FRAUD DETECTION ON APP USAGE WITH RANKING STATISTICS

Arshia Tabassum Assistant Professor in Department of Information Technology in Teegala Krisha Reddy Engineering college.Telangana
N.Manisha UG Scholar in Department of Information Technology in Teegala Krisha Reddy Engineering college.Telangana
A.Sai Kumar UG Scholar in Department of Information Technology in Teegala Krisha Reddy Engineering college.Telangana
K.Sai Krishna UG Scholar in Department of Information Technology in Teegala Krisha Reddy Engineering college.Telangana

Abstract: Ranking fraud the Apps in the popularity list Indeed, it becomes more and more frequent for App developers to use shady means, such a s inflating their Apps' sales or posting phony App ratings, to commit ranking fraud. While the importance of preventing ranking fraud has been widely recognized, there is limited understanding and research in this area. To this end in this paper, we provide a holistic view of ranking fraud and propose a ranking fraud detection system for mobile Apps. Specifically, we first propose to accurately locate the ranking fraud by mining the active periods, namely leading sessions, of mobile Apps. Such leading sessions can be leveraged for detecting the local anomaly instead of global anomaly of App rankings. Furthermore, we investigate three types of evidences, i.e., ranking based evidences, rating based evidences and review based evidences, by modeling Apps' ranking, rating and review behaviors through statistical hypotheses tests. In addition, we propose an optimization based aggregation method to integrate all the evidences for fraud detection.

KEYWORDS*:* Apps, Ranking Fraud Apprehension, Evidence Reckoning, Historical Records, Rating and Review.

I. INTRODUCTION

Web spam refers to all forms of malicious manipulation of user generated data so as to impudence usage patterns of the data. The number of mobile Apps has grown at a breath Taking rate over the past few years. For example, as of the end of April 2013, there are more than 1.6million Apps at Apple's App store and Google Play .

To stimulate the development of mobile Apps, many App stores launched daily App leader boards, which demonstrate the chart rankings of most popular Apps. Indeed, the App leader boar's one of the most important way for promoting mobile Apps. A higher rank on the leader board usually leads to huge number of downloads and million dollars in the revenue. Therefore, App developers tend to explore various ways such as advertising campaigns to promote their Apps in

Margins, column widths, line spacing, and type styles are built-in; examples of the type styles are provided throughout this document and are Identified in italic type, hence within parentheses, following the example. Some components, such as multi-leveled equations, graphics, and tables are not prescribed, although the various table text styles are provided. The formatter will need to create these components, incorporating the applicable criteria that follow. Indeed, our careful observation reveals that mobile Apps are not always ranked high in the leader board, but only in some leading events, which form different leading sessions. Note that we will introduce both leading events Ease of Use and leading sessions in detail later. In other words, ranking fraud usually happens in these leading sessions. Therefore, detecting ranking fraud of mobile Apps is actually to detect ranking fraud within leading sessions of mobile In addition, we develop an unsupervised evidence-aggregation method to integrate these three types of evidences for evaluating the credibility of leading sessions from mobile Apps. Figure 1 shows the framework of our ranking fraud. . Finally, we evaluate the proposed system with real-world App data collected from the Apple's App s tore for a long time period, i.e., more than two years. Experimental results show the effectiveness of the proposed system, the scalability of the detection algorithm as well as some regularity of ranking fraud activities. According to the definitions introduced in, a leading session is composed of several leading events. Therefore, w should first analyze the basic characteristics of leading evens for extracting fraud evidences hence .By the analyzing the Apps' historical ranking records, we observe those that Apps' ranking behaviors in a leading even always satisfy a specific ranking pattern, which consists of the three different ranking phase, namely, rising phase, maintaining phase and recession phase.

II. OBJECTIVE

The ranking based evidences are useful for ranking fraud detection. However, sometimes, it is not sufficient to only use ranking based evidences. For the example, some Apps created by the famous developers, such as Game loft, may have some leading events with large values of e a c h u s e r due to the developers' credibility and the "word-of mouth" advertising effect. Moreover, some of the legal marketing services, such as "limited time discount" , may also result in significant ranking based evidences. To solve this issue, we also study how to extract fraud evidences from Apps' historical rating records.

