Traffic Network Generation Focused on Flash Crowd Anomaly

By

Zahra Saleh

A thesis
Presented to Sharif University of Technology, International Campus, Kish Island
in partial fulfillment of the
Requirements for the Degree of
Master of Science

in

Information Technology

(Computer and Communication Networks)

Supervisor:

Dr. Amir Hossein Jahangir

Kish Island, Iran, 2016

In the name of GOD

Sharif University of Technology

International Campus, Kish Island

This is to certify that the Thesis Prepared,

By: **Zahra Saleh**

Entitled: **Traffic Network Generation Focused on Flash Crowd Anomaly**

and submitted in partial fulfillment of the requirements for the Degree of

Master of Science

complies with the regulation of this university and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

Supervisor:

Dr. Amir Hossein Jahangir ----------

External Examiner:

Dr. Kaveh Kavousi ----------

Internal Examiner:

Dr. Afshin Hemmatyar ----------

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. The work described in this thesis has not been previously submitted for a degree in this or any other university. All and any contributions by others are cited. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners. I understand that my thesis may be made electronically available to the public.

Abstract

Traffic Network Generation Focused on Flash Crowd Anomaly

Zahra Saleh, M.Sc.

Sharif University of Technology, International Campus, Kish Island, 2016

Supervisor: Dr. Amir Hossein Jahangir

Flash Crowd traffic generation can be used as a metrics for measuring the resiliency and performance of a server. Also, it can provide a framework for verification and test of Intrusion detection systems (IDS) and Intrusion protection systems (IPS). Common traffic generation methods mimic timing and content of input traffic or regenerate input traffic by extracting its statistic distribution. So all of them need input traffic, while properties of Flash Crowd are different in the various servers and situations and there is no guaranty in existence of such samples of traffic for all servers.

In this thesis, we introduce and use a new method for traffic generation without the need for input traffic. Therefore, we concentrate on properties of legal user and combine user behavior modeling with Flash Crowd effects. For this purpose, we implement add-ons for Firefox browser in order to capture user behavior. Afterwards, we fit the statistical behavior of users to a well-known distribution and model it by an approximate state

machine. Finally, our traffic is generated online by the combination of this behavior and Flash Crowd effects.

Keyword: Traffic generation, Flash crowd, User behavior analysis.

Acknowledgment

Firstly, I would like to express my sincere gratitude to my supervisor Dr. Jahangir for the continuous support of my thesis study and related research, for his patience, motivation, and knowledge. His guidance helped me in all the time of research and writing of this thesis.

Besides my supervisor, I would like to thank the rest of my thesis committee: Dr. Hemmatyar and Dr. Kavousi for their insightful comments, but also for the hard questions which incented me to widen my research from various perspectives.

My sincere thanks also go to Dr. Arshadi for enlightening me the first glance of research.

Table of contents

Abstract ..... **iv**

Acknowledgment ..... **vi**

Table of contents ..... **vii**

List of Figures ..... **ix**

List of Tables ..... **x**

Chapter One **1**

Introduction **1**

1.1 Definition: 1

1.2 Subject importance and challenges: 3

1.3 Proposed solution: 5

1.4 Structure of this thesis: 6

Chapter Two **7**

Related works **7**

2.1 Network traffic characteristics: 7

2.1.1 Statistical characteristics: 7

2.1.1.1 Long-range dependency and Short-range dependency: 7

2.1.1.2 Heavy-tailed distribution: 8

2.1.1.3 Self-similarity: 9

2.1.1.4 Reasons of self-similarity in network traffic: 9

2.1.2 Temporal properties: 14

2.1.3 Behavioral Oscillation: 14

2.2 Flash Crowd properties: 15

2.3 Related works: 18

2.4 The differences between DDoS and Flash Crowd: 19

2.4.1 Ordinary DDoS: 20

2.4.1.1 Changes in rate of input traffic: 20

2.4.1.2 Changes in entropy of IP addresses: 22

2.4.1.3 Distribution of requests: 23

2.4.1.4 Geographic distributions' users: 25

2.4.1.5 Distribution of requested files: 26

2.4.1.6 Traffic pattern: 26

2.4.1.7 Users' request speed: 27
2.4.2 Advanced DDoS (FE imitators): 30
2.5 Conclusion: 32
Chapter Three 35
Proposed approach 35
3.1 User behavior simulation: 36
3.1.1 Start page selection: 36
3.1.2 Browser behavior simulation: 37
3.1.3 User behavior simulation: 38
3.2 Users grouping according to their trading history with the website and founding parameters like traverse depth, visit time, and number of tabs which are opened by user to any depth: 40
3.2.1 User silence time before concurrent tabs appearance: 41
3.2.2 User silence time after concurrent tabs appearance: 42
3.2.3 Time of page visiting: 47
3.2.3.1 Kolmogorov Smirnov test: 48
3.2.3.2 First step of test: 49
3.2.3.3 Second step of test: 51
3.2.4 User browsing depth: 56
3.2.5 Number of opened tabs in each depth: 58
Chapter Four 61
Implementation and evaluation 61
4.1 Main program: 61
4.2 Traffic generation: 64
4.3 Evaluation and analyzing results: 66
4.3.1 Number of requests: 66
4.3.2 Flow correlation: 69
4.3.3 Response time: 70
4.3.4 User requests speed: 70
Chapter five 71
Conclusion 71
References 73

List of Figures

Figure 2.1 - CAIDA-DDoS [23] 21

Figure 2.2 - World Cup 1998 [23] 21

Figure 2.3 - New IPs appearance rate in CAIDA with 1-minute memory [23] 22

Figure 2.4 - New IPs appearance rate in FE with 1 min memory [23] 23

Figure 2.5 - Requests distribution per each IP in normal traffic [23] 24

Figure 2.6 - Request distribution per each IP in DDoS & FE [23] 25

Figure 3.1 - State machine of users browsing behavior 39

Figure 3.2 - Distribution of users' complete silence time in their familiar websites 50

Figure 3.3 - Distribution of users' partial silence time in their familiar websites 50

Figure 3.4 - User state machine in his familiar website 52

Figure 3.5 - Distribution of users' complete silence time in their unfamiliar websites 53

Figure 3.6 - Distribution of users' partial silence time in their unfamiliar websites 54

Figure 3.7 - User state machine in his unfamiliar website 55

Figure 3.8 - User browsing state machine in the favorite website 59

Figure 3.9 - User browsing state machine in the new website 60

Figure 4.1 - Generated FE traffic 65

Figure 4.2 - Number of requests which server receives per minute 67

Figure 4.3 - User browsing process 70

List of Tables

Table 2.1 - Differences between DDoS & FE ---------- 28

Table 3.1 - Object fetch in different browsers ---------- 38

Table 3.2 - Number of users' page revisiting in their familiar websites ---------- 51

Table 3.3 - Number of users' page revisiting in their unfamiliar websites ---------- 55

Table 3.4 - Users' browsing depth in their familiar websites---------- 56

Table 3.5 - Users' browsing depth in their unfamiliar websites ---------- 57

Table 3.6 - Number of opened tabs in each depth in the familiar websites---------- 58

Table 3.7 - Number of opened tabs in each depth in the unfamiliar website ---------- 60

Chapter One
Introduction

1.1 Definition:

After occurrence of a social event or an important news, sometimes surge of Internet users visit a particular website and so the number of received requests becomes more than server processing capacity. This situation is called Flash Crowd and the server experiences shortage of processing resources such as memory or bandwidth, and cannot respond to user request, or response time becomes more than usual.

It is important to note that every traffic volume growth cannot be considered as Flash Crowd. In this event, legal users and their legal requests are doers of overloading, while attacker bots can increase the traffic load as well. Researchers have suggested various parameters such as rate of requests growth, file distributions, flow correlation etc. for distinguishing between legal requests and attack; we will exhaustively discuss them in Chapter 2.

Various flash events have been categorized according to bellow factors [1]:

1. User properties such as connection speed, geographic distribution and attendance history on the website.

2. The amount of server traffic load which may only be bigger than normal traffic or causes unavailability for the server.

3. The predictive ability of events.

4. Traffic pattern, it is usually bouncing but when the load is more than usual, it may become flat, for example, ticket selling event in World Cup 98.

In another study, flash events have been classified into three major groups [2]:

1. Predicable Flash Event: In this group, flash crowd is Predicable, and furthermore, the peak of the event is almost predictable. (World Cup 98)

2. Unpredictable Flash Event: Unexpected events or news, such as Wikipedia flash crowd at Steve Job's death.

3. Secondary Flash Event: This usually happens in supplementary search about an important event in websites which usually have few visitors.

1.2 Subject importance and challenges:

Todays, one of the important challenges in intrusion detection/prevention systems is the combination of DDoS attacks and FE anomalies because we need to distinguish them in order to block imitative requests and answer to legal users. Therefore, this separation ability is an important factor in performance evaluation of these systems however, such an evaluation needs samples of aforementioned combined traffic itself.

DDoS attacks are often based on UDP protocol or unidirectional streams, so we can regenerate them by means of available input traffic samples and inject them into servers offline. Also, there are traffic generators which are acceptably able to generate these attacks. But Flash Crowd is a kind of traffic which has been generated by legal users and is often based on the TCP protocol. Therefore, because of TCP bidirectional interaction and congestion control mechanism, we cannot use a traffic offline. On the other hand, all of the previous efforts for the generation of this traffic need an input traffic of Flash Crowd. In other words, if we want to use aforementioned traffic generation tools, in the first step we need to have an up-to-date input traffic sample of Flash Crowd. But available valid Flash Crowd traffic samples are very rare and old, in a way that World Cup 98 has been used in most of the researches in this context. Hence it is clear

that using this input traffic sample, because of vast changing in Internet structure, bandwidth, browsers, and browsing habits cannot have trust results.

The difference between users' transmission rate and browsing behavior yields a situation where we could not estimate the number of users that a server can respond. Therefore, this is the other challenge in this area. Hence, if we repeat generation of Flash Crowd in various conditions, we may succeed to estimate an average value for server capacity. By means of these pieces of information, we can define a threshold less than aforementioned value and use tactics for managing Flash Event before it happens.

But as previously mentioned, we cannot inject a captured traffic sample to the server. On the other hand, Flash Crowd is a completely relative event and dependent on server capacity and characteristics, therefore a generator which can generate this event, according to server characteristics needs the traffic sample of Flash Crowd relative to the same server while such a sample may not be available. Even if this sample was available, we would need a generator that we could change its conditions such as user arrival time. But the current traffic generators cannot do so and they always operate with fixed input and output.

In this thesis, we try to solve these issues by generating online Flash Crowd traffic based on server characteristics and user behavior browsing.

1.3 Proposed solution:

Reasons were previously mentioned, forced us to think through a new method for traffic generation which against current tools do not need input traffic sample. In order to reach this goal, we tried to combine legal user behavior properties with differential attributes of Flash Crowd and generate our traffic.

For doing it, we studied about Flash Crowd and its differences with normal traffic and DDoS and add them to our method. Also, we know that Flash Crowd users are legal, therefore we can expect their behavior during browsing like normal traffic with this difference that in this case, due to long response time, the user may give up his request temporally or even forever. Therefore, we decided to model user browsing behavior and consider it in our project.

In our method, we can take into account variable user arrival time and independence from input traffic sample, as we can generate Flash Crowd in various conditions.

1.4 Structure of this thesis:

This thesis is organized in five Chapters. Basic concepts and related works are discussed in the next section. In chapter 3 we present our method. In Chapter 4 results are analyzed, and finally, in Chapter 5 the conclusion and future works are presented.

Chapter Two
Related works

In this Chapter at first, we investigate network traffic properties for preparing required theoretical concepts of our study, then describe Flash Crowd (for short FC, or Flash Event: FE) properties. Next, we discuss differences between FE and DDoS. Finally, we propose our approach for preparing required technical framework of our project.

2.1 Network traffic characteristics:

We can classify network traffic characteristics into three categories:

1. Statistical
2. Temporal
3. Oscillation behavior

We investigate them in the following.

2.1.1 Statistical characteristics:

2.1.1.1 Long-range dependency and Short-range dependency:

A time series with long-range dependency has an auto-covariance function r(k) so that when k approaches infinity then $r(k) \sim k^{-\beta}$ and $0 < \beta < 1$.

In fact, its auto-covariance function decreases power-like during the time, while auto-covariance function in short-range dependency exponentially decays.

Hurst parameter ($0 < H < 1$) is used for measuring the durability of statistical characteristics of time series. This parameter is equal to $H = \beta - ½$ in long-range dependency. Accordingly,, in long-range dependency H is between ½ and 1. When H is equal to ½, this means time series has the short-range dependency. When H is between 0 and ½, time series does not have dependency and it is bouncy. Although self-similarity is seen in both kinds of dependency, but when H approaches 1, both long-range dependency and self-similarity increase.

2.1.1.2 Heavy-tailed distribution:

In probability theory, heavy-tailed distribution is a distribution for which tails are not exponentially bounded. In other words, its tails are heavier than the exponential distribution [3]. Such distributions have the long-range dependency.

Each user request is a URL address or hyperlink and each URL has some objects such as text, image etc. In fact, any user request is for all objects or files which exist inside the requested URL. In a research [4] distribution of

requested file was investigated and they found out this distribution is a heavy-tailed distribution.

The time user does not have any request is called silence time, and has 2 parts: active silence and passive silence. Active silence is the machine processing time; therefore it is short and between 1 millisecond and 1 second, while passive silence is the time user is reading pages. This time is longer than process time.

In another study [5] it has been shown that passive silence follows the heavy-tailed distribution. Also, other researchers model or consider the distribution of active silence as Weibull, and passive silence follows Pareto distribution [6].

2.1.1.3 Self-similarity:

In mathematics a shape is called self-similar when its statistical properties are repeated in smaller scales [7]. In other words, we observe a shape like the main shape when each its small part magnifies. Another study [8] has shown that network traffic is self-similar.

2.1.1.4 Reasons of self-similarity in network traffic:

As already mentioned, distribution of requested files is heavy-tail and we know that heavy-tailed distribution has long-range dependency and as a result, it is self-similar. Hence, researchers believe the distribution of

requested files is one of the most important reasons of self-similar behavior of network traffic.

Also, other studies [5], [6] have shown passive silence is heavy-tail as well. as a result, authors [5] believe the distribution of passive silence is another reason for self-similarity, but as it is lighter than the distribution of requested files, they have taken the distribution of files for the main reason of self-similarity. Some researchers [9] declared transmitting of files having their sizes following heavy-tail is a good sign of self-similarity event, but in another study [18], TCP control mechanism has been enumerated as other reason.

A chaotic system has following properties: [10]

1. Nonlinearity
2. Determinism
3. Butterfly effect
4. Unpredictability

We investigate TCP mechanism from this point of view. TCP mechanism is related to congestion window size and we know changes in this size are nonlinear. Nevertheless TCP has the first condition of a chaotic system. Moreover, in TCP all of the conditions are known, deterministic and

nonrandom, therefore the second condition is satisfied as well. On the other hand, the small change in initial conditions such as bandwidth, the number of connections, buffer size, etc. will cause big changes in its pattern (satisfaction of the third condition). Finally, because of random rate of packet loss, changes of congestion window are unpredictable. Accordingly the last condition is satisfied as well.

In [18] TCP mechanism has been shown as a trajectory. The system changes from a multidimensional state to another where each state is a variable of the system. The system is deterministic, therefore if it goes to the previous state, the previous path will be repeated. This system is stable, for example, dropping a packet does not change its pattern.

Although changing initial conditions of this system can cause complexity, so that it looks as if the system becomes non-periodic, but [18] has shown the number of these states is finite because dimensions are limited and discrete. Hence, the system keeps its periodic conditions but the length of this period may only become longer than usual.

Finally, researchers [18] found that by increasing the ratio of connections' number to bandwidth, TCP converts from a simple periodic system to a system with fractal attractors and self-similarity will increase.

