



ISSN: 2454-9940



**INTERNATIONAL JOURNAL OF APPLIED
SCIENCE ENGINEERING AND MANAGEMENT**

E-Mail :
editor.ijasem@gmail.com
editor@ijasem.in

www.ijasem.in

AdSherlock: An Advanced and Efficient and Deployable Click Based Fraud Detection for Mobile Applications

D Srikar professor1, Adabala Mounika Rajeswari2

Abstract:

Mobile advertising plays a vital role in the mobile app ecosystem. A major threat to the sustainability of this ecosystem is click fraud, i.e., ad clicks performed by malicious code or automatic bot problems. Existing click fraud detection approaches focus on analyzing the ad requests at the server side. However, such approaches may suffer from high false negatives since the detection can be easily circumvented, e.g., when the clicks are behind proxies or globally distributed. In this paper, we present AdSherlock, an efficient and deployable click fraud detection approach at the client side (inside the application) for mobile apps. AdSherlock splits the computation-intensive operations of click request identification into an offline procedure and an online procedure. In the offline procedure, AdSherlock generates both exact patterns and probabilistic patterns based on URL (Uniform Resource Locator) tokenization. These patterns are used in the online procedure for click request identification and further used for click fraud detection together with an ad request tree model. We implement a prototype of AdSherlock and evaluate its performance using real apps. The online detector is injected into the app executable archive through binary instrumentation. Results show that AdSherlock achieves higher click fraud detection accuracy compared with state of the art, with negligible runtime overhead.

Introduction:

Many Android applications are distributed for free but are supported by advertisements. Ad libraries embedded in the app fetch content from the ad provider

and display it on the app's user interface. The ad provider pays the developer for the ads displayed to the user and ads clicked by the user. A major threat to this ecosystem is ad fraud, where a miscreant's

code fetches ads without displaying them to the user or "clicks" on ads automatically. Ad

fraud has been extensively studied in the context of web advertising but has gone largely unstudied in the context of mobile advertising. We take the first step to study mobile ad fraud perpetrated by Android apps. We identify two fraudulent ad behaviors in apps: 1) requesting ads while the app is in the background, and 2) clicking on ads without user interaction. Based on these observations, we

#1 Associate Professor in Computer Science Department Bhimavaram Institute of Engineering and Technology Pennada, Bhimavaram, West Godavari district

#2 M.Tech Scholar, in Computer Science Department Bhimavaram Institute of Engineering and Technology Pennada, Bhimavaram, West Godavari district

developed an analysis tool, MAdFraud, which automatically runs many apps simultaneously in emulators to trigger and expose ad fraud. Since the formats of ad impressions and clicks vary widely between different ad providers, we develop a novel approach for automatically identifying ad impressions and clicks in three steps: building HTTP request trees, identifying ad request pages using machine learning, and detecting clicks in HTTP request trees using heuristics. We apply our methodology and tool to two datasets: 1) 130,339 apps crawled from 19 Android markets including Play and many third-party markets, and 2) 35,087 apps that likely contain malware provided by a security company. From analyzing these datasets, we find that about 30% of apps with ads make ad requests while in running in the background. In addition, we find 27 apps which generate clicks without user interaction. We find that the click fraud apps attempt to remain stealthy when fabricating ad traffic by only periodically sending clicks and changing which ad provider is being targeted between installations.

Detecting click fraud in online advertising: A data mining approach

Click fraud-the deliberate clicking on advertisements with no real interest on the product or service offered-is one of the most daunting problems in online advertising. Building an effective fraud detection method is thus pivotal for online advertising businesses. We organized a Fraud Detection in Mobile Advertising (FDMA) 2012 Competition, opening the opportunity for participants to work on real-world fraud data from BuzzCity Pte. Ltd., a global mobile advertising company based in Singapore. In particular, the task is to identify fraudulent publishers who generate illegitimate clicks, and distinguish them from normal publishers. The competition was held from September 1 to September 30, 2012, attracting 127 teams from more than 15 countries. The mobile advertising data are unique and complex, involving heterogeneous information, noisy patterns with missing values, and highly

imbalanced class distribution. The competition results provide a comprehensive study on the usability of data mining-based fraud detection approaches in practical setting. Our principal findings are that features derived from fine-grained time-series analysis are crucial for accurate fraud detection, and that ensemble methods offer promising solutions to highly-imbalanced nonlinear classification tasks with mixed variable types and noisy/missing patterns. The competition data remain available for further studies

