


Master-thesis

What web objects contained in popular websites are delivered through hyper giants.

Fakultät IV - Elektrotechnik und Informatik
Intelligent Networks / Intelligente Netze (INET)
Research Group of Prof. Anja Feldmann, Ph.D.

Soumya Ranjan Parida
October 17, 2016

Prüfer: Prof. Anja Feldmann, Ph. D.
Betreuer: Ingmar Poese
Juhoon Kim

Die selbständige und eigenhändige Anfertigung versichere ich an Eides Statt.

Berlin, den October 17, 2016 Max Mustermann

Abstract

In a relatively short period of time, the Internet has an amazing impact on almost every facet of our lives. With it, we are able to access new ideas, more information, unlimited possibilities, and a whole new world of communities. In addition, the demand for more and richer content has increased. To fulfill this demand, the Internet has evolved immensely in the last decade. Content become the king. Websites are becoming very rich in content as well as delivering high quality content to their customers. On top of it, introduction of social networking platforms such as Facebook, Twitter; video sharing sites such as Youtube has changed the user interaction with Internet by enabling them to publish their own content and share with each other which led to an exponential growth of Internet traffic.

To adopt this new trend, some companies are following state-of-the-art strategies for distributing their content while offering the best user experience. They have been deploying a large number of scalable and cost effective hosting infrastructure all around the globe. These hosting infrastructure can be composed of a few large data centers, a large number of caches or any combination. Some of these hosting companies handle a major chunk of web traffic, making them the hyper giants of today's Internet. Other websites use the infrastructure of these hyper giants to deliver their content. Such a scenario causes a large amount of traffic to flow through hyper giants, thus creating a huge dependency on them. Hence this thesis focuses on trying to answer the following questions, firstly, whether there any presence of these hyper giants in today's Internet? Secondly, what web objects contained in popular websites are delivered through hyper giants. Finding answers to these research questions help not only to other content distributing companies, content producers, content providers, and ISPs, but also to the research community at large.

In order to conduct this study, 100,000 top ranked websites of Alexa are used. DNS resolution is carried out on all these 100,000 websites along with their embedded links to analyze their second level domains that helps in determining the hosting infrastructure they use. HTTP header analysis is performed on these URLs that helps in finding out the reliance on hyper giants. The experimental results are supported by extensive analysis of data collected from a single vantage point in Germany. The results revealed that there are 26 hyper giants which are collectively contributing to almost 30% percentage of total web traffic on Internet. It is found out that all the hyper giants put together deliver more number of text/HTML files than any other object type.

Contents

List of Figures

List of Tables

Listings

1 Introduction

The Internet is the largest network system in the world and is growing even bigger. Recent study [1] shows that, by 2015 there were more than 3 billion active Internet users in the world. With increase in Internet users and their demand for more and more richer content has led to exponential increase of Internet traffic. High resolution videos, graphic-rich multimedia on-line games, interactive audio and video, high quality audio streaming etc. are contributing to large upsurge in traffic. Consequently, websites having vast and rich content require different strategies to distribute their content all over the world in order to give their users a good quality of experience.

Quality of Experience (QoE) for end users is a key factor in the success of a new Internet service. Response time on web is a really important matter of concern. Longer loading times of websites will result in poor user experience. This will have a severe impact on the digital businesses. It is found out that slow loading websites cost the U.S. e-commerce market more than 500 billion dollar annually. According to e-commerce and conversion statistics, 40% of web users abandon a website if it takes more than 3 seconds to load [2]. There are situations where sometimes, a website receives a huge amount of hits, resulting in the need to handle the peak loads. For this reason, websites host their content using cloud based infrastructure where the required resources can be scaled on-demand. The websites that are highly rich in content, in order to give their users a better experience host their content on content delivery networks. This brings the web content closer to the users, thus reducing the response time. Some other websites use content providers to host their content. Recent studies [3] [4] [5] suggest that some of these hosting infrastructure such as CDNs, Cloud services and Content providers are responsible for a major fraction of Internet traffic, making them hyper giants.

The appearance of hyper giants has influenced the Internet topology. Hyper giants construct peering links with different autonomous systems (ASs). In this way, hyper giants are able to reduce the transit cost of traffic traversing through large ISPs and able to serve content in a faster way. The producers of the content (popular websites) want their content to be delivered to end user in less time for which they rely on hyper giants. Hyper giants build a large infrastructure all around the world to deliver content ensuring a faster response. The websites use these infrastructure to store their content, such as audios ,videos, test/html files etc. Such a scenario causes a large amount of traffic flow from hyper giants. This makes some websites reliant on hyper giants. Therefore these hyper giants can directly impact the way the web objects are delivered in today's Internet. Hence, it is important for the owners of these websites to know the degree to which they rely on hyper giants to deliver their content. It is this symbiosis between these two parties that motivates our work, giving an overview on how far the reach of hyper giants in todays Internet.

By end of this thesis we are able to provide answers for some of the important research questions which can be summarized as follows:

- Identification of hyper giants: A fully automated approach will be purposed to discover hyper giants such as highly distributed content delivery networks, content providers, cloud computing services etc.
- Dependency of content delivery on hyper giants: We quantify the degree of dependency of the popular websites on hyper giants in getting their content delivered.

The remainder of this thesis is structured as follows.

This thesis is separated into 7 chapters.

Chapter 2 starts with overview on the evolution of Internet architecture from early 2000s to current time and the techniques used to analyze the hosting infrastructure.

Chapter 3 discusses the methodology used to identify the hyper giants in today's Internet and the dependency between popular websites on hyper giants.

Chapter 4 discusses on the technologies used for implementing the web crawler. It describes design of the web crawler, followed by the work flow.

Chapter 5 describes the tests conducted following the methods discussed in chapter-3, to extract the results that are required to analyze the hosting infrastructure deployed by the websites.

Chapter 6 summarizes the results to identify the hyper giants and the dependency of popular websites on hyper giants for delivering their content.

Chapter 7 discusses the problems encountered during the thesis work, learnings during this phase, any solutions to overcome the problems encountered. It also summarizes the results and includes possible future work.

2 Background

This chapter gives an overview of the Internet architecture, describes the functionality of each of its elements, with the focus on the role of hyper giants in today's Internet. Different studies related to hyper giants are discussed. Finally, the technical background to Domain name system (DNS) and hypertext transfer protocol (HTTP) that are needed to carried out the required study in this thesis are explained briefly.

2.1 Related studies

The Internet architecture is getting complex with time with introduction of hyper giants. Therefore it is very important to identify them and study their role in Internet. In 2010, Craig Labovitz, then of Arbor Networks [3], first time characterized hyper giant. By placing Google in this list, the author described the hyper giant as, a content provider that makes massive investments in bandwidth, storage, and computing capacity to maximize efficiencies and performance. It is also found that the traffic amount sent by hyper giants is about 30% of the whole amount across the Internet. This concept of hyper giants also aligns with Schmidts [6] assertion which talks about "gang of four" companies which are responsible for the growth and innovation of Internet. Google, Apple, Amazon, and Facebook. In [7], Shavitt and Weinsberg analyzed changes in topological structure, such as betweenness centrality and link density, by focusing on large content providers, also referred to as hyper giants [3] [4]. They create a snapshot of the AS-level graph from late 2006 until early 2011, and then analyze the interconnection trends of the transit and content providers and their implications for the Internet ecosystem. Shavitt and Weinsberg proved that large content providers like Google, Yahoo!, Microsoft, Facebook, and Amazon have increased their connectivity degree during the observed period and are becoming key players in the Internet ecosystem, strengthening the idea that the Internet is becoming flatter. From this analysis, it was found that the structure of the Internet topology has changed from a hierarchical to a flat structure. This is because large content providers construct links with a lot of small ISPs. Unfortunately all the above studies are carried out by accessing the Internet traffic passing through different ASs, not from end user.

2.2 Evolution of Internet Architecture and Rise of hyper giants

The Internet architecture implemented until the early 2000s was based on a multi-tier hierarchical structure. Tier-1 ISPs were on top of the hierarchy followed by the Tier-2 regional ISPs and the Access ISPs at the lower part of the hierarchy connecting the end users.Tier-1 were highly connected to other ISPs and offered transit services to other ISPs in lower layers. Content was distributed through Access ISPs or, in the best cases, through ISPs located at advantageous points. Traffic flows were required to go up and then down in the hierarchy to reach end users shown in Figure 1. Among different network operators, Internet traffic was exchanged at different IP exchange points according to agreements between different layer players where the dissymmetry in traffic was compensated [8] [9] [10].

It can be observed from Figure 1, Tier-1 operators are at top of this hierarchy. Tier-1 network is normally a transit free network that peers with every other Tier-1 network operators. Hence they treat each other equally. While most of the Tier-1 operators offers global coverage, there are some which restricted geographically. Tier-2 ISP is a customer of Tier-1 operators and pays to Tier-1 operators for connectivity to rest of the Internet.

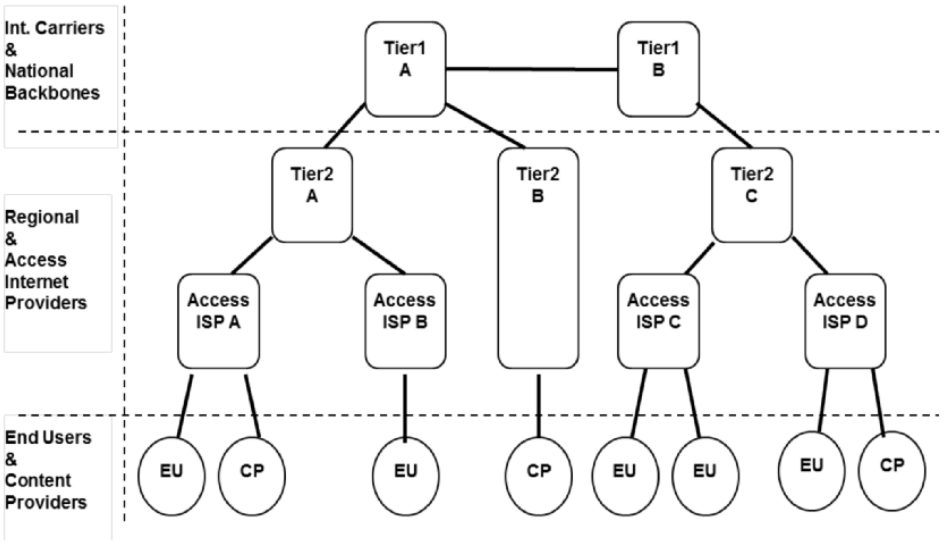

Figure 1: Traditional Hierarchic Internet Structure

They connect to one or more Tier-1 operators, possibly to other Tier-2 operators. Below to the Tier-2 ISPs are local ISPs and Tier-3 ISPs which are connected to the end users. These network operators are customers of the higher Tier ISPs and pays transit fees to connect to the rest of the Internet. In Figure 1, it can be observed that the content hosted in Access ISP A must be passed through different level of hierarchical structure to reach Access ISP D. The content from Access ISP A has to transmit up first to Tier-2 A network operator which forwards content to Tier-1 A operator. Tier-1 A operator and Tier-1 B operator peer with each operator. Tier-1 B operator then transmit it down to Tier-2 C operator which forwards it to Access ISP D operator. Hence the content has to transmit up and down to reach the destination.

