Hindawi
Security and Communication Networks
Volume 2022, Article ID 3043489, 20 pages
https://doi.org/10.1155/2022/3043489

Research Article

BCFDPS: A Blockchain-Based Click Fraud Detection and Prevention Scheme for Online Advertising

Qiuyun Lyu,[1] **Hao Li**,[1] **Renjie Zhou**,[2,3] **Jilin Zhang**,[2,3] **Nailiang Zhao**,[2] **and Yan Liu**[4]

[1]*School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China*
[2]*School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China*
[3]*Key Laboratory of Complex Systems Modeling and Simulation of the Ministry of Education, Hangzhou Dianzi University, Hangzhou 310018, China*
[4]*Zhejiang Panshi Information Technology Co., Ltd., Hangzhou 310015, China*

Correspondence should be addressed to Renjie Zhou; rjzhou@hdu.edu.cn

Received 25 November 2021; Revised 10 March 2022; Accepted 21 March 2022; Published 29 April 2022

Academic Editor: Jiewu Leng

Copyright © 2022 Qiuyun Lyu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Online advertising, which depends on consumers' click, creates revenue for media sites, publishers, and advertisers. However, click fraud by criminals, i.e., the ad is clicked either by malicious machines or hiring people, threatens this advertising system. To solve the problem, many schemes are proposed which are mainly based on machine learning or statistical analysis. Although these schemes mitigate the problem of click fraud, several problems still exist. For example, some fraudulent clicks are still in the wild since their schemes only discover the fraudulent clicks with a probability approaching but not 100%. Also, the process of detecting a click fraud is executed by a single publisher, which makes a chance for the publisher to obtain illegal income by deceiving advertisers and media sites. Besides, the identity privacy of consumers is also exposed because the schemes deal with the plain text of consumers' real identity. Therefore, in this paper, a blockchain-based click fraud detection and prevention scheme (BCFDPS) for online advertising is proposed to deal with the above problems. Specifically, the BCFDPS mainly introduces bilinear pairing to implicitly verify whether a consumer's real digital identity is contained in a click message to significantly avoid click fraud and employs a consortium blockchain to ensure the transparency of the detection and prevention process. In our scheme, the clicks by machines or fraud ones by a human can be accurately detected and prevented by media sites, publishers, and advertisers. Furthermore, ciphertext-policy attribute-based encryption is adopted to protect the identity privacy of consumers. The implementation and evaluation results show that compared with the existing click fraud detection and prevention schemes based on machine learning and statistical analysis, BCFDPS achieves detection of each fraudulent click with a probability of 100% and consumes lower computation cost; furthermore, BCFDPS adds functions of consumers' privacy protection and click fraud detection and prevention, compared to the existing blockchain-based online advertising scheme, by introducing limited communication cost (4,984 bytes) at lower storage cost.

1. Introduction

Nowadays, cost-per-click (CPC) is by far the most popular model used in online advertising [1]. An online advertising system mainly includes four entities [2–4], namely, consumers (Us), advertisers (ADEs), publishers (PUBs), and media sites (MSs). An ad promotion process includes seven steps such as publishing, clicking, paying, and so on, which is shown in Figure 1. The ADE's ad is published by PUB to U on the website of MS, as shown in steps 1–3 in Figure 1. A *click* is counted when a *U* clicks on the ad, as shown in steps 4-5 in Figure 1. Then, ADE needs to pay advertising promotion fees to PUB because of these *clicks*, and PUB also pays advertising *click* fees to MS, as shown in steps 6-7 in Figure 1. There are mainly two types of implementations of online advertising system to publish ads. The first type is the traditional online advertising systems (Google, Twitter, etc.) which mainly rely on centralized servers. Also, inspired by

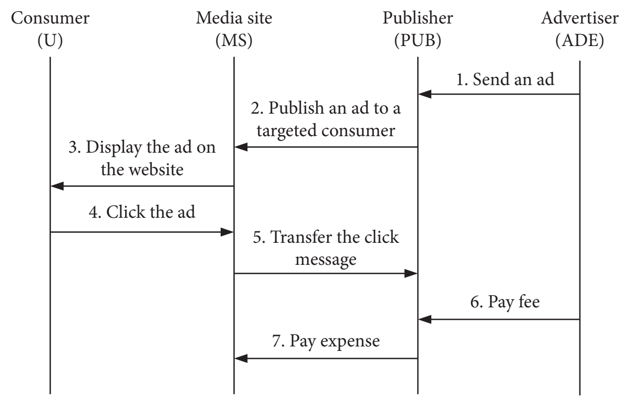


Figure 1: Process of online advertising system.

the tamper-proof and decentralized characteristics of blockchain, the other one is blockchain-based advertising systems [5, 6] which are implemented to achieve the transparency of an advertising business.

For higher revenue in the two types of implementations of online advertising systems, an ad will be published to a targeted *U* who is the potential consumer, which is called ad precision targeting. As we know, two main types of click fraud methods are designed by the attackers in the above two systems to gain extra illegal revenue: *the first type is to generate the repeated click messages by machines*. In detail, malicious advertisers use web crawler, botnet and proxy server, etc. to click an ad by machines [7–9] for exhausting his competitors' budgets. *The second one is click fraud by a human.* Specifically, the malicious publishers and media sites recruit many people to click the same ad frame [10] or abuse the click history of legitimate users [11] to charge advertisers more ad promotion fees [4, 12]. Thus, these fake clicks generate additional budgets for advertisers, but do not create any revenue [13, 14], which undoubtedly disrupts the order of the advertising system.

Therefore, many click fraud detection and prevention schemes have been proposed to predict the authenticity of each click and to maintain the stability of the advertising system. According to the technology adopted, these schemes may be classified into two categories: *machine learning-based scheme* and *statistical analysis-based scheme*. *Machine learning-based schemes* [4, 7–9, 13, 15–22] utilize machine learning algorithms to train models that can judge whether a new click is fraudulent from massive click traffic. For example, in the scheme of [7], a machine learning algorithm based on convolutional neural networks and decision tree is designed to construct a classifier that distinguishes whether a click message is generated by machines or human beings according to the sensors of mobile device. However, the dataset used for training is easy to be mixed with fraudulent clicks in these schemes, causing the process of training a model to be susceptible to adversarial attack [23]. Thence, *statistical analysis-based schemes* [1, 10, 24–32] aim to mitigate the adversarial attacks. For instance, the schemes in [10, 25] predict malicious crowdsourcing platforms by clustering algorithms. Xu and Li [25] used the DP-means clustering method to predict malicious groups, while Tian et al. [10], inspired by the DP-means clustering method, proposed a non-parametric method to solve the problem of malicious coalition fraud. Although they prevent the fraud of short-term malicious crowdsourcing platforms, their approaches are not enough for multiple traffics with long fraud intervals.

Apart from this, in the above two categories of schemes, the click fraud is still in the wild since they predict the click fraud only with a probability which is less than 100%. Also, the transparency of the click fraud detection and prevention process is not achieved, since these fraud detection and prevention algorithms are only implemented within a single central agency (publisher). That is, the publisher could gain illegal income from misreporting the number of the real clicks. Moreover, U's privacy is also leaked since these schemes analyze U's some original identity information such as the username and phone number.

Recently, as a tamper-proof and distributed technology, blockchain has attracted the attention of online advertising systems to significantly increase the trust between consumers and advertisers without additional costs and intermediaries. Specifically, by using a distributed ledger, the data related to the delivery of ads, clicks, and the analysis result of the real click number are all stored in the blockchain, which can be audited and verified by everyone [33]. On the other hand, people's activities in physical space have been transferred to cyberspace increasingly. To build a better cyberspace, a digital identity that maps one-to-one with a physical identity in cyberspace is becoming the focus of the future. To this end, many digital identity infrastructures [34–38] have emerged to better manage user behavior in cyberspace.

Therefore, taking the above merits and problems into account, we introduce the blockchain and the existing digital identity infrastructures to detect and prevent fraudulent clicks for an online advertising system. The main contributions of this paper are summarized as follows.

(1) We propose a blockchain-based click fraud detection and prevention scheme (BCFDPS) for online advertising, which significantly avoids clicking by machines and increases the cost of fraud ones by a human.
(2) Whether a click is fraudulent can be confirmed directly in our scheme rather than predicted with a probability less than 100%. A consumer's digital identity that is one and only mapping to a person in the physical world is embedded in a click message. That is, a fraudulent click does not contain a legitimate digital identity, and many duplicate clicks contain only the same digital identity.
(3) When negotiating the ad billing fee between entities, the problem of tampering with the real number of clicks by media sites and publishers is solved because the transparency of the number in the click fraud detection and prevention process is realized through introducing a consortium blockchain. Specifically, the analysis result of the clicks is periodically recorded by publishers. The media sites and the

advertisers can also verify the result independently through the data in the blockchain.

(4) The risk of leaking consumers' identity privacy from adversaries is alleviated by the bilinear pairing and ciphertext-policy attribute-based encryption without excessively affecting the publisher's accurate target of ads to consumers.

The rest of this article is organized as follows. Related works are discussed in Section 2. Section 3 reviews some preliminaries. Section 4 formulates the problem being addressed. Section 5 describes the proposed BCFDPS in detail. Security is analyzed in Section 6, and an experiment is designed and implemented in Section 7, followed by discussion and conclusion in Sections 8 and 9, respectively.

2. Related Work

2.1. Machine Learning-Based Schemes. Machine learning-based schemes are widely used in advertising fraud detection scenarios with large amounts of click data. User's click features are first extracted, then these features are used to train a model with the training dataset, and further the trained model is evaluated with the test dataset [13]. Oentaryo et al. [15] and Kanei et al. [4] mainly utilized random forest to detect click fraud in online advertising systems. But they are unable to catch coalition attacks involving multiple fraudulent approaches. Then, Wang et al. [16] presented CLUE in 2017, a novel recurrent neural network (RNN)-based online e-commerce transaction fraud detection system, and they deployed the CLUE on JD.com, serving over 220 million active users, to achieve real-time detection of fraudulent transactions. However, the CLUE in [16] will face gradient vanishing or gradient exploding problems when the click traffic is too complicated, leading to a poor fraud detection model. In 2018, Haider et al. [18] used two ensemble learning techniques, bagging and boosting algorithms, to train a model to detect and prevent click fraud. In 2019, support vector machine (SVM), K-nearest neighbor (KNN), AdaBoost, decision tree, and bagging were evaluated to detect a click by Almahmoud et al. [8]. In 2020, gradient tree boosting (GTB) algorithm was used to address the challenges encountered in effectively classifying fraudulent publishers [19]. Nevertheless, the user's identity privacy is exposed in [8, 18, 19] since they used the original identity information of consumers, such as the real username and the address of the visitor, to train models. Then, in 2021, two XGBoost-based schemes [21, 22] were proposed for click fraud detection, but both of them require manual classification of a large amount of click traffic in advance, which is time consuming. Apart from the problems mentioned above, according to the paper of Mikhailov and Trusov [23], all the schemes in this category are prone to adversarial attacks, for which they need a large number of samples as input to train the model. Also, these machine learning-based schemes can only determine whether the click is fraudulent with a probability approaching 100%.

2.2. Statistical Analysis-Based Schemes. Graph-based propagation approaches were first proposed in [24, 27, 29] to analyze the advertising traffic. The main idea of Stitelman et al. [24] is to use the co-visitation network between websites to identify media sites with a large amount of fraudulent traffic, but this approach relies on the fact that the experts have informed views about which websites look reasonable and which do not. Since it is difficult to collect all the users' data in a co-visitation network, Hu et al. [27] analyzed the behavior characteristic of individual mobile advertising user and then reduced malicious user clicks. As a specific deployment of the idea in [27], Dong et al. [29] proposed FraudDroid, a novel hybrid approach, to detect ad frauds in mobile Android apps. It dynamically analyzes applications to build UI state transition graphs, collects their associated runtime network traffic, and then uses it to identify advertising fraud.

Additionally, a pattern-based click fraud detection scheme for mobile applications [32] was designed, and it mainly has two components: *offline pattern extractor* and *online fraud detector*. The *extractor* is responsible for extracting traffic patterns for ad and non-ad traffics, and the *detector* is in charge of monitoring network traffic and detecting click fraud with the traffic patterns. But the schemes in [29, 32] may fail to handle subsequent variant click fraud [39].

Different from the above graph-based and pattern-based analysis schemes, three contextual-based attributes concerning interarrival time (IAT), diurnal activity (DA), and eigenscore (ES) were analyzed in comparison-shopping services to calculate a click's credible score for detecting whether it is fake in [26]. Moreover, Meghanath et al. [28] proposed a new contextual outlier detection technology (ConOut) and applied it to the advertising domain to identify fraudulent publishers. Besides, the work in [30] presents Bag-of-Words algorithm to assess clicks in online advertising system, which is based on the concept of text search methods. However, the outlier detection technology in [28] and the Bag-of-Words algorithm in [30] involve users' real username and address of the visitor, which reveals user's identity privacy.

