Robots Exclusion Protocol

Pooja Jha, Soni Goyal, Tanya Kumari, Neha Gupta

Abstract— World Wide Web (WWW) is a big dynamic network and a repository of interconnected documents and other resources, linked by hyperlinks and URLs. Web crawlers are used to recursively traverse and download web pages for search engines to create and maintain the web indices. Moreover, the need of maintaining the up-to-date pages causes repeated traversal of websites by crawler. Due to this, the resources like CPU cycles, disk space, and network bandwidth, etc., become overloaded which may lead to crashing of website and increase in web traffic. However, websites can limit the crawlers through Robots Exclusion Protocol. It is a mechanism for www servers to indicate to crawlers which part of their server should not be accessed. To implement this protocol, a plain text file called robots.txt is created and placed under root directory of the web servers. This approach was chosen as a crawler can find the access policy with only single document retrieval. Also, it supports auto-discovery of XML sitemaps. Thus, this protocol aids in controlling the crawler's activity.

Index Terms— Robots Exclusion Protocol, robots.txt, Robots Meta tags, web crawler

I. INTRODUCTION

The World Wide Web grows at an exponential rate and is subjected to changes. Almost 52% of the content of the web changes on a daily basis [3].People extensively use search engines as to find the required content. Search engines and many web applications and web data mining agents rely on web crawlers to collect information from the web. Due to this dynamic nature of web and increasing demand of web services, web crawlers keep on traversing the web pages continuously again and again. Hence, the resources like CPU cycles, disk space, network bandwidth etc. of the remote servers get overloaded .This may even lead to crashing of the web sites. According to a report, web crawling contributes about 40% of the current web traffic.[4] Thus, the regulation of activities of the web crawlers has become significant. In order to address this problem, Marjitin Koster proposed Robots Exclusion Protocol in 1994.It is a de-facto standard but it is not owned by any standard body. This protocol is implemented through a file called robots.txt. This file is deployed in the root directory of the website and can be accessed by all the crawlers.

The robots.txt tells search engine spiders that which sections of the website are not to be crawled. This helps in controlling the crawlers and makes web more efficient by refraining the crawler from traversing certain pages. Also, the robots exclusion protocol allows websites to prioritize the access for the crawlers by their name. As a result of these biases, the popular search engines like Google dominate the web while others do not get access to the resources.

Manuscript received March 15, 2014.

Pooja Jha, Department of Information Technology, Bharati Vidyapeeth's College Of Engineering, New Delhi, India.

Soni Goyal, Department of Information Technology, Bharati Vidyapeeth's College Of Engineering, New Delhi, India.

Tanya Kumari, Department of Information Technology, Bharati Vidyapeeth's College Of Engineering, New Delhi, India.

Ms. Neha Gupta, Asst. Professor, Department of Information Technology, Bharati Vidyapeeth's College Of Engineering, New Delhi, India.

However, web crawlers may ignore robots.txt especially malwares that scan the web for security vulnerabilities and email harvesters used by spammers will pay no heed to it. Moreover, this file is publicly available thus all can see which sections are not to be used. The remainder section of this paper is organized as

- Robots Exclusion Protocol
- Types of Exclusion
- Robots.txt
- Applications
- Conflict between Robots Exclusion Protocol and Robots.txt
- Limitations of Robots Exclusion Protocol
- Conclusion

II. ROBOTS EXCLUSION PROTOCOL

Robots Exclusion Protocol [1] is a standard that allows website administrators to specify the search engine crawlers which sections of their website are not to be indexed. This specification is mentioned in a file called robots.txt.

There are two historical descriptions:

1. The original 1994 "A Standard for Robots Exclusion" document.
2. The 1997 Internet Draft Specification "A Method for Web Robots Control".

