RESOURCEFUL AND EMPLOYABLE CLICK FRAUD IDENTIFICATION FOR HANDHELD APPLICATIONS

1 DR. S. VIJAYARANGAM, 2 B. VARUN SAI, 3 B. HEMANTH KUMAR

4 B. BHAVANI SHANKAR, 5 B. LOKESH

1Associate Professor, Department of CSE, Sri Indu College of Engineering and Technology-Hyderabad

2345Under Graduate, Department of CSE, Sri Indu College of Engineering and Technology-Hyderabad

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 project, 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.

Keywords: **Data Security, Fraud Detection**

INTRODUCTION

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. To embed ads in an app, the app developer typically includes ad libraries provided by a third-party mobile ad provider such as AdMob. 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), where the developer and the ad provider get paid from the advertiser when a user clicks on the ad.

A major threat to sustainability of this ecosystem is click fraud, i.e., clicks (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 conducts a large-scale measurement about ad fraud in real world 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 to empirically evaluate eight popular advertising networks by performing real click fraud attacks on them. Results 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, 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, 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 project, we focus on designing a client-side approach to detect click frauds in mobile apps, without coordination from developers.

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 project 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 Ad- Sherlock achieves higher detection accuracy with lower overhead.

LITERATURE SURVEY

This research addresses the critical issue of click fraud in the mobile app ecosystem, which threatens the sustainability of mobile advertising. Existing click fraud detection approaches focus on server-side analysis, leading to potential high false negatives as detection can be evaded through proxies or global distribution of clicks. To combat this, we propose AdSherlock, an efficient and deployable client-side click fraud detection approach for mobile apps. AdSherlock optimizes click request identification by splitting computation- intensive operations into offline and online procedures. In the offline procedure, we generate exact and probabilistic patterns based on URL tokenization. These patterns are then used in the online procedure alongside an ad request tree model for click request identification and subsequent click fraud detection. 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, resulting in higher click fraud detection accuracy compared to state-of-the-art solutions, with negligible runtime overhead.

Click fraud poses a severe threat to the mobile app ecosystem, necessitating robust detection mechanisms. Existing solutions primarily rely on server-side analysis, leaving room for false negatives due to click evasion through proxies and global distribution. In this paper, we introduce AdSherlock, an efficient and deployable client-side click fraud detection system for mobile applications. AdSherlock optimizes click request identification through offline and online procedures, effectively splitting computation-intensive tasks. In the offline phase, we generate exact and probabilistic patterns based on URL tokenization, which are then employed in the online phase along with an ad request tree model for click request identification and subsequent click fraud detection. 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, yielding higher click fraud detection accuracy compared to state-of-the-art solutions, with minimal impact on runtime performance.

The mobile app ecosystem heavily relies on mobile advertising, making it a prime target for click fraud, posing a significant threat to its sustainability. Existing click fraud detection methods predominantly focus on server-side analysis, leading to potential false negatives due to click evasion using proxies and global distribution. To address this, we introduce AdSherlock, a novel and efficient client-side click fraud detection approach for mobile apps. AdSherlock optimizes click request identification by dividing computation- intensive operations into offline and online procedures. In the offline phase, we generate exact and probabilistic patterns based on URL tokenization, which are utilized in the online phase alongside an ad request tree model for click request identification and subsequent click fraud detection. 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, resulting in superior click fraud detection accuracy compared to existing methods, with minimal runtime overhead.

SYSTEM ANALYSIS

EXISTING SYSTEM

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, 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 project, we propose AdSherlock, an efficient and deployable click fraud detection approach for mobile apps at the client side, 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.

Disadvantages:

- This approach not Suitable for client side.
- The click fraud detection should be able to execute under practical user scenarios, instead of a controlled environment dedicated to fraud detection.
- The precision of server-side detection methods is often not high enough in detecting click fraud.
- Server-side systems can have difficulty detecting fraud accurately when malicious users use techniques to hide their activities.

PROPOSED SYSTEM:

We propose two pattern classes: exact patterns and probabilistic patterns. Both of them are built from invariant substrings in the HTTP header. We refer to these substrings as tokens. Exact patterns consist of a set of sequential tokens and match an HTTP request if and only ifthe request contains all tokens in the set with the same ordering. Probabilistic patterns consist of a set of tokens, each of which is associated with an ad score, and a non-ad score. We describe the details of pattern generation in the following sections.