In 2019, a novel inference technique (Clicktok) was developed to isolate click fraud attacks in [31]. Clicktok analyzes the traffic matrix, including matrix decomposition and construction, to propose two defenses, mimicry and bait-click. The mimicry isolates click spam by observing the reuse pattern of legitimate click traffic, and the ad network uses bait-clicks to watermark the channel periodically, which sets off watermark detectors when an attacker harvests and reuses a legitimate clickstream in the channel. But the Clicktok does not have a good ability to prevent the new types of click fraud whose traffic matrix is similar to the one of a normal click. On the other hand, these statistical analysis-based methods are deployed in publishers' devices and they do not achieve the transparency of the click fraud detection and prevention process for each entity in the advertising system. In addition, the statistical analysis-based schemes leak user's identity privacy since they analyze the original data which includes user's real username, address of the user, etc.

2.3. Blockchain-Based Advertising System. Recently, blockchain has been widely adopted in many disciplines owing to its trusted computing model and open nature. Therefore, blockchain-based advertising systems [5, 6] are proposed to provide trust between entities in advertising business. The scheme of Liu et al. [5] develops transparent and accountable vehicular local advertising (TAVLA) by utilizing the message digest, multi-party verification, and smart contract of blockchain. Specifically, the hash of the advertising information database is stored in the blockchain, and the code of the advertising query functions is stored off-chain. A vehicle user first requests the ad off-chain, and then the off-chain query result will be verified and assembled in the blockchain smart contract, and finally, the smart contract sends the result to the user. However, the communication data in this scheme is in plaintext, which is not secure for online advertising systems. Moreover, to improve the low trust caused by the click fraud in the online advertising system, Ding et al. [6] designed and implemented a blockchain-based digital advertising media system (B2DAM) and deployed the business logic based on smart contracts and Hyperledger SDK. But the communication cost between multiple blockchains in their scheme is expensive, as they read and write messages too many times on the blockchains.

All in all, in the above schemes, all real-time interactions of entities are settled in the blockchain, and each ad delivery and click behavior are recorded in the blockchain, so that the throughput is hard to meet the high concurrency in the advertising system. As a result, we periodically record the analysis results of the clicks on the blockchain in our scheme.

3. Preliminaries

3.1. Blockchain. Blockchain records all the transactions which are generated in a peer-to-peer network, and it is actually a decentralized ledger system. In the system, all the blocks include the hash of the previous block; in this way, they are linked together by the hash, and a blockchain is formed. According to the decreasing order of decentralization, the blockchain consists of three categories: public blockchain, consortium blockchain, and private blockchain [40, 41]. The public blockchain is open to all nodes, and everyone can read and write data on it. The consortium blockchain is partially open since it is managed by several organizations and only the authenticated members can access and record data on it. Also, the private blockchain is considered to be centralized as it is fully controlled by a single enterprise or organization. Considering that several enterprises and organizations are included in the advertising system, the consortium blockchain is adopted in our scheme.

3.2. Shamir (t, n) Threshold Secret Sharing Algorithm. A threshold secret sharing algorithm [42] was proposed by Shamir in 1979 to share a master secret in a safe way. In the literature, a trusted center (TC) splits the master secret K into n sub-secrets $(K_1, K_2, K_3, \ldots, K_n)$ and then distributes them to n participants $(U_1, U_2, U_3, \ldots, U_n)$. The master secret K cannot be reconstructed with fewer than t sub-secrets and the specific steps are described as follows. Firstly, a random $(t-1)$-th degree polynomial as $f(x) = a_0 + a_1 x + \cdots + a_{t-1} x^{t-1}$ is generated by the TC, in which the master secret $K = f(0) = a_0$. Then, the TC calculates n sub-secrets $K_i = f(x_i)$, $i = 1, 2, \ldots, n$ and allocates K_i to U_i secretly. Next, when U_i receives K_i, he saves it safely. Finally, the master key K can be recovered by the TC using the *Lagrange interpolation formula*: $f\prime(x) = \sum_{i=1}^{t} K_i \sum_{j=1, j\neq i}^{t} x - x_j / x_i - x_j$, and the master key $K = f\prime(0)$.

3.3. Bilinear Mapping. Assume G_1, G_2, and G_T denote three additive and multiplicative cyclic groups of the same order p, where p is a large prime and q is the generator of G_1, G_2. Besides, ψ: $G_2 \longrightarrow G_1$ is an isomorphism, and G_1, G_2, G_T are equipped with pairing. The bilinear pairing mapping e: $G_1 \times G_2 \longrightarrow G_T$ satisfies the following properties [43, 44].

(1) *Bilinear*: $\forall P \in G_1, Q \in G_2$ and $a, b \in \mathbb{Z}_p^*$, where $\mathbb{Z}_p^* = [1, 2, \ldots,]$; if $e(aP, bQ) = e(P, Q)^{ab}$, the mapping e: $G_1 \times G_2 \longrightarrow G_T$ is said to be bilinear.

(2) *Non-degenerate*: there exists $P \in G_1, Q \in G_2$ such that $e(P, Q) \neq 1_{G_T}$.

(3) *Computability*: $\forall P \in G_1, Q \in G_2$, there is an efficient algorithm to compute $e(P, Q)$.

The group G_T that possesses such a map e is called a bilinear group.

3.4. Ciphertext-Policy Attribute-Based Encryption (CP-ABE). CP-ABE schemes [45, 46] are designed to realize complex access control on encrypted data. In CP-ABE, a party wishing to encrypt a message M specifies a policy by an access tree T, and the private key must meet the policy to decrypt it, where the access tree is constructed by the party and the private key is generated by a set of descriptive attributes S of the decryptors. In the access tree T, each non-leaf node represents a threshold gate, described by its children and a threshold value, and each leaf node x of the tree is described by an attribute y and a threshold value t_x. To facilitate working with the access trees, the parent of the node x is described by $\text{parent}(x)$, and the function $att(x)$ is defined only if x is a leaf node and denotes the attribute associated with the leaf node x. The access tree T also defines an ordering between the children of every node, that is, the children of a node are numbered from 1 to num. The function $\text{index}(x)$ returns such a number associated with the node x.

According to [47], the four algorithms of the bilinear mapping-based CP-ABE scheme are as follows:

(1) *Setup*: this algorithm gives the public parameters PK and master key MK. It chooses a bilinear group G_1 of prime order p with generator g. Next, it chooses two random exponents $\alpha, \beta \in \mathbb{Z}_p$. Then, the public key is published as

$$\mathrm{PK} = \{G_1, g, h, f, l\}, \tag{1}$$

where $h = g^{\beta}$, $f = g^{1/\beta}$, $l = e(g, g)^{\alpha}$, and the master key MK is (β, g^{α}).

(2) *Encrypt* (*PK, M, T*): this algorithm encrypts message M to get ciphertext CT using the public parameters PK and the access tree T. In detail, it first chooses a polynomial p_x for each node x (including the leaves) in the tree T, in which the degree d_x of the polynomial p_x is one less than the threshold value t_x, that is, $d_x = t_x - 1$. Starting with the root node R in T, the algorithm chooses a random $s \in \mathbb{Z}_p$ and sets $p_R(0) = s$. Then, it sets $p_x(0) = p_{\mathrm{parent}(x)}(\mathrm{index}(x))$. Finally, it lets Y be the set of leaf nodes in T, and the ciphertext CT is

$$\begin{aligned} CT = \Big(& T, \widetilde{C} = M \cdot l^s, C = h^s, \forall y \in Y{:}\ C_y = g^{p_y(0)}, \\ & C_y' = (h(att(y)))^{p_y(0)}\Big). \end{aligned} \tag{2}$$

(3) *KeyGen* (*MK, S*): the KeyGen algorithm outputs the private key SK using the master key MK and the attribute set S. Firstly, it chooses $r \in \mathbb{Z}_p$ and $r_j \in \mathbb{Z}_p$ for each attribute $j \in S$. Then, it computes the private key SK as

$$\mathrm{SK} = \Big(D = g^{(\alpha+r)/\beta}, \forall j \in S{:}\ D_j = g^r \cdot h(j)^{r_j}, D_j' = g^{r_j}\Big). \tag{3}$$

(4) *Decrypt* (*CT, SK*): this algorithm decrypts the ciphertext CT with the private key SK for people who satisfy the attribute set S. The decryption procedure is a recursive algorithm in which

$$\begin{aligned} \mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, x) &= \frac{e(D_i, C_x)}{e(D_i', C_x')} \\ &= e(g, g)^{r p_x(0)}. \end{aligned} \tag{4}$$

When x is the root node R in the tree T, it can be concluded that $\mathrm{DecryptNode}(\mathrm{CT}, \mathrm{SK}, R) = e(g, g)^{r p_R(0)} = A$. Then, the message M can be computed by

$$\begin{aligned} \frac{\widetilde{C}}{(e(C, D)/A)} &= \frac{\widetilde{C}}{\left(e\left(h^s, g^{(\alpha+r)/\beta}\right)/e(g, g)^{rs}\right)} \\ &= M. \end{aligned} \tag{5}$$

4. Problem Statement

Many digital identity infrastructures [34–38] have emerged to better manage user's behavior in cyberspace, among which an identity management agency is responsible for generating and maintaining one-to-one mappings between digital identities and physical identities. The one-to-one mapping of digital identity infrastructures can prevent identity-based attacks (Sybil, whitewashing, etc.) in cyberspace. Therefore, based on the existing infrastructures, we designed our blockchain-based click fraud detection and prevention scheme (BCFDPS) for online advertising system. To elaborate our scheme clearly, the main entities and procedures of existing digital identity infrastructures are also included.

4.1. System Model. The proposed BCFDPS consists of seven entities: identity management agency (IMA), entity identity blockchain (EIB), consumer (U), access behavior blockchain (ABB), media site (MS), publisher (PUB), and advertiser (ADE), where the IMA and EIB are the entities of the existing digital identity infrastructures, as shown in Figure 2.

(i) *IMA* generates identities for U, PUB, and ADE, issues their identity licenses, and provides U with a signature on U's masked identity during the registration phase. IMA records the real identities of U, PUB, and ADE in EIB. Note: the IMA belongs to the existing digital identity infrastructures.

(ii) *EIB* is responsible for recording the hash of digital identity in the advertising system other than MS. Also, it is a consortium blockchain maintained by several IMAs. Note: the EIB belongs to the existing digital identity infrastructures.

(iii) U sends the encrypted masked identity and ad click messages to MS whenever he visits MS's website and clicks the ad.

(iv) *ABB* is in charge of recording the information of PUB's advertising bidding, MS's forwarding results of U's click message, and PUB's analysis result of U's access behavior. The ABB is a consortium blockchain, which is controlled by many MSs and PUBs.

(v) *MS* represents the media site between U and PUB. It is responsible for displaying PUB's ad for U and forwarding all the click messages of U for PUB. MS summaries the result of ad bidding and the forwarding information and periodically records them in the ABB. MS can verify the click number independently for detecting and preventing click fraud.

(vi) *PUB* publishes ADE's ad, joins the ad bidding process of MS, analyzes the effective ad clicks generated by U, and records the analysis results in the ABB. PUB can verify the click number independently for detecting and preventing click fraud.

(vii) *ADE* sends an ad to PUB for publishing, and he can also detect and prevent click fraud alone through verifying the number of clicks in the ABB.

4.2. Security Model. In BCFDPS, we have the following security assumptions.

(i) IMA and PUB are semi-honest and they will strictly follow the protocol but are curious about the information.

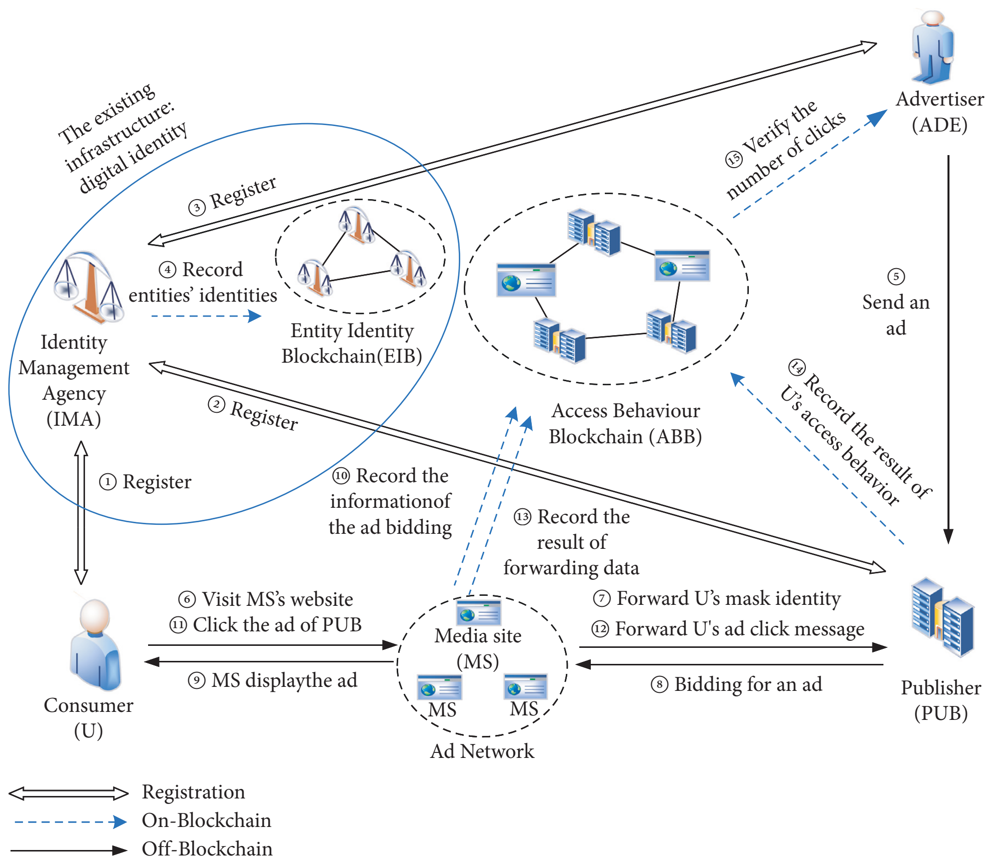

The existing infrastructure: digital identity
③ Register
④ Record entities' identities
Entity Identity Blockchain(EIB)
Identity Management Agency (IMA)
② Register
① Register
⑩ Record the informationof the ad bidding
⑬ Record the result of forwarding data
Access Behaviour Blockchain (ABB)
⑮ Verify the number of clicks
Advertiser (ADE)
⑤ Send an ad
⑭ Record the result of U's access behavior
⑥ Visit MS's website
⑪ Click the ad of PUB
⑨ MS displaythe ad
Consumer (U)
Media site (MS)
MS
MS
Ad Network
⑦ Forward U's mask identity
⑫ Forward U's ad click message
⑧ Bidding for an ad
Publisher (PUB)
Registration
On-Blockchain
Off-Blockchain

Figure 2: System model.

