers should be able to evade the protection that provided by Blockchain. The success on this objective means that some of the attackers' servers will be considered as authorized participant in the federated learning.

- Backdoor the federated learning. The attacker wants federated learning to produce a joint model that achieves high accuracy on both its main task and an attacker chosen backdoor subtask and still retains high accuracy on the backdoor subtask for multiple rounds after the attack.

## 5.5 Attack Mechanism

In this section, we will give our attack mechanism. The attack mechanism shows the resources that is available to attackers the view of how attackers exploit the available resources.

### 5.5.1 Long Range Attack on Blockchain

In the long range attacks on PoS, the attack in theory requires an attacker that controls the majority of stake in the network, but long range attacks can be practically instantiated if the attacker controls/compromises accounts that have no stake at the moment, but have a large stake at some past block. In our implementation, we will apply dense external DoS attack to force some of the client in the blockchain get disconnected. Then the attackers are able to inject some wrong branch to the main chain. Although such kind of branch block will eventually get abandoned by the main chain, this allows an attacker to create forks from past blocks that can overtake the current chain with (past) majority stake. This can be achieved by compromising the private keys of older accounts which no longer have any stake at the moment, but that have accrued majority stake at previous block height. Also we need to notice that accounts that exhibit zero stake might not be as protected as other active accounts—which would further facilitate this attack.

### 5.5.2 Grant Federated Learning Participance

Through long range attack, attackers may not take the majority of blockchain nodes, but should have the ability to alternate the transaction history that stored in the blocks, which means the integration of blockchain has been cracked.

Then attacker will manipulate the transaction history to hide its previous actions and act as normal clients that participate the federated learning. In this stage, we achieve the first objective of our attack. In this stage, the attackers can control the local training process for federated learning. The attacker now can either modify the weights of the resulting model before submitting it for aggregation and can adaptively change its local training from round

to round.

### 5.5.3 Backdoor the Federated Learning

After the attacker grants positions in federated learning participants, further target is to backdoor the federated learning. The attacker can simply train its model on backdoored inputs and each training batch should include a mix of correctly labeled inputs and backdoored inputs to help the model learn to recognize the difference. The attacker can also change the local learning rate and the number of local epochs to maximize the overfitting to the backdoored data.

Because of the non-i.i.d. training data, each local model may be far from global model. Therefore, the attacker can solve for the model it needs to submit local update based on the scaled up the weights of the backdoored local model to ensure that the backdoor survives the averaging and the global model is replaced by backdoored local model. Then the model replacement will happen when the attacker's contribution survives averaging and is transferred to the global model. But here we need to notice that it is a single-shot attack: the global model exhibits high accuracy on the backdoor task immediately after it has been poisoned. At the present, the second attack object has been achieved.

## 5.6 Experiment and Analysis

In this section, we will first introduce the attack model of our proposed method. Then we will show the experiment settings, which includes the experiment environment and the target system. After that we will talk about the experiment result and discuss the attack result in detail.

### 5.6.1 Attack Model

In our designed system, we suppose that the Blockchain is vulnerable to unauthorized attackers and the attackers are able to utilize the vulnerability to modify the Chaines that stored in the image sharing sever without leaving any trace. A successful defence not requires

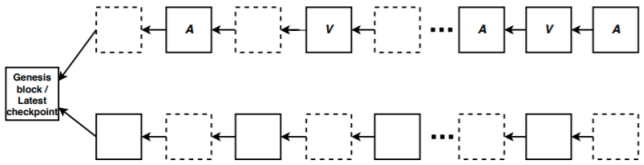


Figure 5.1. Overview of the long range attack process

Table 5.1. Minimum time spent for attacking

| $\theta$ | Attack completion time (years) |
|---|---|
| 12% | 9.3 |
| 14% | 7.0 |
| 16% | 4.7 |
| 18% | 1.4 |

preventing the attacker from altering the image. Given an image sharing server, which is vulnerable to unauthorized attackers, and some Chaines are stored in the server, of which some have been maliciously modified. The defender need to identify all the malicious modified Chaines and when image user trying to fetch the modified image, the defender system should give warning or decline the fetching request to protect the image user from suffering the malicious modified Chaines.