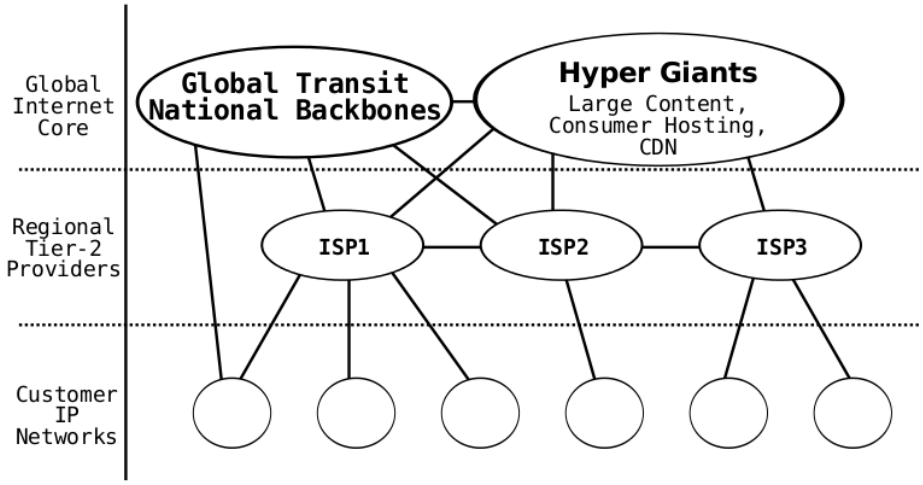

Figure 2: Modern Internet Structure

But with the time, the Internet architecture has changed. Researchers found that now nobody has control over Internet, instead each ISP has control over its network and depend upon the network connected with it. Even during last decade the old pyramidal structure of Internet architecture shown in Figure 1, has been bypassed by big content providers, such as Google, Facebook, Amazon or Yahoo!, and content delivery network operators, such as Akamai. As a result now Internet's backbone has a flatter structure where there are few autonomous systems are playing major role in delivering content. They are connected to each other and have a big footprint by establishing small data centers all over the world. This helps them to get as close as possible to the access networks used by their customers, bypassing intermediate Internet service providers. The trend towards flatter network architectures can also be found in the area of access networks. In Figure 2, it can be observed that content hosted in ISP1 "bypass the network" and easily reach the end users thanks to direct and faster connections. This bypass means improved in network performance in delivering content to the end user and, at the same time, it optimizes network resources by using cache servers near to the end user. The researchers termed these infrastructure providers as "hyper giants" which include large content providers, such as Google and Yahoo, as well as highly distributed CDNs, like Akamai. Hyper giants construct peering links with different autonomous systems (ASs). In this way, they are able to reduce the transit cost of traffic traversing through large ISPs and able to serve content in a faster way.

2.3 Domain name System (DNS)

Domain name system (DNS) is used to translate IP address to corresponding host names. Internally it is maintaining a hierarchal structure of domains. The administration of domains is divided into different zones. The zone information is distributed using authoritative name servers. The top most level of DNS starts with root zone. The root zone of the DNS system is centrally administered and serves its zone information via a collection of root servers. The root servers delegates responsibility for specific zones to some other authoritative name servers which in turn divided responsibility with other authoritative name server. At the end, each site is responsible for its own zone and keep maintain its own database of authoritative name server. The information about a particular domain of a zone is kept in Resource Records (RRs) which specify the class and type of the record as well as the data describing it. Multiple RRs with the same name, type and class are called a resource record set (RRset) [11] [12].

To resolve a host name to IP address, the procedure starts with the end user's stub resolver queries to local name server called caching server. If the caching server can not able to resolve it, it redirects the query to authoritative name server of the domain. If resolver does not know how to contact the corresponding authoritative name server of the domain, it redirects the query to root name server. The root name server again refers the resolver to the authoritative name server responsible for the domain just below the root server. This procedure continues till resolver is able to resolve the domain properly.

A top-level domain (TLD) is one of the domains at the highest level in the hierarchical Domain Name System of the Internet [13]. The top-level domain names are installed in the root zone of the name space. For all domains in lower levels, it is the last part of the domain name, that is, the last label of a fully qualified domain name. For example, in the domain name www.example.com, the top-level domain is ".com". In the Domain Name System (DNS) hierarchy, a second-level domain (SLD) is a domain that is directly below

a top-level domain (TLD). Second level domain normally refers to the organization name that registered the domain name with a domain name registrar.

Listing 1 shows the DNS reply by the resolver when querying the host name "www.bmw.com".

Listing 1: DNS Reply for a host using dig command line tool

```
; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.bmw.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60134
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.bmw.com.                   IN      A

;; ANSWER SECTION:
www.bmw.com.            3600    IN      CNAME   cn-www.bmw.com.edgesuite.net.
cn-www.bmw.com.edgesuite.net. 3600 IN   CNAME   a1586.b.akamai.net.
a1586.b.akamai.net.     20      IN      A       104.121.76.64
a1586.b.akamai.net.     20      IN      A       104.121.76.49

;; Query time: 22 msec
;; SERVER: 127.0.1.1#53(127.0.1.1)
;; WHEN: Fri Oct 14 09:46:54 CEST 2016
;; MSG SIZE  rcvd: 143
```

There are many types of resource records, the most common being A (which gives an IPv4 address for a host name), AAAA (which gives an IPv6 address), MX (which sets the location of a mail server), CNAME (or canonical name, which maps one NAME to another), and TXT (which can include any arbitrary text). In Listing 1, it can be observed that, the host name is "www.bmw.com". The answer section contain a chain of CNAMEs "cn-www.bmw.com.edgesuite.net" and "a1586.b.akamai.net" which resolve into same ARecord set (RRset) with different IP addresses 104.121.76.64 and 104.121.76.49. The host infrastructure correspond to final IP address is normally refer to as ARecord name. In Listing 1, "a1586.b.akamai.net" is the ARecord name for both the IP addresses 104.121.76.64 and 104.121.76.49.

2.4 Hyper text transfer protocol(HTTP)

Hyper text transfer protocol (HTTP) [14] [15] is an application layer protocol mainly used as defector standard to transport content in world wide web. HTTP works on top of the TCP/IP protocol and follows the client server architecture via request-response communication procedure. It allows end-users to request, modify, add or delete resources identified by uniform resource identifiers (URIs).

HTTP message consists of HTTP header which shows the meaning of message and HTTP body which is actual message. HTTP message can be a request message or response message. Both types of message consist of a start-line, zero or more header fields (also known as "headers"), an empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields, and possibly a message-body.

The HTTP client sends a request message to server. There are different types of methods used in HTTP request message like GET, HEAD, POST , PUT ,DELETE, TRACE etc. But in this thesis we have used extensible GET method and the HEAD method. HTTP defines these methods to indicate the desired action to be performed on the identified resource.The GET method is used to retrieve information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data. The HEAD method request for a response identical to that of a GET request but without the response body.The head method transfers the status line and the header section only. The POST method requests the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request-URI in the Request-Line. The actual function performed by the POST method is determined by the server and is usually dependent on the Request-URI. The PUT method requests that the enclosed entity be stored under the supplied Request-URI. If Request-URI refers to an existing resource, then the enclosed entity is considered as the modified resource. The DELETE method requests that the origin server delete the resource identified by the Request-URI. TRACE allows the client to see what is being received at the other end of the request chain and use that data for testing or diagnostic information.

Listing 2: HTTP Get Request message

```
GET / HTTP/1.1
Host: www.example.com
User-Agent: Mozilla/5.0 [...]
Accept: text/html [...]
Accept-Language: en-us
Accept-Encoding: gzip,deflate
Connection: Keep-alive
```

Listing 2 shows HTTP Get request message sample for host "www.example.com". The introductory line in an HTTP request message consists of method, a server-side path, and the HTTP version in use.

Listing 3: HTTP Response message

```
HTTP/1.1 200 OK
Accept-Ranges: bytes
Content-Type: text/html
Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Wed,22 Jul 2009 19:15:56 GMT
Content-Length: 88
<!doctype html>
<html>
[...]
```

Listing 3 shows HTTP response message sample for host “www.example.com”. The introductory line in an HTTP response starts out with the HTTP version in use, followed by a standardized three-digit status code and a textual status description. The status code tells the requester about the success of the query or indicates the reason of an error.

Both request and response messages are followed by multiple header lines. Some header information are valid for request, some are for response and some are valid in both the ways. Since HTTP1.1, the host header is mandatory for request messages. The meta information encompasses information about the file type, the character set in use, preferred language etc. HTTP also allows server to set cookies in client side which help the server to track client requests.

2.5 Conclusion

Within last decade the Internet architecture changed vastly due to the introduction of hyper giants which can be highly distributed CDNs, cloud computing CDNs etc. Todays Internet traffic is dominated by HTTP traffic. Furthermore, to deliver the content fast, DNS protocol is used as the load balancing mechanism by these big hyper giants. DNS acts as entry point for the end users to connect to hosting infrastructure. Hence it is ideal to carry out the study to analyze the hyper giants using the DNS.

3 Methodology

This section discusses the methods to identify the presence of hyper giants in today's Internet and to find out the inter dependency between hyper giants and popular websites. It is commonly known that most of the websites use some kind of hosting infrastructure for distributing their Internet content. This hosting infrastructures can be CDN, cloud computing infrastructure or content providers. Hosting infrastructures were designed to transport and cache large amounts of Internet content, such as HTML code, JavaScript, large files, images, audio, and video. The most appropriate way to identify whether a website is using a hosting infrastructure service is to inspect its HTML code looking for URLs linked to hosting infrastructure. The redirection of a link to an external hosting company will be clear evidence that a particular website is using a hosting infrastructure.

