

# **Using Visual Analytics to Discover Bot Traffic**

**Ignatius Hananto Herlambang**

A thesis submitted in partial fulfilment of the  
requirements of Liverpool John Moores University  
for the degree of Master of Philosophy

**January 2015**

## **Acknowledgements**

First and foremost, I would like to thank my supervisors, Professor Qi Shi and Dr. Bo Zhou, who have given me the support and guidance throughout this research project.

Finally, I would like to express a special and graceful thank to my family which this thesis is dedicated to.

## Abstract

With the advance of technology, the Internet has become a medium tool used for many malicious activities. The presence of bot traffic has increased greatly that causes significant problems for businesses and organisations, such as spam bots, scraper bots, distributed denial of service bots and adaptive bots that aim to exploit the vulnerabilities of a website. Discriminating bot traffic against legitimate flash crowds remains an open challenge to date.

In order to address the above issues and enhance security awareness, this thesis proposes an interactive visual analytics system for discovering bot traffic. The system provides an interactive visualisation, with details on demand capabilities, which enables knowledge discovery from very large datasets. It enables an analyst to understand comprehensive details without being constrained by large datasets. The system has a dashboard view to represent legitimate and bot traffic by adopting Quadtree data structure and Voronoi diagrams. The main contribution of this thesis is a novel visual analytics system that is capable of discovering bot traffic.

This research conducted a literature review in order to gain systematic understanding of the research area. Furthermore, the research was conducted by utilising experiment and simulation approaches. The experiment was conducted by capturing website traffic, identifying browser fingerprints, simulating bot attacks and analysing mouse dynamics, such as movements and events, of participants. Data were captured as the participants performed a list of tasks, such as responding to the banner. The data collection is transparent to the participants and only requires JavaScript to be activated on the client side. This study involved 10 participants who are familiar with the Internet. To analyse the data, Weka 3.6.10 was used to perform classification based on a training dataset. The test dataset of all participants was evaluated using a built-in decision tree algorithm. The results of classifying the test dataset were promising, and the model was able to identify ten participants and six simulated bot attacks with an accuracy of 86.67%. Finally, the visual analytics design was formulated in order to assist an analyst to discover bot presence.

**Keywords:** visual analytics, application-layer, bot traffic.

# Table of Contents

<b>ACKNOWLEDGEMENTS</b>	<b>I</b>
<b>ABSTRACT</b>	<b>II</b>
<b>ACRONYMS</b>	<b>VI</b>
<b>LIST OF FIGURES</b>	<b>VII</b>
<b>LIST OF TABLES</b>	<b>VIII</b>
<b>CHAPTER 1 INTRODUCTION</b>	<b>1</b>
1.1 BACKGROUND	1
1.2 RESEARCH CHALLENGES AND QUESTIONS	2
1.3 RESEARCH METHODOLOGY	3
1.4 RESEARCH AIMS, OBJECTIVES AND NOVELTIES	4
1.4.1 AIMS	4
1.4.2 OBJECTIVES	4
1.4.3 NOVELTIES	4
1.5 RESEARCH SCOPE	5
1.6 PROJECT RESULTS	5
1.7 THESIS OUTLINE	6
<b>CHAPTER 2 LITERATURE REVIEW</b>	<b>7</b>
2.1 VISUAL ANALYTICS	7
2.2 MALICIOUS BOTS	9
2.3 WEBSITE HEATMAP	11
2.4 BROWSER FINGERPRINT	12
2.5 BANNER ADVERTISEMENT	13
2.6 PROOF-OF-WORK SYSTEM	14
2.7 DISCRIMINATING BOTS AGAINST LEGITIMATE FLASH CROWDS TRAFFIC	15
2.8 VISUAL ANALYTICS APPLICATIONS	16
2.9 SUMMARY	18
<b>CHAPTER 3 VATRIX</b>	<b>20</b>
3.1 INTRODUCTION	20
3.2 QUADTREE DATA STRUCTURE	22
3.3 VORONOI DIAGRAM	22
3.4 GENERATING VORONOI DIAGRAM	24
3.5 PRIORITY QUEUING METHOD	25
3.6 VATRIX INTERACTIVE USER INTERFACE	26
3.6.1 DETAILS ON DEMAND	29
3.6.2 WEIGHT SCORE	30
3.6.3 EXTENDED VIEW	31
3.7 VATRIX RESPONSE	33
3.7.1 APACHE WEB SERVER	33
3.7.2 PROCESS MAPPING	34
3.8 SUMMARY	36
<b>CHAPTER 4 X-MAP</b>	<b>38</b>
4.1 CLASSIFICATION	38

<b>4.2</b>	<b>DECISION TREE LEARNING</b>	<b>39</b>
<b>4.3</b>	<b>PROPOSED ALGORITHM X-MAP</b>	<b>43</b>
<b>4.4</b>	<b>PROPOSED ALGORITHM FOR MACHINE IDENTIFICATION</b>	<b>46</b>
<b>4.5</b>	<b>PROPOSED ALGORITHM FOR HEATMAP TRACKER</b>	<b>47</b>
<b>4.6</b>	<b>PROPOSED TECHNIQUE FOR BANNER REACTION</b>	<b>49</b>
<b>4.7</b>	<b>SUMMARY</b>	<b>50</b>
<b>CHAPTER 5 EXPERIMENT SETTINGS</b>		<b>51</b>
<b>5.1</b>	<b>INTRODUCTION</b>	<b>51</b>
<b>5.2</b>	<b>EXPERIMENT METHOD</b>	<b>51</b>
5.2.1	TEST ENVIRONMENT	52
5.2.2	PARTICIPANTS	53
5.2.3	ATTACK SIMULATION	53
<b>5.3</b>	<b>DATA COLLECTION</b>	<b>54</b>
5.3.1	IP AND TCP HEADER	55
5.3.2	HTTP HEADER AND REQUEST BEHAVIOUR	55
5.3.3	PROOF-OF-WORK	55
5.3.4	FORM SUBMISSION FREQUENCY	56
5.3.5	AUTHENTICATED SESSION	56
5.3.6	WEBSITE HEATMAP	56
5.3.7	BANNER REACTION	56
5.3.8	SCREEN RESOLUTION	57
5.3.9	BROWSER FINGERPRINT	57
<b>5.4</b>	<b>DATA PRE-PROCESSING</b>	<b>57</b>
<b>5.5</b>	<b>DATA ANALYSIS</b>	<b>58</b>
<b>5.6</b>	<b>TRAINING DATASET</b>	<b>58</b>
5.6.1	DEPENDENT AND INDEPENDENT VARIABLE	59
5.6.2	DATA CLEANING	59
<b>5.7</b>	<b>VISUAL ANALYTICS REQUIREMENTS</b>	<b>59</b>
<b>5.8</b>	<b>LIMITATIONS</b>	<b>59</b>
<b>5.9</b>	<b>SUMMARY</b>	<b>60</b>
<b>CHAPTER 6 EVALUATION</b>		<b>61</b>
<b>6.1</b>	<b>INTRODUCTION</b>	<b>61</b>
<b>6.2</b>	<b>HTTP REQUESTS</b>	<b>62</b>
6.2.1	LANDING PAGE	62
6.2.2	PRODUCT DETAIL PAGE	63
6.2.3	CATEGORY PAGE	63
6.2.3	LOGIN PAGE	64
<b>6.3</b>	<b>OVERALL PERFORMANCE SCORE</b>	<b>64</b>
<b>6.4</b>	<b>AJAX REQUEST TIMING</b>	<b>65</b>
<b>6.5</b>	<b>SIMULTANEOUS REQUESTS</b>	<b>66</b>
6.5.1	ATTACK SIMULATION	67
<b>6.6</b>	<b>CLASSIFIER PERFORMANCE</b>	<b>68</b>
6.6.1	TESTING ON A TRAINING DATASET	69
6.6.2	BANNER REACTION	70
6.6.3	DISCRIMINATING TRAFFIC	72
<b>6.7</b>	<b>VISUAL ANALYTICS DESIGN</b>	<b>73</b>
6.7.1	INTERACTIVE VISUALISATION	73
6.7.2	LARGE DATASET	75
6.7.3	KNOWLEDGE DISCOVERY	76
6.7.4	INFORMATION OVERLOAD	79
<b>6.8</b>	<b>DISCUSSION</b>	<b>80</b>

<b>6.9 SUMMARY</b>	<b>81</b>
<b>CHAPTER 7 CONCLUSIONS AND FUTURE WORK</b>	<b>84</b>
<b>7.1 CONCLUSION</b>	<b>84</b>
<b>7.2 FUTURE WORK</b>	<b>85</b>
<b>REFERENCES</b>	<b>IX</b>
<b>APPENDIX A – DATA COLLECTION SCRIPTS</b>	<b>XIV</b>
<b>APPENDIX B – TRAINING AND TEST DATASET (ARFF)</b>	<b>XVIII</b>
<b>APPENDIX C – WEBSITE PAGES</b>	<b>XXV</b>
<b>APPENDIX D – ATTACK SIMULATION CLI</b>	<b>XXIX</b>