III. TYPES OF EXCLUSION

The robots exclusion protocol involves three types of exclusion which are as follow:

1. Server-wide exclusion instructs the crawler about certain directories that should not be crawled. This is done via a single robots.txt file that is located in the root directory of a Web site. The syntax is very simple and consists basically in one instruction per line, in which a line indicates which user-agents (the name by which the crawler identifies itself) must obey the limits, and the following lines indicate the directories that must not be downloaded:

User-agent: *
Disallow: /data/private
Disallow: /cgi-bin

2. Page-wise exclusion is done by meta-tags in the pages themselves. Meta- tags are part of the standard HTML syntax and allow a Web page author to associate pairs of the form key=value to Web pages. In this case, they key is “robots” and the value is noindex, meaning “do not index this page”, nofollow meaning “do not follow links from this page”, or both. An example in XHTML :< metaname="robots"content="noindex,nofollow"/>

3. Cache exclusion is used by publishers that sell access to their information. In this case, they allow Web crawlers to index the entire contents of their pages, so they will show after searches for the corresponding keywords. At the same time, they instruct search engines not to show the user a local cached copy of the page,

but only the link to the original document. This is done in the same HTML tag as the page-wise exclusion by using the no cache keyword, for example :< meta name="robots" content="nocache"/>
However, sometimes the crawler will access pages that were not meant to be public, and also legal problems with some of the indexed content may arise. For these reasons, it is important to have a fast way of removing a document from the database of crawled pages
Following diagram illustrates the flow of activity when a crawler is in action:

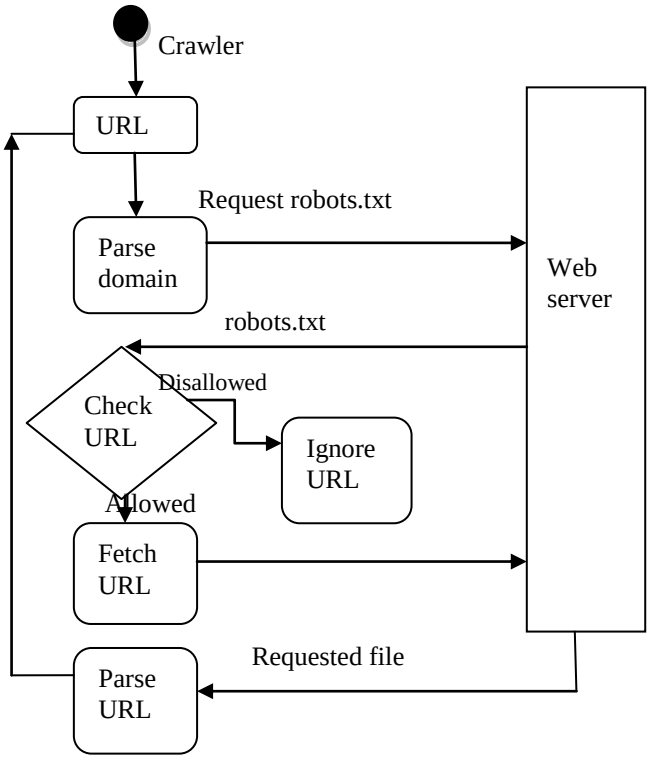

Figure I: Flow of activity

IV. ROBOTS.TXT

The robots.txt file is a text file. The file format is plain text encoded in UTF-8. If the used character encoding results in characters other than those specified in UTF-8, then the file will be parsed incorrectly. An optional Unicode BOM (byte order mark) at the beginning of the robots.txt file is ignored. The file consists of records separated by CR or LF. Each record consists of a field, a colon, and a value. Spaces are optional. Comments can be written using the "#" character; all content after the start of a comment until the end of the record is treated as a comment and ignored. Whitespace at the beginning and at the end of the record is ignored.
A maximum file size may be enforced per crawler. Content which is after the maximum file size may be ignored.
The format for the record is given as [6]:
< field >:< optionalspace >< value >< optionalspace >
The <field> element is case-insensitive. The <value> element may be case-sensitive, depending upon the <field> element. Only valid records will be considered; all other content will be ignored without any warning.
The three types of directives for the <field> are User-Agent, Allow, Disallow and Crawl-Delay. Their descriptions are given as follows:

- User-Agent: This directive is followed by a string str. It tells that any crawler whose name is a superstring str should follow the rules mentioned below. When str is "*" then the rules are applicable to all the crawlers.
- Allow and Disallow: These are followed by a string str .It specifies which sections of the website are allowed and disallowed to visit. However, there's an ambiguity for Disallow: [emptystring] as this can be interpreted as to crawl nothing or anything.
- Crawl-Delay: This directive is followed by a numerical value num .It specifies the minimum time duration for which a crawler should wait between its consecutive visits to the website. Some crawlers do not recognize it.