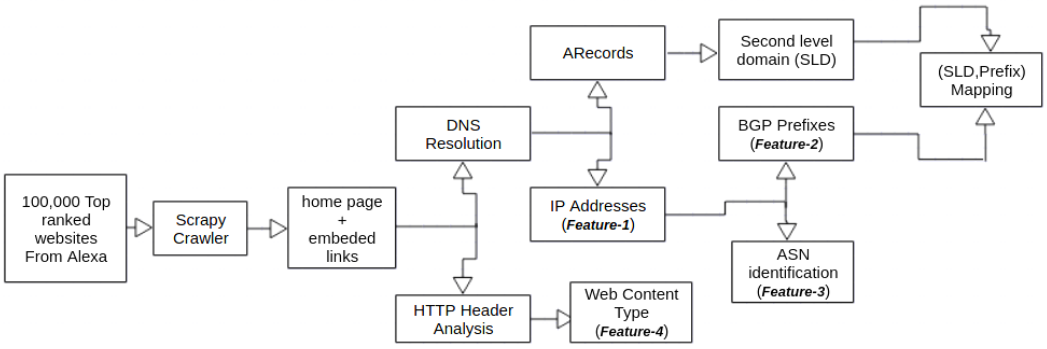

Figure 3: High level approach Part-1

To perform this task, as shown in Figure 3, 100,000 top ranked websites of Alexa [16] will be crawled using scrapy crawler [18] [17] and their HTML codes are inspected to retrieve all their embedded URLs. Then these URLs will be used to find out the link redirection and subsequently the hosting infrastructure. DNS resolution is the known solution to find out the link redirection. It will resolve the web URL into single or multiple IP addresses and their corresponding ARecord names. ARecord names show the hosting infrastructure linked to corresponding IP address. Sometimes hosting companies use different naming convention to represent their ARecords. As an example, a1586.b.akamai.net. is the ARecord name when www.bmw.com is resolved which resolve into same ARecord set (RRset) with different IP addresses 104.121.76.64 and 104.121.76.49. Therefore, to get the desired hosting infrastructure ,the second level domain (SLD) of each ARecord is the best suited option.

The set of IP addresses for a particular SLD shows the degree to which the corresponding hosting infrastructure is connected with different web URLs. The number of BGP prefixes show the network footprint and the number of ASNs show how the infrastructure is distributed. Therefore, to identify the hyper giants, the natural choice for the features to consider are IP addresses, AS Numbers and the BGP prefixes.

Hyper giants build a large infrastructure all around the world to deliver content ensuring a faster response. The websites use these infrastructure to store their content, such as audios ,videos, test/html files etc. Therefore, any kind of disruption in serving these contents from hyper giants will impact the websites which shows the interdependency of hyper giants with the web sites. To know which content type will be impacted more, the

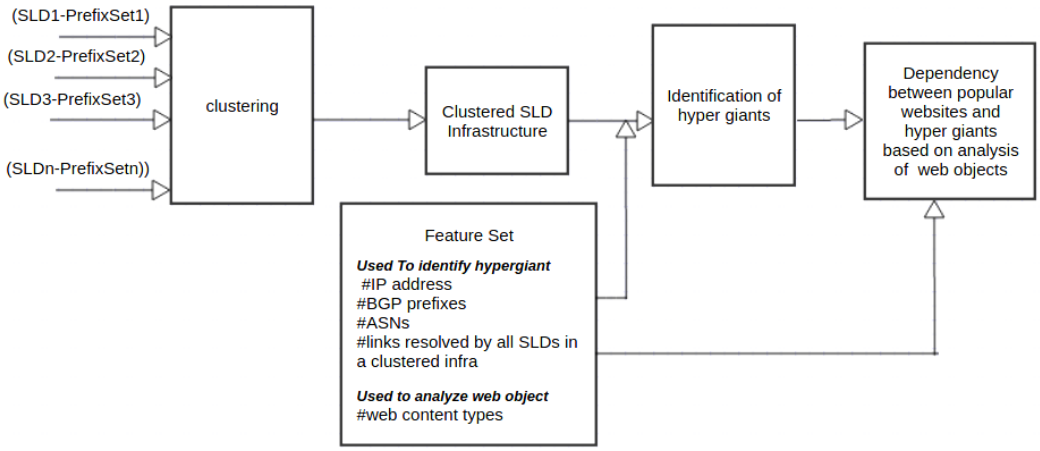

Figure 4: High level approach Part-2

type of content can be studied. To evaluate the type of content,as shown in Figure 4, HTTP header analysis will be carried out on 100000 top ranked websites of Alexa and their corresponding embedded URLs. The web object type retrieved from the HTTP header information determines the content type such as text/html, image, video, audio etc., which will be used to analyze the dependency of websites on hyper giants.

3.1 Identification of hyper giants

To find out the hyper giants in Internet,the hosting infrastructure which are serving popular websites will be observed. This can be determined by analyzing different features of hosting infrastructure such as IP addresses, BGP prefixes, ASN numbers etc. The methods to find out these features will be discussed in the following subsections.

3.1.1 Hosting infrastructure to BGP Prefix Mapping

Listing 4: RIPE RIS bgp prefixes using whois command

```
% This is RIPE NCC's Routing Information Service
% whois gateway to collected BGP Routing Tables, version2.0
% IPv4 or IPv6 address to origin prefix match
%
% For more information visit http://www.ripe.net/ris/riswhois.html
%
% Connected to backend ris-whois05.ripe.net

route:          104.64.0.0/10
origin:         AS35994
descr:          AKAMAI-AS - Akamai Technologies, Inc., US
lastupd-frst:   2016-10-04 11:53Z  80.249.209.167@rrc03
lastupd-last:   2016-10-13 20:44Z  193.203.0.130@rrc05
seen-at:        rrc00,rrc01,rrc03,rrc04,rrc05,rrc06,rrc07,rrc10,rrc11,rrc12,
                rrc13,rrc14,rrc15,rrc16,rrc18,rrc19,rrc20,rrc21
num-rispeers:   160
source:         RISWHOIS

route:          104.121.76.0/24
origin:         AS20940
descr:          AKAMAI-ASN1 , US
lastupd-frst:   2016-07-06 10:09Z  198.32.124.146@rrc16
lastupd-last:   2016-10-13 19:37Z  37.49.237.46@rrc21
seen-at:        rrc00,rrc01,rrc03,rrc04,rrc05,rrc06,rrc07,rrc10,rrc11,rrc12,
                rrc13,rrc14,rrc15,rrc16,rrc18,rrc19,rrc20,rrc21
num-rispeers:   163
source:         RISWHOIS
}
```

From DNS resolution, the hosting infrastructure and corresponding IP addresses are determined. To find out BGP prefix routes of a particular IP address, BGP routing information from RIPE RIS [23] will be used. In listing 4, "route" shows the BGP prefixes of a IP address. From the Listing-2, this can be observed that the IP address 104.121.76.73 can be routed via two different prefixes 104.64.0.0/10 and 104.121.76.0/24. So both the prefixes for this IP will be considered for further analysis. This procedure will be carried out for each IP addresses of hosting infrastructure which will result in (hosting infrastructure, prefixes) mapping.

3.2 Clustering

From the initial analysis on the data collected for analyzing hosting infrastructure, it is found out that some hosting infrastructure are administered by the same company. For example, both the hosting infrastructure akamaiedge.net and akamai.net are administered by the same company, Akamai. Similarly google.com, googleusercontent.com, google-hosted.com and googledomains.com are administered by same company, Google. Generally companies use different ARecord names depending on the type of service they offer and the geographical location from where they offer the services. It is observed that Ama-

zon uses two different ARecord names, namely amazonaws.us-east-1.elb.amazonaws.com, eu-west-1.elb.amazonaws.com for its customers from two geographical locations. It has to be analyzed, whether these ARecord names are pointing to same infrastructure or different. If they are pointing to the same infrastructure, they should be considered as one, else separately. This can be identified by using clustering algorithm by analyzing the BGP prefixes they share.

The clustering algorithm takes a two step approach. In the first step, the prefixes of the hosting infrastructure will be aggregated into set of parent prefixes as explained in 3.2.1. In the subsequent step, the parent prefixes of the hosting infrastructure will be compared with each of the remaining infrastructure for clustering as explained in section 3.2.2.

3.2.1 Prefix Aggregation of hosting infrastructure

From the section 3.1.2, hosting infrastructure mapped to corresponding BGP prefixes are collected. But in this mapping, it is observed that some of the BGP prefixes are subset of other BGP prefixes. For example, googledomains.com has the prefix set '216.239.32.0/19', '216.239.32.0/24', '216.239.34.0/24', '216.239.36.0/24' and '216.239.38.0/24'. Here the prefixes '216.239.32.0/24', '216.239.34.0/24', '216.239.36.0/24', '216.239.38.0/24' are subnet of prefix '216.239.32.0/19'. Hence it is ideal to aggregate them. This way all the child prefixes in each mapping are aggregated to their parent prefixes. By end of this method, each hosting infrastructure will be mapped to a set of parent prefixes. For example, the hosting infrastructure googledomains.com is mapping to the parent prefix '216.239.32.0/19', as all child prefixes got aggregated into '216.239.32.0/19' parent prefix. This procedure is repeated for all the other infrastructure mappings, which generates a complete list of hosting infrastructure to their corresponding aggregated BGP prefixes.These mapping are sorted in the decreasing order of links hosting by each of these hosing infrastructure.The motivation behind doing this, is to allow the merging of smaller infrastructure into the larger one depending on their similarity factor (sim(s1, s2)) which is discussed in next section.

3.2.2 Evaluation of Similarity between two hosting infrastructure

The possibility of clustering any two hosting infrastructures is determined by analyzing the similarity between their aggregated prefixes. This can be done by verifying if one prefix is a parent to the other. While comparing two hosting infrastructures, if any of the hosting infrastructure contain child prefix of other, then the child prefix will be replaced with the parent prefix. This will make two prefix sets homogeneous, thus helping in evaluating the degree of the similarity between them. For comparing two hosting infrastructures, the similarity equation is defined as follows,

The similarity factor sim(s1,s2) is defined as follows,

$$sim(s1, s2) = \frac{|s1 \cap s2|}{|s2|} \tag{1}$$

where s1,s2 are the bgp prefix sets.

If the similarity between two prefix sets are greater than equal to the 70%, then both the hosting infrastructure will be clustered together making them a single clustered infrastructure. It is assumption that, s2 can be clustered with s2, provided 70% of s2's prefixes are from that of s1's. This procedure will be repeated for all the hosting infrastructure. If two infrastructure has similarity factor greater than equal to 0.7, then the child hosting infrastructure will be merged into the parent hosting infrastructure and will be excluded from further comparisons. For example 'googleusercontent.com' matched with 'google.com' with similarity factor 1. This does not mean that both these hosting infrastructure has the same set of prefixes, but it means that, 'googleusercontent.com' shares 100% of its prefixes with 'google.com'. Once this is matched, 'googleusercontent.com' will not be available for any further similarity matching with any other hosting infrastructures. It is assumed that a similarity index of grater than equal to 0.7 served as a good measure to determine the degree of similarity between two hosting infrastructure which can be considered as future work for extensive analysis.A sample example is explained in Listing 5,

Listing 5: Sample example of clustering procedure

```
Let (hosting infrastructure,Prefix set)s are,

(SLD1,[10.0.0.1/24,12.0.0.0/16,192.168.3.0/24])
(SLD2,[192.168.0.0/16,10.0.0.0/16,12.0.0.0/24])
(SLD3,[3.0.0.0/8,12.0.0.0/24,192.168.3.8/24,5.0.0.0/16])
(SLD4,[4.0.0.0/24,6.0.0.0/24,10.0.0.0/16])
(SLD5,[6.0.0.1/16,10.0.0.0/8])

After the clustering procedure, SLD2 and SLD3 will clustered
with SLD1 creating (SLD1,(SLD2,SLD3))mapping and SLD5 will be
clustered with SLD4 to create (SLD4,SLD5) mapping.
```

3.2.3 Evaluation of hyper giants

Once the clustering procedure is performed, a list of parent hosting infrastructure to child hosting infrastructure are available for further analysis. To find out hyper giants presence, the features of each clustered hosting infrastructures will be analyzed. The features include number of links, number of IP addresses, number of BGP prefixes, number of AS numbers. To find out each of these features for a clustered hosting infrastructure, corresponding child hosting infrastructures will be considered. Hence total number of links for a clustered hosting infrastructure will be the sum of all the links served by all child hosting infrastructures. To find out number of BGP prefixes,the set of two prefix sets will be taken. This will result the unique BGP prefixes. Similarly to get the AS numbers, the set of two prefix sets will be taken.The sample example can be found in Listing 6.