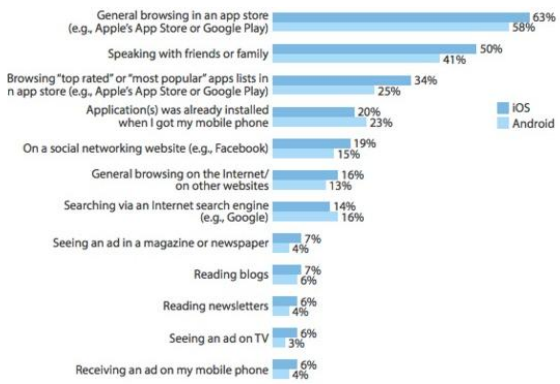

FIGURE:COMPARISIONS OF VARIOUS APPS IN OS

III. PROPOSAL

Besides ratings, most of the App stores also allow users to write some textual comments as App reviews. Such reviews can reflect the personal perceptions and usage experiences of existing users for particular mobile Apps. Indeed, review manipulation is one of the most important perspectives of App ranking fraud. Specification before downloading or purchasing new mobile users often firstly read its historical reviews to ease their decision making, and a mobile App contains more positive reviews may attract more users to download.

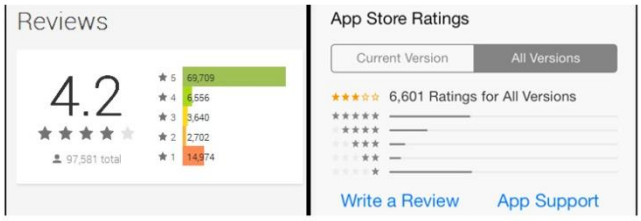

Figure: General rating details of APP.

Indeed, most of the the review manipulations are implemented by both farms due to the high cost of human resource. We define a fraud signature, which denotes the average mutual similarity between the reviews within leading the session *s*. Specifically, this fraud signature can be computed by following steps

This algorithm is used for SSL3 client authentication. In the SSL3 protocol, a concatenation of an MD5 hash and a SHA hash is signed with an RSA private key. CryptoAPI 2.0 and the Microsoft Base and Enhanced Cryptographic Providers support this with the hash type CALG_SSL3_SHAMD5.

To create a CALG_SSL3_SHAMD5 hash

1. Using standard CryptoAPI methodology, create both a MD5 and a *SHA hash* of the target data.
2. Concatenate the two hashes, with the MD5 value leftmost and the SHA value rightmost. This results in a 36-byte value (16 bytes + 20 bytes).
3. Get a handle to a *hash object* by calling CryptCreateHash with CALG_SSL3_SHAMD5 passed in the *Algid* parameter.
4. Set the hash value with a call to CryptSetHashParam. The concatenated hash values are passed as a BYTE* in the *pbData* parameter, and the HP_HASHVAL value must be passed in the *dwParam* parameter. Calling CryptHashData using the handle returned by CryptCreateHash in step 3 will fail.
5. Call CryptSignHash to generate the signature.
6. Call CryptDestroyHash to destroy the hash object

After extracting three types of fraud evidences, the next challenge is the way how to combine them for ranking fraud detection. Indeed, there are many ranking and evidence aggregation methods in the literature, such as permutation based models [17], [18], score based models [11], [26] and Dempster-Shafer rules [10], [23]. However, some of these methods focus on learning a global ranking f all candidates. This is not proper for detecting ranking fraud for new Apps. Other methods are based on supervised learning techniques, which depend on the labelled training data and are hard to be exploited. we propose an unsupervised approach

The following example hashes some data and signs that hash. In a second phase, the hash and its signature are verified. The hash is signed with the user's private key, and the signer's public key is exported so that the signature can be verified.

This example illustrates the following tasks and CryptoAPI functions:

Acquiring a CSP using CryptAcquireContext.

- Getting the user's AT_SIGNATURE key pair using CryptGetUserKey.
- Creating a PUBLICKEYBLOB with the signer's public key to be used in the signature verification process using CryptExportKey.
- Creating a hash object using CryptCreateHash.
- Hashing the data using CryptHashData.
- Signing the hash using CryptSignHash.
- Destroying the original hash object using CryptDestroyHash.
- Making the public key needed to verify the hash available using CryptImportKey.
- Re-creating the hash object using CryptCreateHash and CryptHashData.
- Verifying the signature on the hash using CryptVerifySignature.
- Performing normal cleanup.