Table 1: Syntax for robots.txt

Task	Entry	Notes
Allow robots complete access to the server	User-agent:*	* means all user agents (robots).Because nothing is disallowed, everything is allowed.
Exclude all robots from part of the server	User-agent:* Disallow:/cgi-bin/ Disallow:/tmp/ Disallow:/private/	* means all user agents. The robots should not visit any pages in these directories.
Exclude a single robot	User-agent: BadBot Disallow:* User-agent:* Disallow:/priv/	In this case, the badBot robot is not allowed to see anything. All other agents(*) can see everything. The blank line indicates a new “record”-a new user agent command.
Exclude a robot from a single file	User-agent: WeirdBot Disallow:/links/listing.html User-agent : * Disallow: /tmp/ Disallow:/private/	This keeps the Weirdbot from visiting the listing page in the directory, while all other robots can see everything except the temp and private directories. The * for all other robots should always be at the end of the robots.txt file.

The robots.txt file is to be located on the root of the website host. For example, as to control crawling on all URLs below http://www.abc.com/, the robots.txt file must be located at http://www.abc.com/robots.txt.It can be placed in the sub domains like http://website.abc.com/robots.txt or even on non-standard ports such as http://abc.com:8181/robots.txt .However; it can't be placed in a subdirectory. After, the robots.txt is fetched, following are the possible cases:

1. Full allow: Whole content can be crawled.
2. Full disallow: No content should be crawled

3. Conditional allow: Some content shouldn't be crawled

The accepted protocols for robots.txt are "http" and "https". On http and https, the robots.txt file is fetched using a HTTP non-conditional GET request. Its response will be one of the following [2]:

- 2xx (successful): signals success i.e. "conditional allow" of crawling.
- 3xx (redirection): Redirects will generally be followed until a valid result can be found.
- 4xx (client errors): This is a "full allow" for crawling. This includes 401 "Unauthorized" and 403 "Forbidden" HTTP result codes.
- 5xx (server error): It results in a "full disallow" of crawling. The request is retried until a non-server-error HTTP result code is obtained.

However, handling of a robots.txt file which could not be fetched due to DNS or networking issues is undefined. A robots.txt request is generally cached for up to one day, but may be cached for longer durations when refreshing the cached version is not possible.

V. APPLICATIONS

A robots.txt file provides critical information for search engine spiders that crawl the web. It indicates which parts of the website are not to be crawled. This idea was accepted as it can easily be put into action for all the web servers. Thus, it enhances the efficiency of the web crawlers and helps to reduce the web traffic. Its other uses are:

- Refrain crawlers from traversing non-public parts of a website
- Prevents search engines from trying to index scripts, utilities, or other types of code
- Avoids indexing the duplicate content on a website, such as “print” versions of html pages
- Auto-discovery of XML Sitemaps

VI. CONFLICTS BETWEEN ROBOTS.TXT AND ROBOTS META TAGS

In addition to root-level robots.txt files, robots exclusion directives can be applied at a more granular level through the use of Robots meta tags[5].The robots meta tag is effective after the page is loaded, whereas robots.txt is effective before the page is requested. Following are the possible scenarios when both are in action:

- Case 1: **Robots.txt forbids** indexing of a URL but **meta tag allows** it.
  Outcome: Page will be blocked by robots.txt and meta tag will never be seen by crawler, thus page will be shown in results as a reference rarely.
- Case 2: **Robots.txt allows** indexing of a URL but **meta tags forbids** it.
  Outcome: Page will not be indexed and will not be shown in the search results at all.
- Case 3: **Both Robots.txt and meta tag forbid** the indexing of a URL.
  Outcome: Page will be blocked by robots.txt, meta tag will never be seen by crawler and thus will be ignored, which will allow the page to be shown as snippet with no description in rare occasions in the search results.