Listing 6: Sample example to generate feature set

```
Let ,
p = numner of links served by SLD1
q = number of links served by SLD2
r = number of links served by SLD3

and mapping is , p -> (q,r)
        which means SLD2 and SLD3 are clustered with SLD1
        then number of links of the clustered hosting
        infrastructure is p + q + r
        (Links are counted irrespective of their repetitiveness in other SLDs)

x = numner of IP addresses served by SLD1
y = number of IP addresses served by SLD2
z = number of IP addresses served by SLD3

and mapping is , x -> (y,z)
        which means SLD2 and SLD3 are clustered with SLD1
        then number of IP addresses of the clustered hosting
        infrastructure is set(x,y,z)
        (Non repetitive IP addresses)

Let ,
i = numner of bgp prefixes served by SLD1
j = number of bgp prefixes served by SLD2
k = number of bgp prefixes served by SLD3

and mapping is , i -> (j,k)
        which means SLD2 and SLD3 are clustered with SLD1
        then number of BGP prefixes of the clustered hosting
        infrastructure is set(i,j,k)
        (Non repetitive bgp prefixes)

Let ,
a = number of ASs served by SLD1
b = number of ASs prefixes served by SLD2
c = number of ASs prefixes served by SLD3

and mapping is , a -> (b,c)
        which means SLD2 and SLD3 are clustered with SLD1
        then number of ASNs of the clustered hosting
        infrastructure is set(a,b,c)
        (Non repetitive ASNs)
```

The features of clustered hosting infrastructure will be analyzed further to find out the hyper giants.

To identify the hyper giants, the hosting infrastructures with unique behavior will be separated from the others. To perform this, two main features will be taken, number of links and number of IP addresses. k-means algorithm will be performed on clustered hosting

infrastructure to partition the clustered hosting infrastructure in up to k clusters .The cluster co-efficient value k is chosen as 10.Clusters whose features have high values will be clustered together. On the other hand, smaller infrastructures that use very few links and IP addresses are not sufficiently different, and therefore, can be found in the same cluster. To identify the hyper giants, the clustered hosting infrastructure which clustered separately than the most of the other clustered hosting infrastructure will be considered.

3.3 Web objects delivered from hyper giants to popular web sites

To find out different web objects delivered from hyper giants to popular web sites, HTTP header information for all URLs corresponding to each clustered hosting infrastructure is analyzed. From header information, the content type of each object can be extracted. This information is useful to determine what type of web objects are served by each hyper giant.

3.4 Conclusion

This section discussed about the methods to identify the presence of hyper giants in today's Internet and to find out the inter dependency between hyper giants and popular websites. To execute both the methods,IP addresses, AS numbers, BGP prefixes are used as features for categorical analysis of clustered infrastructure. The section provided the procedures to find out each of the features for hosting infrastructures. At the end, HTTP header information is used to determine the type of web objects served by each hyper giant.

4 Implementation

This chapter discusses the implementation aspects of web crawler. It explains the choice of programming languages, development tools used to implement the crawler engine. Furthermore,emphasis is given on explaining the internal architecture behind the crawler engine,along with the details of functionality performed by each module. Focus is also given on discussing the overall work flow of the crawler engine.

4.1 Choice of programming language

Before going for implementation, it is important to know that the whole process involve two major parts. In the first step, identification of hyper giants will be performed using different features like number of links, number of IP addresses, BGP prefixes etc. In the second step the dependency between hyper giants and popular websites will be discussed based on number of web objects delivered from hyper giants to popular websites. Hence a programming language which is powerful in development as well as data analysis will be chosen. So that any kind of dependency between two steps can be handled easily. Again language should be free, open and usable by anyone who wishes to run this crawler implementation in future. Moreover the language should have good number of open libraries, big community and should have a lot of online documentation. For this purpose python language is chosen for the implementation as it fulfills all necessary requirement. For data analysis pyspark (Python version of spark) will be used.

4.2 Development Tools

For development, it is important to choose correct IDE which allows to write code in less time with minimum effort.Along with this it helps in code completion, syntax highlighting, refactoring. For this purpose "sublime editor" which is free and easy to write python code is chosen.

Now second most important aspect is to choose python web crawling tool which will suit the requirement and can be used in future. There are couple of famous web crawling tools available like urlib, beautiful soup, scrapy etc. But Python Scrapy is the best out there. Scrapy crawling is faster than any other platforms, since it uses asynchronous operations (on top of Twisted). Scrapy has better and faster support for parsing (x)html on top of libxml2. Scrapy is a mature framework with full unicode, redirection handling, gzipped responses, odd encodings, integrated HTTP cache etc. Again it is open source having a big community and documentation.

There are lot of different machine learning tools available but as it is expected that resultant data will be order of some gigabytes, it is better to use that tool which can process data faster.Python and R are popular languages for data analysis due to the large number of modules or packages that are readily available. But traditional uses of these tools are often limiting, as they process data on a single machine where the movement of data becomes time consuming, the analysis requires sampling (which often does not accurately represent the data), and moving from development to production environments requires extensive re-engineering. Spark provides a powerful, unified engine that is both fast and easy to use. Hence Pyspark will be used for initial data analysis and further analysis will

be carried using python and RStudio.

Apart from above tools, some other tools used for the implementation. The following are the list of important softwares and tools used.

- dnspython:it is a DNS toolkit for python.This is used in the code to get the A records of hosts.
- pygeoip :The libarary is used to get the ASN numbers associated with IP addresses.This library is based on Maxmind's GeoIP C API.
- urlparse :this is used to convert a relative url to an absolute url.
- Public suffix List :This is the collection of all registered host names given by all internet users.The Public Suffix List is an initiative of Mozilla, but is maintained as a community resource.It is available for use in any software, but was originally created to meet the needs of browser manufacturers.In our code we use it to get second level domain from a host name.The "effective_tld_namesdat" is the file which is free downloadable from their site.
- IPython :IPython notebook is used for pyspark code writing and execution.
- matplotlib :Python library used to for making different graphs used in this thesis.
- pygal :Pygal is also another python library which we used to make graphs.

4.3 Design of Crawler Engine

In this section,the internal architecture behind the crawler engine will be discussed.The design process involves two parts,first one is the crawler engine which takes 100,000 top ranked websites of Alexa as input,crawl the websites, return the result file as output from the engine and second part is processing of result file using data processing engine which internally use "pyspark" to clean up the crawled data.

4.3.1 Crawler Engine

In this section, the internal architecture of the crawler engine will be discussed. The main objective of crawler engine is to take input data in the format of text file which contain the top 100,000 top ranked websites of Alexa. Then divide this master domain list into multiple sub domain lists with equal weighted websites and process each sub domain list using separate crawler instances. Each crawler instance work independently and store all website information like HTTP header information, DNS resolution analysis etc. in result file. The core of this cralwer engine is "Scrapy" which is used for crawling the website links. In the following sections the architecture of scrapy framework, the main data types of scrapy framework used for implementation will be discussed. In the subsequent section, the overall work flow of the crawler engine will be discussed.

4.3.2 Scrapy Framework

Scrapy framework is one of top ranked open source project, a fast web crawling framework, used to crawl websites and extract structured data from their pages. It provide option for

both focused and broad crawling.In case of focused crawler scrapy crawls a specific domain while in case of broad crawling a large (potentially unlimited) number of domains can be crawled. Hence for the implementation broad crawling will be used. As can be seen from Figure 5 [20],the main components of the framework contain scrapy engine, scheduler, downloaders, spiders, item pipeline, downloader middlewares.

- Scrapy Engine : The engine is responsible for controlling the data flow between all components of the system, and triggering events when certain actions occur.
- Scheduler : The Scheduler receives requests from the engine and enqueues them for feeding them later (also to the engine) when the engine requests them.
- Downloader : The Downloader is responsible for fetching web pages and feeding them to the engine which, in turn, feeds them to the spiders.
- Spiders : Spiders use for to parse responses and extract items from them or additional URLs (requests) to follow.
- Item Pipeline : The Item Pipeline is responsible for processing the items once they have been extracted (or scraped) by the spiders. Typical tasks include cleansing, validation and persistence (like storing the item in a database). For more information see Item Pipeline.
- Downloader middlewares : Downloader middlewares are specific hooks that sit between the Engine and the Downloader and process requests when they pass from the Engine to the Downloader, and responses that pass from Downloader to the Engine.