(ii) *U* is considered as a malicious entity and he will intentionally click on the same ad many times out of profit or curiosity.

(iii) MS is regarded as dishonest. It may deploy click fraud methods and even directly tamper with the statistical results of clicks to try to obtain extra illegal revenue from the PUB.

(iv) ADE is also seen as a malicious entity. He may attempt to deliberately falsify his statistics to reduce the ad expenses from the PUB.

(v) Two consortium blockchains, maintained by multiple IMAs, MSs, and PUBs, respectively, are fast and secure enough in recording transactions. Also, we assume that the standard cryptographic algorithms used in our scheme are secure and unbreakable.

(vi) It is built upon the Canetti–Krawczyk (CK) threat model [48], in which any two parties could communicate via an unauthenticated network. Specifically, an adversary can fully control the communication in a probabilistic polynomial time and try to reveal, track, or even imitate *U* through sniffing and tampering with messages between *U* and MS.

(vii) Corresponding to the physical identity, a person in cyberspace has his one and only digital identity. The *U*'s digital identity in each ad click message is protected by a masked identity and the random numbers, and the click message is generated by *U*'s browser plugin, where the plugin is assumed to be integrated in the browser in advance to protect *U*'s privacy.

4.3. Design Goals. According to the above system model and security model, the design goals of our scheme are as follows.

(1) *No impact on ad precision targeting*: a PUB can still accurately target an ad to a *U* although the *U*'s digital identity is masked. In other words, only the PUB can link the *U*'s masked identity from different click messages.

(2) *Acceptability of ad response speed*: the ad response speed in our scheme is acceptable for a *U*, even though the cryptographic algorithms are used to protect the *U*'s identity privacy in the process of publishing an ad.

(3) *Transparency of ad billing fee*: MS, PUB, and ADE can count the real number of clicks on the same ad in

an independent way. That is, the process of verifying the ad billing fee is transparent between MS, PUB, and ADE.

5. Proposed BCFDPS

The BCFDPS is proposed to detect and prevent click fraud, and it is mainly divided into three phases, as shown in Figure 2. The first phase allows U, PUB, and ADE to register with the IMA and obtain their digital identities and identity licenses. Meanwhile, IMA stores the hash value of their identities on the EIB, as shown in steps ①–④ (note: the four steps belong to the existing digital identity infrastructure). The second phase permits MS and PUB to work together to recommend ADE's ad to U. U clicks the ad that he is interested in and MS records the hash value of the data that it forwarded in the ABB, as shown in steps ⑤–⑬. The last phase lets both PUB and ADE detect and prevent click fraud independently using the data in the ABB, which is shown in steps ⑭ and ⑮.

The detailed process of BCFDPS includes four phases: initialization, registration, ad publishing, and click fraud detection and prevention. To elaborate our scheme clearly, the notations and descriptions of BCFDPS are shown in Table 1.

5.1. Initialization. The digital identity is the cornerstone of cyberspace which is provided and validated whenever a user accesses the network services. The initialization of this section is not exclusive to our scheme. In other words, in order to describe our scheme clearly, the pivotal initialization of the existing digital identity infrastructure is described in this section. Specifically, the IMA performs initialization to generate its public and private keys, and the EIB generates the system public parameters PP. In addition, U, PUB, and ADE generate their public and private keys.

5.1.1. IMA Initialization. IMA initializes its public and private keys PK_{IMA}/SK_{IMA}. Then, IMA publishes PK_{IMA} in the system. In addition, IMA defines PUB's attribute set including but not limited to these attributes $S = \{\{PUB\}, \{Hat \cup Pants \cup Shoes \cup \ldots\}\}$.

5.1.2. EIB Initialization. The EIB performs initialization to generate the system public parameters PP. Firstly, it selects a large prime p, an elliptic curve $E_p(a, b)$, and a base point P with order n under the finite field F_p. Then, it chooses a bilinear group G with generator g and two random numbers $\alpha, \beta \in \mathbb{Z}_p$. Next, it calculates parameters: $h = g^{\beta}$, $f = g^{1/\beta}$, $l = e(g, g)^{\alpha}$, and $MK = (\beta, g^{\alpha})$, and publishes $PP = \{E_p(a, b), P, G, g, hf, l, MK\}$ in the system so that IMA can get PP. Lastly, EIB generates a shared private key x denoting the master key described in Section 3.2, and it uses the Shamir (t, n) threshold secret sharing algorithm [42] to distribute the sub-secrets of x to each IMA.

5.1.3. U, PUB, and ADE Initialization. U, PUB, and ADE also generate their own public and private keys, referred to as PK_U/SK_U, PK_{PUB}/SK_{PUB}, PK_{ADE}/SK_{ADE}.

Table 1: Notations and descriptions.

Notation	Description
PP	Public parameter generated by EIB
UID	U's real digital identity
MUID	U's masked identity
MS	The identity of media site
PUBID	The identity of publisher
ADEID	The identity of advertiser
ID_{ad}	The identity of an ad
x	The shared private key of EIB
S	The publisher's attribute set
U_{Sig}	The signature of U from IMA
PK_e	The public key of entity e
SK_e	The private key of entity e
IL_e	The identity license of entity e
AuS_e	The authentication symbol for entity e
SA_e	The secure authentication for entity e
ts, ts_1, ts_2	The timestamp
$h(\cdot)$	hash function: $\{0,1\}^* \longrightarrow \{0,1\}^n$
$U\|V$	Concatenate operation between U and V
$E_a(b)/D_a(b)$	Encrypt/decrypt message b with key a
$Sig_a(b)$	The signature on message b with key a
M	Message generated after U clicks an ad

5.2. Registration. Similar to Section 5.1, the registrations of U, PUB, and ADE are not exclusive to our scheme. In other words, in order to describe our scheme clearly, the pivotal registration of the existing digital identity infrastructure is described in this section. Specifically, U registers with IMA to obtain his real identity UID, identity license IL_U, and the signature U_{Sig}. Similar to U, PUB registers with IMA to get its digital identity PUBID, identity license IL_{PUB}, and an attribute set S as an ad publisher. Also, ADE receives his digital identity ADEID and identity license IL_{ADE} from IMA.

5.2.1. U Registration (UR)

STEP UR1. IMA collects U's biometric data, e.g., fingerprint, digitalizes the fingerprint to obtain the digitized data, and selects and assembles a set of unique code segments from the code library according to the data, and at the same time, the hash value of the code segments is calculated. The hash value is U's real identity UID. Note that if a U is disguised by a machine or has already registered, the IMA would not generate an identity for the U. In order to issue an identity license IL_U to U, the IMA gathers other's sub-secrets and uses the Shamir (t, n) threshold secret recovering algorithm [42] to recover the shared private key x for calculating $IL_U = UID \cdot h(x) \cdot P$ and U's masked identity $MUID = UID \cdot P$. Then, IMA sends $E_{PK_U}(UID\|IL_U\|U_{Sig}\|PP)$ to U's browser plugin, where U_{Sig} is the signature of U from IMA, shown in (6), and PP refer to the public parameters in EIB. This process is shown in step ① in Figure 2.

$$U_{Sig} = Sig_{SK_{IMA}}(h(IL_U)\|MUID). \quad (6)$$

STEP UR2. At last, IMA records $h(UID)$ in EIB, which is used for accountability when a click fraud happens. This process is shown in step ④ in Figure 2.

5.2.2. PUB Registration (PUBR)

STEP PUBR1. IMA generates PUB's identity PUBID and an attribute set $S = \{\{\text{PUB}\}, \{\text{Hat} \cup \text{Food} \cup \ldots\}\}$ according to the business scope of PUB. Hereafter, similar to *STEP UR1*, IMA generates PUB's identity license $\text{IL}_{\text{PUB}} = \text{PUBID} \cdot h(x) \cdot P$ and sends $E_{\text{PK}_{\text{PUB}}}(\text{PUBID} \| \text{IL}_{\text{PUB}} \| S \| \text{PP})$ to PUB. This process is shown in step ② in Figure 2.

STEP PUBR2. At last, IMA records the $h(\text{PUBID})$ in EIB for supervision when a click fraud appears. This process is shown in step ④ in Figure 2.

5.2.3. ADE Registration (ADER)

STEP ADER1. IMA generates ADE's identity ADEID and identity license $\text{IL}_{\text{ADE}} = \text{ADEID} \cdot h(x) \cdot P$ and sends $E_{\text{PK}_{\text{ADE}}}(\text{ADEID} \| \text{IL}_{\text{ADE}})$ to ADE. This process is shown in step ③ in Figure 2.

STEP ADER2. At last, IMA records $h(\text{ADEID})$ in EIB to supervise when a click fraud arises. This process is shown in step ④ in Figure 2.

5.3. Ad Publishing

5.3. Ad Publishing. To obtain higher revenue, PUB often publishes ads to a targeted U through MS's ad bidding. Then, U clicks the ad that he is interested in.

5.3.1. Publisher Publishes an Ad (PPA). This phase deals with the process that a browser plugin sends U's masked identity in ciphertext to PUB and PUB displays the related ads to the targeted U, as shown in steps ①–⑥ in Figure 3.

STEP PPA1. ADE sends an ad to PUB for publication. Then, ADE and PUB reach a consensus on ADE's ad and create the ad's identity ID_{ad}, as shown in step ① in Figure 3.

STEP PPA2. U sends his masked identity in ciphertext (instead of a real identity in the real world) to PUB for getting the ad that he is interested in, protecting his privacy. Firstly, U visits MS's website, and U's web browser plugin encrypts the secret U_{Sig} through the CP-ABE algorithm to prevent entities other than the collection of PUBs from obtaining U's identity privacy. Then, U sends the ciphertext CT to the MS. After that, the MS directly broadcasts the CT to the PUBs cooperating with the MS. Here is the specific process.

U uses the public parameters PP and defines an access tree T according to the attributes of ads that he is interested in. The format of T is shown in Figure 4, where "1/2" means that PUB must satisfy at least one of the two attributes $\{\text{Hat} \cup \text{Shoes}\}$. Then, U encrypts the secret U_{Sig} to get the ciphertext CT.

$$\begin{aligned} \text{CT} = \Big(T, \widetilde{C} = \left(U_{\text{Sig}} \| ts_1\right) \cdot l^s, C = h^s, \forall y \in Y: \\ C_y = g^{p_y(0)}, C_y' = (h(att(y)))^{p_y(0)}\Big), \end{aligned} \tag{7}$$

where ts_1 is the timestamp, $l = e(g,g)^{\alpha}$, $h \in \text{PP}$, $s \in \mathbb{Z}_p$ is a random number, p_y is a polynomial for each node y in T, and $p_y(0) = s$ and $att(y)$ are the attributes associated with the leaf node y.

After U visits MS's website, U's browser plugin sends CT to MS, which is then directly broadcast to different PUBs by MS. This process is as in steps ② and ③ in Figure 3.

STEP PPA3. Next, PUB decrypts $\widetilde{C}$ in CT to get the secret U_{Sig}. Further, PUB gets U's masked identity MUID from the U_{Sig} and decides whether to bid for an ad according to the MUID, as shown in step ④ in Figure 3. The specific process of getting the U_{Sig} is as follows.

According to the attribute set S obtained as described in Section 5.2.2, PUB uses the master key MK in the public parameters PP to generate the decryption key SK according to

$$\text{SK} = \left(D = g^{(\alpha+r)/\beta}, \forall j \in S: D_j = g^r \cdot h(j)^{r_j}, D_j' = g^{r_j}\right), \tag{8}$$

where $r, r_j \in \mathbb{Z}_p$ are random numbers, $j \in S$ is an attribute, and α, β belong to the public parameter PP.