## Acronyms

<b>ARFF</b>	Attribute-Relation File Format
<b>AJAX</b>	Asynchronous JavaScript and XML
<b>CAPTCHA</b>	Completely Automated Public Turing test to tell Computers and Humans Apart
<b>CLI</b>	Command Line Interface
<b>CPU</b>	Central Processing Unit
<b>NAT</b>	Network Address Translation
<b>DDOS</b>	Distributed Denial of Service
<b>HTML</b>	HyperText Markup Language
<b>HTTP</b>	HyperText Transfer Protocol
<b>HTTPS</b>	HyperText Transfer Protocol Secure
<b>I/O</b>	Input Output
<b>IP</b>	Internet Protocol
<b>OSN</b>	Online Social Network
<b>OTP</b>	One Time Password
<b>ROC</b>	Receiver Operating Characteristic
<b>SNMP</b>	Simple Network Management Protocol
<b>SSL</b>	Secure Socket Layer
<b>TCP</b>	Transmission Control Protocol
<b>VPS</b>	Virtual Private Sever

## List of Figures

Figure 1.1. Research Methodology	3
Figure 2.1. Visual analytics integration [11]	8
Figure 2.2. Visual analytics process [12]	9
Figure 2.3 Application layer attacks [2]	10
Figure 2.4. Distribution of attacks [17]	11
Figure 2.5. Website heatmap [18]	12
Figure 2.6. Browser fingerprint [23]	13
Figure 2.7. Banner advertisement [25]	14
Figure 2.8. Challenge-response [33]	15
Figure 3.1. VATRIX architecture	21
Figure 3.2. An example of Quadtree representation	22
Figure 3.3. A typical Voronoi diagram	23
Figure 3.4. Construct Voronoi	24
Figure 3.5. Constructing Voronoi diagram	24
Figure 3.6. Priority queue	26
Figure 3.7. An example of conventional network traffic visualisation	27
Figure 3.8. Vatrix interface	27
Figure 3.9. Bot selection	28
Figure 3.10. Flash crowds activities	28
Figure 3.11. Heatmap view rendered with d3.js	29
Figure 3.12. Heatmap View	30
Figure 3.13. Weight score selection	31
Figure 3.14. Website heatmap using plugin [59]	31
Figure 3.15. Calendar view	32
Figure 3.16. Histogram view	32
Figure 3.17. Generating View	32
Figure 3.18. Apache server-status	34
Figure 3.19. Apache scoreboard key	35
Figure 3.20. MxN matrix	35
Figure 3.21. Process state visualisation using nvd3.js [65]	36
Figure 4.1 X-Map Architecture	38
Figure 4.2. Observe Visitor	41
Figure 4.3. Prediction model	41
Figure 4.4. A possible training dataset	42
Figure 4.5. A possible classifier	42
Figure 4.6. Classifying bot	43
Figure 4.7. Attribute set	44
Figure 4.8. Traffic classification	45
Figure 4.9. System Flowchart	45
Figure 4.10. Machine identification	46
Figure 4.11 Machine identification flowchart	47
Figure 4.12. Matrix model of heatmap	48
Figure 4.13. Heatmap tracker	49
Figure 4.14 Heatmap tracker flowchart	49
Figure 6.1. Landing page	62
Figure 6.2. Product detail page	63
Figure 6.3. Category page	64
Figure 6.4. Login page	64
Figure 6.5. Overall performance score	65
Figure 6.6. AJAX request timing	65
Figure 6.7. Attack simulation	67
Figure 6.8. Classifier performance	68
Figure 6.9. Decision tree learning curve	69
Figure 6.10. Receiver operating characteristic for the training dataset	72
Figure 6.11. Tree view	73



<i>Figure 6.12. Quadtree representation using d3.js</i>	74
<i>Figure 6.13. Voronoi diagram using d3.js</i>	74
<i>Figure 6.14. Time series format using d3.js</i>	75
<i>Figure 6.15. Heatmap graph example using d3.js</i>	76
<i>Figure 6.16. Latest hits</i>	76
<i>Figure 6.17. Process states visualisation using d3.js</i>	77
<i>Figure 6.18. Group selection using d3.js</i>	77
<i>Figure 6.19. Further exploration</i>	78
<i>Figure 6.20. Time series format using d3.js</i>	78
<i>Figure 6.21. Apache process states</i>	79
<i>Figure 6.22. Visual status [70]</i>	79

## List of Tables

<i>Table 4.1. Predictive Detection model</i>	40
<i>Table 5.1. Participant characteristics</i>	53
<i>Table 5.2. Attack simulation tools</i>	54
<i>Table 5.3. Data collection</i>	55
<i>Table 6.1. Apache benchmark results</i>	66
<i>Table 6.2. Participants requests /sec</i>	66
<i>Table 6.3 X-Map Comparison Table</i>	82
<i>Table 6.4 VATRIX Comparison Table</i>	83

# Chapter 1 Introduction

## 1.1 Background

Application layer attacks are on the rise, and the majority of them target the Hypertext Transfer Protocol (HTTP). A recent study by Incapsula reported that bot traffic has increased up to 61.5% of all website traffic, of which 30.5% is likely to perform malicious activities [1]. Another study by Arbor Networks highlighted a growing trend in the attack threat, impact and frequency [2]. Interestingly, the Cloud Security Alliance listed Distributed Denial of Service (DDoS) as one of the top notorious threats to the Internet [3]. Therefore, a defence system relying on machine learning and artificial intelligence alone is insufficient in defeating adaptive adversaries. In a world of persistent threats, there will always be an open door to new attacks. Attacks that target zero-day vulnerabilities are unstoppable by nature. Similarly, application layer HTTP DDoS attacks, scraper bots and social bots are relatively new types of attack. These attacks are difficult to mitigate because they possess legitimate requests [13, 14]. They are able to perfectly mimic legitimate human behaviour in order to evade detection [6, 7]. However, their sources of addresses are exposed because a complete Transmission Control Protocol (TCP) handshake is required [15]. Although they consume less bandwidths, orchestrated attacks can slow down or even disrupt a network, and they can render the victim website inaccessible with very limited resources [13, 14]. In the presence of genuine flash crowds, a condition where a large number of users are accessing the website simultaneously [34], victims may not even realise that they have been targeted.

Orchestrated bot attacks produce vast amounts of data, and a command line interaction is not an efficient way in responding to these threats. Normally, filtering out the log file in a primitive way, such as line-by-line processing, can be overwhelming. Even an automatically generated script could not provide a measure of certainty in order to determine the final destiny of an incoming packet. Automated blocking will result in a high rate of penalised access for legitimate users. Therefore, actions need to be taken in an effective and efficient manner. Visual analytics, in contrast, allows an insight for understanding complex data with the visibility of attack patterns at multiple scales [44]. The end result

enhances situational awareness for a better decision-making process [45]. Visual analytics is a multidisciplinary field that exploits analytical reasoning techniques, which support innovative visualisation in order to gain insight and reasoning from large dataset [11].

## 1.2 Research Challenges and Questions

Discriminating bot traffic against legitimate flash crowds remains an open challenge to date. The increasing rate of malicious bot attacks has attracted many researches to address this challenge. One way of defence against bots is to use the Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) [37], which enforces human responses to answer a given question and to determine whether the user is a human. On the other hand, previous studies by Yu, Shui et al. have demonstrated that by using statistic based techniques, bots could generate attack traffic that poses similar behaviour to legitimate traffic [4]. Such manipulated traffic follows the patterns based on statistical distributions. The result of this study has shown the difficulties of using statistic based methods to distinguish legitimate traffic against bot generated traffic [4, 5]. In addition, Lee, Myungjin et al. developed *AjaxTracker* that automatically imitates a human interaction [6]. The tool provides automatic workload generation to test specific web applications. Jin, Jing et al. developed an evasive web bot system based on human behavioural patterns [7]. Their system provides a flexible and extensible framework to ensure direct interactions with a web browser. It enables automatic event generation aimed at the application user interface. Guo, Song et al. have demonstrated that *“it is almost impossible to detect mimicking attacks from statistics if the number of active bots of a botnet is sufficient”* [8]. These studies have demonstrated that legitimate browsing behaviour can be perfectly imitated to evade detection [4, 5, 6, 7, 8].

The studies presented so far provide evidence that bots are able to perfectly mimic human behaviour. In order to address these challenges, the following two research questions are raised for this research to answer:

**Question 1.** *Can we find a new solution to classify bot presence against legitimate traffic, without requiring too many user interventions?*

**Question 2.** *By using visual analytics, how can the result of classification be presented in a decision-oriented way?*

### 1.3 Research Methodology

The research was conducted by utilising experiment and simulation approaches. A number of participants were invited in the experiment. The experiment was conducted by capturing website traffic, identifying browser fingerprints, simulating bot attacks and analysing mouse dynamics, such as movements and events, of a participant. A novel visual analytics system was proposed in order to discover bot traffic and provide a measure of certainty for the decision making process.

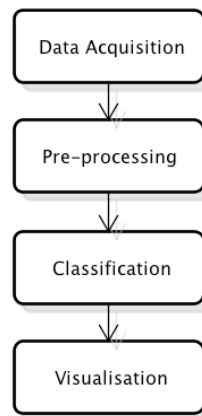


Figure 1.1. Research Methodology

Figure 1.1 illustrates the activities of the research work. In the data acquisition phase, participants' data were collected transparently. The objective is to extract certain characteristics of legitimate visitor and simulated bot attacks. Their browsing behaviour data were captured. The implementation of the data acquisition was written using C, PHP and JQuery.

