Volume 3, Issue 12, December 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Survey on blog bot Detection Techniques

Nikita Priyanka Renato
Department of Computer Science and Engineering,
Karunya University, Coimbatore, India

Abstract: The detection of a blog bot is now become necessary. The main reason for this is because of the improved script programs that are being introduced by programmers. Different approaches are being used to detect the presence of a bot. Each bot has its unique way of mimicking human actions. This paper presents a survey on the various methodologies that are suggested to help in bot detection.

Key words: Blog bot, CAPTCHA, HIPs, C4.5

I. INTRODUCTION

A blog is one of the most common web applications used in today's world. Blogs have the option for other readers to post their comments or views in the comment box below their post. Programmers have introduced automated scripts or programs called blog bots that are programmed to post random comments or spam links to blog sites. Most of the content generated by these bots is not known to the blog owners and the visitors. Blogging software has various methods to avoid postings from such sources. Detecting a human presence proved to be an effective defense against the attacks of these bots. Detection methods based on Human Interactive Proofs (HIPs) [4] usually required direct human user participation, such as using CAPTCHA. The user must enter the same text before the blog site accepts the comment for submission. As the advanced bots' capability keeps increasing for image recognition, CAPTCHA tools add noise to the image to highly distort characters. The main aim of using biometrics is to easily detect if the user is a human or a bot.

The nuisance of the conventional methods led to the need to present a need method based on passive monitoring for the detection of blog bots. This method has two major advantages: Firstly, this method monitors the entire session passively and eliminates any single check points. Secondly, this method is non-interactive and transparent to users. The detection decision doesn't have to be made until the comment has been submitted which helps save system resources. A webpage embedded logger is developed to collect the user activities as inputs from a real active blog site. The biometric features are measured and characterized, and the fundamental differences between human and blog bot in how the web-pages are surfed and comments are posted are discovered. With these results, detection of blog bots can be done accurately. An automatic classification system to detect blog bots based on the user input data consists of two components. A web page embedded logger which runs on the web page on the client browser and server side detector. The web page embedded logger is implemented as java script code in every page of the blog site. Non java script clients are blocked must pass a HIP such as CAPTCHA. The logger is said to be passive because it runs silently and transparently inside the client browser. It collects raw events generated by the user such as Key Press, Key Release, Mouse Move, Mouse Button Press and Mouse Button Release. Each of these events is associated with a java script which is triggered to generate a record in the JSON [5] format. The logger can access the input actions that are only generated by a user in the blog, enabling privacy to the user's data. The confidentiality of user input is maintained by replacing the value of the key strokes with a wildcard character. This avoids additional overhead by encryption.

The server-side detector has three components: the log processor, the classifier and the decision maker. The UI that arrives at the server are raw events. The log processor stores these events in a back end MySQL database. The processor converts these raw events into high-level UI actions. The processor also calculates the entropy rate to measure the complexity of the actions. The action event along with the entropy rate is given to the classifier to classify the actions into action groups. The classifier is based on the C4.5 algorithm [6] which builds a decision tree for classification. The decision tree has two types of nodes, the leaf node labelled with the class value and the interior node which corresponds to the link of a sub tree. The basic idea of the algorithm to is build the tree from the root downward to the leaves. Each interior node must be associated with the attributes that is the most informative. C4.5 applies a greedy search to select the candidate test that maximizes the heuristic splitting criterion. The algorithm is chosen for four main reasons. First, it builds an efficient tree to process large amounts of data in a short time. Second, the algorithm uses the white box approach, which helps in easy understanding and interpretation. Third, both continuous and discrete values can be processed. Last, the tree is pruned from top to down to avoid over fitting and tree height. Once these actions are grouped and determined by the classifier whether the actions are human or bot, the decision maker gives the summary of the classifications. The final decision of the action events is done using a majority voting rule.

This paper focuses on the survey of different detection methods and is given in the following sections. Section II presents the literature survey of different methods and section III concludes with discussions.

II. LITERATURE SURVEY

A. Web robot detection: a probabilistic reasoning approach.

A probabilistic reasoning approach helped to distinguish between bots and human based on the characteristics of the visits in a web-server. This model was introduced to address the problem of robot detection in server access logs. A Bayesian network that classifies the log sessions as human or crawler is constructed, based on the different characteristics of human and crawler behaviour that are witnessed. The probabilistic approach [1] is an ideal approach mainly due to the actual detection problem and the uncertainty in the data. This approach is used as an alternative approach to the decision tree. Adaptive threshold technique is used to extract the data or the web sessions from the access logs. Machine learning techniques are used to determine the various parameters of the probability model. The classification is done based on the maximum posterior probability given all the evidences. This method can be applied to real web server logs.
There is an increasing need to distinguish robots from human for various reasons:

- Click fraud
- Referrer spam
- Performance degradation
- Information protection etc.