Then, PUB uses (9) to decrypt the leaf nodes of the access tree T with SK and CT when PUB's $S = \{\{\text{PUB}\}, \{\text{Hat} \cup \text{Food} \cup \ldots\}\}$ satisfies the attributes which U requires.

$$\text{DecryptNode}(\text{CT}, \text{SK}, x) = \frac{e(D_i, C_x)}{e(D_i', C_x')} = e(g,g)^{r p_x(0)}, \tag{9}$$

where x is a leaf node in T. After PUB obtains all leaf nodes, it uses the Lagrangian interpolation formula to obtain the parent node, and this process is recursive until T's root node is obtained. T's root node is $A = e(g,g)^{r \cdot p_R(0)}$.

Next, PUB can get the secret U_{Sig} by

$$\begin{aligned} \frac{\widetilde{C}}{(e(C,D)/A)} &= \frac{\widetilde{C}}{\left(e\left(h^s, g^{(\alpha+r)/\beta}\right)/e(g,g)^{rs}\right)} \\ &= \left(U_{\text{Sig}} \| ts_1\right). \end{aligned} \tag{10}$$

At last, PUB decrypts U_{Sig} with IMA's public key PK_{IMA} and obtains the masked identity MUID of U. PUB searches its local database to get the portrait of MUID and decides whether to bid for the ad. If a PUB joins in the bidding process, it sends the $\text{ID}_{a\,d}$ and the price fee to the MS. This bidding process will be executed by many PUBs.

STEP PPA4. MS displays the ad of the bid winner and sends the PUBID, ADEID, PK_{PUB}, PK_{ADE}, and ad frame to the U, as shown in step ⑤ in Figure 3.

STEP PPA5. MS periodically (e.g., once a day) records the results of the ad bidding in the ABB sorted by periods and $\text{ID}_{a\,d}$ s. The format of the results is

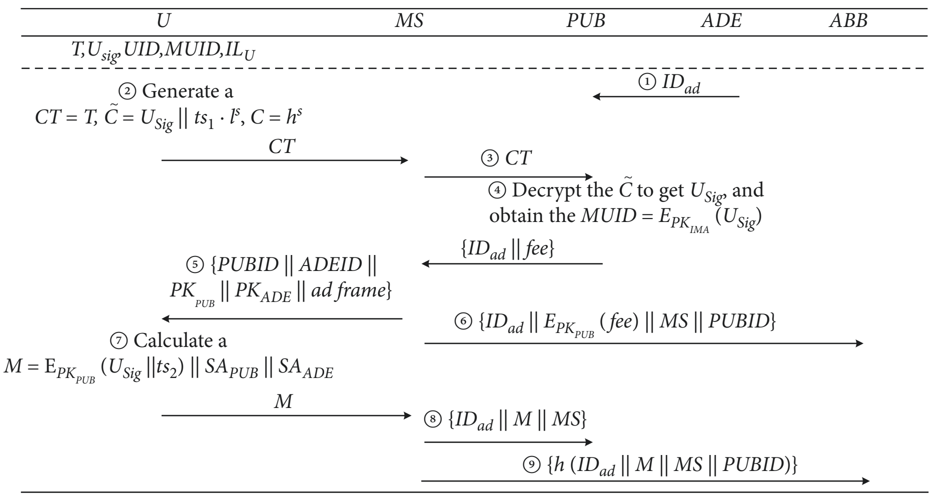

U
MS
PUB
ADE
ABB
$T, U_{sig}, UID, MUID, IL_U$
① ID_{ad}
② Generate a
$CT = T, \tilde{C} = U_{Sig} \| ts_1 \cdot l^s, C = h^s$
CT
③ CT
④ Decrypt the $\tilde{C}$ to get U_{Sig}, and
obtain the $MUID = E_{PK_{IMA}}(U_{Sig})$
$\{ID_{ad} \| fee\}$
⑤ $\{PUBID \| ADEID \| PK_{PUB} \| PK_{ADE} \| ad\ frame\}$
⑥ $\{ID_{ad} \| E_{PK_{PUB}}(fee) \| MS \| PUBID\}$
⑦ Calculate a
$M = E_{PK_{PUB}}(U_{Sig} \| ts_2) \| SA_{PUB} \| SA_{ADE}$
M
⑧ $\{ID_{ad} \| M \| MS\}$
⑨ $\{h(ID_{ad} \| M \| MS \| PUBID)\}$

Figure 3: Process of publishing an ad to a targeted consumer.

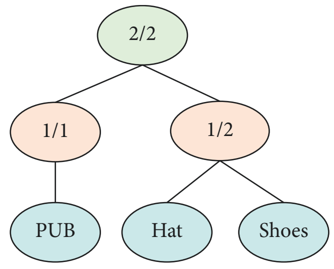


Figure 4: An example of U's access tree T.

$\{ID_{ad} \| E_{PK_{PUB}}(\text{fee}) \| MS \| PUBID\}$, where "fee" is the price that PUB should pay to MS after an ad is clicked once by U, as shown in step ⑥ in Figure 3.

5.3.2. U Clicks the Ad (UCA). U clicks the ad that he is interested in after MS displays it on the website. This section is shown in steps ⑦–⑨ in Figure 3.

STEP UCA1. U's browser plugin gets PUBID, ADEID, PK_{PUB}, and PK_{ADE} from the ad frame and calculates $AuS_{PUB} = e(IL_U \cdot PUBID, P)^{h(IL_U)}$ and $AuS_{ADE} = e(IL_U \cdot ADEID, P)^{h(IL_U)}$. It then embeds the timestamp ts_2 to calculate $SA_{PUB} = E_{PK_{PUB}}(AuS_{PUB} \| ts_2)$ and $SA_{ADE} = E_{PK_{ADE}}(AuS_{ADE} \| ts_2)$. Next, the plugin sends the click message M about the ad to MS. The click message is shown as in equation 6 and as in step ⑦ in Figure 3.

$$M = E_{PK_{PUB}}\left(U_{Sig} \| ts_2\right) \| SA_{PUB} \| SA_{ADE}. \quad (11)$$

STEP UCA2. MS forwards $\{ID_{ad} \| M \| MS\}$ to the PUB who won the bidding and stores $\{ID_{ad} \| M \| MS \| PUBID\}$ in its local database, as shown in step ⑧ in Figure 3.

STEP UCA3. Finally, the data $\{h(ID_{ad} \| M \| MS \| PUBID)\}$ are classified by periods and ID_{ad} s and periodically (e.g., once a day) recorded in the ABB by the MS, as shown in step ⑨ in Figure 3.

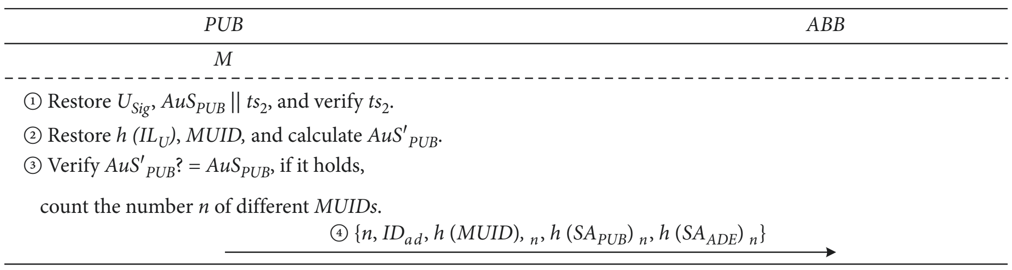
5.4. Click Fraud Detection and Prevention. This phase achieves click fraud detection and prevention between entities in an advertising system based on ABB.

5.4.1. PUB Detects and Prevents Click Fraud (PUBD). To prevent MS from forging the data and ensure the transparency of this ad click analysis process, PUB can detect and prevent fraudulent click, and it is shown in Figure 5.

STEP PUBD1. PUB uses its private key SK_{PUB} to decrypt M from MS to obtain the secret U_{Sig} and $\{AuS_{PUB} \| ts_2\}$ from SA_{PUB}. The PUB verifies the timeliness of the ts_2 to prevent the replay attacks, as shown in step ① in Figure 5.

STEP PUBD2. PUB uses IMA's public key PK_{IMA} to restore $h(IL_U)$ and MUID from the U_{Sig} and then calculates AuS_{PUB} by (12), as shown in step ② in Figure 5:

$$\begin{aligned} e(IL_{PUB}, MUID)^{h(IL_U)} &= e(IL_{PUB}, UID \cdot P)^{h(IL_U)} = e(IL_{PUB}, P)^{UID \cdot h(IL_U)} \\ &= e(IL_U \cdot PUBID, P)^{h(IL_U)} \\ &= AuS'_{PUB}. \end{aligned} \quad (12)$$



① Restore U_{Sig}, $AuS_{PUB} \,||\, ts_2$, and verify ts_2.

② Restore $h\ (IL_U)$, $MUID$, and calculate AuS'_{PUB}.

③ Verify $AuS'_{PUB}? = AuS_{PUB}$, if it holds,

count the number n of different $MUIDs$.

④ $\{n, ID_{ad}, h\ (MUID)_{,\ n}, h\ (SA_{PUB})_{\ n}, h\ (SA_{ADE})_{\ n}\}$

Figure 5: Process of PUB verifying the effective clicks.

STEP PUBD3. If (12) holds, PUB counts the number of different MUID s in AuS_{PUB} s, denoted as n, which is the number of valid advertising clicks in a certain period (e.g., one day). This means that in this period, PUB only pays for n clicks to MS. In this way, the PUB can detect all the fraudulent clicks in the message forwarded by MS. Also, the PUB pays nothing to MS for the repeated MUIDs, so the click fraud by a malicious MS can be prevented. Simultaneously, PUB records U's access behavior information like $\{\text{ID}_{a\ d} \| M \| \text{MUID} \| U_{\text{Sig}} \| \ \text{SA}_{\text{PUB}} \| ts_2 \| \text{SA}_{\text{ADE}} \| \text{MS} \| \text{fee} \| \text{behaviour}\}$ locally, as shown in step ③ in Figure 5.

STEP PUBD4. Finally, PUB periodically (e.g., once a day) records the result $\{n, \text{ID}_{a\ d}, h(\text{MUID})_n, h(\text{SA}_{\text{PUB}})_n, h(\text{SA}_{\text{ADE}})_n\}$ in ABB sorted by periods and $\text{ID}_{a\ d}$s, as shown in step in ④ in Figure 5.

5.4.2. ADE Detects and Prevents Click Fraud (ADED). Similarly, ADE also verifies the results recorded by PUB in the ABB to detect and prevent click fraud, and this section is shown in Figure 6.

STEP ADED1. ADE communicates with PUB to obtain the original access information $\{\text{ID}_{a\ d} \| U_{\text{Sig}} \| \text{MS} \| \text{fee} \| \text{SA}_{\text{ADE}}\}$ of U in the PUB local database. ADE then uses PK_{PUB} to encrypt the fee and compares it with the data on the ABB to prevent PUB's cheating, as shown in step ① in Figure 6.

STEP ADED2. ADE decrypts SA_{ADE} with private key SK_{ADE}, obtains $\{AuS_{\text{ADE}} \| ts_2\}$, and verifies the timeliness of ts_2 to prevent replay attacks, as shown in step ② in Figure 6.

STEP ADED3. Similar to *STEP PUBD2*, ADE also restores $h(\text{IL}_U)$ and MUID from U_{Sig}, and then calculates AuS'_{ADE} by

$$\begin{aligned} e(\text{IL}_{\text{ADE}}, \text{MUID})^{h(\text{IL}_U)} &= e(\text{IL}_{\text{ADE}}, \text{UID} \cdot P)^{h(\text{IL}_U)} \\ &= e(\text{IL}_{\text{ADE}}, P)^{\text{UID} \cdot h(\text{IL}_U)} \\ &= e(\text{IL}_U \cdot \text{ADEID}, P)^{h(\text{IL}_U)} \\ &= AuS'_{\text{ADE}}. \end{aligned} \tag{13}$$

If (13) holds, ADE also counts the number $n\prime$ of different MUIDs in AuS_{ADE}s in a certain period (e.g., one day), as shown in step ③ in Figure 6.

STEP ADED4. ADE reads the data n recorded in ABB by PUB and compares $n\prime$ with the n. If the equation $n\prime = n$ holds, ADE pays fee to PUB according to the n, as shown in step ④ in Figure 6. Therefore, the ADE can detect all the fraudulent clicks in the original access information from PUB. Also, the ADE pays nothing to PUB for the repeated MUIDs, so the fraudulent click by a malicious PUB can be prevented.

5.4.3. MS Detects and Prevents Click Fraud (MSD). MS obtains all the $\{AuS_{\text{PUB}} \| ts_2\}$ from PUB and uses PK_{PUB} to encrypt them successively to get the encrypted result $\text{SA}'_{\text{PUB}} = E_{\text{PK}_{\text{PUB}}}(AuS_{\text{PUB}} \| ts_2)$. Then, MS compares the SA_{PUB}' with the SA_{PUB} in M from MS's local database one by one; if it holds, the data from the PUB are valid. Finally, MS counts the number n'' of the different AuS_{PUB} and verifies if $n'' = n$ holds. If it holds, MS charges PUB fees according to the $n\prime\prime$. As a result, the MS can detect all the fraudulent clicks in the data from PUB. Also, MS cannot charge more PUB for the repeated AuS_{PUB}s, so the fraudulent click by a malicious MS can be prevented.

