# Analyzing Security Level of Web Links using Web Scrapping Tools and BeautifulSoup Environment

**Surjeet Kaur[#1], S.R.Tandan[#2], Priyanka Tripathi[*3], Rohit Miri [#4]**
[#,] Department of Computer Science and Engineering
Dr. C V Raman University, Bilaspur, CG, India
[*,] National Institute of Technical Teachers Training, Bhopal MP, India

***Abstract*—**The way internet is used security is very important. All important notification available in the internet is in form of web links. Web scraping is used to extract URL using Python scripting and BeautifulSoup for proposed method. Security levels of links have been analyized on the basis of number of URL retrieved. Focused on method for URL retrieval technique has been developed and done on online data. URL extraction method has been implemented to classify the web links using Python and BeutifulSoup. On the basis of retrieved URLs analysis, link monitoring for various security reasons and to reduce the load of server by removing dummy links have been suggested.

***Index Terms*— Security Analysis, Web Information Retrieval, BeutifulSoup, Information Processing**

## I. INTRODUCTION

Increasing demand of digital information along with higher productivity and better quality is on a rise in todays era. To get on time delivery of updated information with accuracy, the industry is turning towards Information and Communication Technology ICT. The digital documents are growing at a very high rate because of the extensive usage of electronic media, social networking and internet. The contents of the documents are mainly text, images, and links to name a few. Almost 80% of the contents are represented in the text form [20].

Today Information Retrieval System is a powerful tool and becoming more and more significant in various aspects of human life, especially in industrial, commercial and scientific applications. As a result of scientific achievements and IT Industry development the number of information retrieval systems currently in use for corporate projects are increasing fast. It has become a must to maintain the same pace for the IR systems too. However, IR has been evolving ever since and there has been an increasing interest in developing information extraction systems, still this area needs to be researched and fasten a lot.

Information Processing systems heavily depend on World Wide Web www and vice-versa. Web contains an accumulation of hyperlinks, text and images. Web mining methods consist of incredible framework utilized for data extraction.

The continuous growth in information technology requires a machine which can handle the system and extract the information correctly is the need of the current generation. This concept is classically known as Big Data. The deep investigation of intelligence and meaningful patterns from Big Data is known as Big Data Analytics. A number of researchers and scientists are working in this domain of Big Data using assorted technologies and tools. There are number of approaches by which the live data can be obtained for research and development. One of these approaches is getting data from Open Data Portals. The open data portals provide authentic data sets for research and development in multiple domains. The data sets can be downloaded from these portals in multiple formats including XML, CSV, JSON and many others.

Many times data is not easily accessible – although it does exist. As much as we wish everything was available in CSV or the format of our choice – most data is published in different forms on the web. What if you want to use the data to combine it with other datasets and explore it independently? [1] One of the solutions is Screen Scraping. Screen Scraping is the technique to capture the data that is being displayed in human readable format on the destination terminal and to replicate it at the source terminal for further processing. Screen scraping is sometimes referred to as terminal emulation [2]. Though there are other ways to get the data out of the web i.e., from web-based APIs, such as interfaces provided by online databases and many modern web applications (including Twitter, Facebook and many others). This is a fantastic way to access government or commercial data, as well as data from social media sites [3]. Extracting information from PDFs is beyond the scope of this paper, but there are some tools and tutorials that may help you do it [3]. But the advantage of scraping is that you can do it with virtually any web site — from weather forecasts to government spending, even if that site does not have an API for raw data access. However, screen scraping is not an independent process. Before scraping the output, Crawlers are responsible to navigate to the destination terminal. The search key entered at the source machine, engages the crawlers to navigate through the links on the web. Once the crawlers successfully reaches the correct page that matches up with the search string, scraping process starts.

### (a) What Is Web Scraping?

The automated gathering of data from the Internet is nearly as old as the Internet itself. Although web scraping is not a new term, in years past the practice has been more commonly known as screen scraping, data mining, web harvesting, or similar variations. In theory, web scraping is the practice of gathering data through any means other than a program interacting with an API (or, obviously, through a human using a web browser). This is most commonly accomplished by writing an automated program that queries a web server, requests data (usually in the form of the HTML and other files that comprise web pages), and then parses that data to extract needed information. In practice, web scraping encompasses a wide variety of programming techniques and technologies, such as data analysis and information security. Why Web Scraping? If the only way you access the Internet is