Click fraud is jeopardizing the industry of Internet advertising. Internet advertising is crucial for the thriving of the entire Internet, since it allows producers to advertise their products, and hence contributes to the wellbeing of e-commerce. Moreover, advertising supports the intellectual value of the Internet by covering the running expenses of the content publishers' sites. Some publishers are dishonest, and use automation to generate traffic to defraud the advertisers. Similarly, some advertisers automate clicks on the advertisements of their competitors to deplete their competitors' advertising budgets. In this paper, we describe the advertising network model, and discuss the issue of fraud that is an integral problem in such setting. We propose using online algorithms on aggregate data to accurately and proactively detect automated traffic, preserve surfers' privacy, while not altering the industry model. We provide a complete classification of the hit inflation techniques; and devise stream analysis techniques that detect a variety of fraud attacks. We abstract detecting the fraud attacks of some classes as theoretical stream analysis problems that we bring to the data management research community as open problems. A framework is outlined for deploying the proposed detection algorithms on a generic architecture. We conclude by some successful preliminary findings of our attempt to detect fraud on a real network.

Detecting click fraud in pay-per-click streams of online advertising networks

With the rapid growth of the Internet, online advertisement plays a more and more important role in the advertising market. One of the current and widely used revenue models for online advertising involves charging for each click based on the popularity of keywords and the number of competing advertisers. This pay-per-click model leaves room for individuals or rival companies to generate false clicks (i.e., click fraud), which pose serious problems to the development of healthy online advertising market. To detect click fraud, an important issue is to detect duplicate clicks over decaying window

models, such as jumping windows and sliding windows. Decaying window models can be very helpful in defining and determining click fraud. However, although there are available algorithms to detect duplicates, there is still a lack of practical and effective solutions to detect click fraud in pay-per-click streams over decaying window models. In this paper, we address the problem of detecting duplicate clicks in pay-per-click streams over jumping windows and sliding windows, and are the first that propose two innovative algorithms that make only one pass over click streams and require significantly less memory space and operations. GBF algorithm is built on group Bloom filters which can process click streams over jumping windows with small number of sub-windows, while TBF algorithm is based on a new data structure called timing Bloom filter that detects click fraud over sliding windows and jumping windows with large number of sub-windows. Both GBF algorithm and TBF algorithm have zero false negative. Furthermore, both theoretical analysis and experimental results show that our algorithms can achieve low false positive rate when detecting duplicate clicks in pay-per-click streams over jumping windows and sliding windows.

We present AdSherlock, a productive and deployable snap misrepresentation identification approach at the customer side (inside the application) for versatile applications. AdSherlock parts the calculation

serious tasks of snap demand distinguishing proof into a disconnected system and an online method. In the disconnected strategy, AdSherlock produces both careful examples and probabilistic examples dependent on URL (Uniform Resource Locator) tokenization. These examples are utilized in the online system for click demand recognizable proof and further utilized for click misrepresentation identification along with an advertisement demand tree model.

Advantages:

AdSherlock generates both exact patterns and probabilistic patterns based on URL (Uniform Resource Locator) tokenization.

Finally, AdSherlock instruments the online fraud detector into the app binaries which are then released by the app store. Mobile advertising plays a vital role in the mobile app ecosystem. A recent report shows that mobile advertising expenditure worldwide is projected to reach \$247.4 billion in 2020 [1]. To embed ads in an app, the app developer typically includes ad libraries provided by a third-party mobile ad

provider such as AdMob [2]. When a mobile user is using the app, the embedded ad library fetches ad content from the network and displays ads to the user. The most common charging model is PPC (Pay-Per-Click) [3], where the developer and the ad provider get paid from the advertiser when a user clicks on the ad. A major threat to the sustainability of this ecosystem is click fraud [4], i.e., clicks (i.e., touch events on mobile devices) on ads which are usually performed by malicious code programmatically or by automatic bot problems. There are many different click fraud tactics which can typically be characterized into two types: in-app frauds insert malicious code into the app to generate forged ad clicks; bots-driven frauds employ bot programs (e.g., a fraudulent application) to click on advertisements automatically. To quantify the in app ad fraud in real apps, a recent work MAdFraud [5] conducts a large scale measurement about ad fraud in realworld apps. In a dataset including about 130K Android apps, MAdFraud reports that about 30% of apps make ad requests while running in the background. Focusing on bots-driven