ADVANTAGES:

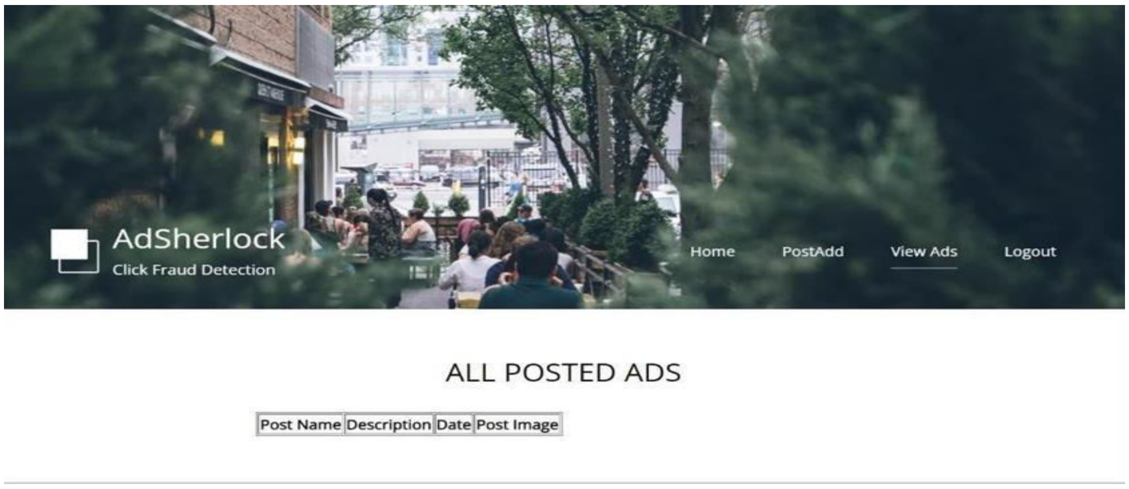
- Two pattern classes: exact patterns and probabilistic patterns. Both of them are builtfrom invariant substring in the HTTP header, helps in robust identification of ad requests.
- This approach suitable for client and server-side.
- AdSherlock achieves higher accuracy in detecting click fraud compared to state- of-the-art methods.
- Running on the client-side allows AdSherlock to access fine-grained user input event data, which aids in more accurate fraud detection.

IMPLEMENTATION

MODULES

- Client Module.
- Time Stamp Upload Key Module
- Time Stamp File Key Module
- Third Party Auditor (TPA) Module
- Cloud Module