through a browser, you're missing out on a huge range of possibilities. Although browsers are handy for executing JavaScript, displaying images, and arranging objects in a more human-readable format (among other things), web scrapers are excellent at gathering and processing large amounts of data (among other things). Rather than viewing one page at a time through the narrow window of a monitor, you can view databases spanning thousands or even millions of pages at once. In addition, web scrapers can go places that traditional search engines cannot. A Google search for "cheapest flights to Raipur" will result in a slew of advertisements and popular flight search sites. Google only knows what these websites say on their content pages, not the exact results of various queries entered into a flight search application. However, a well-developed web scraper can chart the cost of a flight to Boston over time, across a variety of websites, and tell you the best time to buy your ticket. You might be asking: "Isn't data gathering what APIs are for?" Well, APIs can be fantastic, if you find one that suits your purposes. They can provide a convenient stream of well-formatted data from one server to another. You can find an API for many different types of data you might want to use such as Twitter posts or Wikipedia pages. In general, it is preferable to use an API (if one exists), rather than build a bot to get the same data. However, there are several reasons why an API might not exist: You are gathering data across a collection of sites that do not have a cohesive API. The data you want is a fairly small, finite set that the webmaster did not think warranted an API. The source does not have the infrastructure or technical ability to create an API. Even when an API does exist, request volume and rate limits, the types of data, or the format of data that it provides might be insufficient for your purposes. This is where web scraping steps in. With few exceptions, if you can view it in your browser, you can access it via a Python script. If you can access it in a script, you can store it in a database. And if you can store it in a database, you can do virtually anything with that data.

### (b) Building Scrapers

This section focuses on the basic mechanics of web scraping: how to use Python to request information from a web server, how to perform basic handling of the server's response, and how to begin interacting with a website in an automated fashion. By the end, you'll be cruising around the Internet with ease, building scrapers that can hop from one domain to another, gather information, and store that information for later use. To be honest, web scraping is a fantastic field to get into if you want a huge payout for relatively little upfront investment. In all likelihood, 90% of web scraping projects you'll encounter will draw on techniques used in just the next six chapters. This section covers what the general (albeit technically savvy) public tends to think of when they think of "web scrapers": Retrieving HTML data from a domain name Parsing that data for target information Storing the target information Optionally, moving to another page to repeat the process This will give you a solid foundation before moving on to more complex projects in part II. Don't be fooled into thinking that this first section isn't as important as some of the more advanced projects in the second half. You will use nearly all the information in the first half of this book on a daily basis while writing web scrapers.

## II. RELATED WORK

The digital world is growing with a pace that exceeds the speed of any man made fastest prime movers. Here the term growing is used in context to the size of data. At 487bn gigabytes (GB), if the world's rapidly expanding digital content were printed and bound into books it would form a stack that would stretch from Earth to Pluto 10 times. The main contributors to this digital warehouse are social media, government surveillance cameras and plenty of other independent websites which are updated on daily basis such as inventories system of companies, their daily revenues as well as E-Commerce websites that comes up with FMCG's on daily basis. In this digital age, this web data is the most essential resource for any business. The main focus of this paper is to highlight the collection of data through scraping as API's are not available for each and every data source.

Web Monitoring, Scraping and digital forensic is one of the prominent areas in the domain of Big Data and Sentiment Analysis. A number of software products and tools are available in the technology market which are used to guards the network infrastructure and confidential data against cyber threats and attacks. From long time, the monitoring of servers and forensic analysis of network infrastructure is done using packet capturing (PCAP) tools. These activities are performed using PCAP and related tools available in the market which includes open source software as well as commercial products. As far as the fame and usage of the software suites is concerned, the open source market is getting popularity because of the scope of customization and organization specific personalization the software products. In this research paper, an approach is depicted for the fetching and analysis of live data from social media portals and using such approaches the sentiment data analysis can be implemented effectively.

## III. PROPOSED METHODOLOGY AND IMPLEMENTATION

The proposed work focuses on the analysing the web pages. In this work, we have developed working model. By using this methodology web link transform into visual blocks. A visual block is actually segment of webpage. The system is automatic top-down; tag tree independent approach to detect web content structure. Basically, the block-based page content structure is obtained by using python script in BeautifulSoup. Simulation of experimental work shown below

1. Installation of BeautifulSoup and Python
2. Python scripting
3. Execution of script using python
4. Content structure construction