The important point in this view is the sensitivity of the system to initial conditions. If primary conditions change, for example, average number of connections becomes more than what is expected, and the system starts a new period. Accordingly, we can say self-similarity in TCP is due to short-range dependency.

According to what was discussed, we can say TCP network traffic which is made of users' flows aggregation, is a set of time series with the long-rage dependency that some of them are periodic and the others are self-similar chaotic, and TCP is the reason of their self-similar behavior. By the way, the average of these series is a time series with long-range dependency and the reasons of its self-similar behavior are the distribution of files and users' silence periods.

In other words, if we look at the network traffic from very large scales into medium scales, such a way that average of these small series is seen, we will find it long-range dependence. However, if the scale becomes so small, such a way that we face with these series, we will find it short-range dependence. Also, if scale becomes equal to the length of time series, we will be faced with long-range dependency again and traffic will be short-range dependence in smaller scales.

There is some method for testing self-similarity, but among of them, Whittle estimator [19] is more attractive, because it provides a confidence interval. Two kinds of this estimator are used frequently: the first is FGN[1] with H between ½ and 1, and it is only used for processes with long-range dependency, and the second is FARIMA (p , d , q)[2] with d between 0 and ½, which in addition to the long-range dependency, considers constant degree of the short-range dependency as well. d is the correlation function of this model and $d = H - ½$, $-½ < d < ½$.

[4] has focused on long-range dependency, therefore FGN has been used for self-similarity testing. While in [19] both kinds of dependency have been considered and FARIMA has been used.

Although network traffic follows Poisson and Exponential distributions in small scales and Gaussian and Log-normal distributions in large scales, none of them can cover network traffic in all small and large scales. Researchers [19] have found Gamma distribution[3] is able to model network traffic in all scales, as its flexibility based on its shape and scale parameters. Hence, they have used Gamma and FARIMA for self-similarity testing.

[1] Fractional Gaussian noise

[2] Fractionally autoregressive fractionally integrated moving average

[3] $\Gamma\alpha.\beta(x) = \frac{1}{\beta\Gamma(\alpha)}\left(\frac{x}{\beta}\right)^{\alpha-1} \exp(-\frac{x}{\beta})$, depends on two parameters: the shape α and the scale β.

Another research [11] improved classic ARMA model and proposed an algorithm which gets correlation function of the input traffic as the input and corrects this function by periodic matching of this parameter with the current point during process modeling. It has shown that this algorithm works better than FARIMA[4].

2.1.2 Temporal properties:

As you know, network traffic is completely time dependent. Therefore, models that work with temporal data have been proposed. In these models, input traffic sample has been considered as a time series and used for model learning. Models such as ARMA and ARIMA belong to this group.

2.1.3 Behavioral Oscillation:

Behavioral oscillation is the other property of traffic while less attention has been paid to. [12] and [13] have found these properties as important as the others. Protocol behavior, user behavior and network events such as the change in the routing table and TCP mechanism can cause this oscillation.

In [14] oscillations have been classified into two groups: Stationary and Non- Stationary. Researchers of this article believe users' daily activities are the reason of stationary oscillations and non- stationary oscillations are due to attacks, anomalies and E-commerce activities. In this study, a two-

[4] Autoregressive integrated moving average

layer Hidden Markov Model is used for modeling of these oscillations. Stationary oscillations in the first layer and non- stationary oscillations in the second layer have been modeled.

2.2 Flash Crowd properties:

Investigation [15] has shown that in Flash Crowd distribution of requested files is heavy-tail.

In information theory, Shannon entropy is the expected value (average) of the information contained in each message. In fact, the entropy of an event shows the amount of its randomness and disorder. More randomness is equal to more entropy and vice versa.

$$H(X) = -\sum_{i} P(x_i) log_b P(X_i)$$

In another study, [16] the amount of disorder or entropy of users behavioral has been considered as the reason of self-similarity. They believe in Flash Crowd much people follow the same goal in the same time, as result entropy decreases and self-similarity increases. In a research done in Sharif university , [17], it reached the similar result and observed self-similarity increment. Furthermore, this result is matched with achievements of [18]

which expressed increment in the ratio of connections' number to bandwidth, increases self-similarity.

However, in other investigation, [19], different results were observed. In this study, Gamma and FARIMA were used for self-similarity testing, normal traffic has both long and short dependency and was self-similar in all of the scales, while none of the dependencies were observed in some scales of Flash Crowd (250 milliseconds until 2 seconds).

As you can see, these results are in contrast to achievements of before studies [16], [17], [18]. Here two possibilities can be discussed: the first, results of this article are invalid and the second, these results are valid and we must find a justification for this contrast. In following, by assuming the rejection of the first possibility, we try to explain this justification. Validation of this hypothesis will be investigated as future work.

As previously mentioned, some of the researchers [18] expressed increment in the ratio of connections' number to bandwidth, increases self-similarity. In Flash Crowd, file distribution is heavy-tail and the aforementioned ratio is increased as well, but according to [19] in some scales, not only self-similarity is not increased but also it is completely faded. The question is why it happens?

Perhaps we can explain this contrast by definition of a threshold for aforementioned ratio [18]. When this ratio is smaller than the threshold, self-similarity is increased by increment of aforementioned ratio, but when it becomes bigger than the threshold, self-similarity is decreased by increment of aforementioned ratio, until self-similarity completely fades. Since the system is absolutely sensitive to initial conditions, when the number of connections constantly becomes more than what a system expects, the system will constantly change as well. It means TCP periods become so small that dependency completely fades. In other words the system becomes strongly unstable ($0 < H < ½$).

On the other hand, if we investigate details of aforementioned study [17], we will see this reasoning is compatible with its results. Although in this study self-similarity increases but the amount of this increment in Sharif University FE was 0.2 while it was only 0.06 in World Cup 98. Whereas flash crowd of the first is very smaller than the second and according to [16], [18] the increment amount of second must have become more than the first.

Now if we consider offered assumption about the obligation of threshold definition, we can reason the flash crowd of the second has become so much that aforementioned ratio has exceeded the threshold and the amount

of self-similarity has decreased. Perhaps if flash crowd exceeded more and more or the scales became as small as [19], the system was under temporal changes, self-similarity became gradually few and fewer and finally, faded.

2.3 Related works:

Traffic generation tools are divided into two groups:

1) Traffic generation based on input traffic sample: in this class, content and timing of input traffic sample are identically repeated, Such as TCPopera, TCPivo, TCPReplay.

2) Traffic generation based on analytic models: in this class input traffic sample is simulated by using statistic models. Some of them have a simple method and just extract distribution of file size and time intervals from input traffic sample and then generate traffic according to them, such as TG [55], LitGen [56]. Others use more complicated methods and simulate different layers by the help of analytical behavior Models, such as Harpoon [57], Tmix {58] and D-ITG [59]. The others, such as Swing [60], further behavior models, model network behavior as well.

The important point about these tools is that they cannot implement TCP. Although some of them such as D-ITG have tried to approach TCP

implementation by the use of bidirectional flow generation but they practically make a UDP flow with TCP packets format which we can only determine its packet size and inter-arrival time. For TCP generation the help file indicates that one side generates a byte stream based on input traffic rate and waits until another side generates its stream as well. However, this static solution cannot be used as TCP, because of dynamic operation of TCP window. So these tools are not suitable for Flash Crowd generation despite we can use them for DDoS simulation.

Some efforts have been made for Flash Crowd modeling, such as [20] which has proposed a simple model for Flash Crowd with three phases: rise, stability, and decay. Writers of this article model rise and decay linear, but in [21] they have been modeled by a quadratic function. Also, [2] has modeled these phases by an exponential function. These researchers used this model for traffic generation in their next study [22]. This generator can almost simulate Word Cup 98 but its weakness is the need to input traffic sample which is not available in many cases.

2.4 The differences between DDoS and Flash Crowd:

DDoS deliberately performs attacks to waste server resources and disable it. Most of the time, these attacks combine with Flash Crowd or mimic it

for deceiving IPS and IDS tools. Nevertheless, distinguishing between them is an important challenge for the server for better servicing to the legal user and repelling attackers. If a suitable test and verification environment is provided for evaluation of IDS and IPS tools, they can help the network administrator to differentiate between legal users and bots. The first step to provide this environment is to characterize properties of each case be able to differentiate these two anomalies.

2.4.1 Ordinary DDoS:

2.4.1.1 Changes in rate of input traffic:

Although we observe received packets rate growth at the server side in both cases of FE and DDoS, but this growth in FE is gradual while it is rapid and abrupt in DDoS. Also, after peak request rate, the decay is gradual in FE and sudden in DDoS [15], [24], [23].

In [23] received packet rates in CAIDA and World Cup98 were investigated and observed. They showed that required time up to reach the peak was about 2 hours in World Cup98 while it was only about 10 minutes in CAIDA. (Figures 2.1 & 2.2)

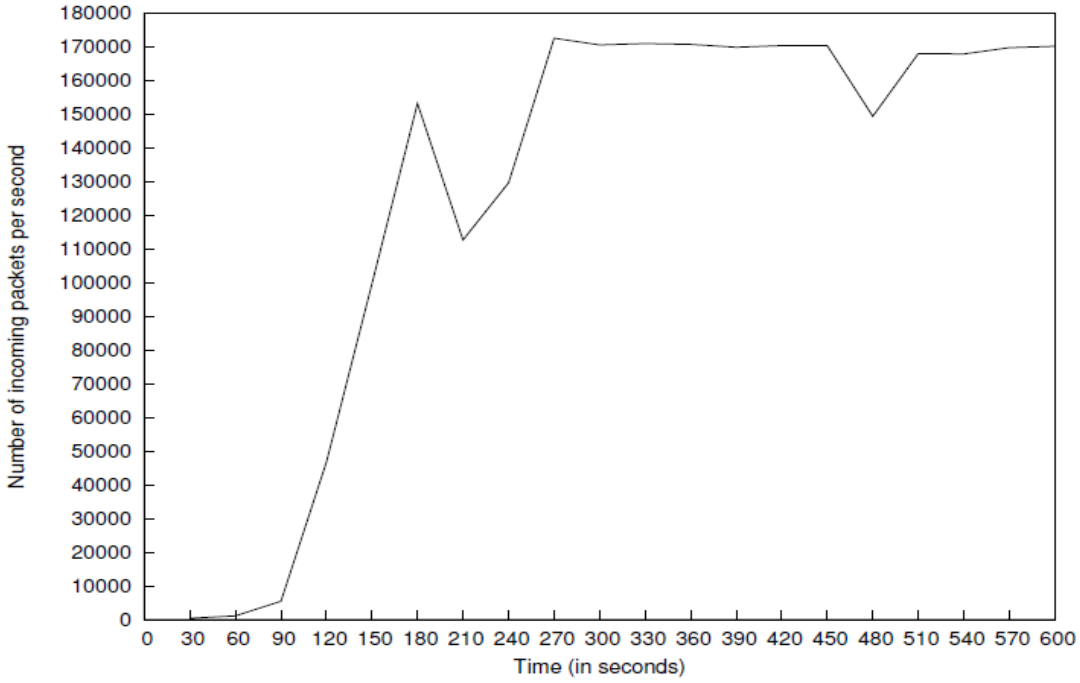

Figure 2.1 - CAIDA-DDoS [23]

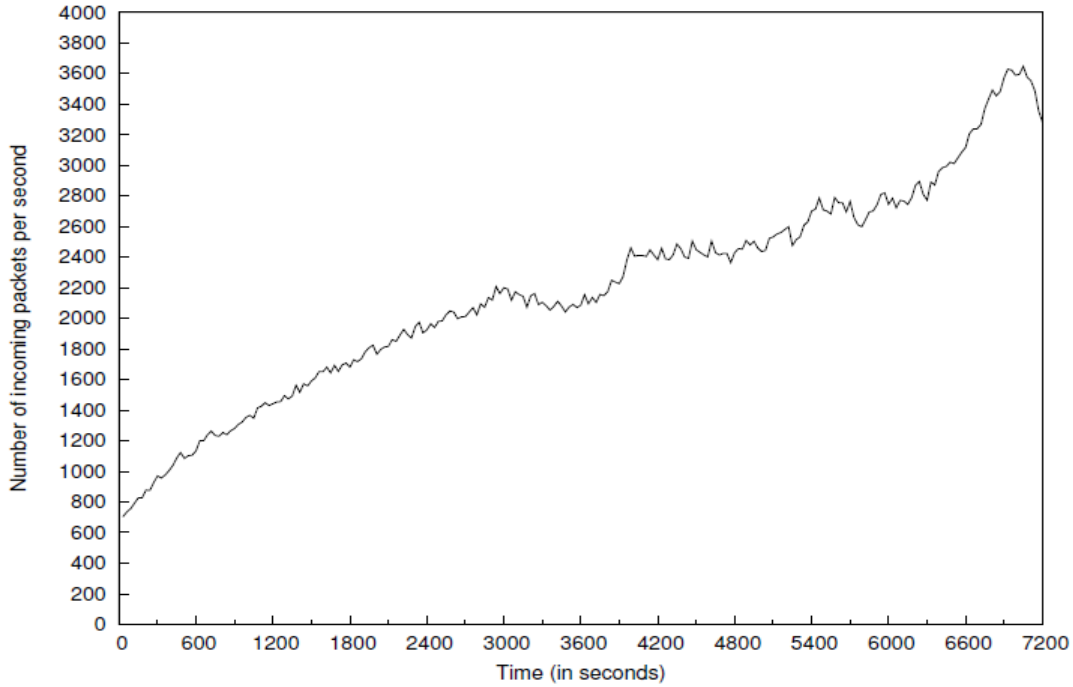

Figure 2.2 - World Cup 1998 [23]

2.4.1.2 Changes in entropy of IP addresses:

DDoS attacks use bots for their goals. Due to bot number limitation and the possibility of their inactivity for reasons such as update and security patches of defensive tools, system reinstall or being turned off [25], these attacks are forced to use their maximum capacity. As a result, we observe a wide range of new IP addresses during the start of these attacks afterward, they repeat or even decrease. However, in FE, new IP addresses are constantly observed because news are gradually propagated between users. [23], [27]. (Figures 2.3 & 2.4)

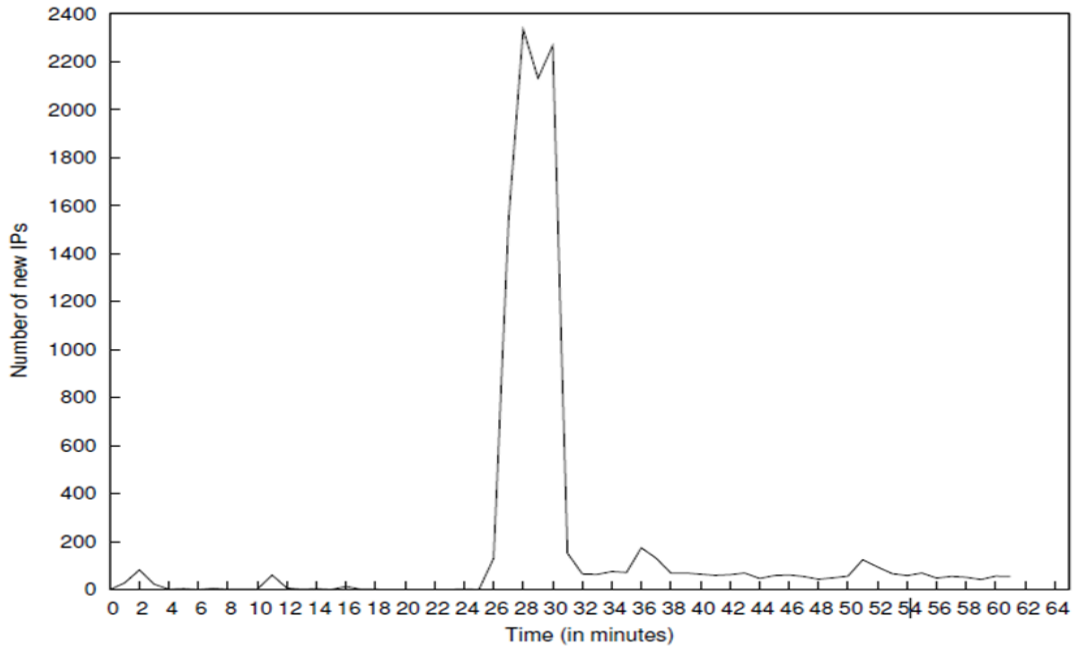

Figure 2.3 - New IPs appearance rate in CAIDA with 1-minute memory [23]