The next phase is the pre-processing, which reads all the data obtained by the previous phase and performs Attribute-Relation File Format (ARFF) conversion. The objective of this phase is to segment and map every input into a probability value. This is to enable the classification to process data.

These data were then normalised and analysed. An initial training dataset was generated by comparing the activity patterns of the participants' legitimate behaviour against simulated bot attacks. To analyse the data, Weka 3.6.10 was used to perform classification based on a training dataset. The chosen algorithm

was J48 Decision Tree C4.5 algorithm [57]. Finally, the visualisation was rendered by using D3.js, a comprehensive JavaScript library for data-driven visualisation.

## **1.4 Research Aims, Objectives and Novelties**

### **1.4.1 Aims**

The aim of this research is to investigate the effectiveness of visual analytics in order to provide a defence mechanism and to discover bots traffic efficiently.

### **1.4.2 Objectives**

The objectives of this research are summarised as follow:

- to collect the data from participants versus the simulated bot attacks in order to obtain a threshold limit value,
- to develop an initial training dataset in order to best classify between legitimate traffic and bots traffic,
- to build an initial model and evaluate the model using a decision tree classifier algorithm,
- to formulate a visual analytics design that is capable of discovering unexpected bots traffic.

### **1.4.3 Novelties**

This research's main contribution is a novel visual analytics system that is capable of discovering bot traffic and providing a measure of certainty for the decision making process. Overall the result of this research contributes to these following novelties:

- VATRIX, a visual analytics system for discovering bots presence.  
A custom algorithm is implemented for generating a voronoi diagram of connected components, which takes a Quadtree data structure.
- X-Map, a standardised method for separating flash crowds from bots traffic.  
An observation input model is developed based on a decision tree algorithm, that analyses user interaction in the application-layer, such as website heatmap data and banner reaction results.

The proposed methods aim to overcome low and slow bandwidth attacks that exploit the vulnerability of an application layer. Several attempts have been made

on the transport and network layers, but they are not specifically designed to mitigate mimicking attacks that masquerade as flash crowds.

All these approaches were incorporated into a single system for interactive visual analytics design in order to overcome the limitations of current approaches, which are not specifically designed to mitigate mimicking attacks that exploit the vulnerability of an application layer.

### **1.5 Research Scope**

CAPTCHA and other automated Turing test methods are effective in restraining the presence of bots [37]. However, an adaptive adversary can evade those protections by hiring human labours to solve the test. This phenomenon has shown that even the most sophisticated CAPTCHA technology will fail to guarantee protection [9, 10]. As a result of this phenomenon, organised labour attacks are outside the scope of this research. This research studies causal relationships of legitimate participants and simulated bot attacks to be identified and analysed. Finally, this research presents the prototype of our visual analytics design.

### **1.6 Project Results**

Data were collected as the participants performed the set list of tasks and while the simulated bot attacks were running. The threshold value for each website page was measured according to all page requests, where the value will be used to indicate a normal legitimate page request and preliminary comparison against simulated bot attacks, as to be described in Section 6.2. The objective is to find the total number of HTTP requests made for each page. A minimal threshold value was then set to indicate the acceptance level of legitimate page requests. An initial training dataset was built based on an observation input model in order to classify incoming traffic.

This research was taken in a fully controlled and structured environment, which will be discussed in Chapter 5. The following are the steps for this research project:

1. Initial website capable of tracking visitor data.
2. Simulated bot attacks
3. Building the machine learning model
4. Constructing the Quadtree data structure

5. Generating the Voronoi diagram
6. Evaluating the system

The results of classifying the test dataset were promising, and the model was able to identify ten participants and six simulated bot attacks with an accuracy of 86.67%. Finally, the visual analytics design was formulated in order to assist an analyst to discover bot presence. The main contribution of this thesis is a novel visual analytics system that is capable of discovering bot traffic. Two methods have been developed. The first is VATRIX, a visual analytics system for discovering bots presence. The second one is X-Map, a standardised method for separating flash crowds from bots traffic, which employs a classification technique to make accurate detection based on visitors' past observations.

The research presented in this thesis was also published in the following conference:

- Herlambang, I. H., Shi, Q. and Zhou, B., *Interactive Visual Analytics for Discovering Bots Traffic*, PGNet, The Annual Postgraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting, pp. 89-94, Liverpool, UK, June 2014.

## 1.7 Thesis Outline

The remaining chapters of this thesis are organised as follows. Chapter 2 discusses the literature review in order to carry out a systematic understanding of the research domain. Chapter 3 presents the proposed visual analytics system. Chapter 4 presents the classification method. Chapter 5 describes the research settings and outlines the overall experiment of the simulation. Chapter 6 provides the evaluation of the system. Finally, Chapter 7 concludes this thesis by highlighting the main contributions and discussing directions for future research.

## Chapter 2 Literature Review

The presence of malicious bots has emerged as real threats to the Internet. Previous studies have demonstrated that legitimate browsing behaviour can be perfectly imitated to evade detection [4, 5, 6, 7, 8]. This research is focused on discriminating bots traffic against legitimate traffic, which can be applied to mitigate malicious bot attacks. The objectives presented in this thesis are suitable for discovering bots traffic, including visualisation. One major problem with application layer attacks is to transparently detect orchestrated bot attacks. The challenges of visualisation are associated with large and complex data.

Using a classification algorithm to distinguish between legitimate traffic and bot traffic has been an interesting research area. With the objective to automatically learn to make accurate predictions based on visitor past observations, this research used browser fingerprint, banner advertisement and other techniques to achieve its goal.

The remaining sections of this chapter provide the background of the research and describe an overview of current work in the area of visual analytics and malicious bots. It discusses the use of a website heatmap, browser fingerprint, banner advertisement and proof-of-work system in order to transparently discriminate bots traffic against legitimate traffic, which is one major challenge in application layer attacks. In addition, the application of visual analytics in the security domain is presented.

### 2.1 Visual Analytics

Visual analytics is a new emerging scientific field. In 2008, Keim et al. published a book in which they described the definition of visual analytics, “*Visual analytics combines automated analysis techniques with interactive visualizations for an effective understanding, reasoning and decision making on the basis of very large and complex data sets*” [11]. Visual analytics tackles the information overload problem and aims to turn them into an opportunity [12]. The broad use of visual analytics is to discover knowledge through visualisation and analytical reasoning.



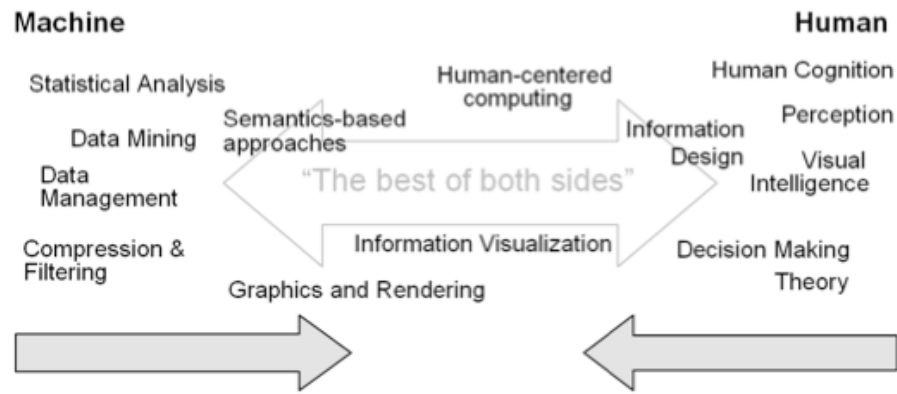


Figure 2.1. Visual analytics integration [11]

Relationship between visual analytics and scientific disciplines has been widely investigated. Keim et al. described that Visual analytics has wide combinations of interdisciplinary research fields, such as data mining, data management and statistics [11]. The integration of these scientific disciplines, as illustrated in Figure 2.1, provides a broader way of analytic reasoning. However, there are a number of important differences between visual analytics and information visualisation. Visual analytics aims to break the barriers of information visualisation limitation by enhancing the knowledge discovery algorithms [11]. In contrast to information visualisation, visual analytics employs the strengths of data analysis algorithms to support interactive analysis tasks [11]. The application areas of visual analytics are significantly different from those of information visualisation, especially, in gaining insight by processing large amounts of data [12].

The visual analytics process has different stages and transitions, which is more extensible compared to conventional information visualisation. As illustrated in Figure 2.2, these are to provide interactions and increase interoperability between stages [12]. The end result will enable users to acquire new knowledge from vast amounts of raw data.