To protect the integrity of the stored Chaines, we propose a smart contract based secure image sharing system. In our system, we firstly propose a secure image hashing algorithm mechanism. The algorithm is to compress and hash the image following the same manner and the compressed Chaines and original Chaines will share the same hash. In our detection model, we aim to find all the Chaines stored in the server that get maliciously modified. If all the modified image are distinguished by our system, our defence is successful. The detail of our system is demonstrated in next section.

## 5.7 Performance Validation

In this section, extensive real-data experiments are conducted to evaluate the performance of our proposed system and traditional system.

### 5.7.1 Experiment Settings

Our experiments adopt CIFAR-10 and Food Chaines (Food-101) as the testing data sets . The prototype of our proposed system is built based on docker. We also adopted Pubma, which is a docker plugins based on chaos monkey [37], to simulate the network environment including latency and packet loss.

To make a comparison, we also apply the same attack to a widely used open source image sharing platform "PINRY" and test how much percent alerted image will be detected.

In addition, our system supports that the uploader can upload new Chaines and overwrite the original Chaines. Under such situation, no matter whether the original image is still benign or has been malicious modified, the new uploaded image can be consider as "fix" corrupt files. To evaluate how the funcion works, we design an experiment to test the system. In the experiment, we firstly execute attack on pre-uploaded Chaines and then let image users try to retrieve all the Chaines stored in the system. Then the image user will record all the reported abnormal Chaines and then ask the image owners to re-upload the reported Chaines. Then the image users will try to retrieve the image again. The result is shown in the table 4.4.

### 5.7.2 Blockchain Security

In our designed system, we applied an Ethereum to record the image hash. This experiment we will show that our local Ethereum is resistant to most of the existed smart contract.

In this part, we will apply several attacks to our system.

Table 5.2. Minimum time spent for attacking

| $\theta$ | Attack completion time (years) |
|---|---|
| 12% | 9.3 |
| 14% | 7.0 |
| 16% | 4.7 |
| 18% | 1.4 |

**Reentrancy attack** Reentrancy attack is a kind of attack that try to steal balance from a wallet. The mechanism is that in all the smart contract, there's a built-in function "fallback".

Normally, there are three ways to transfer between wallets and smart contracts, which are "send()", "Transfer()" and "call.value()". When faced with some error during the transferring process, "send()" and "call.value()" will return false, while transfer() will throw an exception and revert state to what it was before the function call. Then attackers may use the built-in function "fallback" and re-call the transfer() function again, which may lead to duplicated transaction.

In our system, our smart contract checks the balance change first and then call the fallback function. On the other hand, we used the private Ethereum, which means we can set a lower gas limit and make the reentrancy attack harder to implement. The following of our modified smart contract.

# CHAPTER 6

# FUTURE RESEARCH DIRECTIONS

## 6.1 Potential Interesting Problems

### 6.1.1 How to protect image free from one-pixel attack?

So far, we have been talking about detection of one-pixel attack on deep learning. Even though we have proposed two methods for detecting altered pixels and showed that our approaches are efficient, there still several interesting problem that need to be addressed in the future work. In our candidate detection method, we only need to find the pixels that might be altered. But in the real practice, the number of altered pixels are undefined. Since we only interested in the case of one pixel attack, it is very crucial to detect how many pixel get altered in the real attack.

There are man existing works focus on fighting against adversarial attack on image deep learning. While few of works study on mitigating the exact attack route of one-pixel attack. So how to protect image from one-pixel attack is an interesting problem.

### 6.1.2 Efficiency on blockchain enhanced deep learning on image

In our real data experiments of blockchain enhanced deep learning, we find that smart contract is not always as efficient as the I/O volume of IPFS. Though in our virtual network environment we've already taken the the network transaction latency into consideration, it is impossible to simulate the entire network circumstance. In order to make the system always on and efficient, a backup route for possible Ethereum system compromise is needed. The insight is behind that no matter how attacker will intrude the system, which even the smart contract is compromised, the system still keeps an unerasable clue of the attacker but not affecting the user's uploading and downloading experience much. If this work can be done, it will greatly accomplish our previous works.