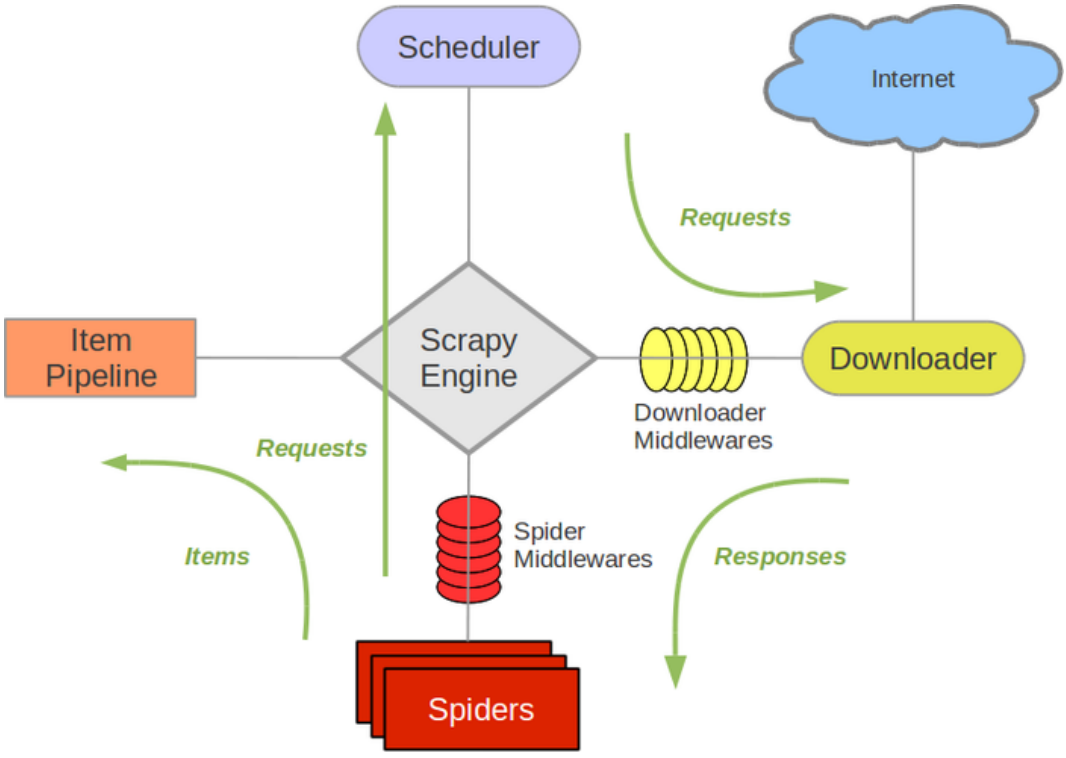

Figure 5: Scrapy Architecture

4.3.3 Scrapy data types

Scrapy uses some specific classes and strings which are used to crawl data. These are written in python and are open source. Scrapy also use twisted framework for multi threading. Some of the classes which are going to be used for implementation, are as follows,

1. Spider :Spiders are the classes used to crawl single or multiple domains using different methods like start_requests(),parse().
    - start_requests() :This scrapy method is used to pass domains to parse method for further crawling process.
    - parse() :he parse method is in charge of processing the response and returning scraped data and/or more URLs to follow. Other Requests callbacks have the same requirements as the Spider class.
2. Twisted Framework:Twisted supports an abstraction over raw threads — using a thread as a deferred source. Thus, a deferred is returned immediately, which will receive a value when the thread finishes. Callbacks can be attached which will run in the main thread, thus alleviating the need for complex locking solutions. A prime example of such usage, which comes from Twisted's support libraries, is using this model to call into databases. The database call itself happens on a foreign thread, but the analysis of the result happens in the main thread.
3. Requests objects:Typically, Request objects are generated in the spiders and pass across the system until they reach the Downloader, which executes the request.A Request object represents an HTTP request, which is usually generated in the Spider and executed by the Downloader, and thus generating a Response.
4. Response objects :Request object returns a Response object which travels back to the spider that issued the request.A Response object represents an HTTP response, which is usually downloaded (by the Downloader) and fed to the Spiders for processing.

4.4 Work flow

In this section the overall work flow of the crawler engine will be discussed. The crawler engine is used to perform multiple operations. It take input data in the format of text file which contain the top 100,000 top ranked websites of Alexa. Then divide this master domain list into multiple sub domain lists with equal weighted websites and process each sub domain list using separate crawler instances. Each crawler instance work independently and store all website information like HTTP header information, DNS resolution analysis etc. in result file. The core of this cralwer engine is "Scrapy" which is used for crawling the website links. The main focus is to create multiple instances of crawler engine. As scrapy is a memory greedy tool,after intensive testing it is decided to create 20 parallel threads which work independently. Each instance of the crawler will take separate input file which contain 5000 domains. Master domain list should be divided properly so that all the crawlers will complete their crawling in almost same time. Hence multiple sublists with equal weight will be created where weight refers to the rank of the websites provided by Alexa. The master domain list contain the website links and the rank of the websites given by Alexa. So after division the first sublist will contain the websites of 1st rank, 21st rank, ..., 99981st rank, second list will contain 2nd rank, 22nd rank, ..., 99982nd rank

websites and so forth for other instances.After the division each sublist will contain 5000 websites with nearly equal weight of website ranks. Each sublist will be sent as input to multiple instances of crawler shown in Figure 6.

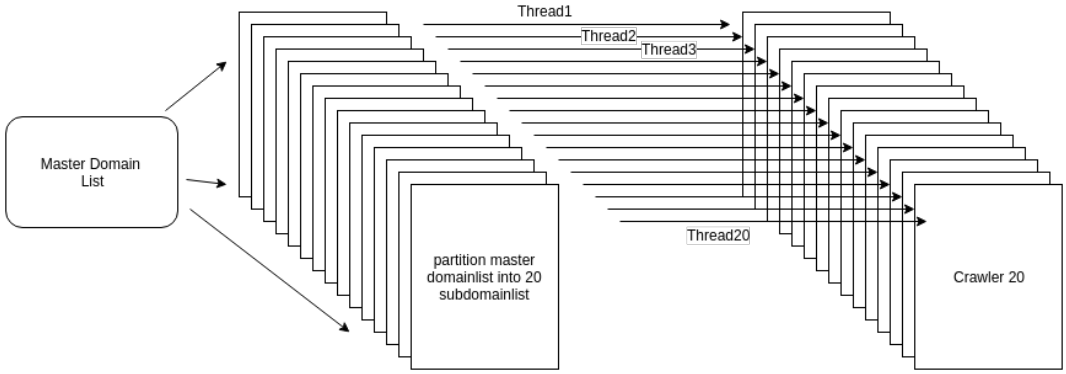

Figure 6: Pre Crawling Step

In the subsequent step each crawler instance will take 5000 domains and parse one by one domain. For each URL crawler engine downloads the corresponding web page, extract the linked URLs, and check each url to see whether the extracted url is a fresh url which has not already been seen . With this architecture a very large number of independent crawls of the white listed domains obtained from Alexa can be crawled. Along with this, the crawler engine also extract HTTP header information, ARecord details ,ASN details.

4.5 Conclusion

This section discusses the softwares used to implement the required functionality. The motivation behind choosing different languages, software tools used during the implementation are discussed. The internal architecture of the crawler engine used for crawling 100,000 top ranked websites is also explained.

5 Measurement

In order to identify the presence of hyper giants in today's Internet and to find out the degree of dependency on hyper giants to delivery Internet web content, the 100,000 top ranked websites of Alexa are crawled. To find out the hyper giants, the hosting infrastructures which are serving popular websites are observed. DNS resolution and HTTP header information also extracted to identify the features related to each website link. The features are number of links served by each hosting infrastructure, number of IP addresses, number of BGP prefixes for each IP address and number of AS numbers. These features are used to cluster the hosting infrastructure following the methodology discussed in chapter 3. This chapter explains the overview of traces which are collected after each step staring from web crawling to the last step, web object identification.

5.1 Web Crawling

To obtain a good coverage of the hosting infrastructure, the 100,000 top ranked websites from Alexa [26] are crawled. Alexa ranks websites based on Internet traffic-users of its tool bar for various web browsers like Google Chrome, Internet explorer, Firefox . Moreover, websites contain a lot of embedded links to images, videos, advertisements etc. which the browser of the user has to download from the various web servers. These embedded links can be from different hosting infrastructure. In our study, such embedded links has to be taken into account, as it might be served from servers other than those serving the home page. To give a better understanding, while crawling facebook.com, the home page is served from Facebook hosting infrastructure while the logo and some other meta data come from Akamai. DNS resolution and HTTP header analysis is carried out on each of these web links including their corresponding embedded links to extract IP addresses, ARecord name information. The scrapy crawler queries the HTTP Get method to local DNS resolver for all the website links and store the results in a trace file.

Three sample traces are given below,

Listing 7: DNS Reply for a host using dig command line tool

```
[55017, 0, 200, 'text/html', 19719,
 'http://parsquran.com',
  'a','parsquran.com', '74.208.215.213', 8560]

[55017, 1, 200, 'text/html; charset=UTF-8', 0,
'http://parsquran.com/site/sitemap.html',
'a', 'parsquran.com', '74.208.215.213', 8560]

[21794, 0, 200, 'text/html', 2014,
'http://gomap.az',
'a', 'gomap.az','85.132.44.164', 29049]
```

Each trace contain 10 different features, as described below.

1. index: index shows the rank of the website.

2. depth_level: 0 or 1. 0 shows that the website is the main page url and 1 shows the embedded links inside main page

3. httpResponseStatus: the HTTP return status code.

4. MIMEcontentType: this is included to know the type of element inside the web page.

5. ContentLength: content length gives the idea about the size of the element. The content type only extracted for the home page link.

6. URL: this is the URL to be crawled by the scrapy engine.This URL can be main page URL or embedded links.Duplicate links are omitted for the same main page by using RFPDupeFilter.RFPDupeFilter is a scrapy class which is used to detect and filter duplicate requests [20].

7. tagType :this shows the HTML element type. for example if an element is embedded in a website like "¡img class="desktop" title="" alt="" src="img/bg-cropped.jpg"¿",then the tagtype will be "img".

8. ARecord=This contain all the aRecord names involved in a website while resolving to the ip address.

Listing 8: DNS Reply for a host using dig command line tool

```
; <<>> DiG 9.10.3-P4-Ubuntu <<>> www.bmw.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60134
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.bmw.com.                   IN      A

;; ANSWER SECTION:
www.bmw.com.            3600    IN      CNAME   cn-www.bmw.com.edgesuite.net
cn-www.bmw.com.edgesuite.net. 3600 IN   CNAME   a1586.b.akamai.net.
a1586.b.akamai.net.     20      IN      A       104.121.76.64
a1586.b.akamai.net.     20      IN      A       104.121.76.49

;; Query time: 22 msec
;; SERVER: 127.0.1.1#53(127.0.1.1)
;; WHEN: Fri Oct 14 09:46:54 CEST 2016
;; MSG SIZE  rcvd: 143
```

From the Listing 8, it can be observed that, a1586.b.akamai.net. is the ARecord name for the the website "www.bmw.com" which will be stored in trace will under ARecord column.

9. destIP :this column stores the corresponding resolved ip addresses of the website.For example,for the website bmw.com,the destIP will store 72.247.184.130 and 72.247.184.137

10. ASN_Number=Field():This column stores the ASN number of a IP address.For this we use maxmind IP to ASN mapping file.

This procedure resulted in a total number of 12120731 unique links, which includes 100,000 top website links of Alexa, along with their embedded links.

5.2 Data Cleanup

From the traces that are collected in previous section, there might exist some traces for which one or more of the fields are missing, making them invalid. Such traces should be excluded from further analysis. This process of data cleaning are carried out by using regular expressions to filter out the invalid entries.

5.3 Hosting infrastructure to BGP mapping

As explained in section 3.1.1, the tests are carried to find out hosting infrastructure to corresponding BGP prefixes set. The IP addresses and corresponding BGP prefixes collected through RIPE RIS for hosting infrastructure, facebook.com is shown in table below.