Also, researchers have found [24] the probability of repeated IP appearance at the beginning of DDoS attack is low while it increases during the time because of bot number limitation. Reciprocally, repeated IPs are numerous at the start of FE, as permanent visitors or users behind proxies and NATs

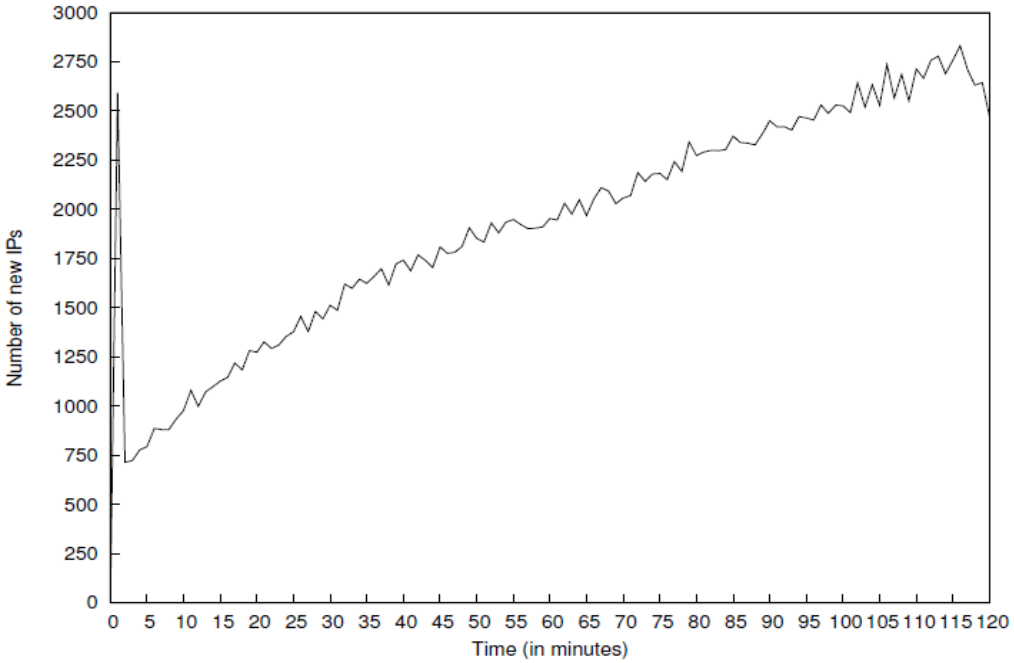

Figure 2.4 - New IPs appearance rate in FE with 1 min memory [23]

might have caused this effect, and this probability increases during the time due to news propagation.

2.4.1.3 Distribution of requests:

Researchers in [15], [23], [26] have figured out that the reason of additive requests rate is deferent in FE and DDoS. This increment is due to user

requests growth in DDoS while in FE, average of user requests is typically constant and this increment is due to users' growth.

Indeed, although users' request rate is more than normal traffic at the start of FE anomalies [26], this rate due to prolongation of response time decreases afterward [15]while the rate of each bot is increased or at least remains constant during the DDoS attacks.

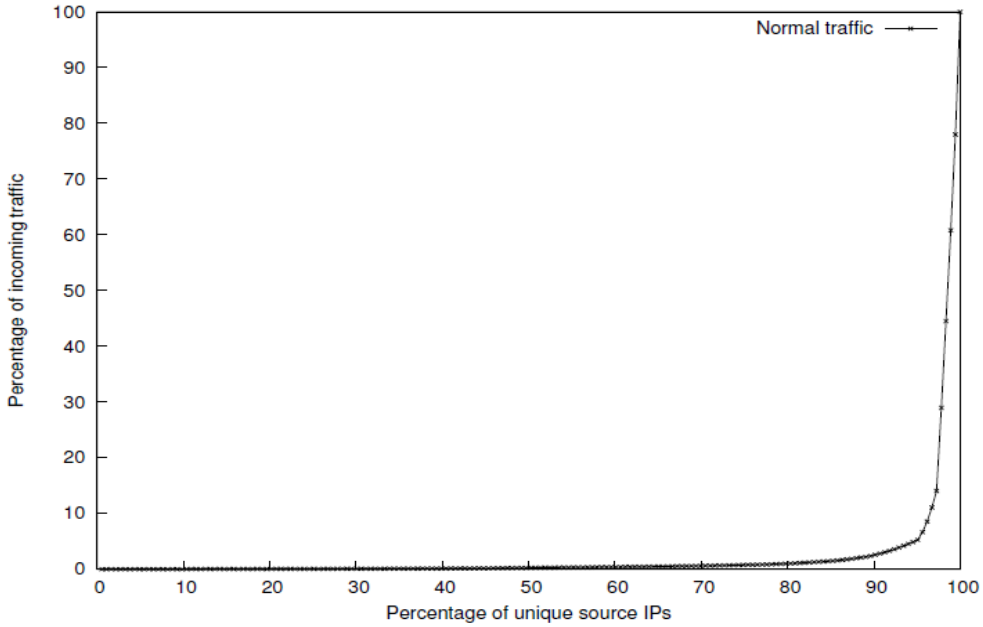

Figure 2.5 - Requests distribution per each IP in normal traffic [23]

The number of requests per IP in normal traffic, DDoS, and FE has been demonstrated in Figures 2.5 and 2.6. As you can see the amount of requests per user of FE in normal traffic does not have considerable increment, while this increment in DDoS is clearly observable.

Another study [23] showed that DDoS traffic comes from all participant IPs in attack monotonically. But in FE, vast part of traffic are made by a small number of IPs belonging to users who are behind proxies and NATs. In fact, big clusters are the main reason of traffic load. Researchers figured out that distribution of traffic load is Hyper-exponential in DDoS and Erlang in FE.

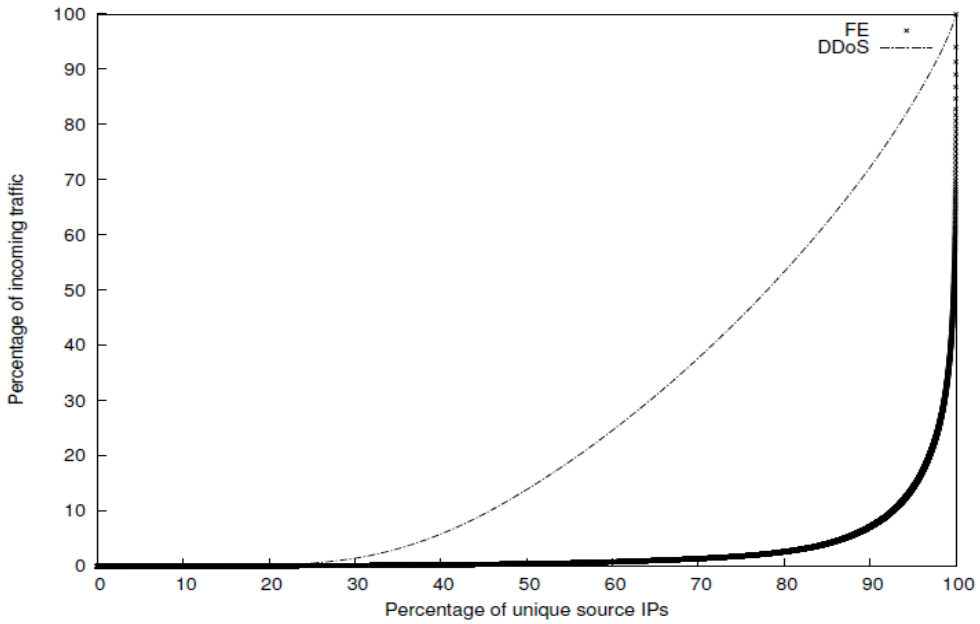

Figure 2.6 - Request distribution per each IP in DDoS & FE [23]

2.4.1.4 Geographic distributions' users:

DDoS participants come from all over the world and number of clusters in these attacks is equal to the number of bots. Therefore, the geographic distribution of IPs is monotonic in these attacks while in FE most of the

clients are in the joint cluster. As a result, the number of clusters in FE is much smaller than the number of users and its distribution is unbiased [15], [24], [27]. Also, a few attacker clusters have already attended in normal traffic while most of the clusters which are active in FE have already been observed in normal traffic [15].

2.4.1.5 Distribution of requested files:

In Flash Crowd, distribution of requested files follows ZIPF rule, but in DDoS requests have limited file numbers without following any rule [15].

2.4.1.6 Traffic pattern:

The behavior of real users (rate of file requests and inter-arrivals time) is not predictable and is completely random. As a result, generated flows by legal users are not dependent. Bot nets need many simultaneous active bots for achieving their goals i.e. sending many requests similar to legal user to the victim [28]. However, this it is not always possible because not only the number of alive bots is limited [29], but also practically at most almost 20 percent of them are simultaneously active [30], [31].

Because of this limitation, each bot must send requests hundreds of times more than a legal user. Indeed, it has to generate the request by its maximum capacity. On the other hand, DDoS attacks are programs which have been already written by attackers and often execute the same

instructions on the Bots [31], [32]. These reasons cause bot flow distributions become similar, and as a result, their pattern will be predictable.

2.4.1.7 Users' request speed:

Researchers [34] have figured out that all of the DDoS participants typically have the same transmission speed because any bot forces to use its maximum capacity for its requests transmission. Whereas FE is made of real and legal users, their transmission speeds are different. Aforementioned differences have been summarized in Table 2.1.

In another study [27], entropy has been used for evaluating the randomness of IP addresses and distinguishing between FE and DDoS. This research has shown that traffic entropy changes are rapid and numerous in DDoS, but the intensity of this increasing decreases next. While in FE entropy changes are sizeable but these changes happen gradually and during the time with soft slop.

Researchers [35] detected suspicious nodes by calculating the difference between the entropy of input traffic before and after anomaly and then investigated the network in terms of the correlation coefficient for determining the probability of network flow. If this probability was more than 60 percent, the flow is the distinguished attack.

Table 2.1 - Differences between DDoS & FE

	Property	FE	DDoS	References
1	Change in input traffic rate	Gradual increment and decrement	Rapid increment and decrement	[1][3][6]
2	Reason of traffic rate increment	User numbers increment	User requests increment	[1][3][8]
3	Entropy change	Gradual growth during the anomaly	Rapid growth in start and consolidation in continues	[1] [5]
4	Requests distribution	Erlang	Hyper exponential	[1][3][5]
5	Users geographic destruction	Unbiased	Monotonic	[3][5][6]
6	Clusters appearance historical	Most of clusters have been previously seen in normal traffic	Only a few of clusters have been previously seen in normal traffic	[3]
7	Requested files distribution	Follows ZIPPF law	Is centralized on limited files	[3]
8	Traffic pattern	Unpredictable	Predictable	[9][14]
9	Users transmission speed	Different transmission speed	Same transmission speed	[7]

In [36] attacks were detected by the calculation of Pearson correlation between input rates and time.

Other researchers [37] believe that if their desirable conditions are provided, they will be able to separate DDoS from FE by coefficient correlation calculation between flows. Said conditions are following: 1) length of flow must be big enough. 2) DDoS must be sufficiently strong in comparison with the other flows. In other words ratio of the signal to noise must be sufficiently high. Anyway, during the attack, number of packets in DDoS flows are more than 10 times higher than normal flows [38].

Researchers [28] figure out that theoretically if Botnet is sufficiently big, the behavior of the legal user can be completely imitated. As a result, separation of the attacker and the legal user is not possible. But as previously discussed, some researchers [29] believe that it is not possible practically, because not only number of alive Bot is limited during attacks but also only 20 percent of them are simultaneously active [30] [31].

Researchers [29] improved standard deviation of Bots aggregation flow is equal to the standard deviation of each Bot flow and is lower than the standard deviation of flow which is made by legal users and used for separating attacks and legal flow.

2.4.2 Advanced DDoS (FE imitators):

Although ordinary DDoS attacks and legal user traffic can be separated by means of packet header analyzing and aforementioned differences, still we face another class of attacks which try to mimic FE. As previously mentioned, it is the essence of FE that requests do not increase suddenly. Accordingly, attacks which try to mimic FE do not need to have large number of concurrent active bots.

Therefore, a DDoS attack with a medium size Botnet can avoid flows similarity by means of various request rate and inter-arrival time by the help of various distributions.

These types of attacks cannot be detected by current detection methods. Hence, we must focus on the behavior of application layer instead of low layer analysis for detection of these attacks. By means of this method, one can differentiate human from machine.

In this regard, in another study [25] a set of decoy hyperlinks has been used for recognizing human from machine. These links are hidden pixels inside of pictures or tables and page isochromatic which are hidden for users. Accordingly, probability of requesting these links by legal user is near zero.

In another study [39], browsing behavior of legal users was modeled by means of hidden Markov chain. For each request traffic sample, standard deviation of its average entropy from the average entropy of model is measured. If it is greater than determined threshold, aforementioned traffic sample is introduced as the anomaly and dropped by intrusion detection filters. Although ordinary DDoS attacks and legal user traffic can be separated by means of packet header analyzing and aforementioned differences, still we face with another class of attacks which try to mimic FE. As previously mentioned it is essence of FE which requests do not increase suddenly. Therefore attack which want to mimic FE does not need to have large number of concurrent active Bots. nevertheless, a DDoS attack with a medium size Botnet can avoid flows similarity by means of various request rate and inter-arrival time by the help of various distributions.

These type of attacks cannot detectable by current detection methods. Therefore it must be centralized on behavior of application layer instead of low layers analyzing for detection of these attacks. By means of this method can be recognized human from machine.

In this regard, in other study [25] set of decoy hyperlinks have been used for recognizing human from machine. These links are hidden pixels inside

of pictures or tables and page isochromatic which are hidden for users. Hence probability of requesting these links by legal user is near zero.

In another study [39] browsing behavior of legal users was modeled by means of hidden Markov chain. For each request traffic sample, standard deviation of its average entropy from the average entropy of model is measured. If it is greater than determined threshold, aforementioned traffic sample is introduced as the anomaly and dropped by intrusion detection filters.

2.5 Conclusion:

We can conclude that all distinctive characteristics of DDoS and FE must be considered to generate a desired FE. Nevertheless, the following issues are important for providing technical framework of our project:

1) The increment rate must be gradual.

2) Two groups of IP must be used: the first, which is representative of big clusters, and are due to using Nat and proxy, and the second is anent singular users. Main weight of requests must be allocated to the first group.

3) Before start of FE generation, normal traffic must be generated and most of cluster IPs must be seen in this normal traffic.

4) In the beginning of anomaly, repeated IPs must be used. These are representative of frequent users. In addition, new IPs must be gradually added during the anomaly. It shows news propagate.

5) Since atone side, most of first kind attacks distinction tools are based on the existence of flow correlation in DDoS and

lack of it in FE, and at the other side, progressive tools for second attacks distinction concentrate on similarity of FE flows and real user flows, therefore aforementioned distinctions must be taken by simulation of legal user behavior. This happens by considering 4 points:

1. Variety in user' transition speed or their active silence time.
2. Variety in users' inactive silence time.
3. Variety in flows length.
4. Imitation and simulation of real user behavior.

As a result, we must only try to simulate real user behavior, because the fourth point consists of all of previous points.

6) Web pages must be accessed based on ZIPF

law. It means 80 percent of views concentrate on only 20 percent of pages.