The effort to identify robots was presented by Menasce, Almedia et al [7,8]. The process was to derive a set of heuristic criteria by observing the navigation patterns of human and crawlers. The disadvantage of the approach is that it is difficult to render identification of the crawlers due to the openness and lack of central control of the internet. It is also hard to derive simple heuristics because a long list of rules will have to be maintained for bot detection. This disadvantage can be overcome by making the system learn to distinguish crawlers from humans. Bayesian networks provide high accuracy in classification, with which this system detects crawler sessions. And due to this high accuracy, the effectiveness of this methodology is proved.

B. Discovery of Web Robot Sessions based on their Navigational Patterns.

The navigational patterns in the click stream of data were used to determine if the user was a robot. Software programs that automatically traverse the hyperlink structure to retrieve and locate information are called as 'web robots'. The navigational pattern [2] of a robot is completely different from that of a human user. These patterns are characterized in terms of

- Types of pages requested
- Length of the session
- Interval between successive HTML requests
- Coverage of the web site etc.

The goal is to build models which will map each of the session automatically into one of the two pre-defined classes, i.e. robots or humans. The methodology that was followed to achieve this goal was first to transform web data into individual sessions. This was done by grouping web log entries according to the user agent fields and the ip address. After this, the features that best describe the properties of each class is extracted by breaking the session into many episodes where each episode will correspond to a request for a HTML file. The third step is to follow a procedure to determine the class of each label of each session. This is done according to their user agents. Then techniques were used to assess the labeling scheme. The classification models are built using an algorithm. Finally a metric is used to evaluate the performance of the models. The results of this method showed that these classification models can be used to detect robots that camouflage and have very similar navigational patterns as other robots. The drawback of this method was that the labeling procedure could introduce errors in the classification models.

C. Securing web Service by Automatic robot detection

Web sites are often visited by agents that are automated know as web robots. These can be malicious or beneficial. This paper uses a special form of turing test to defend the system by deciding if it is a human or robot. Experiments with CoDeeN content distribution network [3] was conducted which showed that 95% of humans were detected in 57 requests and 80% of humans were detected in 20 requests with a maximum false positive rate of 2.4%. A few observations were made which were useful generally. First, most web browsers behave similarly. These patterns are easy to learn, but most web robots deviate from normal browsers. Second, human user behavior differs from a web robot behavior, example human users will be able to follow only visible links but robots are crawlers, and they crawl through all the links of the page. Third, robots need not use a mouse and keyboard to generate such activity. Using the observations, two algorithms were proposed to differentiate humans from bots in real time.

- Human activity detection
- Standard browser detection.

The human activity detection algorithm obfuscates the script that is served to the client. This way if the code is executed by the client, the server can infer that it is a robot and it can run the JavaScript code. The browser detection algorithm is used to see if the user or the client's behavioural pattern will deviate from a normal or a typical browser. If there is deviation, it is assumed that the behaviour is from a robot. The main drawback of this method is that it is not completely immune to the various counter measures done by the attackers. And it needs a large amount of memory, which makes it vulnerable to DoS attacks.

III. CONCLUSION

This paper surveys the different techniques that were used to detect the presence of a web robot or a web crawler. Various methods ranging from a probabilistic approach to using biometrics for detection have been applied. It can be understood that as the complexity of these robots increase, using behavioural biometrics for bot detection has proved to be most efficient.

REFERENCES

[1] Stassopoulou, M.D. Dikaiakos, Web robot detection: a probabilistic reasoning approach, Comput. Netw. 53 (2009) 265–278.

[2] P.-N. Tan, V. Kumar, Discovery of web robot sessions based on their navigational patterns, Data Min. Knowl. Discov. 6 (2002) 9–35.

[3] K. Park, V.S. Pai, K.-W. Lee, S. Calo, Securing web service by automatic robot detection, in: Proceedings of the Annual Conference on USENIX'06 Annual Technical Conference, 2006, pp. 23–23.

[4] K. Chellapilla, K. Larson, P. Simard, M. Czerwinski, Designing human friendly human interaction proofs (hips), in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2005.

[5] Json, Javascript Object Notation <http://www.json.org/> (accessed 08.03.2012).

[6] R. Kohavi, R. Quinlan, Decision tree discovery, in: In Handbook of Data Mining and Knowledge Discovery, University Press, 1999, pp. 267–276.

[7] V. Almeida, D. Menascé, R. Riedi, F. Peligrinelli, R.Fonseca, W. Meira Jr, Analyzing web robots and their impact on caching, in: Proceedings of the Sixth International Workshop on Web Caching and Content Distribution, June 2001, pp. 299–310.

[8] D.A. Menasce, V. Almeida, R. Fonseca, R. Riedi, F. Ribeiro, W. Meira, In search of invariants for e-business workloads, in: 2000 ACM Conference In Electronic Commerce, ACM, 2000, pp. 56–65.