### 6.1.3 How to determine the integrity of blockchain system

In the attack act on blockchain enhanced deep learning process, it is hard to determine whether the blockchain part is compromised or not. In our second work, we simply assuming that the Ethereum will not compromise. However, in reality, there are already 54 security related event happened on public blockchain. What we concerned is to fully track the alter history of images and such function relies on the security of our private Ethereum. Current our image compress and hashing algorithm are stand alone from the Ethereum and if the hash algorithm can be embedded in to smart contract, the system can be still intact after attack on blockchain part.

## 6.2 Attack on blockchain enhanced deep learning

In our future work, we will carry out further research activities along two directions: (i) attempting to distinguish between the benign images and the attacked images in the presence of one-pixel attack; and (ii) mitigating the impact of one-pixel attack by enhancing the resistance to adversarial samples in DNNs. As for the prototype of secure image sharing system, it can be a commercial project under a practical manner. The compression method can be also expanded. Also, the work also require more work on the privacy concern. To achieve a privacy preserved secure trackable image sharing system maybe the next goal.

Also I am currently working on camouflage attack, which is also an deep learning oriented image attack. Such attack will manipulate the source images in an inaudible way and make the scaling algorithm generates adversarial image when applying to the modified source images. Quiring *et al.* [65] analyzes the root cause of camouflage attack and provide two ways to protect image scaling in machine learning from camouflage attack.

# CHAPTER 7

# CONCLUSION

The work "Detection Mechanisms of One-Pixel Attack" proposes two novel methods, *i.e.*, the trigger detection method and the candidate detection method, to detect one-pixel attack that is one of the most concealed attack models. The trigger detection method gives the exact pixel that may be modified by one-pixel attack; the candidate detection method outputs a set of pixels that may be changed in one-pixel attack. Via extensive real-data experiments, the effectiveness of our two methods can be confirmed; especially, the detection success rate of our candidate detection can achieve 30.1%

As a preliminary exploration of one-pixel attack detection, in the work, we consider all the images are attacked and the detection is thus implemented on a dataset full of modified images.

In the work "Blockchain Based Trackable Secure Image Sharing System", we proposed a smart contract based solution for image integrity and system security. Our work utilized a decentralized storage system IPFS and smart contract. To optimize the sharing and verifiying process for images, we proposed an image compression method and hashing method. We also estimate the operational cost of Ethereum and the result is that via using the proposed methods, the computational cost of generating transaction get significantly reduced. Furthermore, our experiment shows that our system has resistant to several common attacks.

## ACKNOWLEDGMENT

This dissertation is partly supported by the National Science Foundation of U.S. (2118083, 1912753, 1704287, 1741277, 1829674).

# REFERENCES

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[2] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, 17(4):2347–2376, 2015.

[3] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR, 2020.

[4] Xianglin Bao, Cheng Su, Yan Xiong, Wenchao Huang, and Yifei Hu. Flchain: A blockchain for auditable federated learning with trust and incentive. In *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*, pages 151–159. IEEE, 2019.

[5] Irad Ben-Gal. Outlier detection. In *Data mining and knowledge discovery handbook*, pages 131–146. Springer, 2005.

[6] Juan Benet. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561*, 2014.

[7] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pages 446–461. Springer, 2014.

[8] Zhipeng Cai and Quan Chen. Latency-and-coverage aware data aggregation scheduling for multihop battery-free wireless networks. *IEEE Transactions on Wireless Communications*, 20(3):1770–1784, 2020.

[9] Zhipeng Cai, Zhuojun Duan, and Wei Li. Exploiting multi-dimensional task diversity in distributed auctions for mobile crowdsensing. *IEEE Transactions on Mobile Computing*, 20(8):2576–2591, 2020.

[10] Zhipeng Cai and Zaobo He. Trading private range counting over big iot data. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 144–153. IEEE, 2019.

[11] Zhipeng Cai, Zaobo He, Xin Guan, and Yingshu Li. Collective data-sanitization for preventing sensitive information inference attacks in social networks. *IEEE Transactions on Dependable and Secure Computing*, 15(4):577–590, 2016.

[12] Zhipeng Cai and Tuo Shi. Distributed query processing in the edge-assisted iot data monitoring system. *IEEE Internet of Things Journal*, 8(16):12679–12693, 2020.

[13] Zhipeng Cai, Zuobin Xiong, Honghui Xu, Peng Wang, Wei Li, and Yi Pan. Generative adversarial networks: A survey toward private and secure applications. *ACM Computing Surveys (CSUR)*, 54(6):1–38, 2021.