SLD Infrastructure	IP address	BGP Prefixes
facebook.com	'173.252.88.112', '173.252.88.113', '173.252.88.104', '173.252.88.66', '173.252.90.132', '173.252.88.73'	'173.252.88.0/21', '173.252.64.0/19'

Table 1: Sample example shows SLD to BGP prefix mapping

From Table 1, it can be observed that, the hosting infrastructure facebook.com in column-1 might have hosted several web links that are resolved to the above stated 8 IP addresses as stated in column-2 which can be reached through the set of prefixes as in column-3.

5.4 Clustering

There are some hosting infrastructure which are administered by same company. For example, both the hosting infrastructure akamaiedge.net and akamai.net are administered by the same company,Akamai. In such cases i tis verified whether these hosting infrastructure share common infrastructure, i which case are clustered together, following the clustering algorithm as described in section 3.2. This procedure is carried out for all the 219604 unique hosting infrastructure, which has resulted in 53852 unique clustered hosting infrastructure.

To find out hyper giants presence, the features of each clustered hosting infrastructures are analyzed. To find out the feature set of each clustered hosting infrastructure, the

corresponding feature set of all the hosting elements involved is aggregate together as described in section 3.2

The Table 2 lists top 20 clustered hosting infrastructure on decreasing order of their number of links they host.

sl. no.	Clustered SLD Infrastructure	SLDs	links	IP addresses	ASNs	prefixes
1	cloudflare.net	17711	1295505	29893	17	78
2	akamaiedge.net	158	597533	3396	9	27
3	google.com	251	240910	195	1	22
4	yunjiasu-cdn.net	6068	210463	5907	21	77
5	us-east-1.elb.amazonaws.com	4254	199109	10885	31	115
6	wpengine.com	3963	136400	4072	19	115
7	anycast.me	2667	116024	2207	3	13
8	ourwebpic.com	722	113746	772	16	16
9	eu-west-1.elb.amazonaws.com	1946	100265	4476	25	31
10	kxcdn.com	1547	82120	1235	7	7
11	edgecastcdn.net	73	79750	383	2	11
12	jiashule.com	1769	79038	2249	28	100
13	cloudflare.com	3	78907	34	1	5
14	alikunlun.com	964	76647	1155	17	45
15	d5nxst8fruw4z.cloudfront.net	3331	75660	1619	3	7
16	incapdns.net	35	66882	382	7	27
17	fastlylb.net	73	65360	201	0	4
18	dynect.net	1	62760	46	2	5
19	ap-northeast-1.elb.amazonaws.com	1308	60343	3105	22	23
20	d2t8dj4tr3q9od.cloudfront.net	2363	59807	1316	8	7

Table 2: top 20 clustered hosting infrastructure

The columns 1-6 from table represent name of the clustered hosting infrastructure, number of hosting infrastructure that are merged into the same cluster, number of links collectively hosted by the cluster, collective set of IP addresses of the cluster, collective number of ASNs of the cluster and collective BGP prefixes of the cluster respectively. From table, it can be observed that 17711 number of SLDs merged with cloudflare.net. From the table it can be observed that two different clustered hosting infrastructure in sl. no. 1, 13, belonging to the same company, cloudflare but remained in different clusters.This is because, they have different infrastructure from each other in terms of BGP prefixes. Same is the case for hosting infrastructure in sl. no. x,y,z administered by Amazon. This might be because of big hosting infrastructure use different infrastructure for various operations.

5.5 Candidate Analysis for hyper giant

From Figure 7 it can be seen that, almost 80% of the total unclustered hosting infrastructure got clustered into first 3.73% of hosting infrastructure. Many of the other hosting infrastructure share their infrastructures with these top 3.73% of clustered infrastructures. Hence there is a possibility to get the hyper giants in these 3.73% of clustered hosting in-

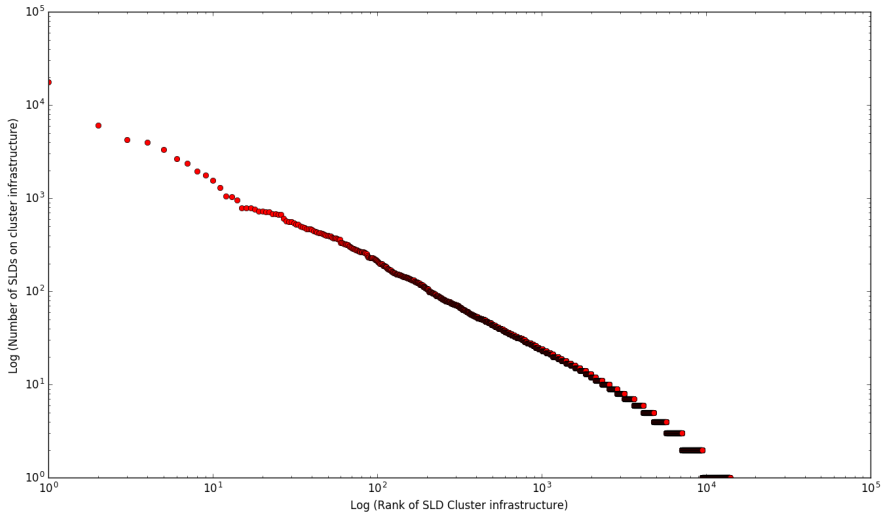

Figure 7: Number of SLDs served by different SLD infrastructure clusters

frastructure. But there are hosting infrastructure who have their own infrastructure in the form of data centers. Hence they don't share any other hosting infrastructure. Like facebook.com who has its own infrastructure in the form of data centers all over the world. In fact from figure ,it is also observed that almost 82.45% of clustered infrastructure who does not share their infrastructure with no more than another hosting infrastructure. It means there are companies who work independently by creating their own infrastructure.This can be data centers all over world. Although these 82.45% of clustered infrastructure do not share their infrastructure with no more than single infrastructure, still some of them serve a large number of links which make them another candidate for hyper giant analysis.

Hence to get a better picture, the clustered hosting infrastructure will be analyzed based on how many links they served. Hence by sorting all clustered hosting infrastructure in their decreasing order of links, it is found that top 5.95 % (=3205) clustered hosting infrastructure serve almost 80% of links and have 78.65 % of hosting infrastructure. Hence hyper giants can be found in this range.

5.6 Collection of web objects

Different web objects embedded in home pages of 100,000 top ranked web sites of Alexa are also collected.After crawling, it is observed that total 285 different types of objects from 13919464 links present in Internet. 38% of all the URLs constitute of object type text, while 42 % of URLs constitute of object type image.

5.7 Conclusion

This chapter outline the measurements obtained after crawling the 100,000 top ranked websites of Alexa. This has resulted in total of 13919464 links which include both the home pages of these top ranked websites and their corresponding embedded links. After applying DNS resolution on these links, 219604 hosting infrastructure are detected. On

clustering these infrastructure, it has resulted in total 53852 clustered hosting infrastructure. It is found that the top 5.95% clustered infrastructure alone are serving 80% of all URLs. It is also found out that 78.65 % of hosting infrastructure pools into the same 5.95% clustered infrastructure. It is for this reason, it can be conveniently assumed that the hyper giants are exists in these top 5.95 % of clustered hosting infrastructure, owing to zipf's law. The measurements revealed that all the URLs constitute 38% of object type text/html, while 42 % of URLs are of the object type image.

6 Results

This chapter analyzes the measurements obtained from chapter 5 to identify hyper giants by studying the features of clustered hosting infrastructure such as number of links, number of IP addresses, number of BGP prefixes, number of AS numbers that belong to cluster. Once hyper giants are determined, the dependency of popular websites on them will be examined by analyzing the type of web objects like images, videos, HTML files etc. they deliver.

6.1 Identifying hyper giants

This section analyses the 3205 clustered hosting infrastructures using their feature set to identify if there are any hyper giants. The hosting infrastructure that has unique features can be used to classify them as hyper giants. Hence some mining technique has to be applied to separate the hyper giants from the rest of the hosting infrastructure. For this purpose, k-means clustering is used to separate infrastructure into different clusters each having it's own characteristics. Furthermore, these clusters are analyzed to identify if any of their characteristics can qualify them to be treated as hyper giants.

K-means algorithm uses, the number of links hosted by each hosting infrastructure and the number of IP addresses each hosting infrastructure contain as it's inputs. The k-means algorithm clusters the given 3205 hosting infrastructure. The value of of k=10 is chosen as a clustering parameter as it gave clusters with unique characteristics. The following graph as shown in Figure 8 represents the distribution of hosting infrastructure into clusters, which is obtained using k-means algorithm.

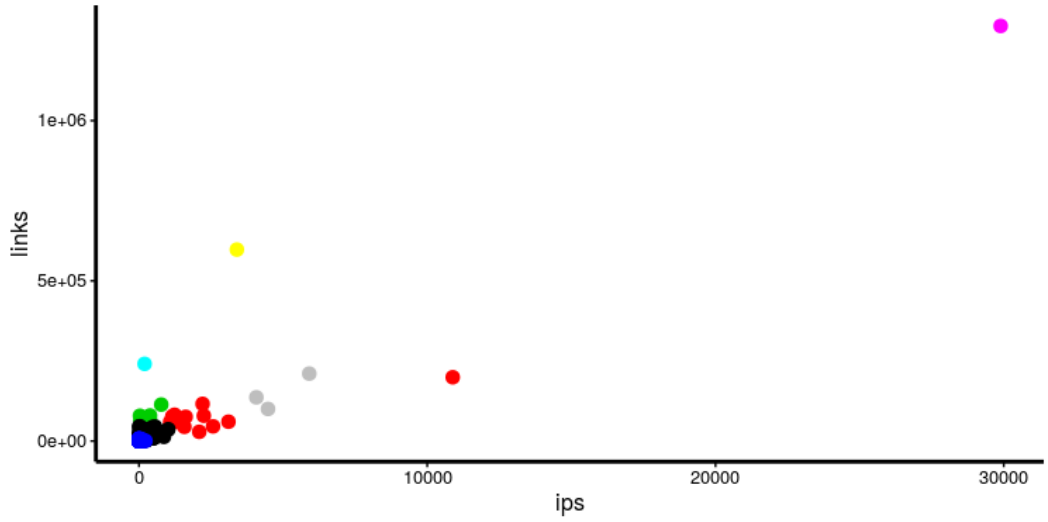

Figure 8: Distribution of hosting infrastructure into clusters

The details of the 10 clusters mined from the 3205 hosting infrastructure that we considered for analysis are categorized into 10 clusters using k-means, whose information is tabulated in Table 3.