Both robots.txt and robots meta tag rely on cooperation from the robots, and are by no means guaranteed to work for every robots. If you need stronger protection from robots and other agents, you should use alternative methods such as password protection.

VII. LIMITATIONS

1. The robots exclusion protocol allows websites to explicitly specify an access preference for each crawler by name. Such biases may lead to a "rich get richer" situation, in which a few popular search engines ultimately dominate the web because they have preferred access to resources that are inaccessible to others.
2. It can easily be exploited by a crawler. Also, by listing directories robots.txt gives a clear pointer to potentially interesting directories. Short of password protection, there is nothing to stop a crawler from exploring these directories. To entirely prevent a page from being added to a search engine’s index even if other sites link to it, one can use a “noindex” robots meta tag and ensure that the page is not disallowed in robots.txt. When spiders crawl the page, it will recognize the “noindex” meta tag and drop the URL from the index.
3. The ambiguous directives and conflicts between robots.txt and robots meta tags often lead to undesired results.
4. Malicious crawler may or may not follow these conventions as nothing concrete has been done to guarantee its usage by the crawlers

These weaknesses of the robots exclusion protocol are particularly disconcerting in light of recent efforts by search engines.

VIII. CONCLUSION

Robots Exclusion Protocol helps in controlling the behavior of crawlers. This protocol deploys a file called robots.txt on the web server. It avoids the crawler from traversing the private sections of the website. Thus, it decreases the web traffic. Although robots.txt helps making web more efficient and regularized but its usage heavily depends on the cooperation from crawlers. Also, the specification of robots.txt file is ambiguous and is open for improvements.

ACKNOWLEDGMENT

Pooja Jha, Soni Goyal and Tanya Kumari would like to thank Ms. Neha Gupta, faculty member of B.V.C.O.E., New Delhi for providing us with valuable resources and insights needed for this paper. She has been a guideline for us throughout this. We really appreciate her valuable time spent for the betterment of this paper.

REFERENCES

1. M. Koster, A method for web robots control. Internet Draft, the Internet Engineering Task Force (IETF), 1996.
2. Maile Ohye, Controlling Crawling and indexing .from:https://developers.google.com/webmasters/control-crawl-index/docs/getting_started.
3. Niraj Singhal,Ashutosh Dixit, R.P. Agarwal and A.K. Sharma, Reducing Network Traffic and Managing Volatile Web Contents Using Migrating Crawlers with Table of Variable Information, World Applied Sciences Journal 19 (5), IDOSI Publications, 2012 pp : 666-673

4. R. A, Bal, S. Nath Novel Approach to Filter Non-Modified Pages at Remote Site without Downloading during Crawling, In: Advances in Recent Technologies in Communication and Computing, 2009. ARTCom '09. International Conference on 27-28 Oct. 2009
5. Wendy Chisholm, Gregg Vanderheiden from Web Content Accessibility Guidelines 1.0 - Guideline 7. September 28, 2007
6. Yang Sun, A Comprehensive Study of the Regulation and Behavior of Web Crawler; Pennsylvania University, 2008.

AUTHORS PROFILE

Pooja Jha is currently pursuing her degree in Bachelor of Technology in Information Technology from Bharati Vidyapeeth's College of Engineering, affiliated to GGSIPU, India. Her area of interest includes robots exclusion protocol and web crawler.

Tanya Kumari is pursuing her B.tech degree in Information Technology from Bharati Vidyapeeth's College Of Engineering, New Delhi affiliated to GGSIPU. Her area of interest is web crawlers.

Soni Goyal is pursuing bachelor of technology in information technology from Bharati Vidyapeeth's College of Engineering, GGSIPU,New Delhi. In past, she has attended National Symposium.

Ms. Neha Gupta, Asst. Professor, Department of Information Technology, Bharati Vidyapeeth's College Of Engineering, New Delhi, India.