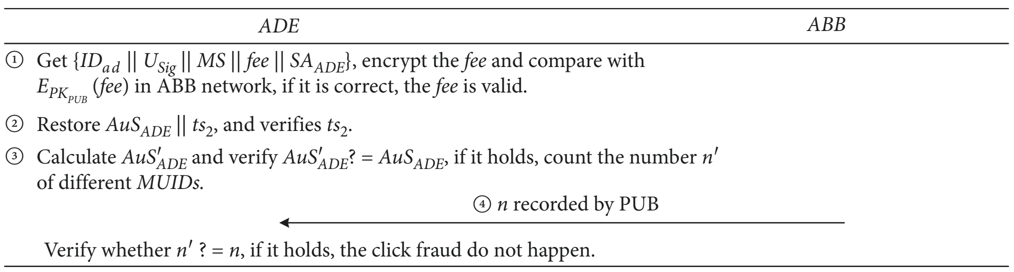
6. Security Analysis

In this section, we first analyze the security of our scheme from three levels: the processing level, the data level, and the infrastructure level, which can be called PDI model-based security [49–52]. Then, we give the informal analysis of security under the security assumptions in Section 4.2. Lastly, we demonstrate that the BCFDPS scheme is provably secure.

6.1. PDI Model-Based Security Analysis. As the one of the latest and most mature blockchain security analysis frameworks for Industry 4.0, the PDI model [49] conducts a comprehensive and detailed analysis of security issues. In the PDI model, the blockchain security is divided into three levels, which are the process level, the data level, and the infrastructure level [51]. Similarly, we also analyze the security of our blockchain-based click fraud detection and prevention scheme according to the three aspects.

6.1.1. The Process Security

(1) *Off-blockchain data processing security*: a large number of data processing operations are run off-



① Get $\{ID_{ad} \,||\, U_{Sig} \,||\, MS \,||\, fee \,||\, SA_{ADE}\}$, encrypt the *fee* and compare with $E_{PK_{PUB}}(fee)$ in ABB network, if it is correct, the *fee* is valid.

② Restore $AuS_{ADE} \,||\, ts_2$, and verifies ts_2.

③ Calculate AuS'_{ADE} and verify $AuS'_{ADE}? = AuS_{ADE}$, if it holds, count the number n' of different *MUIDs*.

④ n recorded by PUB

Verify whether $n'\ ? = n$, if it holds, the click fraud do not happen.

Figure 6: Process of ADE verifying the effective clicks.

blockchain in our scheme, since the data statistical analysis ability of the existing blockchain applications is weak [50]. In our scheme, a *U*'s masked identity MUID and his ad click message *M* are encrypted (denoted as mm) and sent to the MS by *U*'s browser plugin locally. Then, mm is forwarded to a PUB by a MS off-blockchain. Next, the MS, PUB, and ADE can independently count the real click number from mm with ECC and bilinear pairing algorithms. Also, since mm is ciphertext and being processed off-blockchain, it is difficult for an attacker to gather, crack, and modify it. That is, the data processing security off-blockchain is guaranteed in these entities.

(2) *Data processing security in the blockchain*: to implement our scheme in a real-time online advertising scenario, the data processing in the blockchain of our scheme is to periodically read and write content in the access behavior blockchain (ABB) through smart contracts. The ABB is a consortium blockchain that only allows authorized MSs and PUBs to write data, which avoids the unauthorized access. Also, the consensus protocol in ABB guarantees the correctness and consistency of the data when it is written to the ABB, largely eliminating exceptions in data processing and ensuring the security of data processing in the blockchain.

6.1.2. *The Data Security*

(1) *Data tamper-proof*: in our scheme, all original business data are stored in the local servers of MS and PUB, and the aggregated results of the original data are regularly recorded in the consortium blockchain as the form of hash values. In this way, even if attackers obtain the data in the blockchain, they cannot get the original data in the local servers of MS and PUB, so they cannot view or tamper with the original data. On the other hand, blockchain can ensure data consistency in distributed ledgers. Therefore, business data security is achieved whether the data are in the blockchain or not.

(2) *Consumer's identity privacy*: similar to the digital twin in [53–55], a U in our scheme can only obtain his unique digital identity to visit MS's websites and click on PUB's ads. Also, a masked identity MUID, CP-ABE algorithm, and ECC algorithm are utilized by the U to hide his identity, while preserving the ad precision targeting. In addition, nobody except the PUB can mark the U, and no one can reveal U's real digital identity UID. That is, our scheme protects the privacy of consumer's identity.

6.1.3. *The Infrastructure Security*

(1) *System structure security:* the two-level mutual verification between MS, PUB, and ADE maintains the stability of our system structure. For one thing, PUB counts the real and effective clicks from a large number of users' ad click messages *M*s which are forwarded by the MS. Once the *M*s are tampered with or forged by the MS, they cannot pass the verification of PUB. At the same time, the MS can use the ECC algorithm to count the real clicks from the data stored in local database to prevent PUB from forging the amount of the clicks. For the other thing, since the raw data are generated by *U*, the ADE can find anomalies once the PUB adds entries in the raw data. Thus, our scheme has system structure security.

(2) *Cryptographic facilities security*: we use the standard cryptographic facilities to build our system. Specifically, CP-ABE algorithm, bilinear pairing algorithm, and ECC algorithm are used by a *U* to protect his identity. The bilinear pairing algorithm and ECC algorithm are adopted by a PUB and an ADE to detect the fraudulent click, while a MS utilizes the ECC algorithm to detect a click fraud. The security of our scheme relies on these standard cryptographic facilities and we assume that the standard cryptographic facilities used in our scheme are secure and unbreakable.

6.2. *Informal Analysis of Security.*
In this section, we analyze the security of our scheme under the security assumptions in Section 4.2 in an informal way.

6.2.1. *Prevention of a False MUID.*
In Section 5.2.1, a machine cannot obtain a valid MUID since it has no way to pass the IMA biometric authentication. Even if it forges a false MUID, it still cannot generate a valid $U_{\mathrm{Sig}} = \mathrm{Sig}_{\mathrm{SK}_{\mathrm{IMA}}}(h(\mathrm{IL}_U)\|\mathrm{MUID})$ without IMA's private key

SK_{IMA}. That is, the click message, containing an invalid MUID, generated by the machine in phase 5.3.2 will be discarded. Therefore, in our scheme, the number of false MUIDs is not included in the number of valid clicks.

6.2.2. Transparency of Clicks between Entities. PUB decrypts the SA_{PUB} in M to get AuS_{PUB} using SK_{PUB}, then verifies the AuS_{PUB}, and counts the number n of different AuS_{PUB}. Similarly, ADE restores the AuS_{ADE} in SA_{ADE} from PUB's local database using SK_{ADE} to verify the authenticity of AuS_{ADE}, and then ADE counts the number $n\prime$ of different AuS_{ADE}. Although SA_{ADE} of M comes from PUB, AuS_{ADE} in SA_{ADE} is encrypted by PK_{ADE}, and only SK_{ADE} can decrypt it. Therefore, PUB cannot tamper with SA_{ADE}; furthermore, ADE ensures the validity of $n\prime$. MS encrypts the original data $\{AuS_{PUB}\|ts_2\}$ from PUB using PK_{PUB} and compares the encrypted result $SA'_{PUB} = E_{PK_{PUB}}(AuS_{PUB}\|ts_2)$ with SA_{PUB} in M from MS's local database to verify whether the PUB is honest. In this way, PUB, ADE, and MS can verify the number of clicks about the same ad in an independent way.

6.2.3. U's Conditional Unlinkability. First of all, U sends his masked identity in ciphertext CT to MS and MS broadcasts it to PUBs in phase 5.3.1. Then, only PUBs can decrypt U's U_{Sig} from CT since CT is calculated using the CP-ABE algorithm and only attributes S owned by PUBs can generate a decryption key SK. Secondly, U sends his click message M to MS and MS forwards it to PUB in phase 5.3.2. Next, only PUB can reveal U's masked identity MUID from M using its private key for advertising precision marketing. In the entire communication of U, neither the attacker in the channel nor the MS can directly link U's masked identity MUID because both CT and M are encrypted by CP-ABE algorithm or asymmetric cryptographic algorithm and only PUBs can decrypt them. However, even PUB cannot link U's masked identity MUID to U's real identity UID in ABB since PUB does not have the right to write and read in EIB. Thence, the scheme achieves U's conditional unlinkability.

6.2.4. Data Security and Integrity. Firstly, in this scheme, all the commercial contract data, e.g., fee, are encrypted and only the data owner PUB and MS can decrypt these ciphertexts. In addition, all the commercial contract data and the hash value of click result are recorded in the ABB (a consortium blockchain) which is shown in steps ⑩, ⑬, and, ⑭ and any adversary cannot tamper with these data in the consortium blockchain.

6.2.5. Resistance to Replay Attacks. In phases 5.3.1 and 5.3.2, the timestamp ts_1 and ts_2 are included in the message CT and M, and PUB first checks their timeliness to avoid replay attacks. Further, in phases 5.4.2 and 5.4.3, ADE and MS can avoid replay attacks by the timestamp ts_2. Consequently, our scheme is resistant to replay attacks in a great probability.

6.2.6. Resistance to Forgery. For one thing, in phase 5.3.1, MS stores U's CT while it has no ability to construct U's click message M in phase 5.3.2 without a IL_U. For another thing, in phase 5.4.2, PUB records SA_{PUB} and SA_{ADE}, but it cannot forge a SA_{ADE} because it also does not have a IL_U. In other words, the click message M containing MUID can only be generated by U. That is, the BCFDPS can resist forgery attacks.

6.3. Provable Security. The proposed scheme is based on bilinear pairing cryptosystem on elliptic curves (denoted as BPCEC), ciphertext-policy attribute-based encryption (denoted as CP-ABE), and elliptic curve cryptography (denoted as ECC). According to the security characteristics of each module, we show that our scheme meets click fraud detection and prevention and U's conditional unlinkability.

6.3.1. Theorem 1. If the BPCEC, CP-ABE, and ECC algorithms satisfy the basic security properties, then the scheme in this paper can detect and prevent click fraud.

Proof. Define A_{BPCEC} as an adversary who attacks the security of BPCEC algorithm, A_{CP-ABE} as an adversary attacking the security of CP-ABE algorithm, and A_{ECC} as an opponent attacking the security of ECC algorithm. Assuming A_{CF} clicks fraud successfully, a polynomial time algorithm $A_\theta \in (A_{BPCEC}, A_{CP-ABE}, A_{ECC})$ is defined, which has the ability to attack the algorithms of BPCEC, CP-ABE, and ECC. Through the query of A_{CF} and the A_θ's interaction in the click fraud game, A_θ is optimized repeatedly to successfully attack the BPCEC, CP-ABE, and ECC algorithms. That is, if the adversary A_{CF} clicks fraud successfully in the scheme, it means A_θ successfully attacks the security of algorithms of BPCEC, CP-ABE, and ECC with a certain probability.

According to the steps defined above, here are the interactions between algorithm A_θ and the adversary A_{CF}:

STEP 1. Registration phase: through the identity generated in U's registration phase, algorithm A_θ obtains U's digital identity and receives U's identity UID, U's masked identity MUID, U's identity license IL_U, and IMA's signature U_{Sig}. At last, A_θ sends $\{UID, MUID, IL_U, U_{Sig}\}$ to A_{CF}.

STEP 2. Inquiry phase: the adversary A_{CF} can query the algorithm A_θ for polynomial time:

(1) *Generate the ciphertext* CT: A_{CF} visits MS's website, generates the ciphertext CT by the CP-ABE algorithm, and sends CT to MS.
(2) *Generate the click message M*: MA_{CF} generates the click message M which contains a $b \in \{0, 1\}$ randomly selected by A_{CF} through BPCEC and ECC algorithm and A_{CF} then clicks PUB's ad to send M.