7) Before traffic generation, decay hyperlinks must be detected and omitted.

8) User behavior in FE for dissuasion case and temporal leaving of the website must be simulated.

Chapter Three
Proposed approach

In this Chapter, we propose our approach for FE traffic generation according to previous determined purposes. This solution uses a set of executing machines for reaching this goal.

At first, a directional graph is made from the inside links of the web pages. The program launches homepage address as the root of the graph, the inside links of this page are detected and null hyperlinks, which are used for robot detection, are discovered and dropped. Our purpose is to create a traffic load on considered website, other domains links are omitted. Finally, remained links are added to graph as children or neighbors of this node and directional edges connect this node to its neighbors. This procedure is repeated for next nodes.

In FE, 80 percent of requests only concentrate on 20 percent of pages, and accordingly if server logs are available, this graph will be divided into two subgraphs which are included in most visited paths and remained paths. They are sent to all executing machines. Each machine follows first graph paths with probability of 0.8 and the second with probability of 0.2.

Each executive machine is representative of some users and multithreading is used in order to implement it. Main algorithm is executed for every user on considered machine in the form of independent processes.

3.1 User behavior simulation:

As was discussed in Chapter 2 and it is remarked again because of its importance, most of detection tools which are used for distinguishing between first kind DDoS and FE, act on the basis of the existence of dependence or similarity in DDoS flows and their absence in FE. Also, progressive tools which are used for distinguishing between second kind DDoS and FE, concentrate on legal user behavior similarity. Nevertheless, we must try to simulate legal user behavior. In order to reach this goal, following arrangements have been considered:

3.1.1 Start page selection:

Users input a website by means of search engines, following special contexts, typing in browsers, directly bookmarks, and etc. Therefore the start point of browsing is not necessarily the homepage, however, the probability of homepage selection is more than the other pages. Hence, the homepage is considered as the start node with the probability of 0.5 and the others are chosen with equal probability and a total amount of 0.5.

3.1.2 Browser behavior simulation:

Users typically use browsers for accessing web pages. Accordingly, we attribute one browser and one operating system to each thread for pretending to be a legal user, and the following actions are done:

1) Request headers are changed into famous browser headers.

2) For each node, it is added to its neighbor list for simulation of browser refresh button.

3) Parent of current page is added to its neighbor list for simulation of browser back button.

4) As already explained, bookmark operation is simulated.

5) The order of page' objects request is investigated in different famous browsers and applied according to selected browser.

Table 3.1 is the sample of aforementioned order:

Table 3.1 - Object fetch in different browsers

Browser	Background image	Main image	Js objects	Css objects
Firefox	always regular always regular-according code	always regular-according code	always regular-according code	always regular-according code
Google chrome	always regular-according code	Erratic	always regular-according code	always regular-according code
Internet explorer	always regular-according code	Erratic	always regular-according code	always regular-according code

In abovementioned browsers, at first, Css objects then Js objects and main images and finally, background images are requested.

3.1.3 User behavior simulation:

As you know, different users have different behavior for browsing context of a website. For example, some of them open a page and study its total context, then go to a new page. Or some people read a few context of a page and temporally leave it for visiting links inside it and then go back to first page again. This may repeat again and again. In Figure 3.1 browsing behavior of different users has been demonstrated by means of an state machine.

This machine has been limited by depth variable which shows user browsing depth. When user opens a page for reading it, he or she goes to one of the silence states. Partial silence is due to the state that user investigates part of opened page and temporally leaves it. Complete silence is related to the state that a user leaves a page for ever; in this situation, the user may close the tab of this page or not but does not refer to it again.

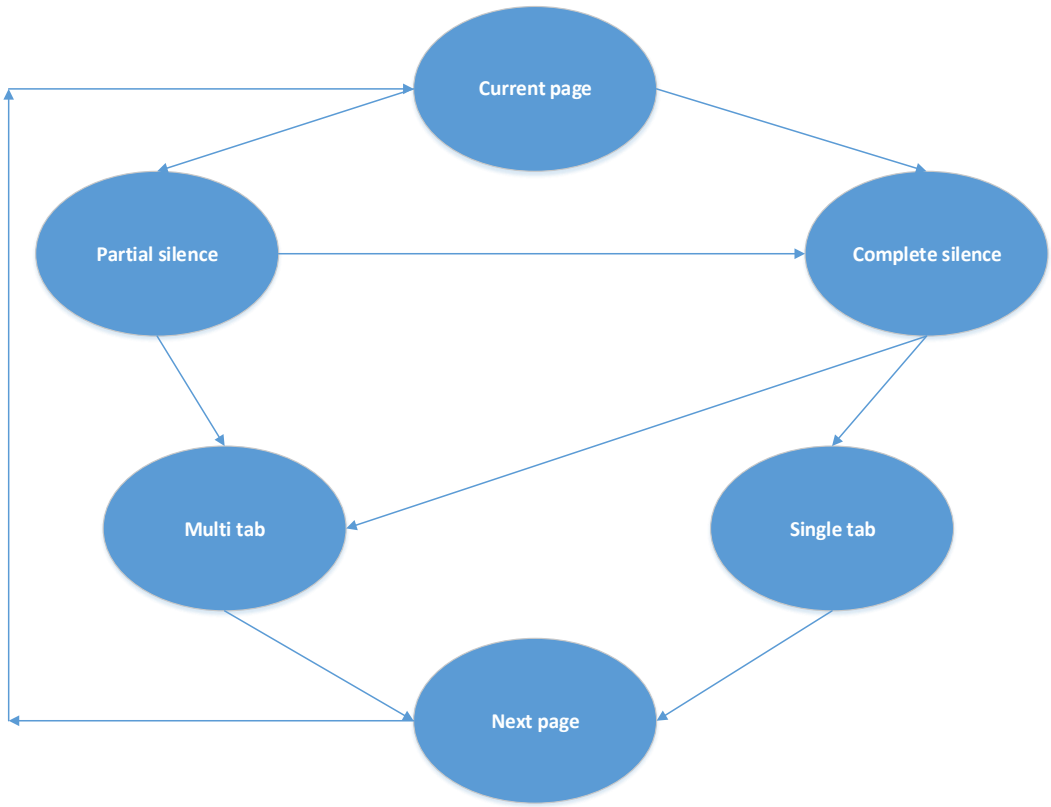

Figure 3.1 - State machine of users browsing behavior

After silence, the user may continue his or her browsing in previous tab and go to single tab state or can open one or more tabs and go to multi tab state. This process can be continued.

3.2 Users grouping according to their trading history with the website and founding parameters like traverse depth, visit time, and number of tabs which are opened by user to any depth:

Studies have been hitherto done about user behavior browsing, and typically concentrate on user silence time only; we investigate them in the following.

A session is set of user requests from a website. The user sends his or her request by typing, clicking on a hyperlink, and etc. This request retrieves a basic file on the server. User browser automatically requests all of attached files as well. The time taken for receiving all attached files is called *active off*. After it, the user reads the requested page; this time period is called *inactive off* or *think time*.

If user does not use concurrent multi-tab and follows his browsing activities in the same tab, active silence will be equal to the time that user spends on the page. Otherwise, we cannot imitate these activities, because user may refer to his previous requested page again and again, or open consecutively some pages and refer to them later. In the next we will

investigate studies done in this context and classify them into two categories, before and after multi-tab appearance:

3.2.1 User silence time before concurrent tabs appearance:

In 1997 researchers [40] observed that active silence time was between 1 millisecond and 1 second, and inactive silence time was between 30 and 3000 seconds. While other study [41] done in the same year on the HTTP traffic, measured the average inactive silence to be about 15 seconds. Also, some researchers [54] found it about 64 seconds.

Other study [42] was done on server side traffic in 1998. It showed user silence time follows Weibull distribution with average of 39.5 s in server side. On the other hand, and as shown in another study [43] in 2002 different results have been observed. In this study researchers did not access to context of webpages, therefore they introduced two definitions for pages' delimitation in traffic sample:

1) When an interruption of more than 1 second occurs between two requests: in this situation Pareto distribution with parameters a=1.29 and k=6.3 is representative of users' silent times. In this distribution, a and k, are shape and scale parameters respectively.

2) When in addition to the last condition, all requested objects are located in a common server: in this situation Pareto distribution with a=0.99 and k=1.07 is suitable.

3.2.2 User silence time after concurrent tabs appearance:

Researchers of [44] investigated YouTube video stream and claimed that average silence time is about 30 seconds. On the other hand, researchers of [45] tracked video streams and figured out that outages between 1 and 4 seconds are due to network delay, while outages more than 8 seconds are due to user silence time.

At last, in a study which was conducted in 2009 [46] collected data from an add-on were perused and silence times were fitted to Weibull and Exponential distributions. This study showed that Weibull distribution was more suitable.

As already discussed, before appearance of multi-tab, silence time which was in fact visit page time has been surveyed in some studies. Some of them found it proportionate with Weibull and the others with Pareto. Besides, other study which was done after multi-tab appearance, used server side data and hence in this study silence time cannot be representative of page visit time. Furthermore, all aspects of user behavior have not been investigated in any research and all of them only

concentrated on user silence time, and there is no consensus about this case. Nevertheless we decided to collect our required dada statistics directly and analyze them to obtain our needed information.

There are two methods for data gathering with each one having its strengths and weaknesses: [47] [48] [49]

1) Data gathering in server side: in this approach number of investigated users is desirable but we do not inform about detail of user browsing. For example, has user continued his browsing in the same tab or opened new tabs. Moreover, server side logs are private and sensitive, hence they are not available always.

2) Data gathering in client side: in this method we can carefully track users' activities but number of users is limited. On the other hand, we need to design special software, compatibility of designed software and different browsers and operating systems are challenging.

In another study [50], it was observed that 30 percent of opened tabs are visited again. In [51] this number was increased to 59.6 percent of opened tabs and it was about 57.4 percent in [48]. Today, parallel browsing and multi-tab using are more common than linear browsing; accordingly, we use the second method of data gathering for user behavior modeling.

All of studies which have investigated on user behavior browsing [47] [48] [52], used an appropriate add-on for collecting users' data. We design a dedicated add-on for Firefox browser and prepare a test to study users browsing habits.

This test was done on 63 people including 43 men and 20 women. It had two steps: in the first step user selects a website that he or she knows and usually visits and browses . In the next step this test is repeated for a website which is new for user. Furthermore, user should not open video files, because we want silence time only represent user behavior and becomes independent of Internet speed.

In 2006 some investigators observed [53] that page presence time was less than 4 seconds for 25 percent of pages and less than 10 seconds for 52 percent of them and more than 2 minutes for 10 percent. At first, researchers supposed that these delays were small because of back button browsers or visiting a page which is seen lately. Hence they omitted these cases, but considered times to be still small. They figured out page presence time has only a few dependences on page words number. Nevertheless, they deduced that the reason of small times is that users do not study the whole page but only scan it. Therefore, because of this independence, difference between selected websites will not have negative effect on our

results and we only ask users to choose websites which are attractive for them.

We therefore designed add-on logs for four-page events:

1) New page opening:

The new page can be opened in the previous tab or in a new tab. A new number is allocated to this page and if it opens in a new tab, a new number will be allocated to this tab as well. Number of previous page containing the link of this new page is logged during this event as parent node. Time of this event is logged as well, and state of page changes to N which means New.

2) Page visiting:

During this event, which happens after total page loading, time is logged and state of page changes to A which means is it Active.

3) Temporal page leaving:

During this event, time is logged and state of page changes to D which means DE-active.

4) Page closing:

During this event, time is logged and state of page changes to C which means Close.

Each event is presented by 4 elements:

1. tab and page number of current page.
2. tab and page number of parent page which is only set during new page opening event and becomes Null in other events.
3. event time.
4. event name.

Each tab and its related page are presented by a decimal number and are separated by a dot. The tab number is at its left and the page number is at the right hand side of the dot.

The following example is part of user browsing behavior which is collected by our add-ons. As can be seen in line 2, page 5 from tab 1 has been left temporarily, and in line 12, the user referred to it again.

1. |1.5, Null ,1472727290658,A|background.js:12:6
2. |1.5, Null ,1472727295176,D|background.js:10:6
3. |5.0, Null ,1472727295176,A|background.js:12:6
4. |5.0, Null ,1472727295177,D|background.js:29:5
5. |5.1, 5.0 ,1472727295177,N|background.js:30:11
6. |5.1, Null ,1472727295177,A|background.js:31:5

7. |5.2, 5.1 ,1472727295177,N|background.js:18:6
8. |5.1, Null ,1472727295263,D|background.js:29:5
9. |5.3, 5.1 ,1472727295263,N|background.js:30:11
10. |5.3, Null ,1472727295263,A|background.js:31:5
11. |5.3, Null ,1472727307365,D|background.js:10:6
12. |1.5, Null ,1472727307366,A|background.js:12:6
13. |1.5, Null ,1472727351279,D|background.js:10:6

We code a program for analyzing these data. It calculates all distances between A and D events for every page in first step and in the next step draws browsing tree for each user by means of N events and represents tree depth and number of opened tabs in each depth. Finally, a distribution is fitted to page visit time and statistical frequency of browsing depth and opened tabs of each depth is determined. In the following, we investigate the results of this analysis:

3.2.3 Time of page visiting:

Number of opened and visited pages of our test was 1078 in first step and 860 in second respectively. Opened pages by new tabs which users did not visit, were omitted. User can open a page, browse it completely and leave it forever (complete silence state in Figure 3.1) or browse it partially and return to it later (partial silence in Figure 3.1). Therefore, we divided pages

into two groups and investigated them by fitting different distributions. Also, probability of user preference in each state was calculated. By studying silence times we found out 99 percent of logged silence times are more than 300 milliseconds. Accordingly we dropped smaller times as user opening fault.

3.2.3.1 Kolmogorov Smirnov test:

This method is used for matching data to a theoretical distribution. It is based on the largest vertical difference between the theoretical and the empirical cumulative distribution function. The null and the alternative hypotheses are:

- H_0: the data follow the specified distribution;
- H_A: the data do not follow the specified distribution.

The hypothesis regarding the distributional form is rejected at the chosen significance level (α) if the test statistics is greater than the critical value obtained from table. The fixed values of α are generally used to evaluate the null hypothesis (H0) at various significance levels. A value of 0.05 is typically used for most applications however, in some critical industries, a lower value of α may be applied.

The P-value, in contrast to fixed α values, is calculated based on the test statistics, and denotes the threshold value of the significance level in the sense that the null hypothesis (H_0) will be accepted for all values of α less than the P-value.

The probability-probability (P-P) plot is a graph of the empirical CDF values plotted against the theoretical CDF values. It is used to determine how well a specific distribution fits the observed data.

This plot will be approximately linear if the specified theoretical distribution is the correct model. We use EasyFit software for this test in our thesis. It shows the reference diagonal line along the graph points.

3.2.3.2 First step of test:

As previously mentioned, in the first step of test, users browse a webpage which they always visit. Collected data are divided into two groups:

1) Pages visited only once during browsing.

2) Pages visited more than once during browsing. It means each user refers to a tab opened previously.