[14] Zhipeng Cai and Xu Zheng. A private and efficient mechanism for data uploading in smart cyber-physical systems. *IEEE Transactions on Network Science and Engineering*, 7(2):766–775, 2018.

[15] Zhipeng Cai, Xu Zheng, and Jinbao Wang. Efficient data trading for stable and privacy preserving histograms in internet of things. In *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, pages 1–10. IEEE, 2021.

[16] Zhipeng Cai, Xu Zheng, Jinbao Wang, and Zaobo He. Private data trading towards range counting queries in internet of things. *IEEE Transactions on Mobile Computing*, 2022.

[17] Zhipeng Cai, Xu Zheng, and Jiguo Yu. A differential-private framework for urban traf-

fic flows estimation via taxi companies. *IEEE Transactions on Industrial Informatics*, 15(12):6492–6499, 2019.

[18] Mark Campbell. Smart edge: The effects of shifting the center of data gravity out of the cloud. *Computer*, 52(12):99–102, 2019.

[19] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma. Pcanet: A simple deep learning baseline for image classification? *IEEE transactions on image processing*, 24(12):5017–5032, 2015.

[20] Yiqiang Chen, Xin Qin, Jindong Wang, Chaohui Yu, and Wen Gao. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

[21] Yongle Chen, Hui Li, Kejiao Li, and Jiyang Zhang. An improved p2p file system scheme based on ipfs and blockchain. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 2652–2657. IEEE, 2017.

[22] Siyao Cheng, Zhipeng Cai, and Jianzhong Li. Curve query processing in wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 64(11):5198–5209, 2014.

[23] Siyao Cheng, Zhipeng Cai, Jianzhong Li, and Hong Gao. Extracting kernel dataset from big sensory data in wireless sensor networks. *IEEE Transactions on Knowledge and Data Engineering*, 29(4):813–827, 2016.

[24] Chuanxiu Chi, Yingjie Wang, Xiangrong Tong, Madhuri Siddula, and Zhipeng Cai. Game theory in internet of things: A survey. *IEEE Internet of Things Journal*, 2021.

[25] Suparna De, Maria Bermudez-Edo, Honghui Xu, and Zhipeng Cai. Deep generative models in the industrial internet of things: a survey. *IEEE Transactions on Industrial Informatics*, 2022.

[26] Tiago M Fernández-Caramés and Paula Fraga-Lamas. A review on the use of blockchain for the internet of things. *Ieee Access*, 6:32979–33001, 2018.

[27] Nicholas Frosst, Sara Sabour, and Geoffrey Hinton. Darccc: Detecting adversaries by reconstruction from class conditional capsules. *arXiv preprint arXiv:1811.06969*, 2018.

[28] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE, 2018.

[29] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[30] J Hao, Yan Sun, and Hong Luo. A safe and efficient storage scheme based on blockchain and ipfs for agricultural products tracking. *J. Comput*, 29(6):158–167, 2018.

[31] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[32] Haya R Hasan and Khaled Salah. Combating deepfake videos using blockchain and smart contracts. *Ieee Access*, 7:41596–41606, 2019.

[33] Zaobo He, Zhipeng Cai, Siyao Cheng, and Xiaoming Wang. Approximate aggregation for tracking quantiles and range countings in wireless sensor networks. *Theoretical Computer Science*, 607:381–390, 2015.

[34] Zaobo He, Zhipeng Cai, and Jiguo Yu. Latent-data privacy preserving with customized data utility for social network data. *IEEE Transactions on Vehicular Technology*, 67(1):665–673, 2017.

[35] Zaobo He, Zhipeng Cai, Jiguo Yu, Xiaoming Wang, Yunchuan Sun, and Yingshu Li. Cost-efficient strategies for restraining rumor spreading in mobile social networks. *IEEE Transactions on Vehicular Technology*, 66(3):2789–2800, 2016.

[36] Shengyuan Hu, Tao Yu, Chuan Guo, Wei-Lun Chao, and Kilian Q Weinberger. A new defense against adversarial images: Turning a weakness into a strength. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 1635–1646. Curran Associates, Inc., 2019.