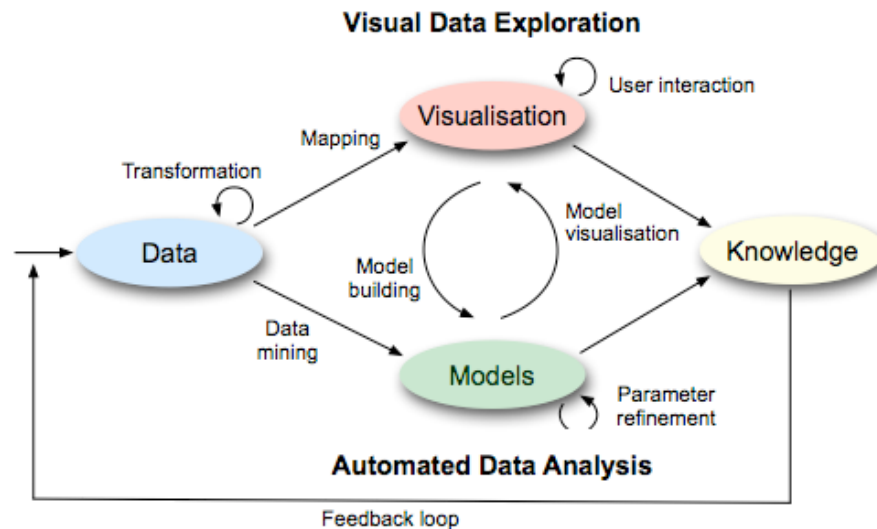


Figure 2.2. Visual analytics process [12]

The initial phase requires data sources integration before visualisation representation [12]. Prerequisite steps need to be taken before going further for applying automated data analysis. Initially, it will preprocess and transform the data in order to gain insight for further analysis. [2]. The preprocessing tasks consist of data cleaning, normalisation, grouping or data source integration [12]. Once the data has been processed and transformed, the next step can progress further to either visualisation mapping or model building. The interactions with automatic methods are characterised through parameter refinement and analysis algorithms selection [12]. This step enables a user to evaluate the model by using data mining techniques in order to verify preliminary results. Key characteristics of this visual analytics process include being able to discover misleading results to ensure a measure of certainty. However, this requires an advanced user to confirm an initial hypothesis that is matched to the automated analysis to reveal hidden results. Finally, the new knowledge can be discovered, by using automatic analysis the interactions between visualisation and model can be performed by human analyst to support their task“ [12].

## 2.2 Malicious Bots

An Internet bot, also identic with robot or bot, is automated software that performs specific tasks. A website crawler, such as Googlebot, is an example of a bot. However, this kind of bot is specific to website indexing tasks and can be regarded as a non-malicious bot. In a world of persistent threats, there are many other malicious bots that can be found on the Internet.

The evolution of scrapper bots, social bots and distributed denial of service bots has a significant impact resulting in financial and reputation consequences. These malicious bots presence has emerged as real threats to the Internet. Large and growing bot traffic has increased up to 61.5% of all website traffic according to a study by Incapsula, which 30.5% of the traffic are likely to perform malicious activities [1].

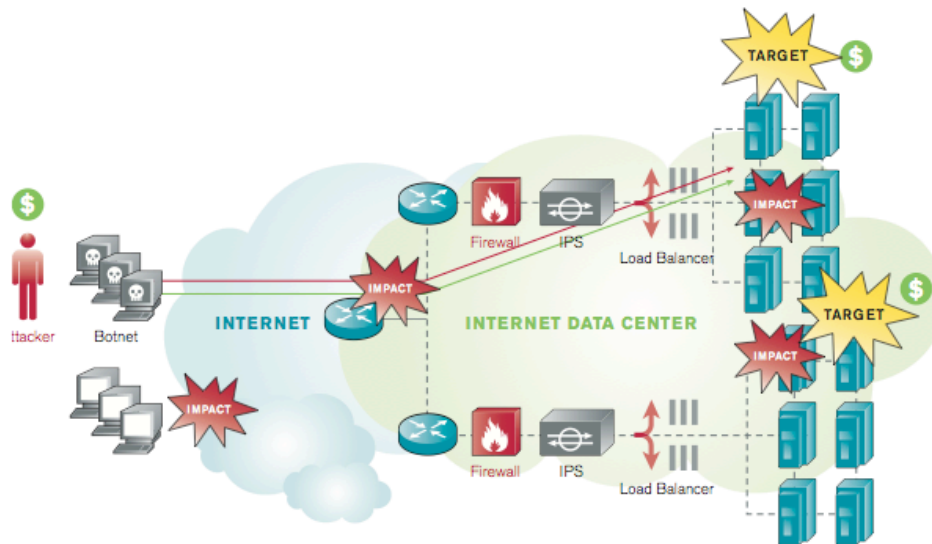


Figure 2.3 Application layer attacks [2]

Adaptive and persistent attacks have proven to evade detection and prevention system, hardware devices, such as firewall and IPS, are insufficient and susceptible to DDoS attacks [2]. Previous study, conducted by Moustis and Kotzanikolaou, has reported that *“more recent DDoS attacks targeting at the HTTP layer can be very effective even with a small number of infected bots”* [13]. Their study has demonstrated that application layer attacks can be deployed with a very limited resource, which only require a small size botnet to disrupt web servers. They examined HTTP SYN-flooding DDoS attacks with a small number of bots by using Slowloris [13]. A similar study, conducted by Zargar et al., has identified several key difficulties in mitigating application layer attacks, because the new adaptive application layer DDoS attacks require small number of bandwidth and target the vulnerabilities of application layer protocol. [14]. Moreover, Durcekova et al. investigated sophisticated attacks aimed at the application layer [15]. They proposed two detection mechanisms for monitoring web traffic in order to discover abnormal burst traffic.

Interestingly, a study by Juniper Networks revealed that DDoS attacks could be purchased with a little cost [16]. The new breed of slow and low application-layer DDoS attacks can stay under a threshold limit of detection systems. A similar study by Prolexic reported that the peak attack bandwidth in 2014 has increased 133% compared to the previous year [17]. The new reflection and amplification attacks are the most prevalent and disruptive compared to the traditional botnet infection method, generating peak traffic of more than 200 Gbps and 53.5 Mpps, [17]. These attacks constitute a real threat to industries. Figure 2.4. illustrates the distributions of attacks targeting key industries, in which 49.80% of the attacks target the media and entertainment industry. This provides insight into the motivation of attackers, which is for press exploitation in order to effectively reach out and recruit others to join their cause [17].

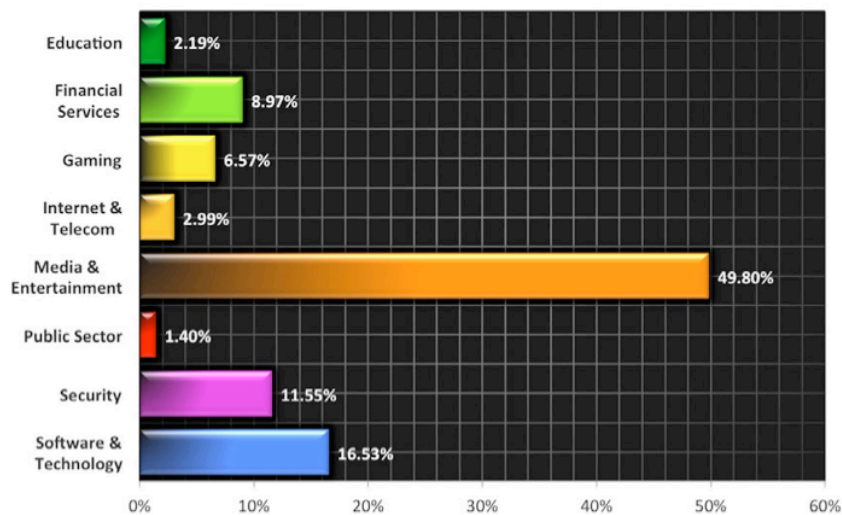


Figure 2.4. Distribution of attacks [17]

A relationship exists in showing the effectiveness of application layer attacks and their impact on service availability. Together these studies provide important insights into the disastrous results of application layer attacks that can be launched effectively.

### 2.3 Website Heatmap

Recently, there are various cloud-based services that offer a website heatmap solution. The open source JavaScript library that produces real-time website heatmaps are also widely available [18]. A website heatmap provides an effective tool in order to analyse visitor behaviour. The technique aims to produce a graphical image that represents visitor interactions with each section of a certain

page. These interactions, for instance, are mouse movement, click density, form usability and scroll movement. An advance website heatmap enables usability study for many mobile and table specific devices.

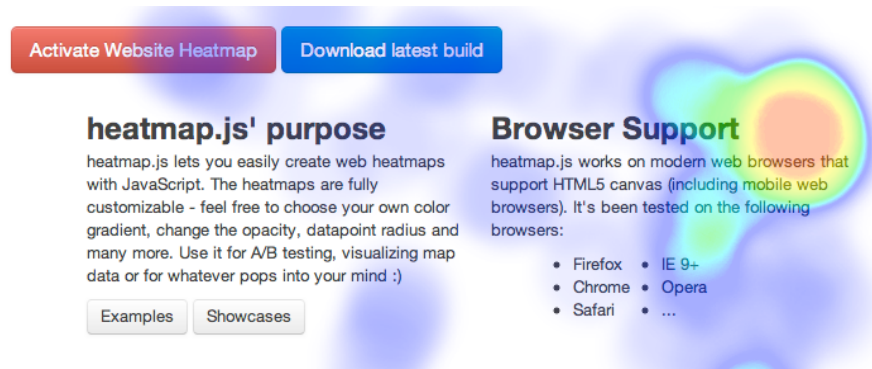


Figure 2.5. Website heatmap [18]

The flexibility of this technique allows data to be created from many different types, such as mouse and keyboard tracking, to enhance usability and design study. One of the limitations with this technique is that it requires JavaScript to be enabled in the browser side, while some requires a HyperText Markup Language (HTML) version 5 enabled browser. Recent Online Social Network (OSN), however, demands significant usage of JavaScript.

