ISSN XXXX XXXX © 2018 IJESC

Research Article Volume 8 Issue No.3

Detecting Malwares and Search Rank Fraud in Google Search Using Rabin Karp Algorithm

Keerthana.B[1], Shaistha Tabasum.S[2], Sivasankari.K[3]
BE Student[1, 2], Assistant Professor[3]
Department of Computer Science and Engineering
Dhaanish Ahmed College of Engineering, Chennai, India

Abstract:
To develop a java application to search web and capture the result page for study. The organic SEO websites to be examined to find out any Fraudulent or immoral ways are whether used by the SEO specialists to promote the website and to suggest the name of such websites for removal, as we cannot remove those websites but we can suggest in an acceptable way with proven research of those websites. The landing page of all websites, the top 10 first given word are to be stored in a file using java application and such websites must be examined with on page SEO techniques used. The algorithm used is **RABIN KARP ALGORITHM**. Java provides most powerful API's like IO a net to do coding related to internet and IO activities like reading, writing and searching the file, counting the keywords, matching.

Keywords: Data Mining, Rabin Karp Algorithm, SQL database, JAVA.

I. INTRODUCTION

We seek to identify both malware and search rank fraud subjects in Google Play. This combination is not arbitrary: we posit that malicious developers resort to search rank fraud to boost the impact of their malware. Fair Play, a novel system that discovers and leverages traces left behind by fraudsters, to detect both malware and apps subjected to search rank fraud. Fair Play correlates review activities and uniquely combines detected review relations with linguistic and behavioral signals gleaned from Google Play app data (87 K apps, 2.9 M reviews, and 2.4M reviewers, collected over half a year), in order to identify suspicious apps. Fair Play achieves over 95 percent accuracy in classifying gold standard datasets of malware, fraudulent and legitimate apps. Prime objective of the current project is to study the web page source Code and to analyze the SEO Techniques used in that landing page to get importance in Google Search. We read the source and store it data base for future reference and study as Google search results may change time to time. Search engine optimization (SEO) is the process of improving the volume and quality of traffic to a web site from search engines via "natural" ("organic" or "algorithmic") search results."The benefits generated by a well planned SEO campaign are numerous. It gives your website exposure and visibility by improving its rankings organically without having to pay the Search Engines, SEO ensures your site is easily navigated and ensures the website is geared towards your visitors to turn them into customers, and there is no point in getting your customers to the site if they don't want to stay. We try to develop an application using the Java Technology to read the search results of Google for a given keyword say 'Top 10 private engineering colleges in Chennai'. To read the search results. Ignore the websites promoted through Ads and considering the top 5 organic listing. Visit the landing page of the top we page and read the HTML source code Count the key words And take a decision based on number of times the subject key word was presented. To get top rank, the ways the tag are used We seek to identify both malware and search rank fraud subjects in Google Search This combination is not arbitrary: we posit that malicious developers resort to search rank fraud to boost the impact of their malware..

1.1 Contributions

We propose, a system that leverages the above observations to efficiently detect Google Play fraud and malware our major contributions are: A Fraud and Malware Detection Approach. To detect fraud and malware, we propose behavioral and linguistic features, that we use Rabin Karp algorithms

1.2: Literature Survey

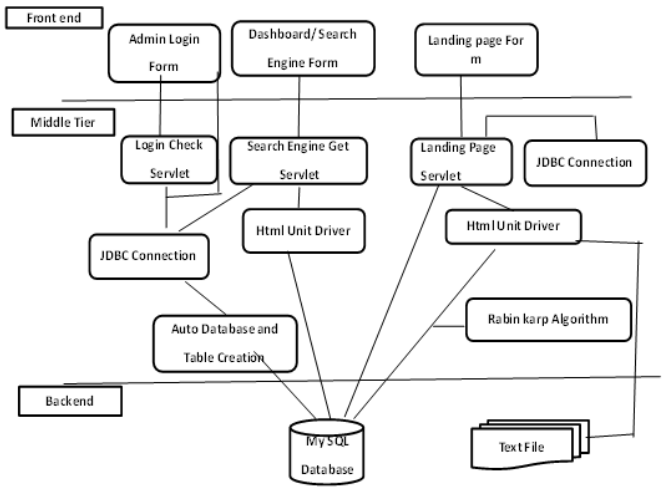
Marco: Detecting Fraudulent Review Behaviors in Yelp. In this they introduced (Malicious Review Campaign Observer), a novel system that leverages the wealth of spatial, temporal and social information provided by Yelp, to detect venues that are targets of deceptive behaviors. Marco exploits fundamental fraudster limitations to identify suspicious venues. First, Marco identifies venues whose positive review timeline exhibits abnormal review spikes, see the adjacent figure. We prove that if a venue has more than 49 genuine reviews, a successful review campaign for that venue will exceed, during the attack interval, the maximum number of reviews of a uniform review distribution. Second, Marco exploits the observation that a venue that is the target of a review campaign is likely to receive reviews that do not agree with its genuine reviews. In addition, following a successful review campaign, the venue is likely to receive reviews from genuine users that do not agree with the venue's newly engineered rating. Marco defines then the disparity of a review for a venue, to be the divergence of the review's rating from the average rating of the venue at the time of its posting. The aggregate rating disparity score of a venue is then the average rating disparity of all its reviews. This is illustrated in the adjacent figure that plots the evolution in time of the average rating against the ratings of individual reviews received by the ``Azure Nail & Waxing Studio" (Chicago, IL). The positive reviews (1 day has a spike of 19, 5-star reviews, shown in red in the upper right corner) disagree with the low rated reviews,