[37] Pooyan Jamshidi, Claus Pahl, Nabor C Mendonça, James Lewis, and Stefan Tilkov. Microservices: The journey so far and challenges ahead. *IEEE Software*, 35(3):24–35, 2018.

[38] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5574–5584, 2017.

[39] Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Blockchained on-device federated learning. *IEEE Communications Letters*, 24(6):1279–1283, 2019.

[40] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[41] David D Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *European conference on machine learning*, pages 4–15. Springer, 1998.

[42] Ang Li, Jiayi Guo, Huanrui Yang, and Yiran Chen. Deepobfuscator: Adversarial training framework for privacy-preserving image classification. *arXiv preprint arXiv:1909.04126*, 2019.

[43] Ji Li, Siyao Cheng, Zhipeng Cai, Jiguo Yu, Chaokun Wang, and Yingshu Li. Approximate holistic aggregation in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 13(2):1–24, 2017.

[44] Kaiyang Li, Guoming Lu, Guangchun Luo, and Zhipeng Cai. Seed-free graph deanonymiztiation with adversarial learning. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 745–754, 2020.

[45] Kaiyang Li, Guangchun Luo, Yang Ye, Wei Li, Shihao Ji, and Zhipeng Cai. Adversarial privacy-preserving graph embedding against inference attack. *IEEE Internet of Things Journal*, 8(8):6904–6915, 2020.

[46] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[47] Yi Liang, Zhipeng Cai, Qilong Han, and Yingshu Li. Location privacy leakage through sensory data. *Security and Communication Networks*, 2017, 2017.

[48] Yi Liang, Zhipeng Cai, Jiguo Yu, Qilong Han, and Yingshu Li. Deep learning based inference of private information using embedded sensors in smart devices. *IEEE Network*, 32(4):8–14, 2018.

[49] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. Deep supervised hashing for fast image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2064–2072, 2016.

[50] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. 2017.

[51] Yunlong Lu, Xiaohong Huang, Yueyue Dai, Sabita Maharjan, and Yan Zhang. Blockchain and federated learning for privacy-preserved data sharing in industrial iot. *IEEE Transactions on Industrial Informatics*, 16(6):4177–4186, 2019.

[52] Chuan Ma, Jun Li, Ming Ding, Long Shi, Taotao Wang, Zhu Han, and H Vincent Poor. When federated learning meets blockchain: A new distributed learning paradigm. *arXiv preprint arXiv:2009.09338*, 2020.

[53] Chuan Ma, Jun Li, Ming Ding, Howard H Yang, Feng Shu, Tony QS Quek, and H Vincent Poor. On safeguarding privacy and security in the framework of federated learning. *IEEE network*, 34(4):242–248, 2020.

[54] Xingjun Ma, Yuhao Niu, Lin Gu, Yisen Wang, Yitian Zhao, James Bailey, and Feng Lu. Understanding adversarial attacks on deep learning based medical image analysis systems. *Pattern Recognition*, page 107332, 2020.

[55] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[56] Nina Narodytska and Shiva Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1310–1318. IEEE, 2017.

[57] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[58] Nishara Nizamuddin, Haya R Hasan, and Khaled Salah. Ipfs-blockchain-based authenticity of online publications. In *International Conference on Blockchain*, pages 199–212. Springer, 2018.

[59] Andrew P Norton and Yanjun Qi. Adversarial-playground: A visualization suite showing how adversarial examples fool deep learning. In *2017 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 1–4. IEEE, 2017.

[60] Fatih Ozyurt, Turker Tuncer, and Engin Avci. A novel probabilistic image authentication methodbased on universal hash function for rgb images. In *2018 International Conference on Computing Sciences and Engineering (ICCSE)*, pages 1–6. IEEE, 2018.

[61] Junjie Pang, Yan Huang, Zhenzhen Xie, Qilong Han, and Zhipeng Cai. Realizing the heterogeneity: A self-organized federated learning framework for iot. *IEEE Internet of Things Journal*, 8(5):3088–3098, 2020.

[62] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

[63] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[64] Yao Qin, Nicholas Frosst, Sara Sabour, Colin Raffel, Garrison Cottrell, and Geoffrey Hinton. Detecting and diagnosing adversarial images with class-conditional capsule reconstructions. *arXiv preprint arXiv:1907.02957*, 2019.