Web ranking spam refers to any deliberate actions which bring to selected WebPages an unjustifiable favourable relevant importance .Web spam detection,which comprehensively introduces the principles and algorithm in the literature. Indeed, the work of Web ranking spam is mainly based on the analysis of ranking principles of sear engines, such as Page Rank and query term

frequency. This different from ranking fraud detection for mobile Apps

A hashed message authentication checksum (HMAC) is typically used to verify that a message has not been changed during transit. Both parties to the message must have a shared secret key. The sender combines the key and the message into a string, creates a digest of the string by using an algorithm such as SHA-1 or MD5, and transmits the message and the digest. The receiver combines the shared key with the message, applies the appropriate algorithm, and compares the digest thus obtained with that transmitted by the sender. If the digests are exactly the same, the message was not tampered with.

This example demonstrates the following tasks and CryptoAPI functions:

- Acquiring a handle to a cryptographic service provider by calling CryptAcquireContext.
- Deriving a symmetric key from a byte string by calling CryptCreateHash, CryptHashData, and CryptDeriveKey.
- Using the symmetric key to create an HMAC hash object by calling CryptCreateHash and CryptSetHashParam.
- Hashing a message by calling CryptHashData.
- Retrieving the hash by calling CryptGetHashParam.

IV. CONCLUSIONS

In this paper, we developed a ranking fraud detection system for mobile Apps. Specifically, we first showed that ranking fraud happened in leading sessions and provided a method for mining leading sessions for each App from its historical ranking records. Then, we identified ranking bas evidences, rating based evidences and review based evidences for detecting ranking fraud .Moreover , we proposed an optimization based aggregation method to integrate all the evidences for evaluating the credibility of leading sessions from mobile Apps. An unique perspective of this approach that all the evidences can be modelled by statistical hypotheses tests, thus it I s easy to be extended with other evidence from domain knowledge to detect ranking fraud. Finally, we validate the proposed system with extensive experiment on real-world App data collected from the Apple's App stores. Experimental results showed the effectiveness of the propos approach. In the future, we plan to study more effective fraud evidences and analyze the latent relationship among rating, review and rankings. Moreover, we will extend our ranking fraud detection approach with other mobile App is related services, such as mobile Apps recommendation, for enhancing user experience.

REFERENCES

[1]. http://en.wikipedia.org/wiki/cohen'skappa.

[2]. http://en.wikipedia.org/wiki/information retrieval.

[3]. https://developer.apple.com/news/index.php ?id=02062012a

[4]. http://www.ibtimes.com/apple-threatens-crackdown-biggestapp-store-ranking-fraud-406764.
http://www.lextek.com/manuals/onix/index. html.

[5]. http://www.ling.gu.se/˜lager/mogul/porter-stemmer.

[6]. L. Azzopardi, M. Girolami, and K. V. Risjbergen. Investigating the relationship between language model perplexity and ir precision recall measures. In Proceedings of the 26th International Conference on Research and Development in Information Retrieval (SIGIR'03), pages 369–370, 2003.

[7]. D. M. Blei, A. Y. Ng, and M. I. Jordan. Lantent dirichlet allocation. Journal of Machine Learning Research, pages 993– 1022, 2003

[8]. Y. Ge, H. Xiong, C. Liu, and Z.-H. Zhou. A taxi driving fraud detection system. In Proceedingsof the 2011 IEEE 11th International Conference on Data Mining, ICDM '11, pages 181–190, 2011.

[9]. D. F. Gleich and L.-h. Lim. Rank aggregation via nuclear norm minimization. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '11, pages60–68, 2011.

[10]. T. L. Griffiths and M. Steyvers. Finding scientific topics. In Proc.of National Academy of Science of the USA, pages 5228–5235, 2004.

[11]. G. Heinrich. Parameter estimations for text analysis. Technical report, University of Lipzig, 2008.

[12]. N. Jindal and B. Liu. Opinion spam and analysis. In Proceedings of the 2008 International Conference on Web Search and Data Mining, WSDM '08, pages 219–230, 2008.

[13]. J. Kivinen and M. K. Warmuth. Additive versus exponentiated gradient updates for linear prediction. In Proceedings of the twenty seventh annual ACM symposium on Theory of computing, STOC '95,pages 209– 218, 1995.

[14]. A. Klementiev, D. Roth, and K. Small. An unsupervised learning algorithm for rank aggregation. In Proceedings of the 18th European conference on Machine Learning, ECML '07, pages 616–623, 2007