generating a high average rating disparity. Third, Marco detects both venues that receive large numbers of fraudulent reviews, and venues that have insufficient genuine reviews to neutralize the effects of even small scale campaigns. In preliminary work we have used the features we extract to classify 7,435 venues we collected from Miami, San Francisco and New York City. The table on the right shows our results. The black numbers represent the number of venues of a certain type we have collected from each city. The red numbers between parentheses represent the number of such venues that Marco has detected as suspicious. We observe that San Francisco has the highest concentration of deceptive venues: Marco flags almost 10% of its car repair and moving companies as suspicious.

1.3 Results

By using this algorithm it first Study web page source Code and then analyze SEO Techniques. SEO is the Google's Algorithm. It helps us to understand the code. After this process it goes to the landing page in that place we have lot of links are present and the links are stored in data base for future reference. The Google search results which change time to time and count the keywords. Tags used by SEO are <title>, <meta>, <Header> etc. Then we can give suggestion to Google to remove the Fake activities.

II. PROCESS DIAGRAM:



III. WHAT RABIN KARP ALGORITHM STATES

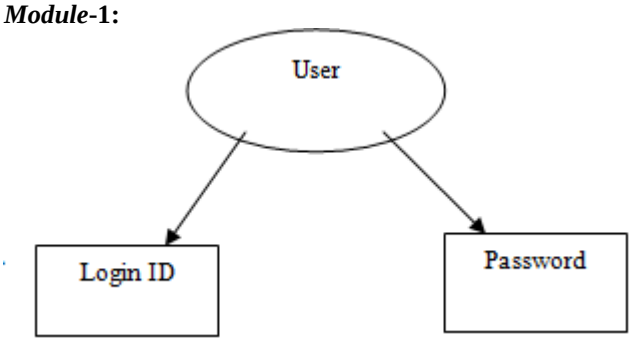
The Rabin-Karp algorithm is a string searching algorithm that uses hashing to find patterns in strings. A string is an abstract data type that consists of a sequence of characters. Letters, words, sentences, and more can be represented as strings. String matching is a very important application of computer science. If you've ever searched through a document for a in this module a Java program is written to count the key word in the source code of the Landing page. All the five links will show 0 before clicking the count button and after clicking the count button the field is updated with number of times the key word is found. This is very important to analyze the SEO Technics…

IV. HOW ITS WORKS:

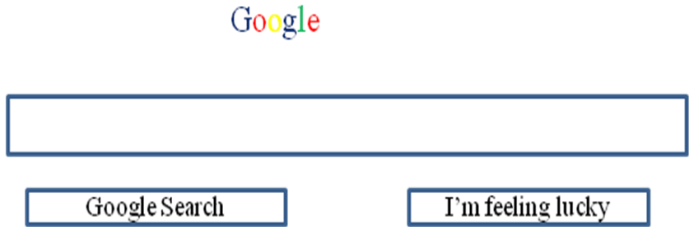
Being a web project that too being an online, we need authentication to get into landing page or Dash Board. The user id and password are hard coded, that is written inside code itself. No database is used as we have only one user.

This is the prime module developed using java servlet to search the web, a Google search simulation program. The result page of the Google is saved in your folder and displayed as a result page for the user. The top five links are stored in mysql database for further research they involve JDBC connectivity. This is the key module developed using Java Servlet to Visit the landing page or Dash Board and read the HTML Source code. The source codes are saved in a file as well in MySql Database for further reference. JDBC code is required and Selinium web driver is also used in Eclipse IDE that plays a vital role in 2nd and this module In this module a Java program is written to count the key word in the source code of the Landing page. All the five links will show 0 before clicking the count button and after clicking the count button the field is updated with number of times the key word is found. This is very important to analyze the SEO Technics… Here we write a Java Program to search the website to find out the string Like 'on the basis of 'key word. If no such key word found in that page then the website will be treated as biased or less helpful. Suggestions to Google are planned as Google always welcome suggestions from anyone.