A study by Huang and White discovered the correlation between mouse movement and search results, which will be examined by a visitor [19]. Although the website heatmap technique could predict high presence of a human, but currently it is not widely applied for detecting bots presence. However, previous studies have demonstrated that legitimate browsing behaviour can be perfectly imitated to evade detection [4, 5, 6, 7, 8]. Currently, no empirical studies on the potential benefits of website heatmaps have been conducted to detect the presence of bots.

#### 2.4 Browser Fingerprint

Tracking users can vary in techniques, such as generating browser cookies or using browser fingerprints. Previous study by Eckersley has shown that web browser attributes can be used to transparently fingerprint clients, even when browser cookies are disabled by default [20]. The result indicated that browser fingerprinting enables web browser identification without relying on *User Agent* string headers. Browser fingerprinting provides a method to discriminate machines

behind a single source address, such as Network Address Translation (NAT) and proxy server.

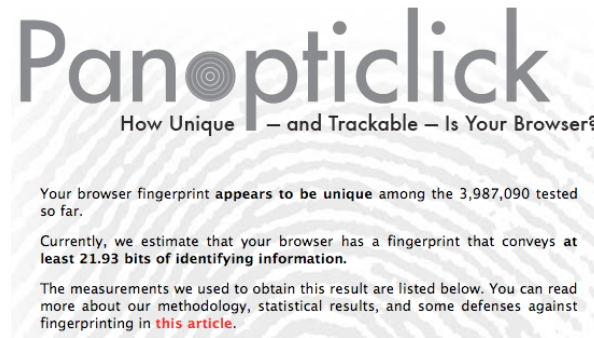


Figure 2.6. Browser fingerprint [23]

The approach is based on research study that at least one in 286,777 browsers shares the same fingerprint [20]. The demonstration estimates that under current active session only one in 3,987,090 browsers will share the same fingerprint [23]. The browser fingerprinting technique allows browsing sessions to be linked together. The finding suggests that browser fingerprint can be effectively used to replace browser cookies in order to track visitor uniqueness [20]. A similar study, conducted by Mulazzani et al., confirmed the reliability and efficiency of browser fingerprinting in terms of bandwidth and computational overhead [21]. Moreover, Mowery et al. study has shown that it is difficult to simulate or impersonate browser fingerprint unique session that belongs to other user. [22].

However, browser attributes are constantly changing, due to plugin installation and setting configuration. In making decisions based on dynamic attributes, it is important to present to the analyst the results and impact of that uncertainty. This is to ensure unique identification between clients.

## 2.5 Banner Advertisement

Website banner advertisement is becoming pervasive, and the ability to attract customers has been adopted by online news websites, such as Forbes and InformationWeek [24, 25]. The banner can be in a form of full page or only at a small fraction of size. Typically, a banner resides in a floating area, while others prefer an embedded banner in a single page. A banner has a closing or a skip link, which forces a visitor to close them in order to continue browsing the website. A proof-of-concept demonstration of this situation is taken from the

InformationWeek website, as illustrated in Figure 2.7 below. This condition requires the visitor to respond or wait in order to access the main news.

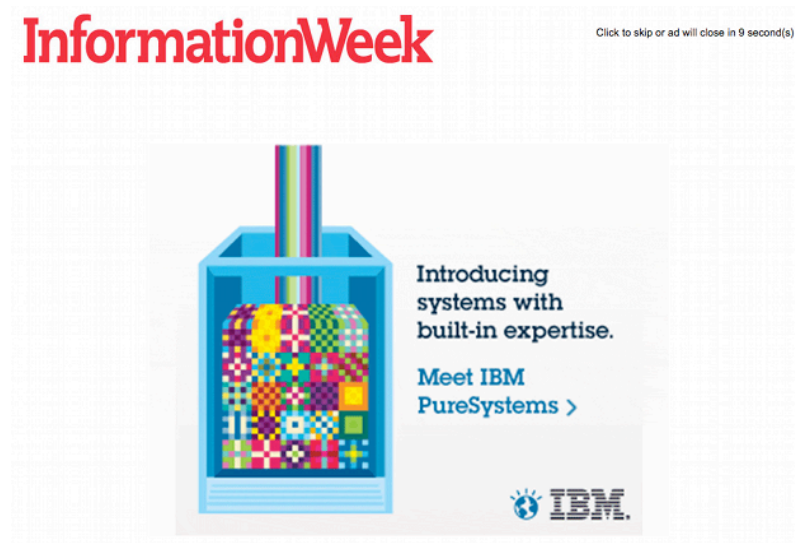


Figure 2.7. Banner advertisement [25]

So far, several studies have been conducted to investigate visitor behaviour towards advertisement [26, 27]. Similarly, Kitts et al. proposed a method for click fraud bot detection, by calculating mix adjusted traffic and bot signatures [28, 29]. However, no empirical studies of the potential benefits of user tracking advertisement reaction have been conducted to provide a measure of certainty of a human presence. Such methods would have a high possibility to replace CAPTCHA in the future [29]. While the results are still uncertain, however, this allows a combination of security protection with the opportunity of banner advertisements, which can lead to beneficial revenue behind them.

## 2.6 Proof-of-work System

Kaiser and Feng defined the goal of a proof-of-work system, it aims to maximize the amount of workload that adversaries must conduct and minimize the legitimation process on the server side. [30]. Typically, a proof-of-work system enforces clients in committing their computational resources. According to a study conducted by Pandey and Rangan, it has been shown that this technique effectively protects system resources against denial of service attacks [31]. In their study, a variety of algorithms, Token Bucket and Fair Queue Algorithm, were used to prevent brute force attacks. However, a previous study by Laurie and Clayton argues that proof-of-work itself can be a better solution for e-mail spams [32].

They examined the approach from both economic and security perspectives. Their study indicated that an uncomplicated scheme for email challenge-response is not a guaranteed solution to mitigate spams. Despite the wide usage of the proof-of-work system for security purposes, however, determining how much processing power and how much work are required in order to provide stability remains uncertain. An active monitoring system, that supports visualisation, would be an ideal solution in providing insight. Figure 2.8 illustrates challenge-response - a part of the proof-of-work system.

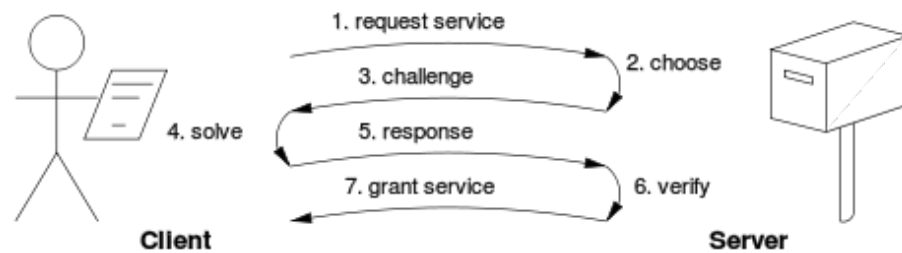


Figure 2.8. Challenge-response [33]

In the application of user behaviour tracking, the proof-of-work system can be combined with session tokens or One Time Password (OTP). This is to ensure protection against brute force and replay attacks. A randomly generated JavaScript code, containing the task to solve, can be securely delivered to the client by using Secure Socket Layer (SSL). Hardening methods such as code obfuscation can also be applied. However, the key challenge is to maintain integrity of each session in order to ensure such accuracy.

## 2.7 Discriminating Bots against Legitimate Flash Crowds Traffic

The increasing rate of malicious bot attacks has attracted many researches to address this challenge. One major difficulty in application layer attacks is to distinguish bots from legitimate flash crowds traffic, “*A flash crowd is a surge in traffic to a particular Web site that causes the site to be virtually unreachable*” [34]. Several attempts have been made on the transport and network layers to provide solutions. Yu et al. reported the effectiveness of their method by using flow correlation coefficient [35]. The result of Thapngam et al.’s study reported that linear discriminant analysis could approach the problem with a high rate of accuracy [36]. However, these methods are not specifically designed to mitigate



mimicking attacks that exploit the vulnerability of an application layer [4, 5, 6, 7, 8].

A number of studies on the application layer have been conducted in order to discriminate bots against legitimate flash crowds traffic. Kandula et al. developed *Kill-Bots* in order to guard web servers against DDoS attacks [37]. During intense traffic spikes, *Kill-Bots* enforces clients to solve graphical tests before allowing them access to the server. One major drawback of this approach is that the method requires CAPTCHA puzzles to be solved.

Xie and Yu proposed their method based on user browsing behaviour [38, 39]. Their method applied a Hidden Semi-Markov Model to detect anomaly of a user browsing behaviour. The browsing behaviour was observed by the request rate, page viewing time and request sequence. The result presented in their study reveals the effectiveness of defence against application layer DDoS attacks. However, previous studies have demonstrated that legitimate browsing behaviour can be perfectly imitated to evade detection [4, 5, 6, 7, 8].

Oikonomou and Mirkovic proposed their method by using human behaviour modelling [40]. They investigated request dynamics of a human interaction in order to learn request patterns. In addition, a deception approach, by embedding invisible objects, was used to detect bots presence. However, their method cannot effectively defence against flash crowds attacks. Yu et al. have demonstrated the difficulties to detect mimicking attack, especially when the number of active bots is massive [8].