As observed from Table 3, the number of hosting infrastructures inside clusters 1, 2, 4, 5,

cluster No.	No. of hosting infrastructure	Link density of cluster
1	1	1295505
2	3	149042
3	3015	1386
4	1	597533
5	8	72584
6	1	240910
7	130	13642
8	34	16735
9	1	199109
10	11	40648

Table 3: k-means clustering

6, 9 and 10 are very sparsely distributed, while in clusters 3,7 and 8, the hosting infrastructure is very densely distributed. Thus the clusters 1, 2, 4, 5, 6, 9, 10 can be treated separately from the other clusters owing to the fact that their hosting infrastructure distribution in these two groups are completely different.

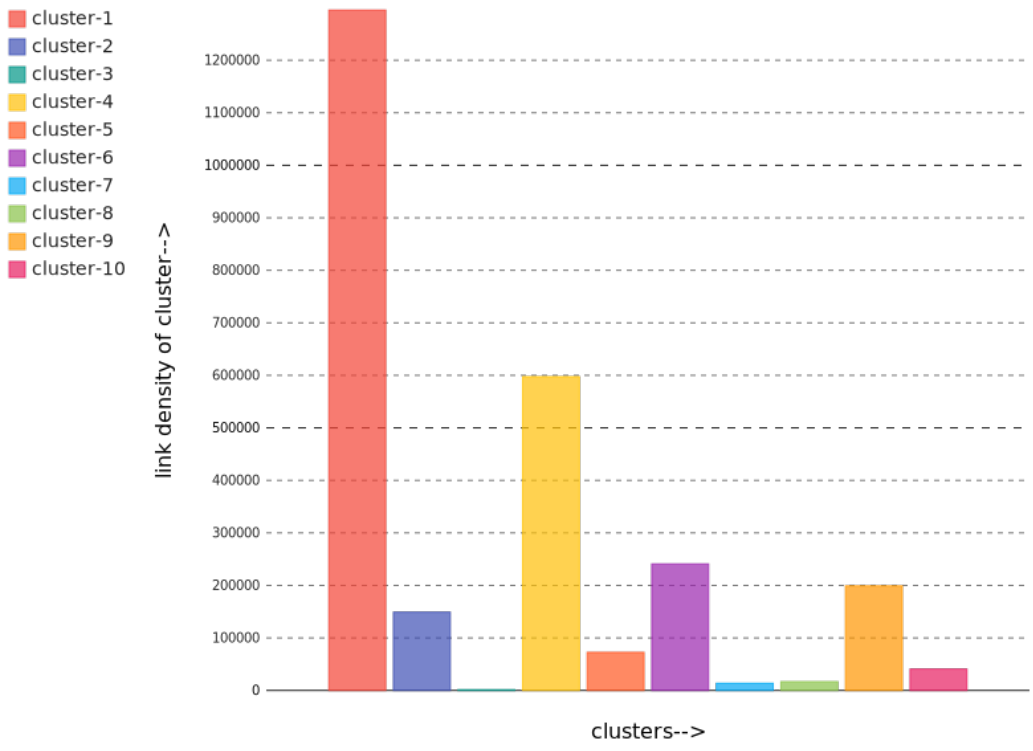

Figure 9: link density per cluster

sl. no.	Hyper giants
1	alikunlun.com
2	d2t8dj4tr3q9od.cloudfront.net
3	ap-northeast-1.elb.amazonaws.com
4	ap-southeast-1.elb.amazonaws.com
5	cdntip.com
6	d5nxst8fruw4z.cloudfront.net
7	eu-west-1.elb.amazonaws.com
8	fastlylb
9	google.com
10	jiashule.com
11	kxcdn.com
12	pbwstatic.com
13	akamaiedge.net
14	anycast.me
15	cloudflare.net
16	cloudflare.com
17	dynect.net
18	edgecastcdn.net
19	incapdns.net
20	netdna-cdn.com
21	ourwebpic.com
22	us-east-1.elb.amazonaws.com
23	wpengine.com
24	us-west-2.elb.amazonaws.com
25	windows.net
26	yunjiasu-cdn.net

Table 4: 26 hyper giants

Link density of a cluster can be defined as the average no of links of the cluster, which is obtained by diving the number of links hosted by cluster with the total number of hosting infrastructure in the cluster. Figure 9 shows the distribution of Link density for each cluster. X-axis of the graph shown in Figure 10 represents the cluster number, while the Y-axis represents the Link density of the cluster.

From the graph show in Figure 9, It is clear that clusters 1, 2, 4, 5, 6, 9,10 have relatively higher link densities compared to the other clusters. Hence these clusters can be grouped together, which here afterwards referred to as group-1, while the rest of the clusters 3,7,8 can be grouped together, and be called here afterwards as group-2.

Since, we regrouped the total 10 clusters in 2 groups, it has to be analyzed further to identify which one of these two groups can be qualified as a hyper giant. This can evaluated by analyzing the link density of each group. Link density of a group can be calculated by summing the total number of links hosted by each cluster of the group and dividing this sum with total number of hosting infrastructure of the group. In this case, group-1 has 7 clusters, who has a cumulative total of 4088821 links and 26 hosting infrastructure. While in group-2, which has 3 clusters, has a cumulative total of 6959026 links and 3179 hosting infrastructure. Hence, the link density of the group-1 is 157262.36 (4088821/26), and for group-2 is 2189.06 (6959026/3179). From this, it is observed that their link densities are

approximately in the ratio of 72:1, thus qualifying the members of group-1 to be called as hyper giants.

26 hosting infrastructure are concluded as hyper giants shown in Table 4, which are collectively hosting 33,73% of the total web links crawled.

6.1.1 Conclusion

From the above section,we are able to identify total 26 hyper giants when analyzed from a single vantage point in Germany. These hyper giants are collectively hosting 33,73% of the total number of links crawled. As the study has considered 100,000 top ranked websites, it can concluded that these hyper giants are hosting approximately 33,73% of Internet web traffic.

6.2 Popular websites dependency on hyper giants

Hyper giants build a large infrastructure all around the world to deliver content ensuring a faster response. The websites use these infrastructure to store their content, such as audios ,videos, test/html files etc. Therefore, any kind of disruption in serving these contents from hyper giants will impact the websites which shows the interdependency of hyper giants with the web sites. This section focuses on finding how hyper giants influence the delivery of web objects of Internet web traffic.

6.2.1 Cumulative object type distribution in hyper giants

The goal of this study is to analyze how hyper giants influence the delivery of web objects. This can be observed by visualizing the top web objects hosted by hyper giants to web objects delivered by all hosting infrastructure. Figure 10 depicts the comparison between the number of web objects delivered by all hosting infrastructure to hyper giants. The x-axis in the graph shows the different web object type delivered and the y-axis represents the number of web object type delivered through hyper giants and all the hosting infrastructure. For better understanding, all the web object types are divided into major 4 parts. “text”, “image”, “application”, “other”. The “text” object type contains different object types like text/html, text/css etc. The “image” object contain objects like image/png, image/jpeg etc. Similarly the “application” object contain objects like application/javascript, application/asp etc.

From the Figure 10, it can be observed that, out of total 12120731 links crawled from 100,000 top ranked web sites of Alexa, 9750438 are text object type. 1741064 are of image type, 620573 are of application type and 8656 are of other type while considering all hosting infrastructure. Similarly out of 9750438 text objects, hyper giants delivering 3183272 text objects which is 32,64 % of all text objects. For image type, it is observed that hyper giants are delivering more objects comparison to the that of all hosting infrastructure. Out of 1741064 image objects, 682962 image objects delivered by hyper giants which is 39,22 % of all image type. For application type, out of 620573 application type web objects, hyper giants are delivering 219801 application object types which is 35,41 % of all application type web objects. Similarly for other type web objects, hyper giants are delivering 2786 out of 8656 web objects delivered through all hosting infrastructure.

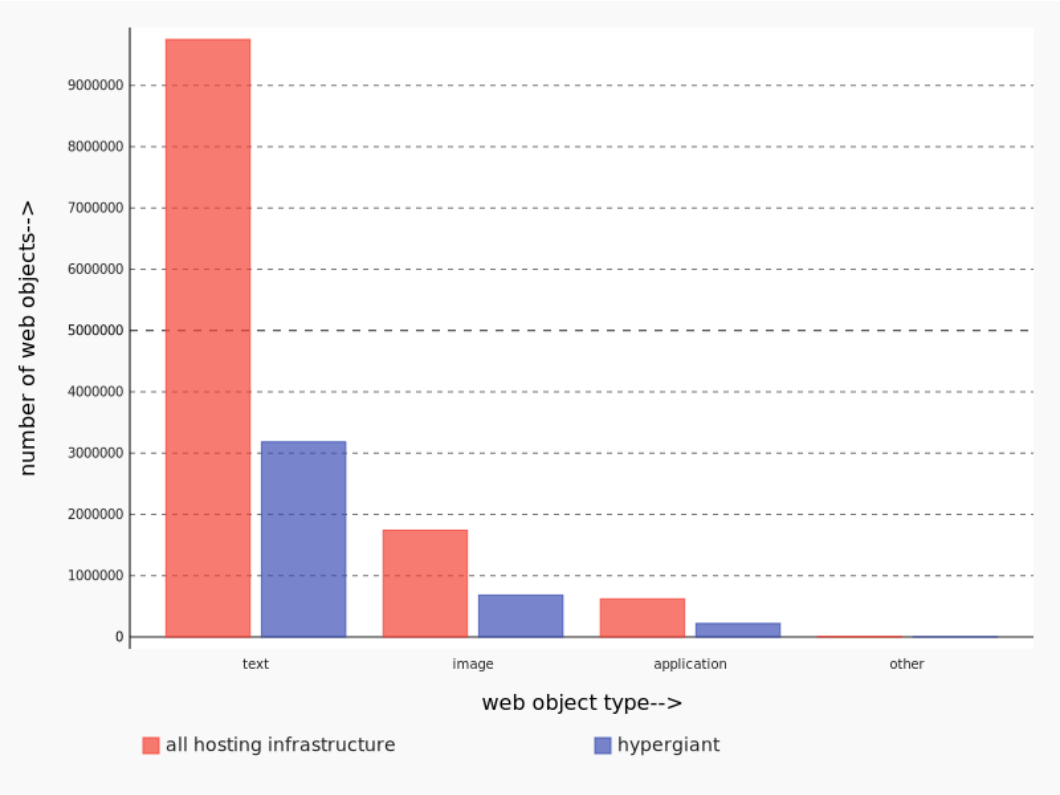

Figure 10: top 10 object percentage used by all hyper giants

From the Figure 10, it can be observed that, hyper giants are delivering almost same percentage of web objects for all the 4 types of web content. Only for the image type, it is slightly better. Hyper giants host overall 4088821 web objects which is almost 33,73 % of all web objects types host by all the infrastructure. Hence the percentage of "text", "image", "application", "other". The "text" objects types are almost percentage wise align with the overall percentage of web object delivered through hyper giant.