STEP 3. Verification phase: PUB verifies U's click message M and outputs $b\prime$ using the ECC and BPCEC algorithms. If $b\prime = b$ exists, it indicates that the adversary A_{CF} successfully carried out the click fraud attack. The success probability of the adversary A_{CF} is

$$\begin{aligned}
\mathrm{Adv}_{A_{\mathrm{CF}}}(k) &= Pr\left[\mathrm{Exp}_{A_{\mathrm{CF}}}(k) = 1\right] \\
&= Pr\left[A_{\mathrm{CF}}(\mathrm{verify}) = 1|b = 1\right]\cdot Pr[b = 1] + Pr\left[A_{\mathrm{CF}}(\mathrm{verify}) = 0|b = 0\right]\cdot Pr[b = 0] \\
&= \frac{1}{2}Pr\begin{bmatrix} A_{\mathrm{BPCEC}}(\mathrm{verify}) = 1, \\ A_{\mathrm{CP-ABE}}(\mathrm{verify}) = 1, |b = 1 \\ A_{\mathrm{ECC}}(\mathrm{verify}) = 1, \end{bmatrix} + \frac{1}{2}Pr\begin{bmatrix} A_{\mathrm{BPCEC}}(\mathrm{verify}) = 0, \\ A_{\mathrm{CP-ABE}}(\mathrm{verify}) = 0, |b = 0 \\ A_{\mathrm{ECC}}(\mathrm{verify}) = 0, \end{bmatrix} \\
&< \frac{1}{2}\left(Pr\left[A_{\mathrm{BPCEC}}(\mathrm{verify}) = 1|b = 1\right] + Pr\left[A_{\mathrm{CP-ABE}}(\mathrm{verify}) = 1|b = 1\right] + Pr\left[A_{\mathrm{ECC}}(\mathrm{verify}) = 1|b = 1\right]\right) \\
&+ \frac{1}{2}\left(Pr\left[A_{\mathrm{BPCEC}}(\mathrm{verify}) = 0|b = 0\right] + Pr\left[A_{\mathrm{CP-ABE}}(\mathrm{verify}) = 0|b = 0\right] + Pr\left[A_{\mathrm{ECC}}(\mathrm{verify}) = 0|b = 0\right]\right) \\
&= Pr\left[\mathrm{Exp}_{A_{\mathrm{BPCEC}}}(k) = 1\right] + Pr\left[\mathrm{Exp}_{A_{\mathrm{CP-ABE}}}(k) = 1\right] + Pr\left[\mathrm{Exp}_{A_{\mathrm{ECC}}}(k) = 1\right] \\
&= \mathrm{Adv}_{A_{\mathrm{BPCEC}}}(k) + \mathrm{Adv}_{A_{\mathrm{CP-ABE}}}(k) + \mathrm{Adv}_{A_{\mathrm{ECC}}}(k).
\end{aligned} \tag{14}$$

If an attacker A_{BPCEC} successfully attacks the BPCEC algorithm, an attacker $A_{\mathrm{CP-ABE}}$ successfully attacks the CP-ABE algorithm, and an attacker A_{ECC} can successfully attack ECC algorithm, A_{CF} can carry out the click fraud attack successfully. However, the probability of $A_{\mathrm{BPCEC}}, A_{\mathrm{CP-ABE}}$, and A_{ECC} successfully attacking the BPCEC, CP-ABE, and ECC algorithms is almost $1/n$, respectively; then, A_{CF} wins in the click fraud attack game of BCFDPS scheme with a probability of $3/n$. But, according to the assumptions that BPCEC, CP-ABE, and ECC algorithms satisfy the basic security properties, it is concluded that the probability of A_{CF} successfully attacking can be ignored, so the scheme can detect and prevent click fraud. □

6.3.2. Theorem 2. If all the crypto-algorithms such as BPCEC, CP-ABE, and ECC satisfy the basic security features, then U's conditional unlinkability can be achieved in the BCFDPS.

Proof. Define A_{BPCEC} as an adversary who attacks the linkability of MUID of BPCEC algorithm, $A_{\mathrm{CP-ABE}}$ as an adversary attacking the linkability of U_{Sig} of CP-ABE algorithm, and A_{ECC} as an opponent attacking the linkability of U_{Sig} of ECC algorithm. Assuming A_{CP} (except PUB) links U's masked identity MUID successfully, a polynomial time algorithm $A_\tau \in (A_{\mathrm{BPCEC}}, A_{\mathrm{CP-ABE}}, A_{\mathrm{ECC}})$ is defined, which has the ability to attack the algorithms of BPCEC, CP-ABE, and ECC. During the communication process of U, MS, and PUB, two messages CT and M are encrypted by the algorithms BPCEC, CP-ABE, and ECC. Therefore, for the adversary A_{CP}, the probability of successfully linking many different messages to the same U is

$$\begin{aligned}
Adv_{A_{CP}}(k) &= Pr\left[\mathrm{Exp}_{A_{\mathrm{CP}}}(k) = 1\right] \\
&= Pr[\mathrm{Ver}(\mathrm{CT}) = 1]\cdot Pr\left[\mathrm{Ver}\left(U_{\mathrm{Sig}}\right) = 1\right]\cdot Pr\left[\mathrm{Ver}\left(AuS_{\mathrm{PUB}}\right) = 1\right] \\
&= Pr\left[\mathrm{Exp}_{A_{\mathrm{CP-ABE}}}(k) = 1\right]\cdot Pr\left[\mathrm{Exp}_{A_{\mathrm{ECC}}}(k) = 1\right]\cdot Pr\left[\mathrm{Exp}_{A_{\mathrm{BPCEC}}}(k) = 1\right] \\
&= \mathrm{Adv}_{A_{\mathrm{CP-ABE}}}(k)\cdot \mathrm{Adv}_{A_{\mathrm{ECC}}}(k)\cdot \mathrm{Adv}_{A_{\mathrm{BPCEC}}}(k).
\end{aligned} \tag{15}$$

Therefore, if the attacker A_{BPCEC} successfully attacks the BPCEC algorithm, the attacker $A_{\mathrm{CP-ABE}}$ successfully attacks the CP-ABE algorithm, and the attacker A_{ECC} successfully attacks the ECC algorithm, then A_{CP} wins in the conditional unlinkability simulation attack game. However, according to the assumptions about these security features, the probability of A_{CP} successfully attacking can be ignored. As a result, the scheme accomplishes U's conditional unlinkability. □

7. Implementation and Evaluation

We evaluate our scheme in terms of computation, communication, storage, and Ethereum gas cost based on JPBC library [56] and Ethereum.

In the proposed scheme, four phases of initialization, registration, ad publishing, and click fraud detection and prevention are involved. Because the first two phases happen rarely, they are not implemented in this section and we mainly focus on the phases of ad publishing and click fraud detection and prevention in which an ad is published and the click fraud is detected and prevented.

7.1. Computation Cost

7.1.1. Evaluation of Our Scheme. We mainly focus on the phases of the ad publishing and click fraud detection and prevention in this section. We execute evaluation tests to get the time cost of meta-operations and the evaluation test is based on a PC (Intel Core i5-9400F CPU @ 2.90 GHz, 16 GB RAM @ 2667 MHz and Windows 10×64). We use JDK 1.8, JPBC library [56], to support efficient bilinear pairing operations.

To achieve persuasive expression of computation comparison, the symbols and parameters are introduced: T_{C_Enc} is the encryption algorithm in CP-ABE scheme, T_{C_KG} denotes the key generation algorithm in CP-ABE scheme, T_{C_Dec} means the decryption algorithm in CP-ABE scheme, T_{E_Enc} expresses the encryption algorithm in ECC, T_{E_Dec} signifies the decryption algorithm in ECC, and T_{bp} represents the bilinear pairing operation. Their time cost is as follows: $T_{C\_Enc} = 146.41$ ms, $T_{C\_\mathrm{KG}} = 118.80$ ms, $T_{C\_De\ c} = 33.12$ ms, $T_{E\_Enc} = 3.17$ ms, $T_{E\_\mathrm{Dec}} = 0.36$ ms, and $T_{bp} = 6.79$ ms. In addition, the time cost of hash function and concatenate operation is small, and we do not take this into account in computation cost. The detailed computation costs for each phase are illustrated in Table 2.

In phase 5.3.1, U is required to perform one encryption algorithm in CP-ABE scheme and PUB needs to execute one key generation algorithm in CP-ABE scheme, one decryption algorithm in CP-ABE scheme, and one decryption algorithm in ECC, that is, the running time is $T_{C\_Enc} + T_{C\_\mathrm{KG}} + T_{C\_\mathrm{Dec}} + T_{E\_\mathrm{Dec}} = 298.69$ ms. According to Ma et al. [57], the response speed of publishing an ad in our scheme is in the acceptable threshold (150 ~ 600 ms) and is lower than the one in [6] which closes to 400 ms. In phase 5.3.2, U computes three encryption algorithms in ECC and two bilinear pairing operations. Therefore, the execution time to generate a click message is $3T_{E\_Enc} + 2T_{bp} = 23.09$ ms, and it has no effect on the user experience. Further, in phase 5.4.1, PUB is required to run three decryption algorithms in ECC and one bilinear pairing operation, that is $3T_{E\_\mathrm{Dec}} + T_{bp} = 7.87$ ms. In summary, the computation cost from publishing an ad for U (phase 5.3.1) to verifying the effective clicks by PUB (phase 5.4.1) is $298.69 + 23.09 + 7.87 = 329.65$ ms, where the time cost of one click fraud detection and prevention is only 7.87 ms. After PUB counts the effective clicks, ADE and MS will also verify the clicks to ensure their profit. Similar to PUB, ADE performs three decryption algorithms in ECC and one bilinear pairing operation, that is, $3T_{E\_\mathrm{Dec}} + T_{bp} = 7.87$ ms. For the MS, it executes one encryption algorithm in ECC to detect a click fraud, which is $T_{E\_Enc} = 3.17$ ms. From Table 2, it can be seen that the CP-ABE algorithm increases the run time in phase 5.3.1, but it protects U's privacy from MS and the sniffer of a channel. In addition, it should be noted that the computation overhead of PUB in phases 5.3.1 and 5.4.1 can be improved at the publisher with powerful computing clusters. Moreover, distributed and parallel optimization techniques for verifiable computations can also be adopted to further enhance publisher's performance in publishing the ad to a U who is the potential consumer of the ad.

On the other hand, blockchain is introduced in our scheme; in order to demonstrate the practical performance of our blockchain-based scheme, we evaluate the execution cost of our smart contract based on a public Ethereum testnet (Rinkeby). We used Chrome v89.0 explorer with the plugin MetaMask and Remix which is a browser-based IDE to connect the contract between Ethereum and the program simulated. Rinkeby testnet was started by the Ethereum team in April 2017 and it uses Clique PoA (Proof of Authority) consensus protocol. Importantly, it is immune to spam attacks, as Ether supply is controlled by several trusted parties and only they can write transactions in the blockchain, which makes it like a consortium blockchain; thence, the waiting time for transaction confirmation is relatively short to be ignored.

We deploy smart contracts on Rinkeby to record the transaction data and count the gas cost of smart contracts on deployment and recall. The gas cost of our scheme is shown in Table 2. In our scheme, a smart contract is only deployed once in phase 5.3.1 and the gas cost of deploying the contract is 89, 003. Additionally, in phases 5.3.1, 5.3.2, and 5.4.1, the cost of recalling the contract to write 256, 32, and 128 bytes of analysis result on blockchain is 27,054, 23,470, and 25,006 gas, respectively. All in all, judging from the evaluation results, our scheme is feasible in practice.

7.1.2. Comparison of the Computation Cost in Click Fraud Detection and Prevention Process. As far as we know, the click fraud detection and prevention schemes that use blockchain are hardly found. Therefore, we choose the click fraud detection schemes [8, 16, 18, 29, 31] which do not use blockchain and compare the computation costs with them in publisher's click fraud detection and prevention process (phase 5.4.1), and the comparison result is shown in Table 3.

It can be seen from Table 3 that Almahmoud et al. [8] utilized SVM, KNN, etc. to detect a fraudulent click by machines, and the time taken to build the model of the generated 500 instances is 10 ms, while the time taken to classify a single instance whether legitimate or illegitimate is 50 ms with a precision of 95.10%. The scheme in [16] uses

Table 2: Computation cost of our scheme in ad publishing and click fraud detection and prevention.

Phases	Time (ms)					Gas on contracts		
	U	MS	PUB	ADE	Total	Deploy	Call	Total
5.3.1: PPA	$T_{C\_Enc}= 146.41$	0	$T_{C\_\mathrm{KG}} + T_{C\_\mathrm{Dec}} + T_{E\_\mathrm{Dec}}= 152.28$	0	298.69	89,003	27,054	116,057
5.3.2: UCA	$3T_{E\_Enc} + 2T_{bp}= 23.09$	0	0	0	23.09	0	23,470	23,470
5.4.1: PUBD	0	0	$3T_{E\_Dec} + T_{bp}= 7.87$	0	7.87	0	25,006	25,006
5.4.2: ADED	0	0	0	$3T_{E\_Dec} + T_{bp}= 7.87$	7.87	0	0	0
5.4.3: MSD	0	$T_{E\_Enc}= 3.17$	0	0	3.17	0	0	0

Table 3: Comparison of computation cost in click fraud detection and prevention.

Schemes	Methods	Precision (%)	Preparation time	Verification time (ms)
[8]	Machine learning (SVM, KNN, etc.)	95.10	10 ms	50
[16]	Machine learning (RNN)	33.80	12 h (roughly 6–8 epochs)	—
[18]	Machine learning (bagging and boosting)	96.29	≈ 800 s	—
[29]	Statistical analysis (UI transition graphs)	≈ 93	216.7 s	400
[31]	Statistical analysis (traffic matrix analysis)	89.34	—	—
Ours	Blockchain (identity authentication)	100	0	7.87

Method refers to the algorithms or ideas used in these schemes. Precision indicates the credibility of a click traffic detection result. Preparation time denotes the time cost to train a model or analyze a pattern of a click fraud. Verification time describes the time cost to detect a click fraud using the model or pattern.

recurrent neural network to train a model with more than 1.6 million sessions so that the typical training duration is 12 hours (roughly 6–8 epochs), but the precision is 33.80%. The dataset of the scheme in [18] contains 393,708 deliveries (243,650 ok deliveries and 150,058 fraud deliveries), and the time required to train classifier with 10 features is about 800 seconds with a precision rate of 96.29%. Dong et al. [29] utilized 12,000 ad-supported apps, and an average of 216.7 seconds was spent to construct the UI transition graphs and an average of 400 ms was spent to detect the ad frauds. The dataset in [31] is from a university campus network between June 2015 and November 2017 with total of 217,334,190 unique clicks. After training, the precision is 89.34%. Table 3 shows that the preparation times of schemes in [16, 18, 29] are longer than ours because their schemes are based on machine learning and statistical analysis, and they need to spend more time training machine models and analyzing the pattern of the click traffic, while the preparation time is not included in our scheme. The verification time of a click fraud in the schemes in [16, 18, 31] is not explained, but in the scheme in [29], it is 400 ms, which is obviously higher than ours. In summary, our scheme is the best one for publishers to detect and prevent a click fraud.