## **2.8 Visual Analytics Applications**

Visual analytics approach can significantly support the process of knowledge discovery from very large and complex datasets. Considerable amounts of literature have been published on the topic of visual analytics.

Zhang and Huang studied visual analytics model to detect DDoS flood attacks [41]. They proposed Density-Workload Model with three coefficients in order to investigate the attacks and their impact level. Their study analyses the relationship between system performance and attack density by using clustered visualisation. A graph model, consisting of nodes and edges, was mainly used to represent the flood attacks. The result has shown that the model produces high accuracy to measure

different types of flood attacks. However, their method of visual analytics model has a number of limitations. This model lacks the ability to identify bot attacks and flash crowd attacks, which have different traffic patterns with flood attacks. An interactive respond to handle flood attacks is missing in their study. In the absence of intense traffic spikes, it is very difficult to identify low-bandwidth and slow HTTP attacks with this technique. They limited issue with this method is that it does not identify legitimate flash crowds traffic against flood attacks.

Zhao et al. developed *NetSecRadar*, a real-time visual analytics system to monitor network security events [42]. A radial graph, composed of hosts and attack correlation, was mainly used to aid monitoring Intrusion Detection System (IDS) alerts. Their system supports filtering, animation and direct interaction with the user. Preliminary result has shown the ability to illustrate attacks and visually correlate the events. However, *NetSecRadar* is IDS dependent. The ability to identify malicious bot attacks largely depends on the IDS database. Therefore, a key feature to discriminate flash crowds against DDoS attacks is missing.

Fischer et al. developed *VisTracer*, a visual analytics tool to investigate routing anomalies [43]. They introduced novel glyph-based and graph-based visual representations. The interesting part of their study is the ability to distinguish between legitimate routing changes and malicious activities from legitimate addresses. However, the visualisation is only specifically designed to detect routing anomalies on large traceroute datasets, which has not been applied to discriminate malicious bots traffic.

Keim and Fischer developed *VACS*, a novel visual analytics suite for cyber security [44]. The system allows visual exploration to identify suspicious host behaviour by using *Graph Viewer* and *Hierarchical ClockMap*. Their study has demonstrated the ability to identify and explain unusual activities in the network by using visual analytics. The evidence presented in this study suggests that *VACS* is best used on a large display to enhance situational awareness. However, no explanation is given on how to detect unusual happenings inside the network.

Shurkhovetsky et al. studied visual analytics for network security [45]. The study investigated visualisation tools to assist network forensic analysts. They use a combination of multiple interactive visualisations to provide a global and detailed

view of network activities. However, the visualisation makes no attempt to differentiate between various types of DDoS attacks. Another weakness is that the detection method for suspicious events inside the network was not fully described.

Yassem presented a visual analytics approach to network security hygiene [46]. The research investigated visualisation methods and techniques to enhance situational awareness. The visualisation implements a stacked bar chart, hierarchical edge bundle and Hilbert curve into a single dashboard. However, it is unclear how situational awareness can be enhanced from an analyst perspective. The research fails to fully define what types of attacks were detected by the visualisation. Another weakness is that the visualisation fails to provide user interactions, which is an important feature for visual analytics.

Mansmann et al. presented a real-time visual analytics system for dynamic event data streams [47]. A loosely coupled modular visual analytics system was introduced. The extensible framework allows internal threats identification and suspicious events investigation. The result has shown that *“it can be used to smoothly switch between historic events and the most recent events for monitoring purposes and relate incoming data with historic events”* [47]. The visualisation system enables suspicious user behaviour tracking and recognises abnormal patterns. Although this is the most comprehensive modular system of visual analytics produced so far. However, there are limits to how far the idea of burst and anomalous behaviour can be identified, which is missing in this framework. In this study, no attempt was made to classify malicious bot attacks.

## 2.9 Summary

After conducting the above literature review, the following shows a summary of the identified weaknesses with the existing work:

- Several attempts have been made on the transport and network layers, but they are not specifically designed to mitigate mimicking attacks that exploit the vulnerability of an application layer.
- Some existing solutions such as *Kill-Bots* offer effective protection but this relies on CAPTCHA puzzles to be solved. This method only focuses on preventing DDoS attacks that masquerade as flash crowds, which is not suitable for low bandwidth attacks.

- Most of the existing solutions still rely on the use of statistic-based methods that can be perfectly imitated to evade detection.
- The existing solutions lack the ability to identify bot attacks and flash crowd attacks, which have different traffic patterns with flood attacks.
- In the absence of intense traffic spikes, it is very difficult to identify low-bandwidth and slow HTTP attacks with the existing solutions.

This chapter has outlined the literature review and explained the necessary understanding of the visual analytics, malicious bots, website heatmap, browser fingerprint, banner advertisement and proof-of-work system. It was found that related studies have been conducted in the domains of visual analytics and malicious bot attacks. Although these studies did not evaluate specific classification methods for adaptive bot attacks, they provide valuable insight for our research study. A combination of visual analytics and traffic discrimination techniques could provide a measure of certainty and give a strong indicator of human or bot presence, which is missing in the current proposed methods. Therefore, the limitations of current approaches have motivated this research.

There are several studies on the effective use of CAPTCHA in order to discriminate the presence of bots. However, existing studies have not dealt with the use of banner advertisement to predict the presence of a legitimate visitor, based on the response behaviour. The next chapter will describe our research methodology, which was defined using experiment and simulation approaches for undertaking the research to fill these gaps.

## Chapter 3 VATRIX

In the previous chapter, an overview of visual analytics and malicious bots attacks has been introduced. In addition, the challenges of this research area have been discussed. Furthermore, the applications of visual analytics and the technique used in order to discriminate bot traffic have been identified. Recent work in the domain of visual analytics and network security has presented techniques that allow an analyst to monitor network security events, such as analysing the flow of DDoS flood attacks. However, the prior work in network security visualisation mostly applies graphs directly without the use of any classification technique.

On the other hand, visual analytics is an emerging scientific field with the ability of interactive visualisation to provide collaboration among analysts. The approach presented in this chapter aims to create an interactive visual analytics system, VATRIX, that automatically discriminates bot traffic and provides a rich user interface in a multiple dashboard view. VATRIX embraces the use of a Voronoi diagram, Quadtree data structure and priority queuing technique in order to provide a defence mechanism and to allow an analyst to gain insight from large amounts of data. VATRIX also employs X-Map in order to classify incoming traffic and discriminate them against bot traffic, which will be discussed in Chapter 4, whilst also producing a system that observes visitor behaviour.

### 3.1 Introduction

This chapter presents the visualisation techniques employed in VATRIX - Visual Analytics Through Responsive Interactive X-Map, which was mainly developed during this research study. It covers the user interface, mapping specification and architecture of the entire system. The aim of VATRIX is to provide a defence mechanism, by using interactive visual analytics, with fair allocation across all the visitors in the presence of attacks. By using advanced interaction the advantage is that it does not increase the amount of representation complexity. Therefore, action can be taken more effectively and efficiently. The next section will present the VATRIX architecture. The architecture consists of two main parts and three stages, the data acquisition and visualisation. Data acquisition was used to collect computing and network resource usage data. On the other hand, visualisation presents interactive graph to the end user in order to explore the large datasets.

VATRIX is a web application using HTML5, JavaScript, mixed libraries and several toolkits. To produce the visualisation D3.js was used.

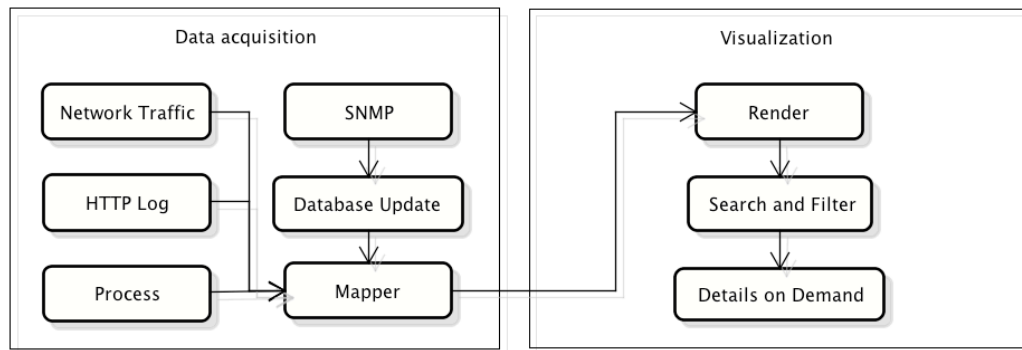


Figure 3.1. VATRIX architecture

Stage 1, the data acquisition stage aims to gather all traffic entering the network. It filters all incoming HTTP and HTTPS requests inside the network. It is expected to handle over millions of packets per second. Several raw data are extracted to provide details on demand in the visualisation stage, for instance IP address and source port. In the application layer, visitor data, such as website heatmap, browser fingerprint and other attributes are also extracted. This stage is constantly comparing current profile with previous history profile to avoid duplication and performance overhead.

Stage 2, the classification objective is to classify incoming network traffic, especially HTTP and HTTPS requests. The classification is based on visitor behaviour training data set. It analyses previous patterns to identify legitimate or illegitimate behaviour and combining the results with details on demand information to provide a measure of certainty. The result of this classification will be visualised according to the final class.