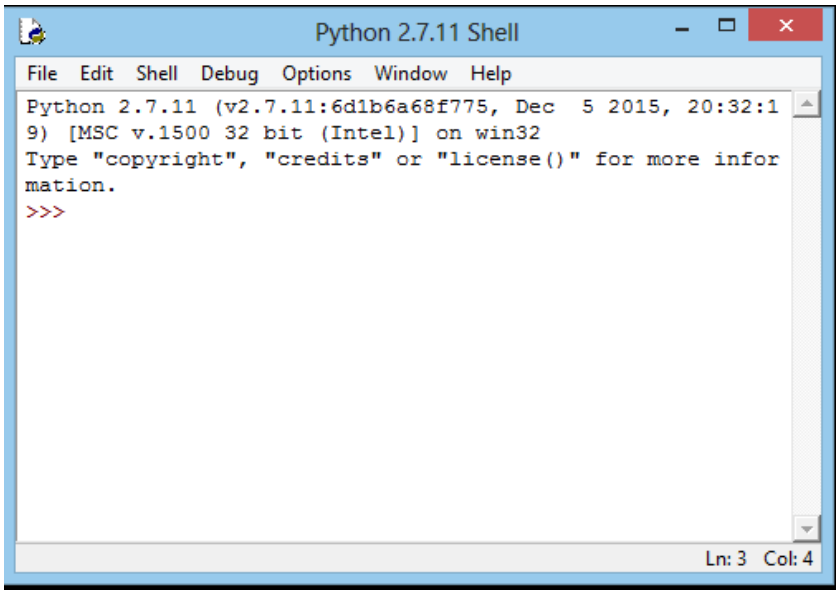


**Figure 1.1 Initialization of Python IDLE (GUI Interface)**

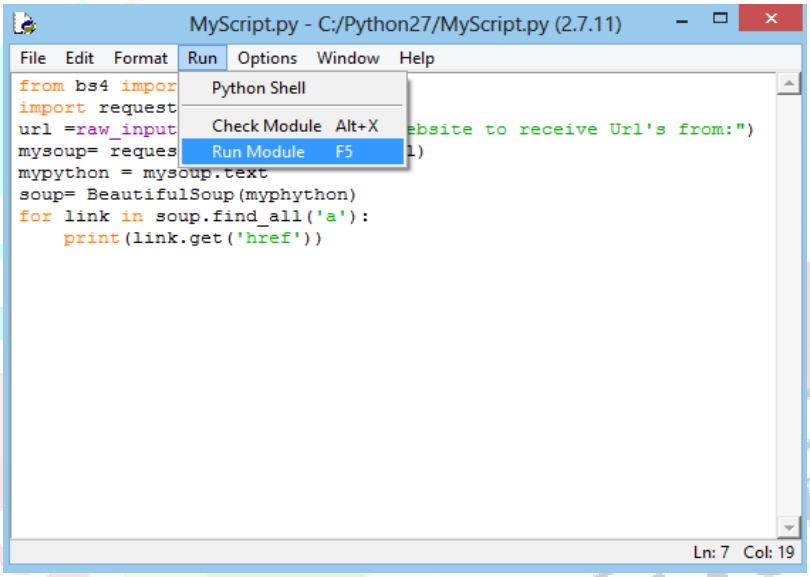


**Figure 1.2 URL Extraction Method**

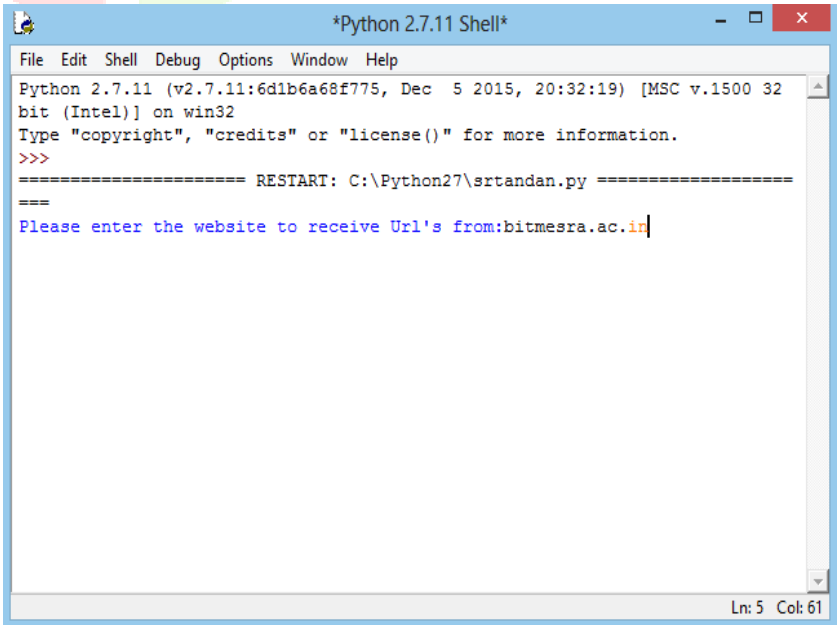


**Figure 1.3 Retrieval Process of Website**

**TABLE 1.1 URL EXTRACTIONS OF VARIOUS WEBSITES**

| S.No | Name of Website | No. of Link Retrieved | Organization |
|---|---|---|---|
| 01 | bitmesra.ac.in | 246 | Birla Institute of Technology, Mesra Ranchi, INDIA |
| 02 | nitttrbpl.ac.in | 82 | National Institute of Technical Teachers Training, Bhopal, INDIA |
| 03 | cvru.ac.in | 230 | Dr C V Raman University, Bilaspur, INDIA |
| 04 | web.mit.edu | 69 | Massachusetts Institute Technology, Cambridge, USA |
| 05 | nasa.gov | 0 | National Aeronautics and Space Administration, USA |
| 06 | isro.gov.in | 148 | Indian Space Research Organization, INDIA |
| 07 | barc.gov.in | 123 | Bhabha Atomic Research Centre, INDIA |
| 08 | fbi.gov | 152 | Federal Bureau of Investigation, USA |
| 09 | bell-labs.com | 108 | Bell Laboratory, USA |
| 10 | tcs.com | 0 | Tata Consultancy Services Ltd, INDIA |
| 11 | infosys.com | 148 | Infosys Consultants Pvt. Ltd, INDIA |
| 12 | iitb.ac.in | 248 | Indian Institute of Technology, Bombay, INDIA |
| 13 | iiitnr.ac.in | 138 | International Institute of Information Technology, Naya Raipur, INDIA |
| 14 | mciindia.org | 139 | Medical Council of India, INDIA |
| 15 | drdo.gov.in | 0 | Defence Research Development Organization, New Delhi, INDIA |
| 16 | bseindia.com | 72 | Bombay Stock Exchange, Bombay, INDIA |
| 17 | aajtak.intoday.in | 776 | Aaj Tak News Channel, Noida, INDIA |

The data given in Table 1.1 are retrieved from various websites. The data given are the numerical values which is number of URL links retrieved. While analyzing the individual website links based on the data retrieved following are the observations.

i. It is a very fruitful technique to obtain whole links.
ii. The website also contains dummy links which are not in use.
iii. The security level of the website can also be evaluated using this.
iv. While analyzing the website like NASA and DRDO (Defence Research Development Organization) links extraction method failed to fetch any link as of highly secured sites.