7.2. Communication and Storage Cost

7.2.1. Evaluation of Our Scheme. Our scheme is embedded in the advertising system and many entities in the system need to send data to publish an ad and store data as evidences to pay for fees. To evaluate the feasibility of our scheme in practice, we simulate the scheme in terms of ad publishing and click fraud detection and prevention, and the results of communication and storage cost are shown in Table 4. Specifically, we assume that the output size of the general hash function (h) is 256 bits, the size of an elements in the elliptic curve is 256 bits, the size of an element in a bilinear group is 1,024 bits, the length of identities is 256 bits, and the timestamp size is 112 bits.

In phase *5.3.1*, an ADE first sends an ad's identity ID_{ad} to a PUB, then a U transmits a ciphertext CT containing his own MUID to a MS, the MS further forwards the ciphertext CT to the PUB, and after the PUB decrypts and obtains the MUID, the PUB sends the ID_{ad} and bidding fee to the MS; next, the MS displays the ad frame for the U. The communication cost of U, MS, PUB, and ADE is $\mathrm{CT} = 1,508$, $\mathrm{CT} + ad\,\mathrm{frame} = 2,508$, $\mathrm{ID}_{ad} + \mathrm{fee} = 33$, and $\mathrm{ID}_{ad} = 32$ bytes. Also, U stores 259-byte parameters $\{T, U_{\mathrm{Sig}}, \mathrm{MUID}, \mathrm{IL}_U\}$ to compute ad click messages M faster. To make it easier to publish the ad, the MS stores the $\{\mathrm{ID}_{ad}, \mathrm{fee}, \mathrm{PUBID}\}$ that are 65 bytes, the PUB reserves $\{\mathrm{CT}, \mathrm{MUID}, \mathrm{ID}_{ad}, \mathrm{fee}, \mathrm{MS}\}$, which are 1,637 bytes, and the ADE keeps his 32 bytes $\{\mathrm{ID}_{ad}\}$. Similarly, in phase *5.3.2*, the contents of the communication of U, MS, PUB and ADE are $M = 484$ bytes, $\mathrm{ID}_{ad} + M + \mathrm{MS} = 548$ bytes, 0, and 0, respectively. The storage cost of them is 0, $\{M\} = 484$ bytes, $\{M\} = 484$ bytes, and 0 separately. The click fraud is detected and prevented by the MS, PUB, and ADE in an independent way in phases *5.4.1*, *5.4.2*, and *5.4.3*, and the processed results are also stored. Specifically, the PUB writes a total of $n + \mathrm{ID}_{ad} + \mathrm{Info} + ts = 143$ bytes of data in the ABB and it consumes 175 bytes to store $\{n, \mathrm{ID}_{ad}, \mathrm{MS}, \mathrm{Info}, ts\}$. The ADE receives $\mathrm{SA}_{\mathrm{ADE}} + ts + U_{\mathrm{Sig}} = 255$ bytes to verify the click messages, and the ADE stores $\{n, \mathrm{ID}_{ad}, \mathrm{PUBID}, ts\} = 79$

Table 4: Communication and storage cost of our scheme in ad publishing and click fraud detection and prevention.

Phases	Communication cost (bytes)					Storage cost (bytes)				
	U	MS	PUB	ADE	Total	U	MS	PUB	ADE	Total
5.3.1: PPA	1,508	2,508	33	32	4,081	259	65	1,637	32	1,993
5.3.2: UCA	484	548	0	0	1,032	0	484	484	0	968
5.4.1: PUBD	0	0	143	0	143	0	0	175	0	175
5.4.2: ADED	0	0	0	255	255	0	0	0	79	79
5.4.3: MSD	0	142	0	0	142	0	207	0	0	207

bytes of data. Moreover, $\mathrm{AUS}_{\mathrm{PUB}} + ts = 142$ bytes of message are obtained by the MS to detect the click fraud, and it stores $\{n, \mathrm{ID}_{a\,d}, \mathrm{PUBID}, \mathrm{AUS}_{\mathrm{PUB}}, ts\} = 207$ bytes of result.

For the data presented in Table 4, the communication and storage cost in our scheme is mainly consumed in phases *5.3.1* and *5.3.2*. A total of about 8,000 bytes are used, which is negligible in today's common online advertising systems.

7.2.2. Comparison of the Communication Cost in Publishing and Clicking an Ad. We did our best to search for current blockchain-based online advertising click fraud detection and prevention schemes but only found two blockchain-based online advertising schemes [5, 6] which do not realize the detection and prevention of click fraud. Additionally, Ding et al. [6] were mainly concerned about the throughput of the blockchain transactions, and they did not give details of sending the advertising messages. Therefore, from the perspective of scheme similarity, we only make a comparison in the processes of "Publisher publishes an ad (phase *5.3.1*)" and "U clicks the ad (phase *5.3.2*)" with a vehicular local advertising system of Liu et al. [5]. Table 5 visually describes the communication cost in the processes of publishing an ad and clicking an ad.

In the scheme of Liu et al. [5], a PUB directly sends an ad to a U, and the U then clicks on the ad. ADE and MS are not included in the process of publishing and clicking on the ad, so the cost of ADE and MS is 0. To obtain an ad in the scheme of Liu et al. [5], a U needs to send his local position of $2 \times 8 = 16$ bytes, five attributes of $5 \times 10 = 50$ bytes, and a number of 1 byte to a PUB, in which the communication of a U is $16 + 50 + 1 = 67$ bytes. Also, the PUB returns two positions of $2 \times 8 = 16$ bytes and forty attributes of $40 \times 10 = 400$ bytes to the U, in which the communication of a PUB is $2 \times 16 + 400 = 432$ bytes. However, in their scheme, the click fraud still exists since they did not verify the authenticity of the click. Also, the privacy of U's locations and interests is leaked to the sniffer in the channel because the communication data are in plaintext. In our scheme, we are able to detect and prevent click fraud while protecting the identity privacy of the U. Specifically, an ADE first sends an ad $\mathrm{ID}_{a\,d}$ to a PUB, a U sends a ciphertext CT to the MS, and the MS forwards the CT to the PUB for getting an ad. Then, the PUB sends a $\mathrm{ID}_{a\,d}$ and a price fee to the MS, and the MS displays the $a\ d$ to the U. Next, the U clicks on an ad and sends a click message M to the MS, and the M is forwarded to the PUB. In these steps, the U's communication cost includes a CT and a M, which is $1,508 + 484 = 1,992$ bytes, the

Table 5: Comparison of communication cost (bytes) in publishing and clicking an ad.

	[5]	Ours
Consumer (U)	67	$1,508 + 484 = 1,992$
Media site (MS)	0	$2,508 + 548 = 3,056$
Publisher (PUB)	432	33
Advertiser (ADE)	0	32
Total	499	5,113

communication cost of the MS contains a CT, an $a\ d$ frame, a $\mathrm{ID}_{a\,d}$, an identity MS, and a M, which is $1,508 + 1,000 + 32 + 32 + 484 = 3,056$ bytes, the PUB's communication cost consists of a $\mathrm{ID}_{a\,d}$ and a fee, which is $32 + 1 = 33$ bytes, and the ADE only sends 32 bytes of $\mathrm{ID}_{a\,d}$. The communication cost of ours is higher than that of Liu et al. [5] since we add some additional authenticity information in the click message to detect and prevent click fraud. Moreover, the communication data are encrypted by the CP-ABE algorithm to protect U's privacy from the transmission medium.

When we place our scheme and Liu et al.'s scheme [5] with the same level of U's privacy protection and without regarding to click fraud detection and prevention, the ad publishing steps in our scheme can be modified as follows: a U needs to send UID to the MS, then the MS forwards UID to the PUB, next, the PUB sends $\mathrm{ID}_{a\,d}$ and fee to the MS, and finally, the MS displays $\mathrm{ID}_{a\,d}$ for the U. As a result, during these steps, the total communication content within the system is $\{\mathrm{UID}, \mathrm{UID}, \mathrm{ID}_{a\,d}, \mathrm{fee}, \mathrm{ID}_{a\,d}\} = 32 + 32 + 32 + 1 + 32 = 129$ bytes, which is significantly lower than 499 bytes of Liu et al.'s scheme. That is, we add $5,113 - 129 = 4,984$ bytes of communication overhead for U's privacy protection and click fraud detection and prevention. Also, the overhead (4,984 bytes) added to our scheme is acceptable in the background that the mainstream network bandwidth is above 3 MB/s (the average bandwidth of a 4G network is 3 MB/s).

7.2.3. Comparison of the Storage Cost in Publishing and Clicking an Ad. Besides, the comparison of storage cost when a publisher publishes an ad and a consumer clicks the ad is also shown in Figure 7.

In the processes of publishing and clicking an ad, Liu et al.'s scheme [5] does not involve the advertiser and the media site, that is, the storage cost of them is 0. Also, to request an ad faster, the U stores his ad query in advance, in which the storage cost of U is 67 bytes. After publishing an ad to the U, the PUB records the result of the ad query, and according to the experimental

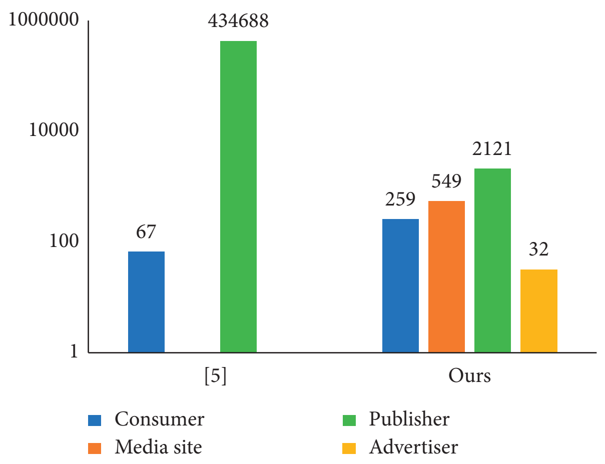


FIGURE 7: Comparison of storage cost (bytes) when a publisher publishes an ad and a consumer clicks the ad.

result, the total length is $12 \times 8 \times 128 + 3 \times 22 \times 50 \times 128 = 434,688$ bytes. For our scheme, the access tree (T), the signature from IMA (U_{Sig}), the masked identity (MUID), and the identity license (IL_U) of U are stored in U's browser plugin in advance, which is a total of $60 + 71 + 64 + 64 = 259$ bytes. The MS is responsible for forwarding messages and retaining the forwarding results $\{\text{ID}_{a\,d}, \text{fee}, \text{PUBID}, M\}$, so its storage cost is $32 + 1 + 32 + 484 = 549$ bytes. Additionally, the PUB stores $\{\text{CT}, \text{MUID}, \text{ID}_{a\,d}, \text{fee}, \text{MS}, M\}$ forwarded by the MS, which is a total of $1,508 + 64 + 32 + 1 + 32 + 484 = 2,121$ bytes. Also, the ADE only stores 32 bytes of ad information $\{\text{ID}_{a\,d}\}$. In a word, the total storage cost of our scheme is significantly lower than that of Liu et al. [5] because they need to store all the similarity results between multiple ads and one consumer.

8. Discussion

Our scheme addresses the challenging problems encountered in online advertising click fraud detection and prevention, namely, incompletely reliable detection results, tampering with the number of real clicks by the PUB itself (the PUB can count the real click number), and leakage of consumer's identity privacy. However, it still has some shortcomings that need to be solved.

First of all, although an entity identity blockchain (EIB) exists in our scheme, fraudulent adversaries have not been held accountable in our current scheme. Specifically, the EIB is designed as a consortium blockchain that records the digital identity hash of entities which can serve as evidence to hold malicious entities accountable when a fraudulent click fraud occurs. To restrain the malicious entities, an accountability system needs to be designed in the future.

Secondly, the time spent by MS to detect and prevent click fraud is slightly higher. In detail, when a MS detects click fraud, it needs to use the PK_{PUB} to encrypt the $\{AuS_{\text{PUB}} \| ts_2\}$ successively and then compare the encrypted result with the SA_{PUB} in its local database one by one. As a result, to reach an agreement with PUB on ad billing fees, the time cost for MS to detect real clicks may be high in a certain period. Therefore, our future research will focus on reducing the time cost of MS in its detection process.

Lastly, the problem of consumers' partial data loss may still exist. In our scheme, we assume that the parameters obtained by registration such as the user's identity license IL_U are secretly stored in his browser plugin, so how to prevent the leakage of parameters from the plugin also needs to be further studied.

9. Conclusion

In this paper, we proposed a blockchain-based click fraud detection and prevention scheme (BCFDPS) for online advertising to avoid clicking by machines and increases the cost of fraud ones by a human. Specifically, a click fraud by a malicious machine is significantly avoided since a consumer's immutable digital identity is embedded in the click message with the bilinear pairing algorithm and the machine does not have a digital identity to generate a valid click message. Also, the cost of click fraud by a human increases because many valid clicks by the same recruited person can only be counted once. Additionally, the introduced consortium blockchain maintains all the hash values of analysis result of consumers' click messages to achieve the transparency of the click fraud detection and prevention process for each entity in the advertising system. Further, the identity privacy of consumers is protected from media sites, advertisers, and the sniffers in the channel by ciphertext-policy attribute-based encryption. Our implementation and evaluation demonstrate the advantages of BCFDPS in computation and storage cost, and the Ethereum gas cost is limited. Additionally, to protect the user's identity privacy, the communication cost is moderately increased.