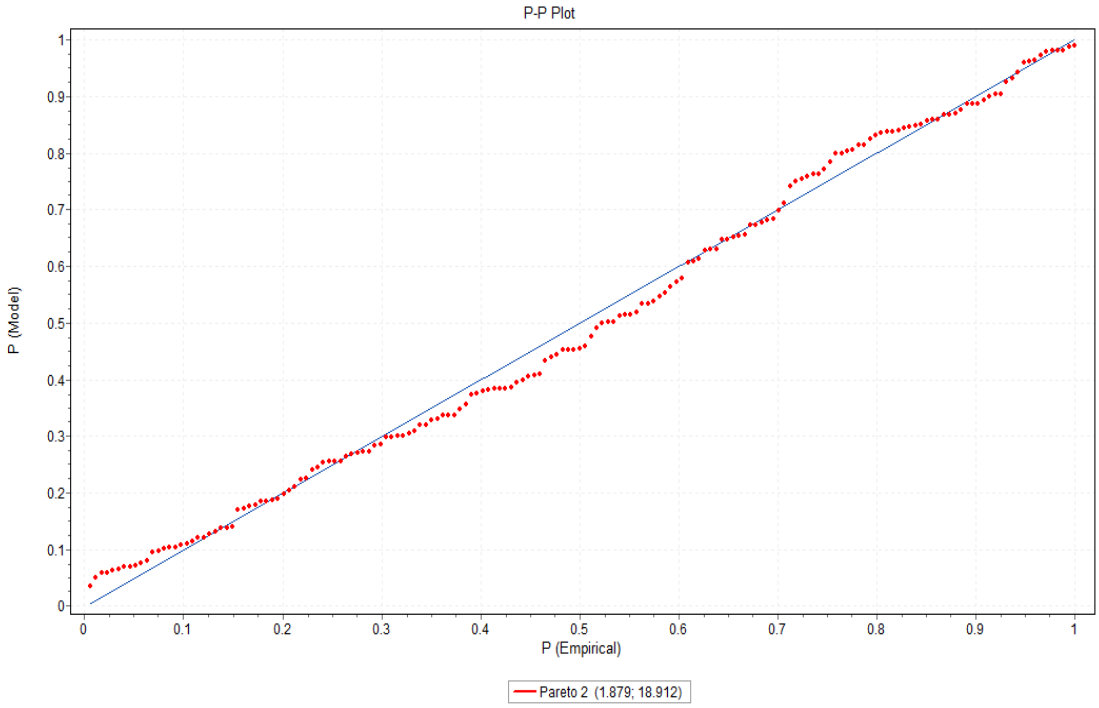

Figure 3.2 - Distribution of users' complete silence time in their familiar websites

K-S test fitted silence time of the first group to Pareto distribution with demonstrated parameters as shown in Figure 3.2 and p-value=0.24824 which shows null hypothesis is true.

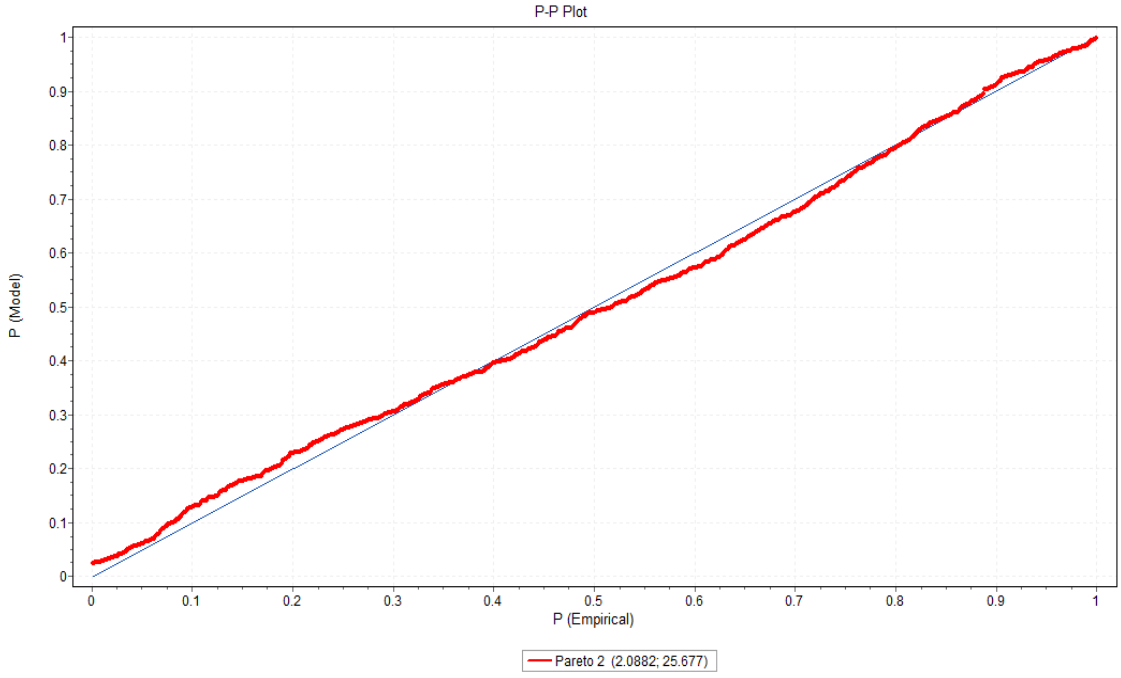

Figure 3.3 - Distribution of users' partial silence time in their familiar websites

Also, in Figure 3.3 fitted distribution to silence time of second group has been demonstrated with p-value=0.77274.

Among 1226 visited pages, 1167 pages belong to the first group and 59 pages were in the second group. Accordingly, when users are familiar with a page, he goes to complete silence state with probability of 0.95 and go to partial silence state with probability of 0.05. Also, the amount of page revisiting and change of state machine from partial silence to complete silence (Figure 3.4) was determined by means of Table 3.2

Table 3.2 - Number of users' page revisiting in their familiar websites

Number of users' page revisiting	Numerical frequency	Probability
1	41	0.69
2	8	0.13
3	5	0.084
4	2	0.033
5	1	0.016
6	1	0.016
7	1	0.016
sum	59	1

3.2.3.3 Second step of test:

In this step, users browse a webpage which has not been visited yet. Collected data are divided into two groups:

1) Pages were visited only once during browsing.

2) Pages were visited more than once during browsing. It means user again refers to a tab opened previously.

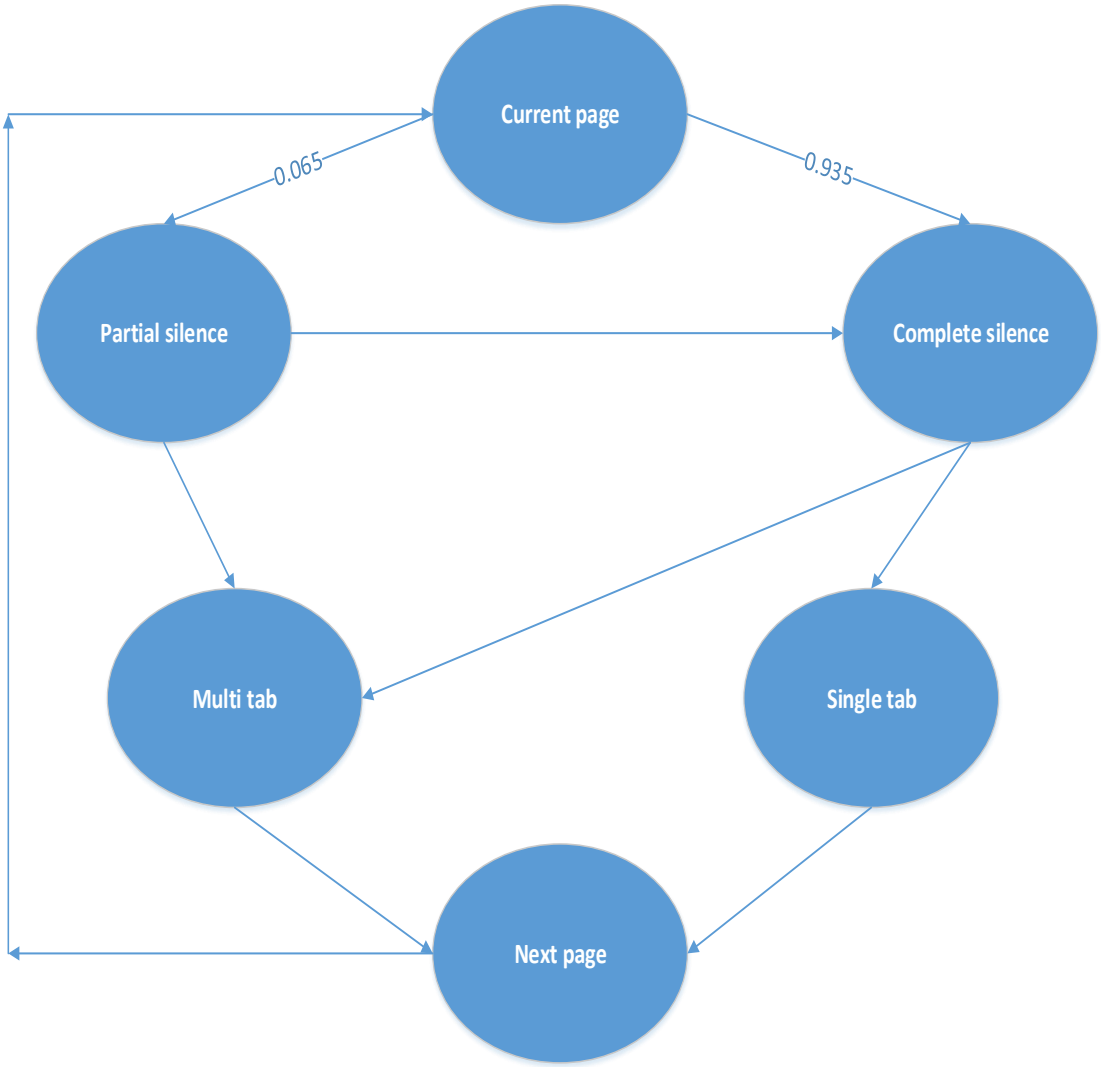

Figure 3.4 - User state machine in his familiar website

K-S test fitted silence time of first group to Pareto distribution with demonstrated parameters in Figure 3.5 and p-value=0.36531 which shows null hypothesis is true. Also, in Figure 3.6 fitted distribution to silence time of second group has been demonstrated with p-value=0.18437.

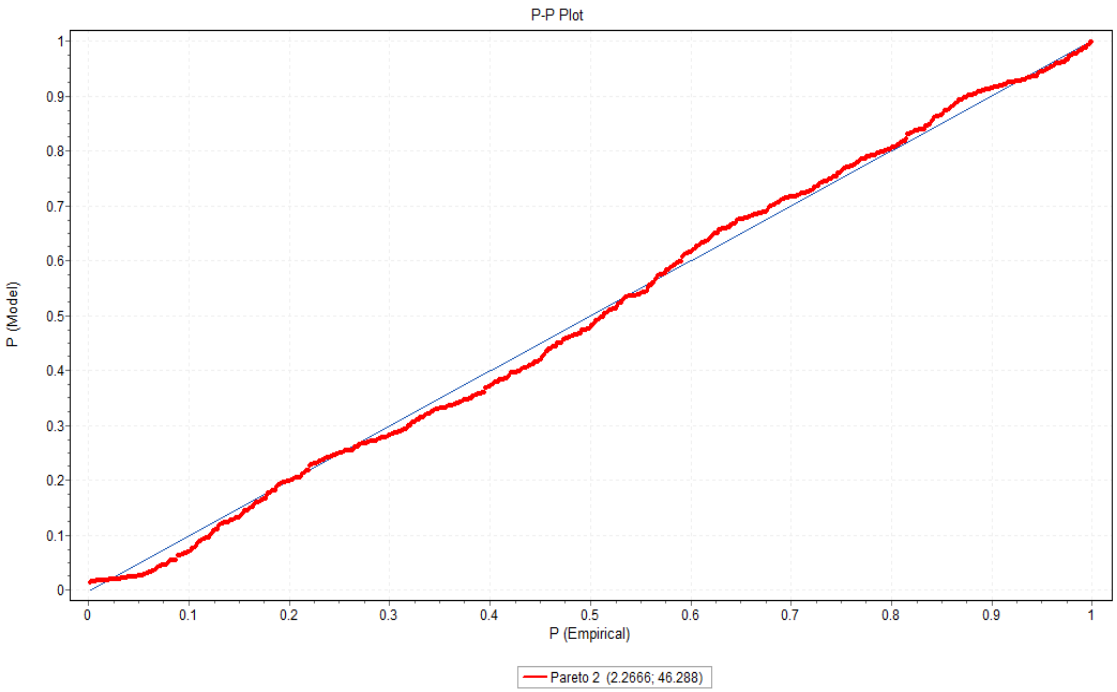

Figure 3.5 - Distribution of users' complete silence time in their unfamiliar websites

Among of 1043 visited pages, 948 pages belong to first group and 95 pages were in second group. Accordingly, when users are familiar with a page, go to complete silence state with probability of 0.91 and go to partial silence state with probability of 0.09.

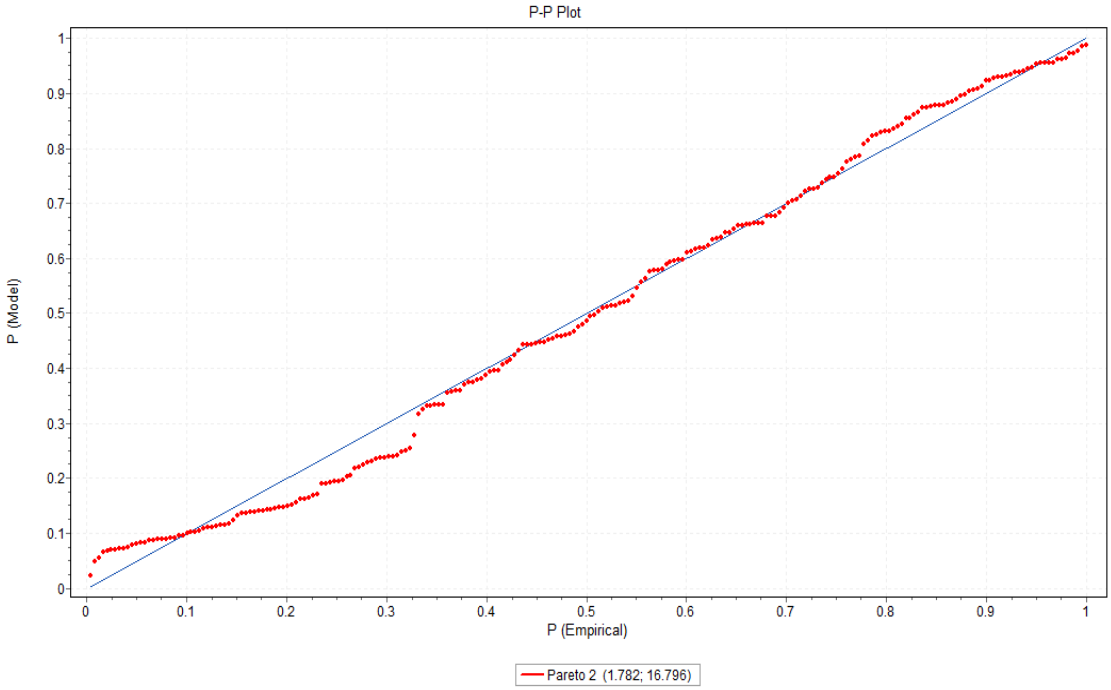

Figure 3.6 - Distribution of users' partial silence time in their unfamiliar websites

Also, amount of page revisiting and change of state machine from partial silence to complete silence (Figure 3.7) was determined by means of Table 3.3.

Therefore, derived random numbers from aforementioned distributions are considered as visit time or complete silence in main algorithm.