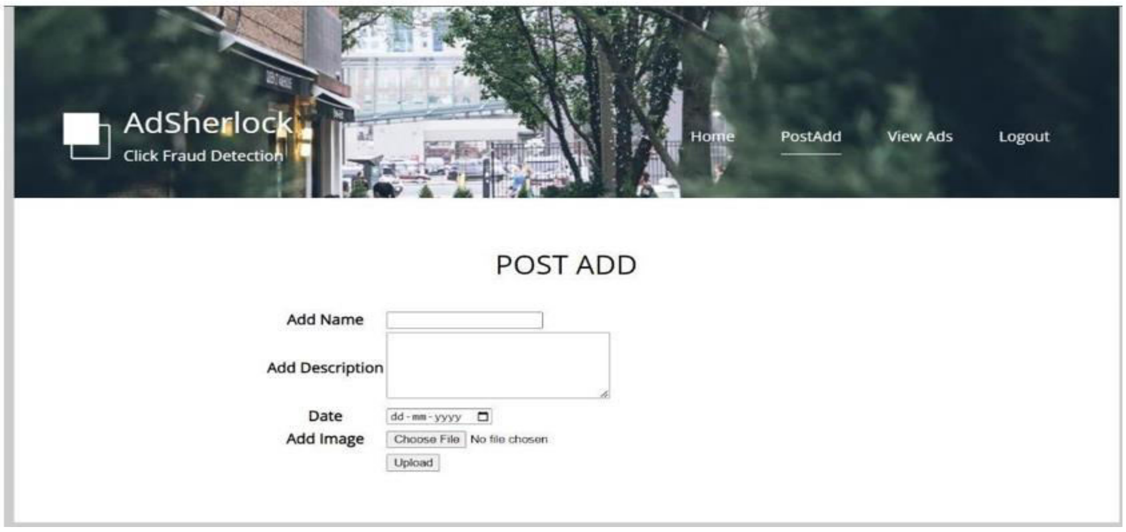
RESULTS



ALL POSTED ADS

Post Name	Description	Date	Post Image

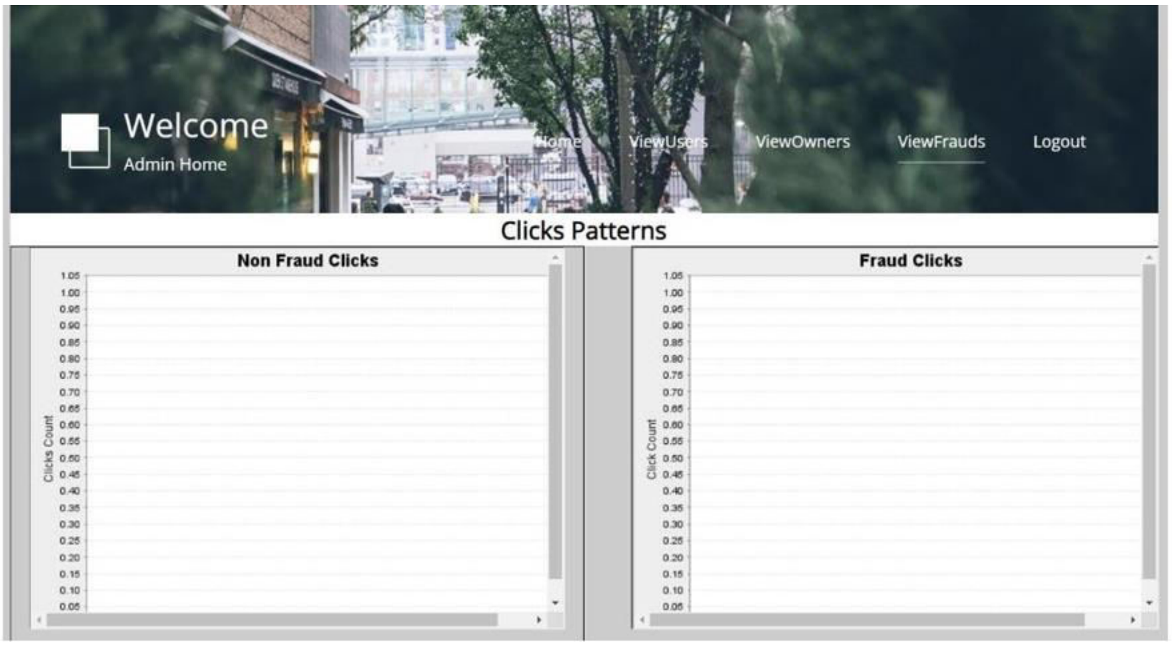
HOME SCREEN



POST ADD

Add Name	
Add Description	
Date	dd-mm-yyyy
Add Image	Choose File No file chosen
	Upload

POST ADS PAGE



VIEW FRAUDS

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.

FUTURE SCOPE:

Enhanced Machine Learning Algorithms: Advanced machine learning algorithms, particularly deep learning and AI, will play a crucial role in identifying patterns and anomalies that signify fraudulent activity. Implementing real-time detection mechanisms that use predictive analytics to flag fraudulent clicks as they happen.

Improved Data Collection and Analysis: Leveraging big data to collect and analyze vast amounts of data from handheld devices, improving the accuracy of fraud detection. Analyzing legitimate user behavior to better distinguish between genuine clicks and fraudulent ones.

Cross-device and Cross-platform Detection: Developing systems that can track and identify fraud across multiple devices and platforms, providing a holistic view of user interactions.

Utilizing cloud computing to aggregate and process data from various sources for more comprehensive fraud detection.

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. Ettedgui, 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.