click fraud, another recent work uses an automated click generation tool ClickDroid [4] to empirically evaluate eight popular advertising networks by performing real click fraud attacks on them. Results [4] show that six advertising

networks out of eight are vulnerable to these attacks. Aiming at detecting click frauds in mobile apps, a straightforward approach is a threshold-based detection at the server side. If an ad server is receiving a high number of clicks with the same device identifier (e.g., IP address) in a short period, these clicks can be considered as fraud. This straightforward approach, however, may suffer from high false negatives since the detection can be easily circumvented when the clicks are behind proxies or globally distributed. In the literature, there are also more sophisticated approaches [6], [7] focusing on detecting click frauds at the server-side. The precisions of these server-side approaches, however, are not sufficient enough for the click fraud problem. For example, in a recent mobile ad fraud competition [6], the best three approaches achieve only a precision of 46.15% to 51.55% using various machine learning techniques. Given the insufficient precision of server-side approaches, a natural question comes up: how about client-side approaches? In fact, compared with the server-side approaches, it is easier to tell whether there is an actual user input at the client side. However, the attacker of the click fraud could be the app developers themselves, since the developers will get paid for those fraudulent ad clicks. Due to this conflict-of-interest problem, we cannot assume the existence of coordination from developers in designing a client-side approach for click fraud detection, e.g., a click fraud detection SDK.

Therefore, in this paper, we focus on designing a client-side approach to detect click frauds in mobile apps, without coordination from developers. There are two major challenges in designing such a system. First, for a mobile client, its resources are constrained in terms of computation, memory, and energy. Therefore, the proposed approach must perform the complete fraud

detection process efficiently, without causing significant overhead. This means that we need to design new algorithms to detect click frauds since existing machine-learning algorithms used by server-side approaches are not suitable for the client side. Second, the click fraud detection should be able to execute under practical user scenarios, instead of a controlled environment dedicated to fraud detection. In MAdFraud [5], a controlled environment (i.e., only one app is running and the HTTP requests are collected for offline analysis) is used to measure the ad fraud behavior of a vast number of apps. However, in our case, the click fraud detection should happen inside the mobile

client without outside support, i.e., be deployable in real-world scenarios. In this paper, we propose AdSherlock, an efficient and deployable click fraud detection approach for mobile apps at the client side. Note that as a client-side approach, AdSherlock is orthogonal to existing server-side approaches. AdSherlock is designed to be used by app stores to ensure a healthy mobile app ecosystem. AdSherlock's high accuracy helps market operators to fight both in-app frauds and bots-driven frauds. Note that, AdSherlock can also be used by any third parties to detect in-app frauds. For example, ad providers can employ AdSherlock to check whether apps embedding their libraries have in-app fraudulent behaviors. To achieve these goals, AdSherlock relies on an accurate offline pattern extractor and a lightweight online fraud detector.

AdSherlock works in two stages. At the first stage, the offline pattern extractor automatically executes each app and generates a set of traffic patterns for efficient ad request identification, i.e., extracts common token patterns across different ad requests. Specifically, after tokenization of the network requests, AdSherlock generates both exact patterns and probabilistic patterns for robust matching. Using the offline pattern extractor, AdSherlock can perform the computation and I/O intensive pattern generation operations in an offline manner, without degrading the online fraud detection operations. At the second stage, the online

fraud detector as well as the generated patterns are instrumented into the app and run with the app in actual user scenarios. Inside the app, AdSherlock uses an ad request tree model to identify click requests accurately and efficiently. Since the online fraud detector runs inside the app, it can obtain the fine-grained user input events which are further employed for click fraud detection. We implement AdSherlock and evaluate its performance using real apps. Results show that AdSherlock achieves higher click fraud detection accuracy compared with state of the art, with negligible runtime overhead. The contributions of this paper are summarized as follows:

We present the design and implementation of AdSherlock, the first system which can achieve efficient and deployable click fraud detection at the client side.




We propose a pattern generation mechanism that generates patterns for ad requests and non-ad requests with high accuracy.