Table 3.3 - Number of users' page revisiting in their unfamiliar websites

Number of users' page revisiting	Numerical frequency	probability
2	6	0.063157895
3	6	0.063157895
4	3	0.031578947
5	3	0.031578947
7	2	0.021052632
Sum	95	1

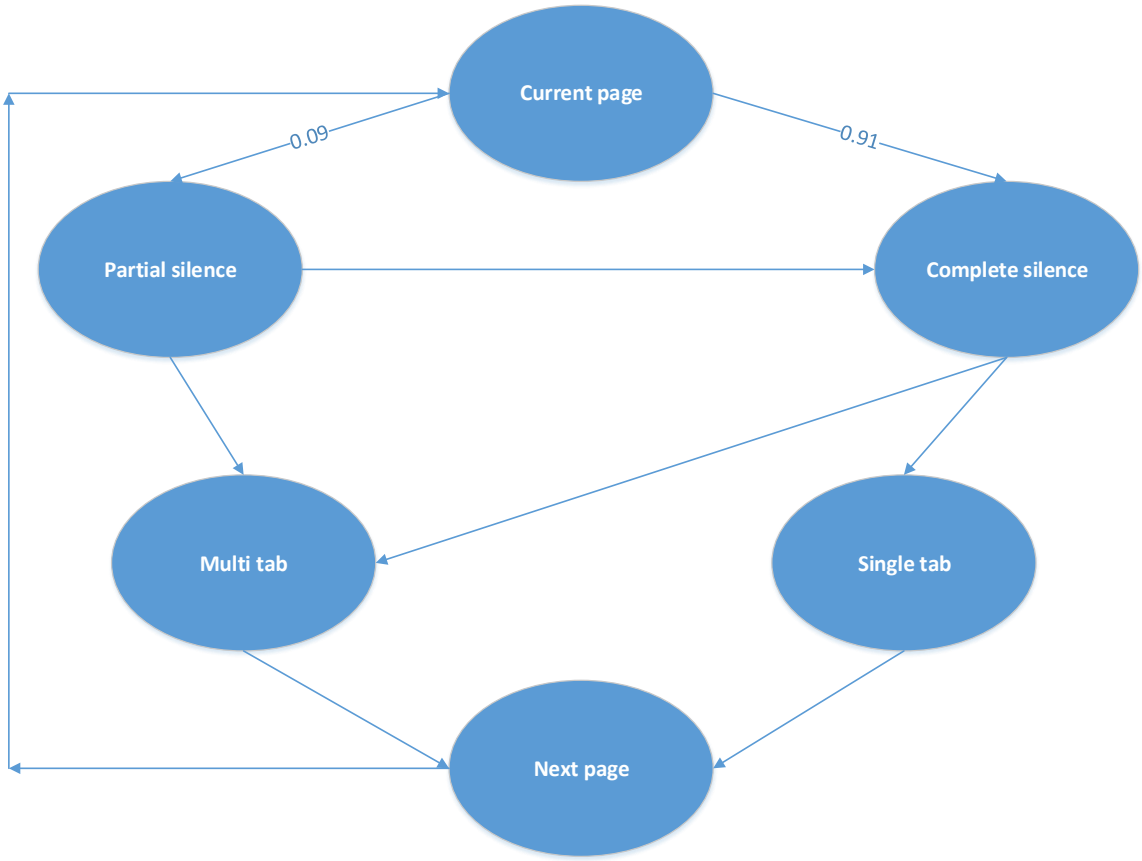

Figure 3.7 - User state machine in his unfamiliar website

3.2.4 User browsing depth:

Depth of user browsing is demonstrated in Tables 3.4 and 3.5 for familiar and unfamiliar websites respectively.

Table 3.4 - Users' browsing depth in their familiar websites

browsing depth	Numerical frequency	probability	CDF
2	1	0.016129	0.016129
3	1	0.016129	0.032258
4	5	0.080645	0.112903
5	3	0.048387	0.16129
6	4	0.064516	0.225806
7	9	0.145161	0.370968
9	3	0.048387	0.419355
10	2	0.032258	0.451613
11	3	0.048387	0.5
12	3	0.048387	0.548387
13	2	0.032258	0.580645
14	3	0.048387	0.629032
15	1	0.016129	0.645161
16	3	0.048387	0.693548
17	3	0.048387	0.741935
19	3	0.048387	0.790323
21	1	0.016129	0.806452
24	1	0.016129	0.822581
25	1	0.016129	0.83871
26	1	0.016129	0.854839
29	2	0.032258	0.887097
30	1	0.016129	0.903226
32	2	0.032258	0.935484
38	1	0.016129	0.951613
45	1	0.016129	0.967742
52	1	0.016129	0.983871
63	1	0.016129	1

Table 3.5 - Users' browsing depth in their unfamiliar websites

browsing depth	Numerical frequency	probability	CDF
1	2	0.03225807	0.032258072
3	8	0.12903226	0.12903226
4	9	0.14516129	0.27419355
5	4	0.06451613	0.33870968
6	7	0.11290323	0.45161291
7	4	0.06451613	0.51612904
8	5	0.08064516	0.5967742
9	4	0.06451613	0.66129033
10	2	0.03225807	0.6935484
11	2	0.03225807	0.72580647
12	1	0.01612903	0.7419355
13	2	0.03225807	0.77419357
14	1	0.01612903	0.7903226
15	1	0.01612903	0.80645163
16	1	0.01612903	0.82258066
21	3	0.0483871	0.87096776
22	1	0.01612903	0.88709679
24	1	0.01612903	0.90322582
26	1	0.01612903	0.91935485
27	1	0.01612903	0.93548388
33	1	0.01612903	0.95161291
62	1	0.01612903	1

As you can see 80 percent of users in the case of unfamiliar website, go forward less than 15 steps, while 36 percent of them go forward more in their favorite website. More than half of users go forward less than 7 steps

in the case of the unfamiliar website while half of them go forward more than 11 steps in their favorite website.

However, in each state, depths which have most statistical frequency and cover about 70 percent of users are chosen. As a matter of fact, they are used as browsing depth and determine the end point of state machine.

Table 3.6 - Number of opened tabs in each depth in the familiar websites

Number of opened tabs	Numerical frequency	probability	CDF
0	1119	0.91272431	0.912724
1	79	0.06443719	0.977162
2	8	0.00652529	0.983687
3	7	0.00570962	0.989396
4	6	0.00489396	0.99429
5	3	0.00244698	0.996737
6	1	0.00081566	0.997553
7	2	0.00163132	0.999184
8	1	0.00081566	1

3.2.5 Number of opened tabs in each depth:

Tables 3.6 and 3.7 represent number of tabs opened by the user in each depth for familiar and unfamiliar websites respectively. Also, user browsing state machine is completed by means of these tables (Figures 3.8 & 3.9).

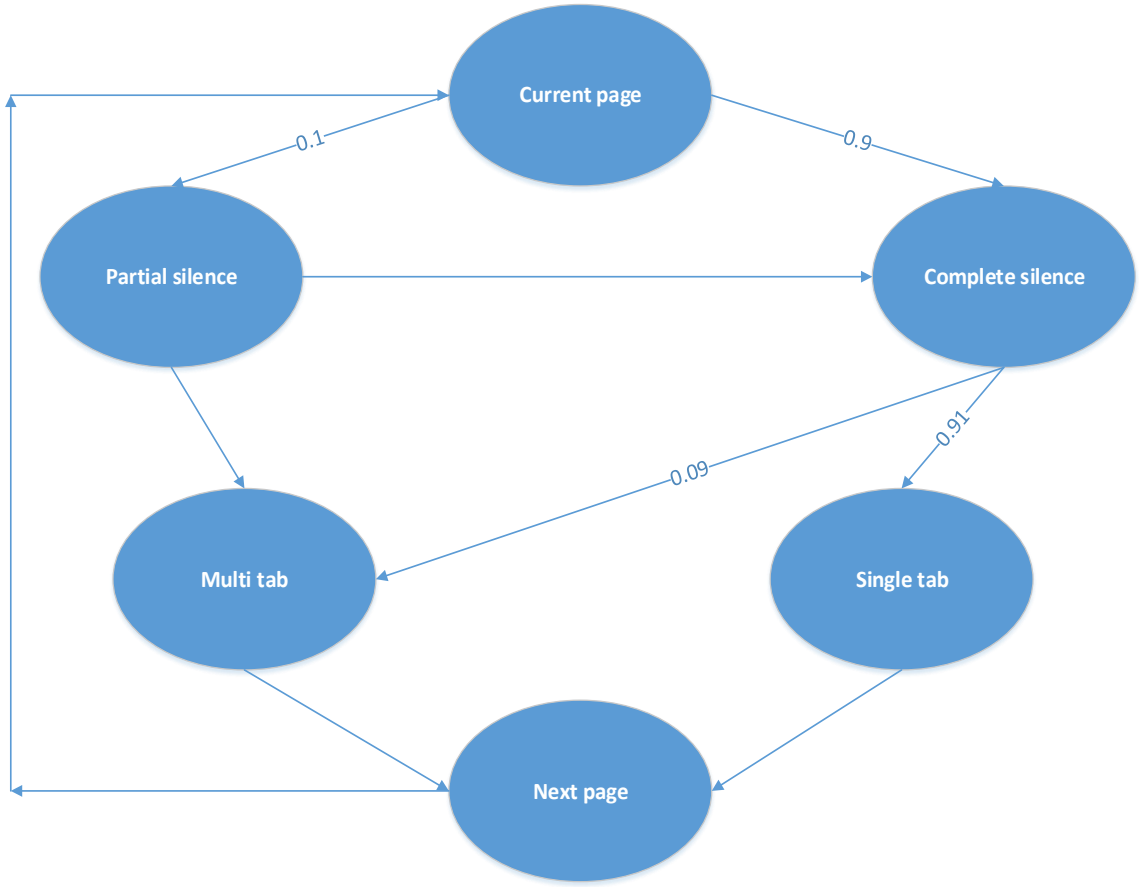

Figure 3.8 - User browsing state machine in the favorite website

As already mentioned, user browsing depth is limiter factor of state machine in change from next page state to current page state. In each state, depths which have most statistical frequency and cover about 70 percent of users are selected and used as browsing depth in main algorithm.

As it was observed in comparison with unfamiliar websites, users stay less time in pages of their favorite websites, while their browsing depth is more in their favorite website.

Table 3.7 - Number of opened tabs in each depth in the unfamiliar websites

Number of opened tabs	Numerical frequency	probability	CDF
0	878	0.84180249	0.841802
1	116	0.11121764	0.95302
2	27	0.02588686	0.978907
3	10	0.00958773	0.988495
4	5	0.00479386	0.993289
5	1	0.00095877	0.994247
6	1	0.00095877	0.995206
7	2	0.00191755	0.997124
8	2	0.00191755	0.999041
9	0	0	0.999041
10	0	0	0.999041
11	0	0	0.999041
12	1	0.00095877	1

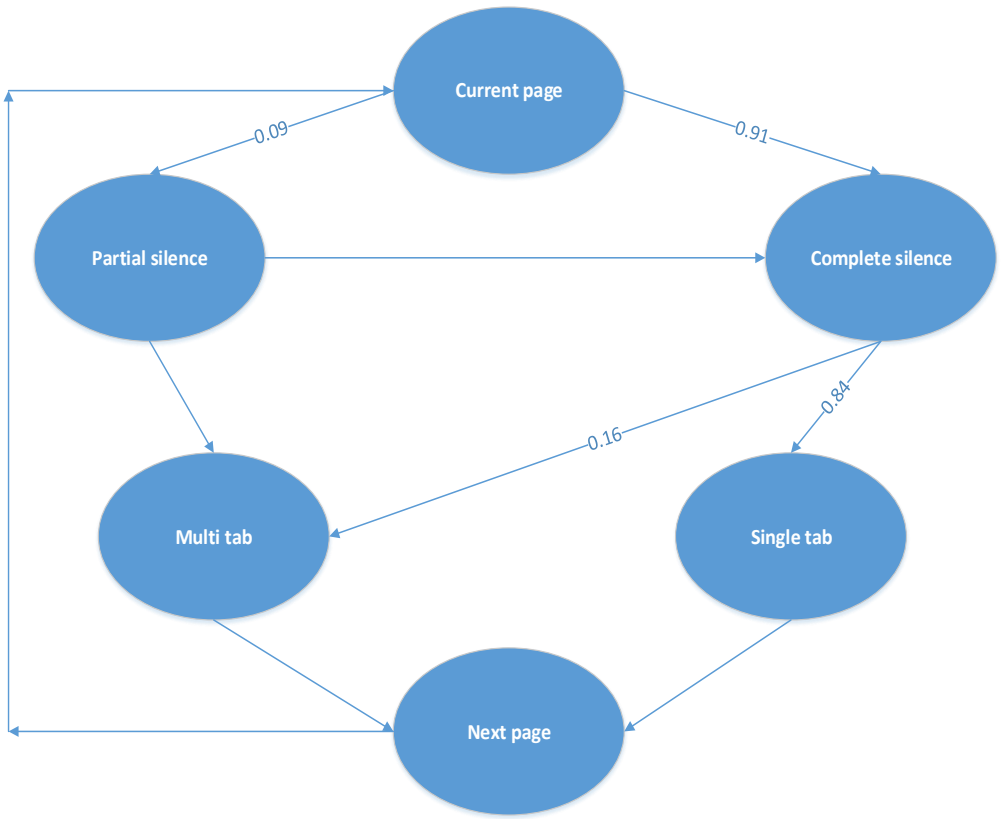

Figure 3.9 - User browsing state machine in the new website

Chapter Four
Implementation and evaluation

As mentioned in the previous chapter, at first a graph of page links is composed by means of an independent program. Null links that are used for trapping bots are detected and dropped. Links to other domains are dropped as well. This program is called by main algorithm which is described next.

4.1 Main program:

We use LoadPage class for loading inside object of pages. This class firstly requests basic HTML file and extracts its inside objects. Then CSS pages are requested and background images are extracted. Finally, objects are requested from the server according to the kind of browser. Also, we use single_tab and multi_tab which select one or more pages, respectively and make an object of LoadPage.

The program executes in the form of multi-threading. Each thread is the symbol of one user which independently executes the main program. Every user has a number and each page which is opened by user is located in opened pages' list with a page number. Opened pages' list is the list of

available selections for continuing user browsing process. It means user may choose one of them and follow some of its inside hyperlinks[5] by clicking on them. When a page is visited by the user, this page is added to visited pages' list as well. Visited pages' list specifies page state and shows whether a page is in partial silence state or not. Until page k related to user n has not gone to complete silence state, its number stays in both lists but after that it is deleted from them. Main program algorithm is briefly described in the following.

Main program algorithm

```
Select a, b
Select delay_time
Select agent
Select start_node
Current_node=start_node
Add Current_node into open_page_list
Select user mode
Select silence_mode from Figures 3.4 & 3.7
Select depth from Tables 3.4 & 3.5
While depth ≠0
    X=load_page
    Create neighbor_list of Current_node
    If neighbor_list == 0
        Go back homepage
    If Current_node is not in visit_page_list (complete_silence)
        Add Current_node in to visit_page_list
```

[5] In every page, there are some hyperlinks to other pages, they are called inside hyperlinks.

```
    If user-mode == permanent
        Select tab_num from Tables 3.6 & Figure 3.8
        If silence_mode == partial
            Select referrer_num from Table 3.2
    Else:
        Select tab_num from Table 3.7 & Figure 3.9
        If silence_mode == partial
            Select referrer_num from Table 3.3
    If length of neighbor_list<tab_num
        Edit tab_num
    If silence_mode == complete
        Delete current_node from open_page_list
        Delete current_node from visit_page_list
        If user_mode == permanent
            Select Silence_time from Figure 3.2
        Else
            Select Silence-time from Figure 3.5
If tab_num == 0
    Call single_tab
    Add next_node into open_page_list
    Current_node=next_node
Else
    Call multi_tab
    Add multi_tab output into open_page_list
    Select next_node from open_page_list
If Current_node is in visit_page_list (partial_silence)
    Refferer_num -1
    If Refferer_num == 0
        Delete current-node from open_page_list
        Delete current-node from visit_page_list
    If user_mode == permanent
```

```
        Select Silence_time from Figure 3.3
    Else
        Select Silence_time from Figure 3.6
    Call multi_tab
    Add multi_tab output into open_page_list
    Select next_node from open_page_list
depth -1
```

4.2 Traffic generation:

The described algorithm was executed on a cluster computer[6] by means of multithreading. Each thread represents a client. Besides, Sharif University website at Kish Island campus was downloaded by HTTrack[7] website copier and placed on an Apache server. All of aforementioned clients send their requests to this server for simulating Flash Crowd.