Stage 3, the visualisation stage renders all the classification results. This stage provides interaction to the analyst. Analyst is now able to discover bots presence. Analyst can quickly block or route suspicious requests from the interface to divert malicious bots activities, for example forwarding requests to puzzle server with a banner challenge. All other legitimate requests will not be affected.

### 3.2 Quadtree Data Structure

A data structure for visualization is of significant interest in representing elements including points, areas and lines. A quadtree data structure is “a tree data structure in which each internal node has exactly four children [68]. The node has a similarity with a binary tree. The partitioning of this data structure suits well for mapping multiple IP addresses, as illustrated in Figure 3.2. Each leaf node of the data structure has four branches, which each region represents a partition of one octet of a valid IP address.

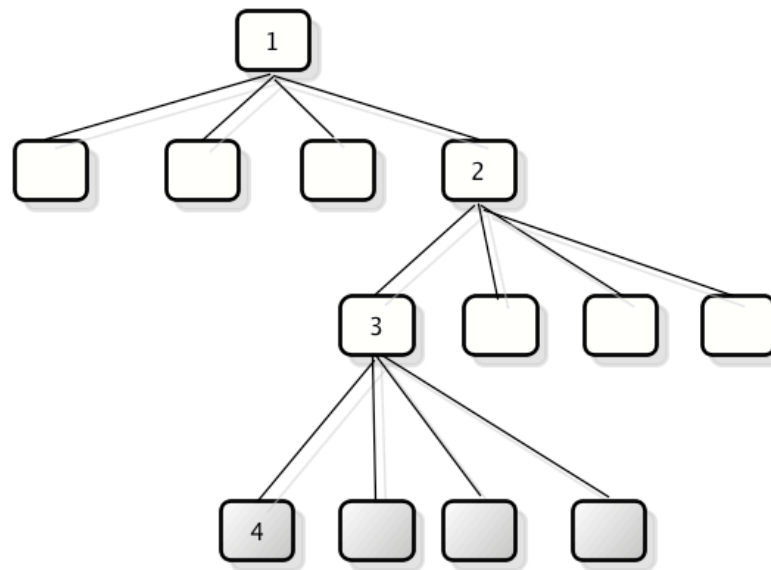


Figure 3.2. An example of Quadtree representation

The data structure allows adding, deleting or replacing elements dynamically according to practical situation. By using this data structure, the disadvantage of redundant IP address mapping can be enhanced because current visualisation tools are restricted to represent constraints of massive list of IP addresses. Thus, the rendering for the Voronoi diagram can be grouped into four different regions making the first octet of the IP address to be adjacent for each group.

### 3.3 Voronoi Diagram

Voronoi diagrams are considered to be the most useful technique in the area of visualisation and computational geometry, “a Voronoi diagram is a way of dividing space into a number of regions.” [69]. The definition of a Voronoi diagram consists of a set of  $n$  distinct points,  $P = \{P_1, P_2, P_3 \dots, P_n\}$  [69], where the  $P$  set can be defined as the division of a region. The diagram can be presented

as the division of a plane into multiple regions showing vertices and edges. A connected line segments of the boundary regions are the voronoi edges and each endpoints of the edges would represent voronoi vertices [78]. The application of the diagram can be plotted to the bound cells for the points of a dataset to represent all elements that will be visualised. This enables interactive visualisation to be presented into multiple regions and manage in a more compact way. Figure 3.3 illustrates a typical Voronoi diagram also known as a point diagram.

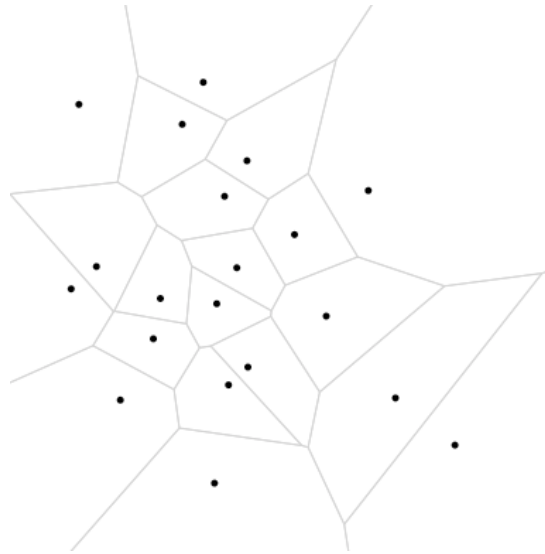


Figure 3.3. A typical Voronoi diagram

The division of space for any point that lies in the region has a site that is nearest to each other for rendering the final visualisation. This is to enable linking of multiple points, such that all points in the region are closed to each other. Thus, it supports grouping and filtering for a large dataset because a Voronoi diagram provides a useful representation for generating a minimal number of elements [69]. For all layers in the Voronoi diagram, the following requirements apply:

- The region contains at least one point to represent the first octet of the IP address.
- The layer contains a finite number of points.
- The diagram is represented in a two dimensional grid.
- Each point that is closest to each other contains a unique value, which is the first octet of the IP address. For instance, the same first octet of the IP address that will be grouped together will be adjacent to each other.



### 3.4 Generating Voronoi Diagram

Generally, constructing a Voronoi diagram is time-consuming, especially for a large dataset with a huge number of pixels. In this section, a custom, fast and efficient algorithm is implemented for generating a Voronoi diagram of connected components, which takes a quadtree data structure, as illustrated in Figure 3.4. The proposed algorithm for constructing the area of a Voronoi diagram is:

```
procedure ConstructVoronoiArea
1: v ← initialiseArea
2: I ← parsePacketQueue
3: for each incoming_packet of I do
4:   h ← parseHeader(I)
5:   Q ← quadtree_grow (h)
6:   if firstEntry == null then
7:     merge-data-points [h]
8:     group-into-clusters [h]
9:     calculate_timing
10:  else
11:    appendNewVertices(v)
12:    appendNewCoordinates(v)
13:  endif
14: end for
15: r = API_generateVoronoi(v, Q)
16: if r >= 1 then
17:   API_drawDiagram(v, Q)
18: endif
```

Figure 3.4. Construct Voronoi

To illustrate the above algorithm, the steps can be visualised as follows:

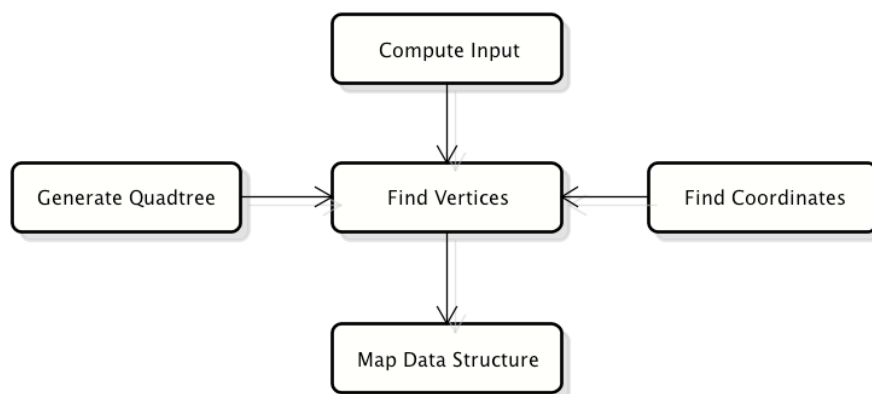


Figure 3.5. Constructing Voronoi diagram

The Voronoi diagram takes a quadtree data structure, which is a group of four different regions, for all IP address entries as computed by using a decision tree algorithm to perform the classification task for making detection based upon past observations. The chosen algorithm was J48 Decision Tree C4.5 algorithm. Then the Voronoi diagram is calculated in an optimal time for every created layer. Therefore, in the generated Voronoi diagram of all dataset the vertices are presented, where a region label indicates the first octet of the IP address. By using these vertices, multiple layers of Voronoi diagrams can be generated to provide visualisation interaction. These layers are stored in main memory, which can be used when required. In order to plot an IP address into the diagram, the initial step is to obtain all octets from the quadtree data structure (set IP). To generate the first order of the block in the diagram, it starts with first IP set and mapped each cell found in the structure of  $X = \{x_1, x_2, x_3, x_4\}$ , where the order of the cell will follow the structure of the quadtree data structure. For a given set of IP address, the farthest-point divides the plane into several cells where the same point of the octet will be adjacent to each other. For instance, an entry of IP address in class C network will be adjacent and formed together into one single group of cell.

The role is to divide continuous space into mutually disjoint subspaces according to the nearest rule. Therefore, a selection of multiple regions, containing multiple layers, is possible by decomposing the space into regions around each point.

### **3.5 Priority Queuing Method**

One of the most time-consuming aspects of analysing visitor behaviour is to prioritise incoming traffic. The system must capture every behaviour and perform classification. For instance, a huge amount of traffic may be taken before decision can be made. In order to optimise the task, in this section, the priority queuing method is presented to classify visitor behaviour. The method has a role to prioritise incoming request patterns of a legitimate visitor by controlling the order of activities. The queue is implemented in an abstract data type, which has functionality for enqueueing and dequeuing operations. Figure 3.6 illustrates the queuing and dequeuing process.