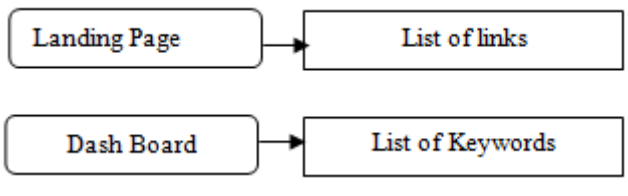
***Module*-1:**



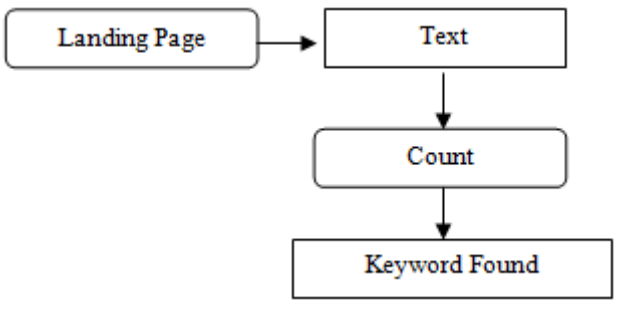
Module-2



Module-3



Module-4



V. CONSTRUCTION:

Initially we create a login form which consists of user name and password. Once you have logged into the form then you are able to enter into the webpage. In webpage there are two default options like landing page and dashboard options which is used to see the search results. The landing page consists of list of links which are searched earlier. These links are used to see who are all involved in immoral acitivities.these links are stored in sql database using landing page servlet and jdbc connection. The dashboard which displays the keywords that are used to highlight that link in top. we use rabin karp algorithm to identify the duplication of link or fake links. In rabin karp algorithm we use string function is used to identify fake links. The Rabin-Karp algorithm is a string searching algorithm.. It uses hashing to find patterns in strings. It can also be used to detect plagiarism by comparing strings in document with strings in document. A practical application of the algorithm is detecting plagiarism. After detecting the plagiarism we suggest the Google to take our suggestion about that link. Since the link contains some malware activities which could be ignored by the Google to have a better search links for the Google users.

VI. APPLIED ALGORITHM:

e h_p(for pattern p)
Compute h_t(for the first substring of t with m length)
For i= 1 to $n - m$
If h_p= h_t
Match $t[i....i+ m]$ with p,
if matched return 1 Else
$h_t = (d\left(h_t - t\left[i+1\right].d^{m-1}\right) + t\left[m+i+1\right]) \bmod q$
End
Suppose, t= 2359023141526739921 and p = 31415,
Now, h_p= 7 (31415 = 7 (mod 13))
Substring beginning at position 7 = valid match

VII. STRING MATCHING:

Rabin-Karp string searching algorithm calculates a numerical (hash) value for the pattern p, and for each m-character substring of text t. Then it compares the numerical values instead of comparing the actual symbols. If any match is found, it compares the pattern with the sub stringby naive approach. Otherwise it shifts to next substring of t to compare with p.We can compute the numerical (hash) values using Horner's rule. The running time of the Rabin Karp algorithm is O((n-m+1)m) in the worst case, since the Rabin Karp algorithm explicitly verifies ever valid shift. The applications of this algorithm are Text Processing, Bioinformatics, and Compression.
Let us assume, h0 = k
h1 = d
□
k □ p [1] :dm□1
_
+ p [m + 1]
Suppose, we have given a text t = [3, 1, 4, 1, 5, 2] and m = 5, q = 13;
t0 = 31415
So t1 = 10(31415 - 105□1.t[1]) + t[5+1]
= 10(31415 □ 104:3) + 2
= 10(1415) + 2 = 14152
Here p and substring ti may be too large to work with conveniently. The simple solution is,
we can compute p and the ti modulo a suitable modulus q.
So for each i,
hi+1 = (d
□
hi □ t [i + 1] :dm□1
_
+ t [m + i + 1]) mod q
The modulus q is typically chosen as a prime such that d.q _ts within one computer word.
Algorithm
Compute hp (for pattern p)
Compute ht (for the _rst substring of t with m length)
For i = 1 to n □ m
If hp = ht
Match t[i : : : :i + m] with p, if matched return 1
Else
ht = (d
□
ht □ t [i + 1] :dm□1
_
+ t [m + i + 1]) mod q
End
Suppose, t= 2359023141526739921 and p = 31415,
Now, hp = 7 (31415 = 7 (mod 13))

substring beginning at position 7 = valid match This algorithm has a signi_cant improvement in average-case running time over naive ap-proach.