We also propose an efficient method for online click fraud detection based on an ad request tree model.

We implement AdSherlock and compare its performance with the state-of-art approach. Results show that AdSherlock achieves higher detection accuracy with lower overhead. The rest of this paper is organized as follows. Click Fraud Detection in Web Advertising in the context of Web advertising, researches on click fraud detection mainly focus on bots-driven click frauds. These approaches are usually performed at the server-side,

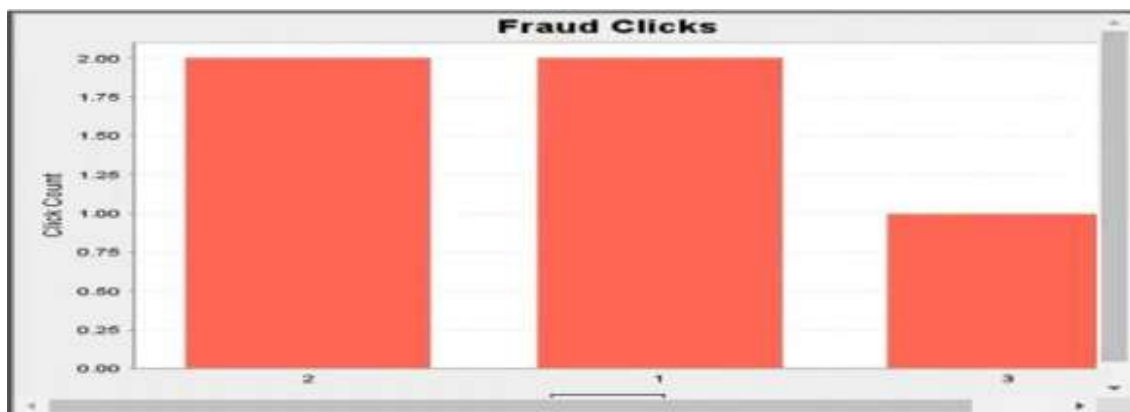
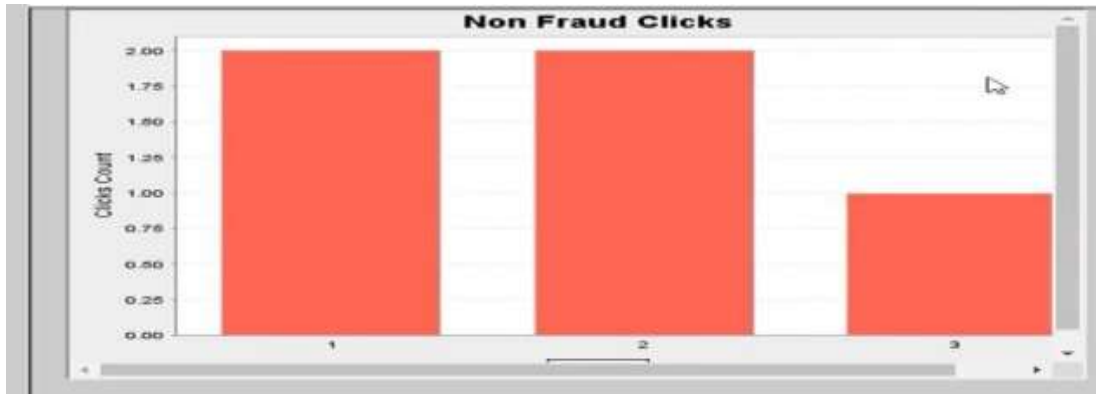
Results:

ALL POSTED ADS

| Post Name | Description | Date | Post Image |
|----------------|---|------------|---|
| samsung mobile | new lunching samsung mobile | 2021-02-23 |  |
| laptop | Acer is the best laptop we are going to offer you 2000 discount | 2021-02-23 |  |
| pendrive | offering 50 rps on pendrives | 2021-02-24 |  |

analyzing network traffic and characterizing the features of click fraud behaviors. [8] and [9] aggregate ad traffics across client IP address and cookie IDs to observe the client who has deviated ad traffic behaviors. SBotMiner [10] detects search engine bots by looking for anomalies in query distribution. However, such server-side approaches are not robust against sophisticated bots who can vary their IP addresses and other traffic features.