Data Availability

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Qiuyun Lyu was responsible for conceptualization, methodology, analysis, and writing. Hao Li was responsible for conceptualization, methodology, software, and writing. Renjie Zhou was responsible for revision and funding acquisition. Jilin Zhang was responsible for methodology and validation. Nailiang Zhao was responsible for validation and analysis. Yan Liu was responsible for review and editing.

Acknowledgments

This study was partially supported by the Zhejiang Provincial Key Technology Research and Development Program under grant no. 2019C03134 and National Key Technology Research and Development Program of China under grant no. 2019YFB2102100.

References

[1] M. Gabryel, "Data analysis algorithm for click fraud recognition," in *Proceedings of the International Conference on Information and Software Technologies*, pp. 437–446, Springer, Vilnius, Lithuania, Octomber 2018.

[2] J. Estrada-Jiménez, J. Parra-Arnau, A. Rodríguez-Hoyos, and J. Forné, "On the regulation of personal data distribution in online advertising platforms," *Engineering Applications of Artificial Intelligence*, vol. 82, pp. 13–29, 2019.

[3] Z. Gharibshah and X. Zhu, "User response prediction in online advertising," *ACM Computing Surveys*, vol. 54, no. 3, pp. 1–43, 2021.

[4] F. Kanei, D. Chiba, K. Hato, and M. Akiyama, "Precise and robust detection of advertising fraud," in *Proceedings of the 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, pp. 776–785, IEEE, Milwaukee, WI, USA, July 2019.

[5] D. Liu, J. Ni, X. Lin, and X. Shen, "Transparent and accountable vehicular local advertising with practical blockchain designs," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15694–15705, 2020.

[6] Y. Ding, D. Luo, H. Xiang, W. Liu, and Y. Wang, "Design and implementation of blockchain-based digital advertising media promotion system," *Peer-to-Peer Networking and Applications*, vol. 14, no. 2, pp. 482–496, 2021.

[7] C. Shi, R. Song, X. Qi, Y. Song, B. Xiao, and S. L. ClickGuard, "Exposing hidden click fraud via mobile sensor side-channel analysis," in *Proceedings of the ICC 2020-2020 IEEE International Conference on Communications (ICC)*, July 2020.

[8] S. Almahmoud, B. Hammo, and B. Al-Shboul, "Exploring non-human traffic in online digital advertisements: analysis and prediction," *Computational Collective Intelligence*, Springer, in *Proceedings of the International Conference on Computational Collective Intelligence*, pp. 663–675, March 2019.

[9] G. Thejas, K. Boroojeni, K. Chandna, I. Bhatia, S. Iyengar, and N. Sunitha, "Deep learning-based model to fight against ad click fraud," in *Proceedings of the 2019 ACM Southeast Conference*, pp. 176–181, Switzerland, May 2019.

[10] T. Tian, J. Zhu, F. Xia, X. Zhuang, and T. Zhang, "Crowd fraud detection in internet advertising," in *Proceedings of the 24th International Conference on World Wide Web*, Switzerland, May 2015.

[11] B. Stone-Gross, R. Stevens, A. Zarras, R. Kemmerer, C. Kruegel, and G. Vigna, "Understanding fraudulent activities in online ad exchanges," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, pp. 279–294, Berlin, Germany, November 2011.

[12] J. Hu, T. Li, Y. Zhuang, S. Huang, and S. Dong, "A C Approach for Fraud Detection in mobile Advertising," *Security and Communication Networks*, vol. 2020, Article ID 1656460, 2020.

[13] F. Kanei, D. Chiba, K. Hato, K. Yoshioka, T. Matsumoto, and M. Akiyama, "Detecting and understanding online advertising fraud in the wild," *IEICE - Transactions on Info and Systems*, vol. 103, no. 7, pp. 1512–1523, 2020.

[14] IAB US benchmarking study, "What Is an Untrustworthy Supply Chain Costing: The U.S. Digital Advertising Industry?," 2015, https://www.iab.com/wp-content/uploads/2015/11/IAB_EY_Report.pdf.

[15] R. Oentaryo, E. Lim, M. Finegold et al., "Detecting click fraud in online advertising: a data mining approach," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 99–140, 2014.

[16] S. Wang, C. Liu, X. Gao, H. Qu, and W. Xu, "Session-based fraud detection in online e-commerce transactions using recurrent neural networks," *Machine Learning and Knowledge Discovery in Databases*, Springer, in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 241–252, Septmber 2017.

[17] R. Mouawi, M. Awad, A. Chehab, H. Imad, H. El, and A. Kayssi, "Towards a machine learning approach for detecting click fraud in mobile advertizing," in *Proceedings of the International Conference on Innovations in Information Technology (IIT)*, pp. 88–92, IEEE, Al Ain, United Arab Emirates, November 2018.

[18] C. M. R. Haider, A. Iqbal, A. H. Rahman, and M. S. Rahman, "An ensemble learning based approach for impression fraud detection in mobile advertising," *Journal of Network and Computer Applications*, vol. 112, pp. 126–141, 2018.

[19] D. Sisodia and D. Sisodia, *Gradient Boosting Learning for Fraudulent Publisher Detection in Online Advertising*, Data Technologies and Applications, United Kingdom, 2020.

[20] J.-A. Choi and K. Lim, "Identifying machine learning techniques for classification of target advertising," *ICT Express*, vol. 6, no. 3, pp. 175–180, 2020.

[21] N. P. Gohil and A. D. Meniya, "Click ad fraud detection using XGBoost gradient boosting algorithm," in *Proceedings of the International Conference on Computing Science, Communication and Security*, pp. 67–81, Springer, Gujarat, India, February 2021.

[22] G. Thejas, S. Dheeshjith, S. Iyengar, N. Sunitha, and P. Badrinath, "A hybrid and effective learning approach for click fraud detection," *Machine Learning with Applications*, vol. 3, Article ID 100016, 2021.

[23] E. Mikhailov and R. Trusov, "How Adversarial Attacks Work," 2017, https://blog.ycombinator.com/how-adversarial-attacks-work/.

[24] O. Stitelman, C. Perlich, B. Dalessandro, R. Hook, T. Raeder, and F. Provost, "Using co-visitation networks for detecting large scale online display advertising exchange fraud," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1240–1248, Illinois, Chicago, USA, Augest 2013.

[25] J. Xu and C. Li, "Detecting crowdsourcing click fraud in search advertising based on clustering analysis," in *Proceedings of the 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing*, pp. 894–900, IEEE, Beijing, China, August 2015.

[26] S.-C. Lee, C. Faloutsos, D.-K. Chae, and S.-W. Kim, "Fraud detection in comparison-shopping services: patterns and anomalies in user click behaviors," *IEICE - Transactions on Info and Systems*, vol. 100, no. 10, pp. 2659–2663, 2017.

[27] J. Hu, J. Liang, and S. D. iBGP, "A bipartite graph propagation approach for mobile advertising fraud detection," *Mobile Information Systems*, vol. 2017, Article ID 7602384, 12 pages, 2017.

[28] M. Meghanath, D. Pai, and L. A. ConOut, "Contextual outlier detection with multiple contexts: application to ad fraud," in *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 139–156, Springer, France, September 2018.

[29] F. Dong, H. Wang, L. Li et al., "Automated ad fraud detection for android apps," in *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 257–268, November 2018, https://doi.org/10.1145/3236024.3236045.

[30] M. Gabryel and K. Przybyszewski, "The dynamically modified BoW algorithm used in assessing clicks in online ads," *Artificial Intelligence and Soft Computing*, Springer, in *Proceedings of the International Conference on Artificial Intelligence and Soft Computing*, pp. 350–360, April 2019.

[31] S. Nagaraja and R. S. Clicktok, "Click fraud detection using traffic analysis," in *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, Florida, Miami, May 2019.

[32] C. Cao, Y. Gao, Y. Luo et al., "Efficient and deployable click fraud detection for mobile applications," *IEEE Transactions on Mobile Computing*, vol. 20, no. 4, pp. 1285–1297, 2020.

[33] N. Kshetri and J. Voas, "Online advertising fraud," *Computer*, vol. 52, no. 1, pp. 58–61, 2019.

[34] L. Lv Z. Wang and M. Yang, "Research on trusted identity architecture in cyberspace based on eid," *Netinfo Security*, vol. 9, pp. 97–100, 2015.

[35] P. Tammpuu and A. Masso, "Transnational digital identity as an instrument for global digital citizenship: the case of Estonia's E-residency," *Information Systems Frontiers*, vol. 21, no. 3, pp. 621–634, 2019.

[36] A. Abraham, K. Theuermann, and E. Kirchengast, "Qualified eID derivation into a distributed ledger based IdM system," in *Proceedings of the 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*, pp. 1406–1412, IEEE, New York, NY, USA, August 2018.

[37] G. Shahaf, E. Shapiro, and N. Talmon, "Genuine personal identifiers and mutual sureties for sybil-resilient community growth," in *Proceedings of the International Conference on Social Informatics*, pp. 320–332, Springer, Doha, Qatar, November 2020.

[38] G. Shahaf, E. Shapiro, and N. Talmon, "Genuine Personal Identifiers and Mutual Sureties for Sybil-Resilient Community Formation," 2019, https://arxiv.org/abs/1904.09630.

[39] T. Zhu, Y. Meng, H. Hu, X. Zhang, M. Xue, and H. Zhu, "Dissecting Click Fraud Autonomy in the Wild," 2021, https://arxiv.org/abs/2105.11103.

[40] Y. Wang, A. Zhang, P. Zhang, and H. Wang, "Cloud-assisted EHR sharing with security and privacy preservation via consortium blockchain," *IEEE Access*, vol. 7, Article ID 136719, 2019.

[41] M. Shen, J. Zhang, L. Zhu, K. Xu, and X. Tang, "Secure SVM training over vertically-partitioned datasets using consortium blockchain for vehicular social networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5773–5783, 2019.

[42] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[43] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," *Advances in Cryptology - CRYPTO 2001*, Springer, in *Proceedings of the Annual International Cryptology Conference*, pp. 213–229, August 2001.

[44] D. Galindo, "Boneh-Franklin identity based encryption revisited," in *Proceedings of the International Colloquium on Automata, Languages, and Programming*, pp. 791–802, Springer, Portugal, August 2005.

[45] V. Goyal, A. Jain, O. Pandey, and A. Sahai, "Bounded ciphertext policy attribute based encryption," in *Proceedings of the International Colloquium on Automata, Languages, and Programming*, pp. 579–591, Springer, Iceland, July 2008.

[46] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Proceedings of the International Workshop on Public Key Cryptography*, pp. 53–70, Springer, Italy, March 2011.

[47] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP'07)*, pp. 321–334, IEEE, Berkeley, CA, USA, May 2007.

[48] R. Canetti and H. Krawczyk, "Analysis of key-exchange protocols and their use for building secure channels," *Lecture Notes in Computer Science*, Springer, in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 453–474, May 2001.

[49] J. Leng, M. Zhou, J. Zhao, Y. Huang, and Y. Bian, "Blockchain Security: A Survey of Techniques and Research Directions," *IEEE Transactions on Services Computing*, Article ID 9271868, 2020.

[50] J. Leng, G. Ruan, P. Jiang et al., "Blockchain-empowered sustainable manufacturing and product lifecycle management in industry 4.0: a survey," *Renewable and Sustainable Energy Reviews*, vol. 132, Article ID 110112, 2020.

[51] J. Leng, S. Ye, M. Zhou et al., "Blockchain-secured smart manufacturing in industry 4.0: a survey," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 237–252, 2020.

[52] J. Leng, P. Jiang, K. Xu et al., "Makerchain: a blockchain with chemical signature for self-organizing process in social manufacturing," *Journal of Cleaner Production*, vol. 234, pp. 767–778, 2019.

[53] Q. Liu, J. Leng, D. Yan et al., "Digital twin-based designing of the configuration, motion, control, and optimization model of a flow-type smart manufacturing system," *Journal of Manufacturing Systems*, vol. 58, pp. 52–64, 2021.

[54] Q. Liu, H. Zhang, J. Leng, and X. Chen, "Digital twin-driven rapid individualised designing of automated flow-shop manufacturing system," *International Journal of Production Research*, vol. 57, no. 12, pp. 3903–3919, 2019.

[55] J. Leng, H. Zhang, D. Yan, Q. Liu, X. Chen, and D. Zhang, "Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 3, pp. 1155–1166, 2019.

[56] V. Boyko, P. MacKenzie, and S. Patel, "Provably secure password-authenticated key exchange using Diffie-Hellman," *Advances in Cryptology - EUROCRYPT 2000*, Springer, in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 156–171, May 2000.

[57] Y. Ma, Y. Liu, B. Xu, and J. Zhi, "Impacts of Waiting on mobile Users – Case Study of Digital Novels," *Data Analysis And Knowledge Discovery*, vol. 2, no. 8, 2018.