VII. ADVANTAGES

1. To find out the content quality in Google.
2. To help in finding relevancy of landing page.
3. To help Google's search algorithm for better accuracy

VIII. CONCLUSION & FUTURE ENHANCEMENT

Internet and websites now plays predominant role in Business and all walks of life. Google search becomes a popular way of searching the we Hence every one having website would like to see his website on the top Ranks This leads to chances of growing SEO techniques. When people search for your products and services, you obviously want to appear as high in the search engine rankings as possible. So we try to analyze the source code of that web page to find out the seo techniques used and try to suggest fraudulent ones. We can't change the results but can suggest such websites. .

IX. REFERENCES:

[1]. Google Play. [Online]. Available: https://play.google.com/

[2]. E. Siegel, "Fake reviews in Google Play and Apple App Store," Appentive, Seattle, WA, USA, 2014.

[3]. Z. Miners. (2014, Feb. 19). "Report: Malware-infected Android apps spike in the Google Play store," PC World. Available: http:// www.pcworld.com/article/2099421/report-malwareinfectedandroid-apps-spike-in-the-google-play-store.html

[4]. S. Mlot. (2014, Apr. 8). "Top Android App a Scam, Pulled From Google Play," PCMag. Available: http:/ /www. pcmag.com/ article2/0,2817,2456165,00.asp

[5]. D. Roberts. (2015, Jul. 8). “How to spot fake apps on the Google Play store,” Fortune. Available: http://fortune. com/2015 / 0 7/08/ google-play-fake-app/

[6]. Freelancer.[Online]. Available:http://www.freelancer. com

[7]. Fiverr. [Online]. Available: https://www.fiverr.com/

[8]. BestAppPromotion. [Online]. Available: www. bestrevi ewapp. com/

[9]. G. Wang, et al., “Serf and turf: Crowdturfing for fun and profit,” in Proc. ACM WWW, 2012. [Online]. Available: http://doi.acm. org/10.1145/2187836.2187928

[10]. J. Sahs and L. Khan, “A machine learning approach to Android malware detection,” in Proc. Eur. Intell. Secur. Inf. Conf., 2012, pp. 141–147.

[11]. B. Sanz, I. Santos, C. Laorden, X. Ugarte-Pedrero, P. G. Bringas, and G. _ Alvarez, “Puma: Permission usage to detect malware in android,” in Proc. Int. Joint Conf. CISIS12-ICEUTE Special Sessions, 2013, pp. 289–298.

[12]. J. Ye and L. Akoglu, “Discovering opinion spammer groups by network footprints,” in Machine Learning and Knowledge Discovery in Databases. Berlin, Germany: Springer, 2015, pp. 267–282.

[13]. L. Akoglu, R. Chandy, and C. Faloutsos, “Opinion Fraud Detection in Online Reviews by Network Effects,” in Proc. 7th Int. AAAI Conf. Weblogs Soc. Media, 2013, pp. 2–11.

[14]. Android market API, 2011. [Online]. Available: https://code. google.com/p/android-market-api/

[15]. E. Tomita, A. Tanaka, and H. Takahashi, “The worst-case time complexity for generating all maximal cliques and computational experiments,” Theory. Comput. Sci., vol. 363, no. 1, pp. 28–42, Oct. 2006. [Online]. Available: http://dx.doi.org/ 10.1016/j.tcs.2006.06.015

[16]. K. Makino and T. Uno, “New algorithms for enumerating all maximal cliques,” in Proc. 9th Scandinavian Workshop Algorithm, 2004, pp. 260–272.

[17]. T. Uno, “An efficient algorithm for enumerating pseudo cliques,” in Proc. ISAAC, 2007. [Online]. Available: http:// dl.acm.org/ citation.cfm?id=1781574.1781621

[18]. S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python. Sebastopol, CA, USA: O’Reilly, 2009.

[19]. B. Pang, L. Lee, and S. Vaithyanathan, “Thumbs Up? Sentiment classification using machine learning techniques,” in Proc. ACL-02 Conf. Empirical Methods Natural Lang. Process., 2002, pp. 76–86.

[20] J. H. McDonald, Handbook of Biological Statistics, 2nd ed. Baltimore, MD, USA: Sparky House Publishing, 2009. [Online]. Available: http://udel.edu/ _mcdonald/statintro.html