6.2.2 Individual object type distribution in hyper giants

In this section, different types of web objects delivered through the identified hyper giants will be analyzed. In today's Internet ,content plays a vital role. Hence it is important to observe what kind of data mostly delivered by each hyper giants .

The Table 5 shows the list of all 26 hyper giants and their corresponding object distribution in terms of percentage of the total web objects they serve. It can be observed from table that most of the hyper giants deliver very high percentage of text files which contain text/html,text/css etc. But there are few exceptions, such as the hosting infrastructure "netdna-cdn.com", "d5nxst8fruw4z.cloudfront.net" and "netdna-cdn.com" are providing very high number of images compare to other hyper giants.In comparison to other hyper giants, it can be observed that "netdna-cdn.com" host highest number of "application" type of objects.

Country Name	text	image	application	other
dynect.net	94.71	3.76	1.52	0.01
ap-northeast-1.elb.amazonaws.com	90.09	6.44	3.43	0.03
cdntip.com	89.55	8.79	1.66	0.0
yunjiasu-cdn.net	89.07	8.56	2.36	0.01
jiashule.com	88.89	8.73	2.38	0.0
eu-west-1.elb.amazonaws.com	88.29	6.76	4.93	0.02
us-east-1.elb.amazonaws.com	87.73	7.19	5.0	0.08
incapdns.net	87.39	7.74	4.85	0.02
ap-southeast-1.elb.amazonaws.com	86.74	8.72	4.43	0.11
ourwebpic.com	85.37	12.65	1.98	0.01
pbwstatic.com	84.46	15.29	0.25	0.0
us-west-2.elb.amazonaws.com	84.21	8.68	7.04	0.07
wpengine.com	81.15	12.01	6.79	0.05
windows.net	80.1	13.23	6.39	0.28
anycast.me	78.82	15.91	5.19	0.08
cloudflare.net	78.63	15.26	6.05	0.06
alikunlun.com	78.37	18.54	3.07	0.02
cloudflare.com	76.34	15.24	8.39	0.03
kxcdn.com	75.79	17.9	6.27	0.05
akamaiedge.net	75.5	18.63	5.8	0.07
google.com	75.07	23.2	1.72	0.01
fastlylb.net	69.02	22.43	8.4	0.14
d2t8dj4tr3q9od.cloudfront.net	50.26	41.64	7.77	0.33
d5nxst8fruw4z.cloudfront.net	32.83	57.3	9.43	0.44
netdna-cdn.com	19.44	59.04	21.14	0.37

Table 5: web object distribution for hyper giants

6.2.3 Conclusion

From the section, it can be inferred that the percentage of “text”, “image”, “application”, “other” are almost percentage wise align with the overall percentage of web object delivered through hyper giant compare to web object delivered through all the hosting infrastructure which is 33,73 %. For object distribution, most of the hyper giants deliver more text files compare to other types of objects, except for “d5nxst8fruw4z.cloudfront.net” and “netdna-cdn.com” who delivers more image type of objects.

7 Conclusion

This thesis conducted an empirical analysis to find out the presence of hyper giants on Germany's Internet ecosystem. It aims to answer the following research questions: Firstly, whether there is any presence of hyper giants in today's Internet? Secondly, what web objects contained in popular websites are delivered through hyper giants?

To answer the first question, we introduced an automated process to find out the hosting infrastructure in today's Internet as well as cluster these hosting infrastructure to find out the presence of hyper giants. To perform this task, 100,000 top ranked websites of Alexa are crawled using scrapy engine and their HTML codes are inspected to retrieve all their embedded URLs. Using DNS resolution and HTTP header analysis, different features of hosting infrastructure are collected such as number of links, number of IP addresses, number of BGP prefixes, number of ASNs. A clustering algorithm is presented which will help to find out which hosting infrastructure are sharing infrastructure with each other using features of hosting infrastructure. The advantage of this automated approach is that, it uses SLD and BGP prefixes to perform the tasks, hence this procedure can be used in any other research. The study reveals that there are 26 hyper giants present in Internet when analyzed from a single vantage point in Germany, which are collectively contributing to nearly 30% percentage of total web traffic on Internet. Some well known hosting infrastructure are detected in these 26 hyper giants, such as Google, Akamai, Amazon, Cloudflare etc.

The thesis has also focused on finding out, what web objects contained in popular websites are delivered through hyper giants. Using HTTP header analysis, different web object types such as text/html, image, video, audio etc. are collected. It is found that hyper giants focuses on delivering more number of text/HTML object files than other object types. This reveals that any disruption on hyper giant infrastructure may leave many web pages being unloaded. It is found that cloudflare.net delivered maximum percentage of "application" type web objects compared to other type of objects that it hosts. Google, Akamai are hosting more text/html object type files compared to other object types. cloudfront.com and netdna-cdn.com are delivering more image files compare to other web object types they host. This information will help the websites, which are focused on delivering a particular kind of content. For example video sharing sites are mainly focused on video content where as blogs focus more on text/html objects.This information might be of some value for this kind of web sites while they are making a decision on where to host their content.

The data is collected at a single vantage point in Germany. Furthermore this thesis is an important step towards answering some of the very crucial questions for CDNs, content providers etc. Moreover it will help the research community to discover the Internet architecture changes with time. They also can able to track the hyper giants and dependency of popular websites in these hyper giants.

The following areas can be investigated further which are not covered in the current scope of this thesis.

- Measurements are performed on a single vantage point in Germany. Hence the identification of hyper giants, their role and the degree of dependency of popular websites on hyper giants might change when the whole procedure will be carried out from multiple vantage points across the world.

- Secondly while clustering the hyper giants,the k-means parameter is taken by going through very small number of observations.Hence in future this can be tested more precisely which might help to cluster the hosting infrastructures at a granular level.
- In the clustering algorithm, a similarity index of 0.7 is chosen to decide that two infrastructure can be clustered together. While this is an assumption, further analysis can be done on choosing the right similarity index.

8 List of Acronyms

CDN	Content delivery network
SLD	Second level domain
ASN	Autonomous system number
BGP	Border gateway protocol
HTTP	Hypertext transfer protocol
HTML	Hypertext markup language
DNS	Domain name system
IP	Internet protocol
QoS	Quality of service
XML	Extensible markup language
ISP	Internet service provider
IXP	Internet exchange point
GGC	Google global cache
CNAME	Canonical name
RR	Resource record
URIs	Uniform resource identifiers
URL	Universal resource locator
IDE	Integrated development environment

References

[1] Internet Society Global Internet Report 2015. *http://www.internetsociety.org/globalinternetreport/assets/download/IS_web.pdf.*

[2] Internet stats & facts for 2016 *https://hostingfacts.com/internet-facts-stats-2016/*

[3] C. Labovitz, S. Lekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian. *Internet Inter-Domain Traffic.* In Proc. ACM SIGCOMM, 2010.

[4] I. Poese, B. Frank, B. Ager, G. Smaragdakis, and A. Feldmann. *Improving Content Delivery using Provider-Aided Distance. Information.* In ACM IMC, 2010.

[5] Manuel Palacin, Miquel Oliver, Jorge Infante, Simon Oechsner and Alex Bikfalvi *The Impact of Content Delivery Networks on the Internet Ecosystem.* Journal of Information Policy, Vol. 3 (2013), pp. 304-330

[6] Schonfeld, E. Eric Schmidts *Gang Of Four:Google, Apple,Amazon,And Facebook.* TechCrunch.Retrieved from https://techcrunch.com/2011/05/31/schmidt-gang-four-google-apple-amazon-facebook/

[7] Y. Shavitt and U. Weinsberg *"Topological trends of Internet content providers"* in Proceedings of the 4th Annual Workshop on Simplifying Complex Networks for Practitioners (SIMPLEX '12), pp. 13–18, ACM, Lyon, France, April 2012.

[8] Internet evolution *https://atos.net/content/dam/global/ascent-whitepapers/ascent-whitepaper-internet-evolution.pdf*

[9] Bernhard Ager,Wolfgang Mhlbauer,Georgios Smaragdakis,Steve Uhlig *Web Content Cartography.*

[10] Bernhard Ager *Impact of Location on Content Delivery.* http://people.ee.ethz.ch/ bager/papers/A-ILCD-11.pdf

[11] Benjamin Frank, Ingmar Poese, Georgios Smaragdakis, Anja Feldmann, Bruce M. Maggs, Steve Uhlig, Vinay Aggarwal, Fabian Schneider *Collaboration Opportunities for Content Delivery and Network Infrastructures.* 2013

[12] P. Mockapetris *Domain Name System.* https://www.ietf.org/rfc/rfc1034.txt

[13] Postel, Jon. *"Domain Name System Structure and Delegation"* Network Working Group. Retrieved 7 February 2011.

[14] K RISTOL , D., AND M ONTULLI , L. *HTTP State Management Mechanism.* RFC 2109.

[15] K RISTOL , D., AND M ONTULLI , L. *HTTP State Management Mechanism.* RFC 2965.

[16] Alexa *http://www.alexa.com/topsites*

[17] Scrapy *http://doc.scrapy.org.*

[18] *Technology: How and Why We Crawl the Web.* Alexa. Archived from the original on April 2, 2014. Retrieved November 6, 2011.

[19] G. Maier, A. Feldmann, V. Paxson, and M. Allman. *On Dominant Characteristics of Residential Broadband Internet Traffic.* In Proc. ACM IMC,2009.

[20] Nygren, R. K. Sitaraman, and J. Sun. *The Akamai Network:A Platform for High-performance Internet Applications.* Syst. Rev., 44:219, August 2010.

[21] R. Krishnan, H. Madhyastha, S. Srinivasan, S. Jain,A. Krishnamurthy, T. Anderson, and J. Gao. *Moving Beyond End-to-end Path Information to Optimize CDN Performance.*

[22] Yuval Shavitt, Udi Weinsberg. *Topological Trends of Internet Content Providers.* SIMPLEX 12: Proceedings of the Fourth Annual Workshop on Simplifying Complex Networks for Practitioners (2012): 13-18.

[23] Xenofontas Dimitropoulos, Dmitri Krioukov, Marina Fomenkov, Bradley Huffaker, Young Hyun, kc claffy, George Riley *AS Relationships: Inference and Validation.* SIGCOMM Computer Communications Review 37, no. 1(2007): 31-40.

[24] A. Gerber and R. Doverspike. *Traffic Types and Growth in Backbone Networks.* OFC/NFOEC, 2011.

[25] Mike Axelrod *The Value of Content Distribution Networks and Google Global Cache.*

[26] Netflix Open Connect *https://openconnect.netflix.com/en/*

[27] *http://www.hpl.hp.com/research/idl/papers/ranking/adamicglottometrics.pdf.*

[28] Lada A. Adamic ,Bernardo A. *Huberman Zipfs law and the Internet* . Glottometrics 3, 2002,p 143-150

[29] RIPE NCC *RIPE Routing Information Service.* http://www.ripe.net/ris/.