[6] A cluster machine in Advanced Computer Network Lab. of Sharif University. This computer is composed of 31 Beagle Bone boards, each having about 400 MB of Ram. All of them run an operating system based on Linux. This OS is loaded by the 32th board which is a Raspberry Pi.

[7] It allows you to download a World Wide Web site from the Internet to a local directory, building recursively all directories, getting HTML, images, and other files from the server to your computer.

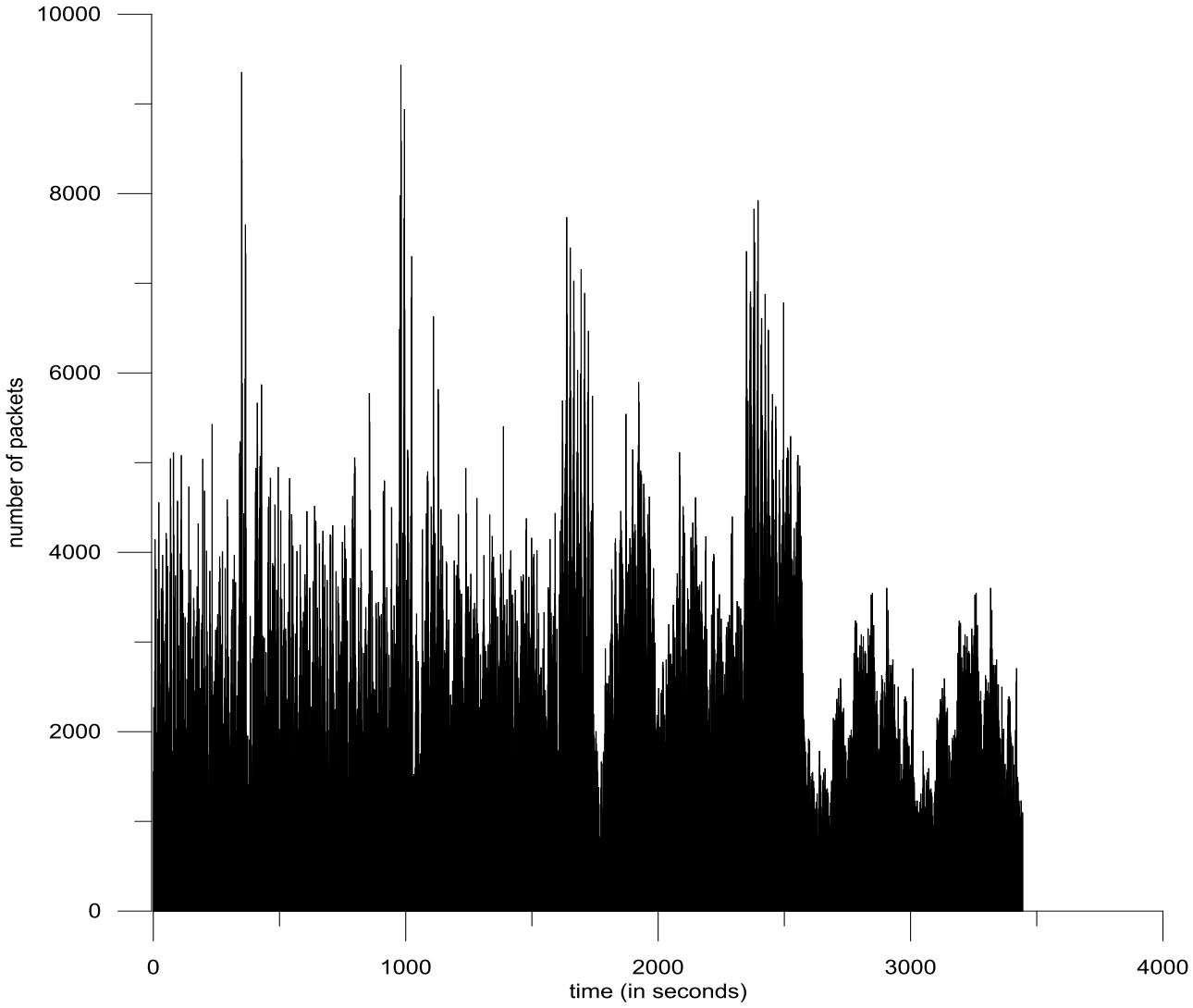

Figure 4.1 - Generated FE traffic

Each board, which we call a machine, can simultaneously execute 32 threads, therefore the total numbers of users in our project is 992. Flash Crowd is a relative event depending on server capacity, and the number of users or the volume of transmitted traffic. For instance, a fixed traffic can be considered FE for a low capacity or slow server while it may be considered as a normal traffic for another high capacity server.

As the number of users participating to generate the flash event is limited (992) we decrease our server capacity by reducing the CPU power, but for having actual results, we do not overspend, so that generated traffic cannot make server out of reach. However, it caused prolongation of response time and we faced connection reset. We captured generated traffic by means of tcpdump[8]. It is demonstrated in Figure 4.1.

4.3 Evaluation and analyzing results:

4.3.1 Number of requests:

For analyzing generated traffic number of received requests per minute was calculated in server side Figure 4.2. As was discussed in Chapter 2, requests number increment in FE is due to users' number growth. On the other hand, when FE has occurred in large volume such as what happed in Word Cup 98, the number of users increases with each passing second. Hence, we face request increment in every second. But here due to the limitation of number of users, we could not have this rapid growth. Otherwise, traffic rapidly gets to top or peak and it becomes similar to DDoS. Nevertheless, we increased the number of users gradually during the time and used minute unit instead of second in our analysis.

[8] A traffic capture tool in Linux command line

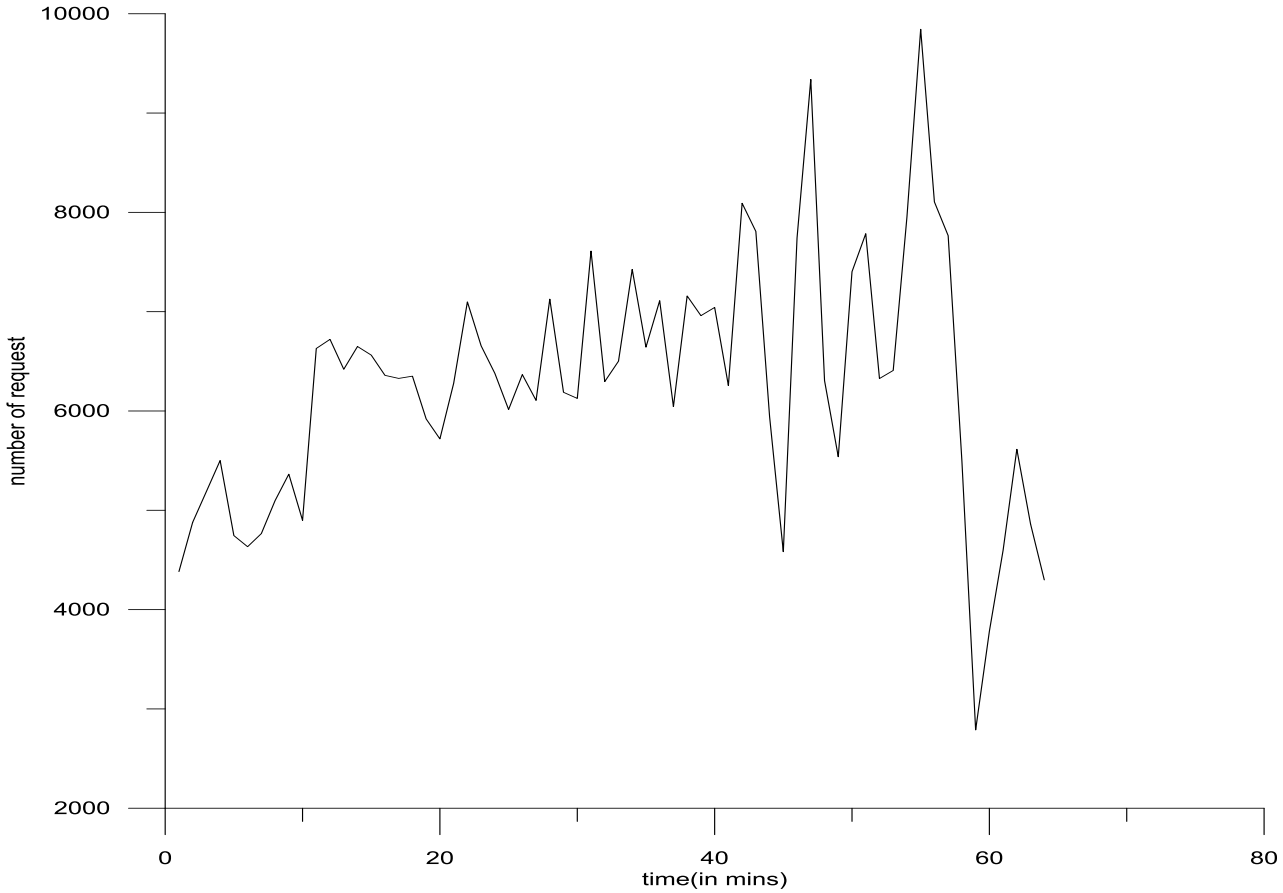

Figure 4.2 - Number of requests which server receives per minute

Figure 4.2 demonstrates that the number of requests increases gradually; this is one important property of FE and was observed in World Cup 98, as shown in Figure 4.3. On the other hand, as you can see in Figure 4.4, DDoS gets to peak in less than 5 minutes, and thereafter the traffic volume is almost constant. However, request' rate in World Cup 98 was gradually and has oscillation, which represented users' natural behavior.

It means, we suppose that in t2, the number of users has increased in comparison with t1, therefore we face increment in request rate, until part of users goes to silence phase, now we face decrement in request rate. In this state, number of requests can even be smaller than t1.

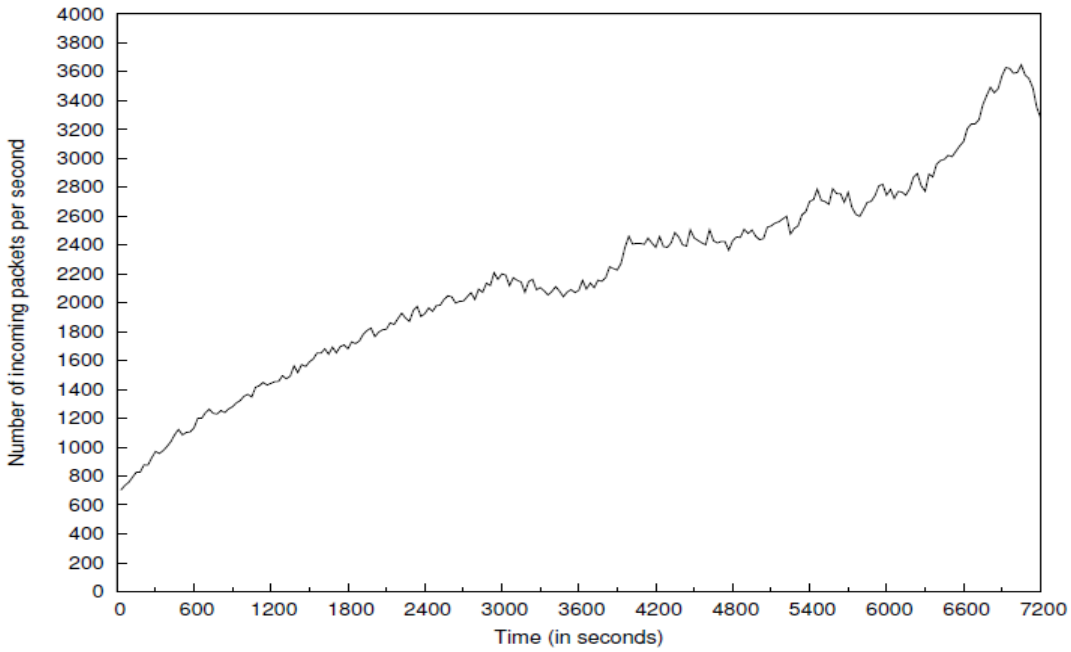

Figure 4.3 - World Cup 1998 [23]

Therefore, we have a general rate increment during the time but we observe oscillations as well, because of natural user behavior. As you can see in Figure 4.3, this silent and natural user behavior are not observed in an attack, and as a result, the rate is almost constant, because here the goal is only overloading.

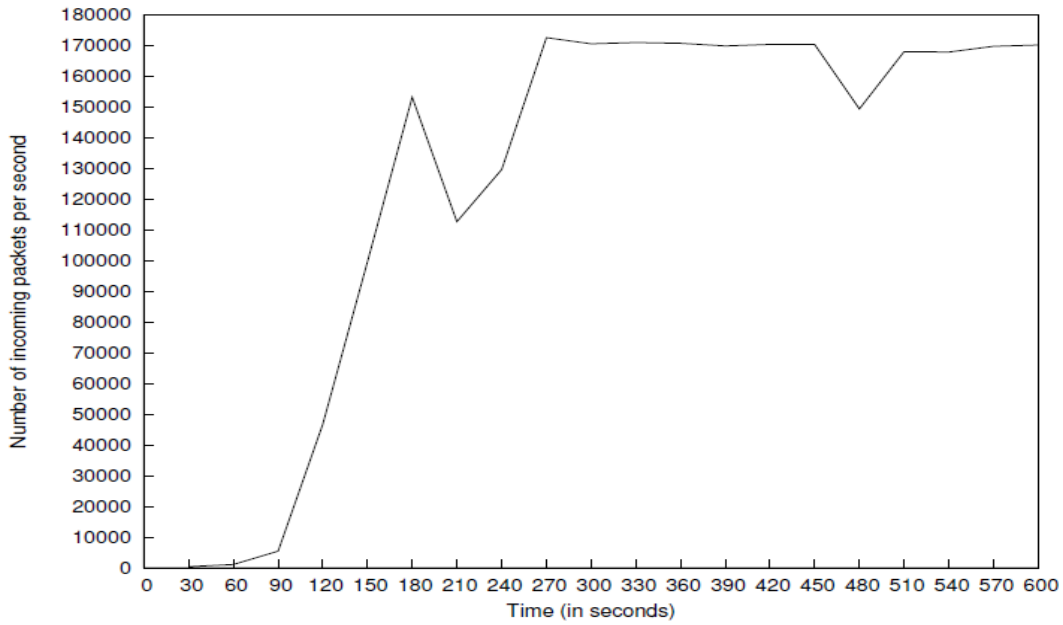

Figure 4.4 - CAIDA-DDoS [23]

4.3.2 Flow correlation:

As mentioned, some of studies have distinguished between attack and FE by means of flow correlation. In Figure 4.5 user browsing process is demonstrated. N, or the browsing depth, is a random number and comes from statistical analysis which was discussed. Silence times are random numbers from aforementioned Pareto distribution. Also, this chain may contain some referring to previous pages with random repeated number. Accordingly, possibility of flow similarity is improbable in this continuous probability space, while in studies, less than 60 percent similarity has been considered as FE.

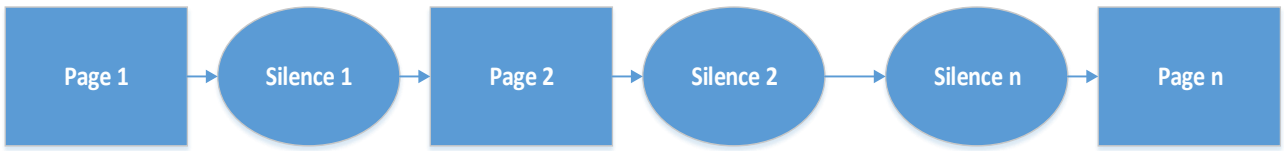

Figure 4.3 - User browsing process

4.3.3 Response time:

Although our FE was not so powerful for much slowing down or stopping the server, but response time for 2 users in machine 2 and 3 became so long that their connection was reset. Figure 4.6 shows overloading and FE in server side.