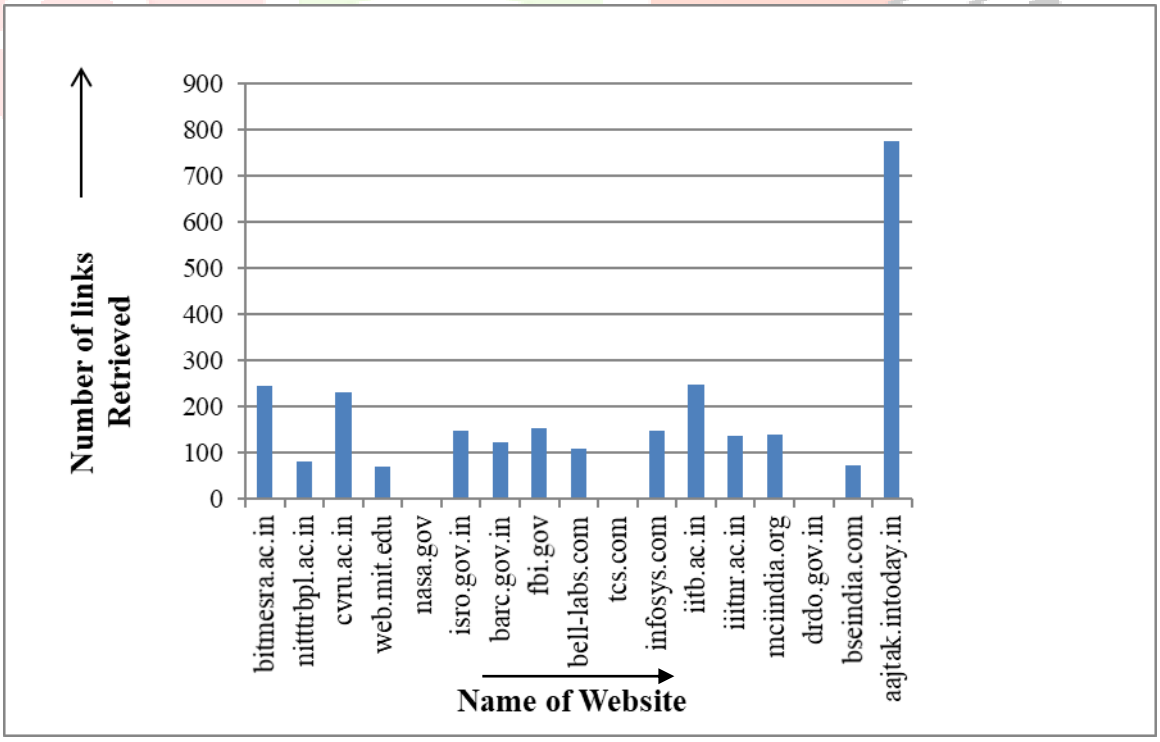


**Figure 1.4 URL Extraction Chart**

## IV. CONCLUSIONS

Web information retrieval is one of the challenging areas of research because web information changes frequently. An increasing use of internet, social media services is the resulting of turning towards analysis of big data. Web information is mostly in an unstructured format. The developed method is useful to retrieve the unstructured data and make it useful. The combination of BeutifulSoup and Python retrieve the web links efficiently. It is useful to analyze the visibility and nature of the website. The method works efficiently while retrieving web contents compared to existing techniques. Advantage of python based scripting method is fast and easy to deal with complex URLs.. The developed method is useful while analyzing security level of web links.

## ACKNOWLEDGMENT

I would like to thank my research guide Mr. S. R. Tandan for his valuable support during research work. I would like to thank all respected faculty members of Department of Computer Science and Engineering, Dr. C V Raman Institute of Science and Technology.

## REFERENCES

[1] Making data on the web useful: scraping
[2] Screen Scraping: Techopedia
[3] Getting Data from the Web: Data Journalism Handbook
[4] urllib2 — extensible library for opening URLs: https://docs.python.org
[5] Beautiful Soup Documentation – www. crummy.com
[6] Crawling the Web, Gautam Pant, Padmini Srinivasan and Filippo Menczer.
[7] Web Crawler: A Review , Md. Abu Kausar , V. S. Dhaka Dept, Sanjeev Kumar Singh
[8] Web Scraping, Wikipedia.
[9] Text Categorization by Fabrizio Sebastiani Dipartimento di Matematica Pura e Applicata Universita di Padova ` 35131 Padova, Italy
[10] Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis.Foundations and trends in information retrieval, 2(1-2), 1-135.
[11] Pang, B., & Lee, L. (2004, July). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of the 42nd annual meeting on Association for Computational Linguistics (p. 271). Association for Computational Linguistics.
[12] Wilson, T., Wiebe, J., & Hoffmann, P. (2005, October). Recognizing contextual polarity in phrase-level sentiment analysis. In Proceedings of the conference on human language technology and empirical methods in natural language processing (pp. 347-354). Association for Computational Linguistics.
[13] Pak, A., & Paroubek, P. (2010, May). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In LREC (Vol. 10, pp. 1320-1326).
[14] Nasukawa, T., & Yi, J. (2003, October). Sentiment analysis: Capturing favorability using natural language processing. In Proceedings of the 2nd international conference on Knowledge capture (pp. 70-77). ACM.
[15] Liu, B. (2010). Sentiment analysis and subjectivity. Handbook of natural language processing, 2, 627-666.
[16] Mullen, T., & Collier, N. (2004, July). Sentiment Analysis using Support Vector Machines with Diverse Information Sources. In EMNLP (Vol. 4, pp. 412-418).
[17] Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. (2011, June). Sentiment analysis of twitter data. In Proceedings of the Workshop on Languages in Social Media (pp. 30-38). Association for Computational Linguistics.
[18] Manu Bansal, "Sentiment Analysis from Social Media Live Feeds Using Unstructured Data Mining" International Journal of Computing and Corporate Research ISSN (Online): 2249-054x Volume 5 Issue 5 September 2015
[19] Rahul Dhawani, Mrudav Shukla, Priyanka Puvar, Bhagirath Prajapati A Novel Approach to Web Scraping Technology International Journal of Advanced Research in Computer Science and Software Engineering Volume 5, Issue 5, MAY 2015 ISSN: 2277 128X
[20] George Foreman (2003), An Extensive Empirical Study of Feature Selection Metrics for Text Classification. Journal of Machine Learning Research, pp.1289–1305.