[65] Erwin Quiring, David Klein, Daniel Arp, Martin Johns, and Konrad Rieck. Adversarial preprocessing: Understanding and preventing image-scaling attacks in machine learning. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 1363–1380, 2020.

[66] K Shankar and Mohamed Elhoseny. Multiple share creation with optimal hash function for image security in wsn aid of ogwo. In *Secure Image Transmission in Wireless Sensor Network (WSN) Applications*, pages 131–146. Springer, 2019.

[67] Pradip Kumar Sharma, Jong Hyuk Park, and Kyungeun Cho. Blockchain and federated learning-based distributed computing defence framework for sustainable society. *Sustainable Cities and Society*, 59:102220, 2020.

[68] Madhuri Siddula, Yingshu Li, Xiuzhen Cheng, Zhi Tian, and Zhipeng Cai. Anonymiza-

tion in online social networks based on enhanced equi-cardinal clustering. *IEEE Transactions on Computational Social Systems*, 6(4):809–820, 2019.

[69] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011.

[70] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[71] Zice Sun, Yingjie Wang, Zhipeng Cai, Tianen Liu, Xiangrong Tong, and Nan Jiang. A two-stage privacy protection mechanism based on blockchain in mobile crowdsourcing. *International Journal of Intelligent Systems*, 36(5):2058–2080, 2021.

[72] Guanhong Tao, Shiqing Ma, Yingqi Liu, and Xiangyu Zhang. Attacks meet interpretability: Attribute-steered detection of adversarial samples. In *Advances in Neural Information Processing Systems*, pages 7717–7728, 2018.

[73] Antonio Tenorio-Fornés, Viktor Jacynycz, David Llop-Vila, Antonio Sánchez-Ruiz, and Samer Hassan. Towards a decentralized process for scientific publication and peer review using blockchain and ipfs. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019.

[74] Shixin Tian, Guolei Yang, and Ying Cai. Detecting adversarial examples through image transformation. In *AAAI*, 2018.

[75] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[76] S Vimal and SK Srivatsa. A new cluster p2p file sharing system based on ipfs and blockchain technology. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–7, 2019.

[77] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723. IEEE, 2019.

[78] Chenyu Wang, Zhipeng Cai, and Yingshu Li. Sustainable blockchain-based digital twin management architecture for iot devices. *IEEE Internet of Things Journal*, 2022.

[79] Jinbao Wang, Zhipeng Cai, Yingshu Li, Donghua Yang, Ji Li, and Hong Gao. Protecting query privacy with differentially private k-anonymity in location-based services. *Personal and Ubiquitous Computing*, 22(3):453–469, 2018.

[80] Jinbao Wang, Zhipeng Cai, and Jiguo Yu. Achieving personalized $k$-anonymity-based content privacy for autonomous vehicles in cps. *IEEE Transactions on Industrial Informatics*, 16(6):4242–4251, 2019.

[81] Peng Wang, Zhipeng Cai, Donghyun Kim, and Wei Li. Detection mechanisms of one-pixel attack. *Wireless Communications and Mobile Computing*, 2021, 2021.

[82] Shufen Wang. Blockfedml: Blockchained federated machine learning systems. In *2019 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*, pages 751–756. IEEE, 2019.

[83] S. Wu, G. Li, L. Deng, L. Liu, D. Wu, Y. Xie, and L. Shi. $l1$ -norm batch normalization for efficient training of deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 30(7):2043–2051, 2019.

[84] Zhen Wu, Yining Liu, and Xingxing Jia. A novel hierarchical secret image sharing scheme with multi-group joint management. *Mathematics*, 8(3):448, 2020.

[85] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. Dba: Distributed backdoor attacks against federated learning. In *International Conference on Learning Representations*, 2019.

[86] Zuobin Xiong, Zhipeng Cai, Qilong Han, Arwa Alrawais, and Wei Li. Adgan: Protect your location privacy in camera data of auto-driving vehicles. *IEEE Transactions on Industrial Informatics*, 17(9):6200–6210, 2020.

[87] Zuobin Xiong, Zhipeng Cai, Daniel Takabi, and Wei Li. Privacy threat and defense for federated learning with non-iid data in aiot. *IEEE Transactions on Industrial Informatics*, 18(2):1310–1321, 2021.