4.3.4 User requests speed:

As mentioned, different users send their requests with different speeds in FE. However, during the attacks, due to bots shortage, attackers are forced to use total available capacity so that they can only execute programs with the same requests speeds in their bots.

In this project, we managed to get a variety of speeds by creation of different delays and different inter arrival time requests.

Chapter five

Conclusion

Our proposed traffic generator generates various real like flows by mimicking and simulating user browsing behavior. On the other hand, it combines user behavior characteristics with special FE properties especially, when the user gets tired due to long response time, and leaves website. It generates a Flash Crowd based on server characteristics and its available files without the need for input traffic sample.

Since this generator produces online traffic, network changes which vary congestion control window, face user's online reaction and hence, traffic flow seems real. Also, this generator is not related to input traffic sample nevertheless, we can produce various traffic according to capacity of different servers by means of increment or decrement in number of participant machines, while it was not provided in other studies until now. This possibility helps researchers studying this anomaly in controlled conditions. Finally, this tool prepares a desired platform for evaluation of prevention and detection intrusion tools.

As future work, we will try to investigate user behavior when facing FE more precisely and add more details to our project. Since this approach needs to capture user behavior in client side, we are forced to handle both

side of communication, which means creating hot content and setting user browsers, or we must use an easier method for instance, when government declares an important event such as online census or completing tax return. These events can turn to FE.. Therefore, we can capture behavior of users or coffee net managers (who input user data online) and investigate them however, we must moderate our final result because they concentrate on special group of society.

Also, as was discussed in Chapter 2, we intend to study self-similarity in FE, as researchers still have not reached an agreement on it, by generating and studying FE traffics in various scales and characteristics.

References

[1] R. Pal, S. Kumar, R. L. Sharma, “A Detailed Classification of Flash Events: Client, Server and Network Characteristics,” *Computer Science & Service System (CSSS), 2012 International Conference,* pp. 11-13 Aug. 2012.

[2] Bhatia S, Mohay G, Schmidt D, Tickle A, “Modelling web-server flash events,” *Network computing and applications (NCA), 2012 11th IEEE International Symposium*, pp. 79-86, 2012.

[3] Asmussen, S. R. "Steady-State Properties of GI/G/1 Applied Probability and Queues,” *Stochastic Modelling and Applied Probability*, vol. 51. pp. 266–30, 2003.

[4] T. Karagiannis, M. Molle, M. Faloutso, “Long-range Dependence: Ten Years of Internet Traffic Modeling,” *Internet Computing, IEEE*, vol. 8, no. 5, pp. 57-64, 2004.

[5] M.E. Crovella and A. Bestavros, “Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes,” *IEEE/ACM Transactions on Networking,* vol. 5, pp. 835-846, December 1997.

[6] Sh. Deng, “Empirical model of WWW document arrivals at access links,” *Proceedings of the 1996 IEEE International Conference on Communication*, June 1996.

[7] Mandelbrot, Benoit B, "How long is the coast of Britain? Statistical self-similarity and fractional dimension," *New Series 156 (3775). Science*, pp. 636–63, May 1967.

[8] W. E. Leland, M. S. Taqqu, W. Willinger and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Trans. Networking*, vol. 2, no. 1, pp. 1-15, 1994.

[9] K. Park, Gi.T. Kim, and M.E. Crovella, “On the relationship between file sizes, transport protocols, and self-similar network trance,” *Proceedings of the Fourth International Conference on Network Protocols (ICNP'96)*, pp. 171-180, October 1996.

[10] W. Ditto and T. Munakata, “Principles and Applications of Chaotic Systems,*”* *Comm. of the ACM*, Vol. 38, No. 11, Nov 1995.

[11] M. Laner, Ph. Svoboda, M. Rupp, “Parsimonious Fitting of Long-Range Dependent Network Traffic Using ARMA Models,” *IEEE Communications Letters*, VOL. 17, NO. 12, December 2013.

[12] R. Smith,” Internet Traffic Periodicities and Oscillations: A Brief Review”, 2009.

[13] N. Li and S. Yu, “Periodic Hidden Markov Model-Based Workload Clustering and Characterization,” *Proc. IEEE Eighth International Conf. Computer and Information Technology*, pp. 378-383, 2008.

[14] Y. Xie, J. Hu, Y. Xiang, S. Yu, S. Tang and Y. Wang, "Modeling oscillation behavior of network traffic by nested hidden Markov model with variable state-duration," *IEEE Trans. Parallel Distrib. Syst*., vol. 24, no. 9, pp. 1807-1817, 2013.

[15] J. Jung, B. Krishnamurthy, and M. Rabinovich, “Flash crowds and denial of service attacks: Characterization and implications for CDNs and websites*,” ACM Proceedings of the 11th international conference on World Wide Web*, pp. 293–304, 2002.

[16] J. Pan, H. Hu and Y. Liu, "Human behavior during Flash Crowd in web surfing," *Physica A: Statistical Mechanics and its Applications,* vol. 413, p. 212–219, 2014.

[17] H.Ranjbar, “Distinguishing DDoS attacks from Flash Crowds,” M.Sc. Thesis, Sharif University of Technology Department of Computer Engineering, Sep 2015. (in Persian).

[18] A. Veres, M. Boda, “The Chaotic Nature of TCP Congestion Control,” *Infocom 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1715-1723, 2000.

[19] A. Scherrer, N. Larrieu, P. Owezarski, P. Borgnat, and P. Abry, "Non gaussian and long memory statistical characterizations for Internet traffic with anomalies*," IEEE Trans. Dep. and Secure Comp*, vol. 4, no. 1, pp. 56-70, Jan. 2007.

[20] I. Ari, B. Hong, E. Miller, S. Brandt, and D. Long, “Managing flash crowds on the Internet,” in *Proceedings of 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems (MASCOTS). IEEE*, pp. 246–249, 2003.

[21] P. Wendell and M. Freedman, “Going viral: flash crowds in an open CDN,” in *Proceedings of the 2011 ACM SIGCOMM conference*. ACM, pp. 549–558, 2011.

[22] S. Bhatia, D. Schmidt, G. Mohay and A. Tickle, "A framework for generating realistic traffic for distributed denial of service attacks and flash events", *Computers & Security*, vol. 40, no. 0, pp. 95-107, 2014.

[23] S. Bhatia, G. Mohay, A. Tickle and E. Ahmed, “Parametric Differences Between a Real-world Distributed Denial-of-Service Attack and a Flash Event,” *Sixth International Conference on Availability, Reliability and Security, 2011 IEEE Conference*. pp. 210-217, Aug 2011.

[24] Q. Le, M. Zhanikeev and Y. Tanaka, “Methods of Distinguishing Flash Crowds from Spoofed DoS Attacks,” *Next Generation Internet Networks, 3rd EuroNGI Conference, IEEE*, pp. 167-173, May 2007.

[25] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "My Botnet is bigger than yours: why size estimates remain challenging," *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, 2007.

[26] S. Bhatia, G. Mohay, D. Schmidt, A. Tickle, “Modelling Web-server Flash Events,” *Network Computing and Applications (NCA), 2012 11th IEEE International Symposium*, pp. 79-86, Aug 2012.

[27] D. Mahajan and M. Sachdeva, “Distinguishing DDoS Attack and Flash Event using Real World Datasets with Entropy as an Evaluation Metric,” *IEEE Machine Intelligence and Research Advancement (ICMIRA), 2013 International Conference*, pp. 90-94, Dec. 2013.

[28] S. Yu, G. Zhao, S. Guo, Y. Xiang, and A. Vasilakos, “Browsing behavior mimicking attacks on popular websites,” *INFOCOM, 2012 Proceedings IEEE*, pp. 947–951, March 2011.

[29] Sh. Yu, S. Guo, I. Stojmenovic, “Can We Beat Legitimate Cyber Behavior Mimicking Attacks from Botnets*?,” Infocom, 2012 Proceedings IEEE*, pp. 2851-2855 March 2012.

[30] M.A. Rajab, J. Zarfoss, F. Monrose and A. Terzis, "A Multifaceted Approach to Understanding the Botnet Phenomenon," *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pp. 41-52, October 2006.

[31] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "My Botnet is bigger than yours: why size estimates remain challenging," in *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, 2007.

[32] V.L.L. Thing, M. Sloman, and N. Dulay, "A Survey of Bots Used for Distributed Denial of Service Attacks," *Proc. SEC*, pp. 229-240, 2007.

[33] C.Y. Cho, J. Caballero, C. Grier, V. Paxson, and D. Song, "Insights from the Inside: A View of Botnet Management from Infiltration," *Proc. Third USENIX Conf. Large-Scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More (USENIX LEET*), 2010.

[34] R. Pal, S. Kumar, "A Detailed Classification of Flash Events: Client, Server and Network Characteristics," *Computer Science & Service System (CSSS), 2012 International Conference, IEEE*, pp. 960-963, Aug 2012.

[35] T. Swaroopa Rani, V. Sindhura, G. RamaKoteswara Rao, K. Pranathi, "Discerning Flooding Attack from Flash Crowd based on traffic patterns using entropy detection method," *IEEE International Conference, Electrical, Computer and Communication Technologies (ICECCT)*, pp. 1-6, March 2015.

[36] Th. Thapngam, Sh. Yu, W. Zhou and G. Beliakov, "Discriminating DDoS Attack Traffic from Flash Crowd through Packet Arrival Patterns," *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference*, pp. 952 – 957, April 2011.

[37] Sh. Yu, W. Zhou, S. Member, W. Jia, S. Guo, Y. Xiang, and F. Tang, "Discriminating DDoS Attacks from Flash Crowds Using Flow Correlation Coefficient," *Parallel and Distributed Systems, IEEE Transactions on*, Vol. 23, no. 6, pp. 1073 – 1080, June 2012.

[38] D. Moore, C. Shannon, D.J. Brown, G.M. Voelker and S. Savage, "Denial-of-Service Activity," *ACM Trans. Computer Systems*, vol. 24, no. 2, pp. 115 – 139, 2006.

[39] Yi. Xie and Shun-Zheng. Yu, "A Large-Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviors*," IEEE/ACM Transactions on Networking*, Vol. 17, no. 1, pp. 54-65, Feb. 2009.

[40] Mark E. Crovella, A. Bestavros, "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes," IEEE*/ACM Transactions on Networking*, Volume 5, no. 6. Dec 1997.

[41] Bruce A. Mah, "An Empirical Model of HTTP Network Traffic," *Proceedings of IEEE Infocom, Kobe, Japan*. Apr. 1997.

[42] H. Choi and J. Limb, "A Behavior Model of a Web Traffic", *Proc. Int'l Conf. Network Protocols (ICNP '99)*, Sep. 1999.

[43] N. Vicari, Prof. Dr. P. Tran-Gia, "Modeling of Internet Traffic: Internet Access Influence User Interference and Tcp Behavior," *Würzburg university, ISSN*, p. 1432 – 8801**,** 2003.

[44] P. Gill, M. Arlitt, Z. Li, A. Mahanti. "Characterizing User Sessions on Youtube," *Proceedings of ACM/SPIE MMCN. San Jose, USA*. Jan 2007.

[45] J. Shaikh, M. Fiedler, P. Arlos and D. Collange, "Modeling and analysis of web usage and experience based on link-level measurements," *24th International Teletraffic Congress (ITC 24)*, pp. 1-8, 2012.

[46] C. Liu, R. W. White and S. Dumais, "Understanding web browsing behaviors through weibull analysis of dwell time," *Proc. 33rd Int. ACM SIGIR Conf. Res. Develop. Inform. Retrieval*, pp. 379-386, 2009.

[47] Labaj, M., Bieliková, M, "Modeling parallel web browsing behavior for web-based educational systems," *Proc of the 10th Int. Conf. on Emerging eLearning Technologies and Applications - ICETA 2012*, pp. 229–234. 2012.

[48] J. Huang and R.W. White, "Parallel browsing behavior on the web," Proc. *of the 21st ACM Conf. on Hypertext and hypermedia-HT'10*, pp. 13-17, 2010.

[50] P. Dubroy and R. Balakrishnan, "A Study of Tabbed Browsing Among Mozilla Firefox Users," in *Proc. of the 28th int. conf. on Human factors in comp. systems - CHI '10*, pp. 673–682, 2010.

[51] H. Zhang and S. Zhao, “Measuring Web Page Revisitation in Tabbed Browsing,” *Proc. of the 2011 annual conf. on Human factors in computing systems - CHI ’11*, pp. 1831–1834, 2011.

[52] Becchetti, L., Castillo, C., Donato, D., Leonardi, S., and Baeza-Yates, R, “Link-Based Characterization and Detection of Web Spam,” *Proceedings of AIRWeb ’06*, pp. 1-8, 2006.

[53] H. Weinreich, H. Obendort, E. Herder and M. Mayer, “Off the Beaten Tracks: Exploring Three Aspects of Web Navigation,” *15th international conference on World Wide Web*, 2006.

[54] M. Arlitt. Characterizing Web User Sessions, *ACM SIGMETRICS Performance Evaluation Review*. Vol. 28, no. 2. Sep 2000.

[55] http://www.postel.org/tg/

[56] C. Rolland, J. Ridoux and B. Baynat, "Litgen a lightweight traffic generator: Application to p2p and mail wireless traffic," *LNCS*, vol. 4427, pp. 52-62, 2007.

[57] http://cs.colgate.edu/~jsommers/harpoon/

[58] http://www.isi.edu/nsnam/ns/doc/node563.html

[59] http://traffic.comics.unina.it/software/ITG/

[60] K. Vishwanath and A. Vahdat, "Swing: Realistic and responsive network traffic generation," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 712-725, June 2009.

به نام حضرت دوست

چکیده:

تولید ترافیک شبکه با تمرکز بر ترافیک نابهنجار هجوم ناگهانی

زهرا السادات صالح، کارشناسی ارشد

دانشگاه صنعتی شریف، پردیس بین الملل، جزیره کیش، ۱۳۹۵

استاد راهنما: دکتر امیرحسین جهانگیر

تولید ترافیک هجوم ناگهانی [9]از یک سو می‌تواند به عنوان یک سنجه [10]جهت تعیین میزان ظرفیت یک سرور استفاده شود و از سوی دیگر بستری برای تست و ارزیابی سیستم های کشف [11]و پیشگیری [12]نفوذ، فراهم می‌آورد. روش‌های معمول تولید ترافیک که یا عیناً دنباله ترافیک ورودی را تقلید می‌کنند و یا با محاسبه‌ی خصوصیات آماری دنباله‌ی ترافیکی به بازتولید آن می‌پردازند، نیازمند دراختیار داشتن دنباله‌ی ترافیکی ورودی هستند، این در حالی است که از یک سو مشخصات هجوم ناگهانی بر روی سرورهای مختلف متفاوت است و از سوی دیگر هر سروری الزاماً چنین پدیده ای را مشاهده نکرده است تا دنباله ای از آن در اختیار داشته باشد.

بنابراین در این پژوهش سعی شده است از روش جدیدی برای تولید ترافیک استفاده شود، که نیازمند وجود دنباله‌ی ورودی نباشد. برای نیل به این هدف از خصوصیت اصلی این ترافیک یعنی

[9] Flash Crowd
[10] Metrics
[11] Intrusion detection system
[12] Intrusion prevention system

قانونی بودن کاربران حاضر در این ناهنجاری و مدل سازی رفتار کاربران واقعی در ترکیب با خصوصیات ذاتی هجوم ناگهانی بهره گرفته شده است. بدین منظور یک افزونه بر روی مرورگر فایرفکس طراحی شده و به کمک آن رفتار مرورگری کاربران مختلف ثبت و توزیع های آماری متناسب تعیین و به کمک یک ماشین وضعیت رفتار مرورگری هر کاربر مدل شده است. در نهایت با ترکیب این رفتار با خصوصیات هجوم ناگهانی، ترافیک مورد نظر به صورت آنلاین تولید می‌شود.

کلمات کلیدی: تولید ترافیک، هجوم ناگهانی، تحلیل رفتار