Different from them, AdSherlock is a client-side method exploiting the property of click events on the end device which is hard to bypass. Moreover, these server-side methods need to collect sufficient ad traffics for analysis while AdSherlock does not need. From the client-side, AdSherlock can detect and prevent click fraud promptly. Others works such as [11] and [12] focus on detecting duplicate clicks, where a publisher inflates its clicks by clicking on the same ad many times. These server-side methods can be viewed as a supplementary on AdSherlock in that they can detect click fraud performed by real humans. FcFraud [13] is the latest work on click fraud detection in web advertising from the user side and is very related to our work. It identifies ad clicks and examines whether real mouse events accompany them. However, it needs to collect a bundle of HTTP requests for the ad request classifier which will cause unbearable overhead for Andriod apps. AdSherlock, on the other hand, focuses on click fraud detection in mobile applications.



Conclusion

AdSherlock is an efficient and deployable click fraud detection approach for mobile apps at the client side. As a client-side approach, AdSherlock is orthogonal to existing server-side approaches. It splits the computation intensive operations of click request identification into an offline process and an online process. In the offline process, AdSherlock generates both exact patterns and probabilistic patterns based on url tokenization. These patterns are used in the online process for click request identification, and further used for click fraud detection together with an ad request tree model. Evaluation shows that AdSherlock achieves high click fraud

detection accuracy with a negligible runtime overhead. In the future, we plan to combine static analysis with the traffic analysis to improve the accuracy of ad request identification and explore attacks designed to evade AdSherlock.

References:

1. "Mobile advertising spending worldwide." [Online]. Available: <https://www.statista.com/statistics/280640/mobile-advertisingspending-worldwide/>
2. "Google admob." [Online]. Available: <https://apps.admob.com/>
3. M. Mahdian and K. Tomak, "Pay-per-action model for online advertising," in Proc. of ACM ADKDD, 2007.
4. G. Cho, J. Cho, Y. Song, and H. Kim, "An empirical study of click fraud in mobile advertising networks," in Proc. of ACM ARES, 2015.
5. J. Crussell, R. Stevens, and H. Chen, "Madfraud: Investigating ad fraud in android applications," in Proc. of ACM MobySys, 2014.
6. R. Oentaryo, E.-P. Lim, M. Finegold, D. Lo, F. Zhu, C. Phua, E.-Y. Cheu, G.-E. Yap, K. Sim, M. N. Nguyen, K. Perera, B. Neupane, M. Faisal, Z. Aung, W. L. Woon, W. Chen, D. Patel, and D. Berrar, "Detecting click fraud in online advertising:"

A data mining approach,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 99–140, 2014.

7.B. Kitts, Y. J. Zhang, G. Wu, W. Brandi, J. Beasley, K. Morrill, J. Etedgui, S. Siddhartha, H. Yuan, F. Gao, P. Azo, and R. Mahato, *Click Fraud Detection: Adversarial Pattern Recognition over 5 Years at Microsoft*. Cham: Springer International Publishing, 2015, pp. 181–201.

8. A. Metwally, D. Agrawal, and A. El Abbadi, “Detectives: detecting coalition hit inflation attacks in advertising networks streams,” in *Proc. of ACM WWW*, 2007.

9. A. Metwally, D. Agrawal, A. El Abbad, and Q. Zheng, “On hit inflation techniques and detection in streams of web advertising networks,” in *Proc. of IEEE ICDCS*, 2007.

10. F. Yu, Y. Xie, and Q. Ke, “Sbotminer: large scale search bot detection,” in *Proc. of ACM WSDM*, 2010.

[11] L. Zhang and Y. Guan, “Detecting click fraud in pay-per-click streams of online advertising networks,” in *Proc. of IEEE ICDCS*, 2008.

[12] A. Metwally, D. Agrawal, and A. El Abbadi, “Duplicate detection in click streams,” in *Proc. of ACM WWW*, 2005.

[13] M. S. Iqbal, M. Zulkernine, F. Jaafar, and Y. Gu, “Fc fraud: Fighting click-fraud from the user side,” in *Proc. of IEEE HASE*, 2016.

[14] B. Liu, S. Nath, R. Govindan, and J. Liu, “Decaf: detecting and characterizing ad fraud in mobile apps,” in *Proc. of USENIX NSDI*, 2014.