[88] Zuobin Xiong, Wei Li, Qilong Han, and Zhipeng Cai. Privacy-preserving auto-driving: a gan-based approach to protect vehicular camera data. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 668–677. IEEE, 2019.

[89] Zuobin Xiong, Honghui Xu, Wei Li, and Zhipeng Cai. Multi-source adversarial sample attack on autonomous vehicles. *IEEE Transactions on Vehicular Technology*, 70(3):2822–2835, 2021.

[90] Honghui Xu, Zhipeng Cai, Ruinian Li, and Wei Li. Efficient citycam-to-edge cooperative learning for vehicle counting in its. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[91] Honghui Xu, Zhipeng Cai, and Wei Li. Privacy-preserving mechanisms for multi-label image recognition. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 16(4):1–21, 2022.

[92] Honghui Xu, Zhipeng Cai, Daniel Takabi, and Wei Li. Audio-visual autoencoding for privacy-preserving video streaming. *IEEE Internet of Things Journal*, 9(3):1749–1761, 2021.

[93] Xuehu Yan, Yuliang Lu, and Lintao Liu. A general progressive secret image sharing construction method. *Signal Processing: Image Communication*, 71:66–75, 2019.

[94] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong,

Daniel Ramage, and Françoise Beaufays. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

[95] Jun Yu, Zhenzhong Kuang, Baopeng Zhang, Wei Zhang, Dan Lin, and Jianping Fan. Leveraging content sensitiveness and user trustworthiness to recommend fine-grained privacy settings for social image sharing. *IEEE transactions on information forensics and security*, 13(5):1317–1332, 2018.

[96] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*, 2019.

[97] Lichen Zhang, Zhipeng Cai, and Xiaoming Wang. Fakemask: A novel privacy preserving approach for smartphones. *IEEE Transactions on Network and Service Management*, 13(2):335–348, 2016.

[98] Xu Zheng and Zhipeng Cai. Privacy-preserved data sharing towards multiple parties in industrial iots. *IEEE Journal on Selected Areas in Communications*, 38(5):968–979, 2020.

[99] Xu Zheng, Zhipeng Cai, Jianzhong Li, and Hong Gao. Location-privacy-aware review publication mechanism for local business service systems. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pages 1–9. IEEE, 2017.

[100] Xu Zheng, Zhipeng Cai, and Yingshu Li. Data linkage in smart internet of things systems: a consideration from a privacy perspective. *IEEE Communications Magazine*, 56(9):55–61, 2018.

[101] Xu Zheng, Zhipeng Cai, Jiguo Yu, Chaokun Wang, and Yingshu Li. Follow but no track: Privacy preserved profile publishing in cyber-physical social systems. *IEEE Internet of Things Journal*, 4(6):1868–1878, 2017.

[102] Xu Zheng, Ling Tian, and Zhipeng Cai. A fair and rational data sharing strategy towards two-stage industrial internet of things. *IEEE Transactions on Industrial Informatics*, 2022.

[103] Xu Zheng, Ling Tian, Guangchun Luo, and Zhipeng Cai. A collaborative mechanism for private data publication in smart cities. *IEEE Internet of Things Journal*, 7(9):7883–7891, 2020.

[104] Kang Zhou, Shenghua Gao, Jun Cheng, Zaiwang Gu, Huazhu Fu, Zhi Tu, Jianlong Yang, Yitian Zhao, and Jiang Liu. Sparse-gan: Sparsity-constrained generative adversarial network for anomaly detection in retinal oct image. In *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, pages 1227–1231. IEEE, 2020.

[105] Lijing Zhou, Licheng Wang, Yiru Sun, and Pin Lv. Beekeeper: A blockchain-based iot system with secure storage and homomorphic computation. *IEEE Access*, 6:43472–43488, 2018.

[106] Saide Zhu, Zhipeng Cai, Huafu Hu, Yingshu Li, and Wei Li. zkcrowd: a hybrid blockchain-based crowdsourcing platform. *IEEE Transactions on Industrial Informatics*, 16(6):4196–4205, 2019.

[107] Saide Zhu, Wei Li, Hong Li, Ling Tian, Guangchun Luo, and Zhipeng Cai. Coin hopping attack in blockchain-based iot. *IEEE Internet of Things Journal*, 6(3):4614–4626, 2018.