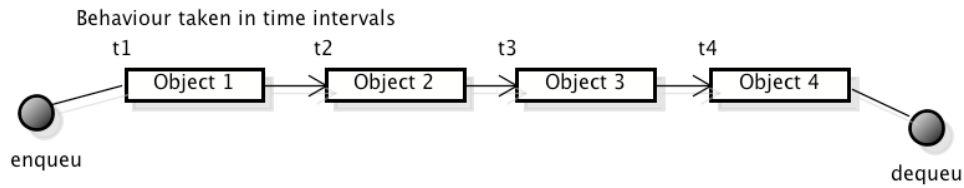


Figure 3.6. Priority queue

The result of the priority queuing method enables traffic to be grouped in one object and constructed in a prioritised way. Thus, classification can be performed more effectively, especially for large datasets. There are several implementations of First In First Out (FIFO) queues which have more advantage over fixed length arrays, such as a singly or doubly linked list [73]. This enables new elements to be added dynamically without any capacity constraint. To construct the priority queue, it can be visualised as follows:

1. A buffer starts with an empty element.
2. An object is added into the queue.
3. The object will be processed to perform classification.
4. The object will be removed from the queue upon successful classification.

This shows how the queue buffer is managed to support classification. The data structure of an object can be defined as follow:

```
typedef struct {
    int      size;
    int      first;
    int      last;
    vType    *elems; } QueueBuffer;
```

### 3.6 VATRIX Interactive User Interface

Our research in visual analytics combines interactive information visualisation and automated analysis techniques. This allows the development of an interactive user interface and the ability to render large amounts of data. The purpose of the user interface in VATRIX is to allow an analyst to interactively gain insight from large amounts of data. The diagrams presented in this section use Voronoi diagrams, a Quadtree data structure and priority queueing method in combination with d3.js for the presentation.

In the absence of intense traffic spikes, it is very difficult to justify whether a system is under attack. As network traffic produces vast amounts of data, using

conventional visualisation tools alone, as illustrated in Figure 3.7, can be difficult to grasp a good feel of the data, especially when trying to identify the class of the traffic.



Figure 3.7. An example of conventional network traffic visualisation

The VATRIX visual analytics system provides a main dashboard view to represent human, bots and unclassified traffic. This behaviour is rendered in a real-time mode by the help of X-Map. It visually represents the current network traffic. Therefore, the visualisation in VATRIX is presented as illustrated in Figure 3.8, so that fast interactive responses can be made, such as blocking all bots traffic or taking further investigation. Using this mechanism, it allows a high-level traffic priority for legitimate users accessing the website. It can help to get further insights into the structure of data and interactively explore each plot.

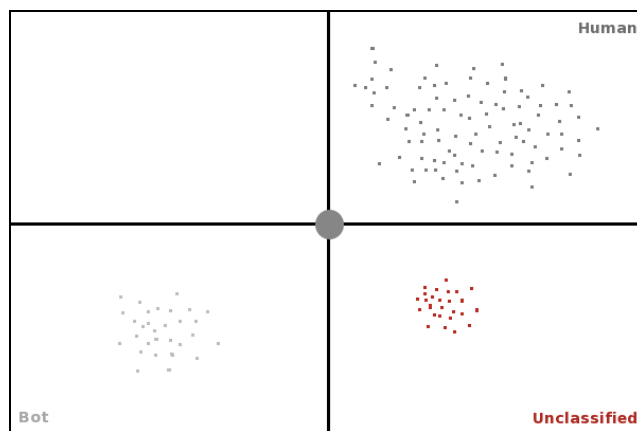


Figure 3.8. Vatrix interface

Given vast amounts of network traffic, an analyst could interactively select part of the data to determine its destiny or visualise it in detail. The analyst could grasp more comprehensive details without being constrained by large data.

The output of the current selection, as presented in Figure 3.9, can be exported in a CSV or text format containing the source of addresses and observation data. This brings a simple action without changing the representation of the overall visualisation.

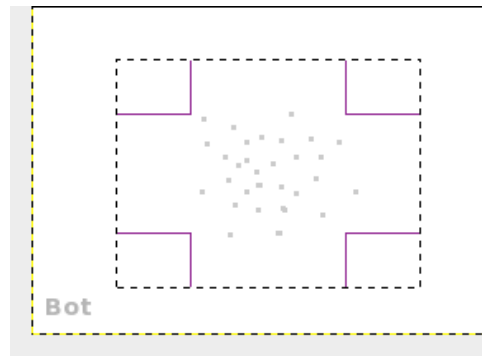


Figure 3.9. Bot selection

From the current selection, there are other features that can be made, such as:

- Export to CSV
- Export to Text
- Block Traffic
- Redirect Traffic
- Run Custom Script
- Assign Class
- Expand Visualisation

Each of its pixels represents a single client connection. The colour of a pixel represents the maximum value of activities; the dark colour of the pixel represents a high peak of activities. The pixels are arranged so that connections are close to each other; adjacent traffics are mapped to adjacent pixels. This gives fair visualisation allocation when flash crowds might occur in a network. The flash crowds activities shown in Figure 3.10 below illustrate all possible drawn pixels when incoming network traffic is in high load. The diagram shows overcrowded activities, which are intended to improve by using Quadtree and Voronoi diagrams.

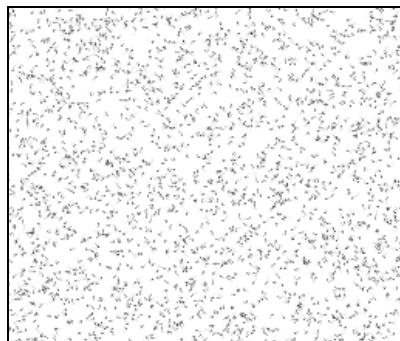


Figure 3.10. Flash crowds activities

The next section describes the details on demand feature. It gives an analyst information details and control over an incoming network connection. The graph is produced by using a heatmap diagram.

### 3.6.1 Details on Demand

Another feature that is supported by the user interface is details on demand capability, which is the key for the visual analytics system. Details on demand can show which activities are intense. The prototype interface follows a visual analytics approach by using a two-dimensional representation of a graphical heatmap. It aims to provide a better understanding to show the intensity requests of the current situation.

As illustrated in Figure 3.11, pixels cycle their colours over their lifetime, and the animation speed level depends on their activities. The pixel colour is used for showing the intensity of an activity; a darker colour represents an active request. Therefore, it is possible to see exactly how request loads are progressing according to their pixel colours. Suppose an analyst wants to explore the data payload. He could then select the interesting part of an unusual network traffic pattern; represent it in pixels; explore the header with its payload for further checking.

This heatmap view acts as a complement for the VATRIX main dashboard view and it represents an overall incoming connection coming to the network. It aims to reduce the drawing complexity as illustrated in the previous Figure 3.10.

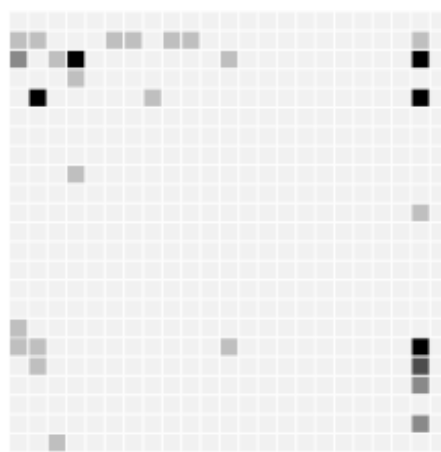


Figure 3.11. Heatmap view rendered with d3.js

The procedure starts by initialising the heatmap graph using the third party library, D3.js [58]. It actively monitors incoming requests. Each connection will be

rendered on a different pixel location according to its request load, which can be differentiated by its colour. The following procedure is presented to generate the heatmap view:

```
procedure GenerateHeatmapView
1: H ← API_drawHeatmap()
2: while(conn[] = grabIncomingRequest())
3:   for each conn of I do
4:     for x=1 to MAX_ROW
5:       for y=1 MAX_COL
6:         if conn.requestRate > HIGH then
7:           H.drawPixel(x, y, COLOR.DARK)
8:         else
9:           H.drawPixel(x, y, COLOR.LIGHT)
10:        endif
11:      end for
12:    end for
13:  end for
14: end while
```

Figure 3.12. Heatmap View

The heatmap view is firstly initialised before any pixels are appended to the diagram. For each incoming connection, it will be represented and drawn to the heatmap element. Each new incoming connection will cycle through the graph where the intensity of the request rate will determine the colour of its pixel. An analyst can view further details of the connection regarding the probability score, which will be presented in the next section. This enables an analyst to gain insight regarding the current connection that traverse through the network.

### 3.6.2 Weight Score

The weight score is a collection of calculated values for each visitor data, where the value ranges between 0 and 1. The score is generated based upon visitor observation, which will be discussed in Section 5.3. An entry of each incoming connection corresponding to a certain task can be interpreted as an activity score to be classified and rendered by the system. The graph contains connection information details, which are rendered by the previous Heatmap view. The graph, which is produced by using d3.js, aims to complement the Heatmap view by adding weight scores for all pixels. For instance, a high score on user interaction – between 0 and 1 - represents a higher likelihood that the traffic belongs to human.

These techniques provide a measure of certainty and can assist an analyst to emphasise that the current traffic belongs to a particular class. It determines the intensity level of activities, for example the value of request load. On a mouse hover event, the behaviour activities are presented with a activity weight score graph, as illustrated in Figure 3.13. This type of representation can be extended with a custom ordering of rows and columns, such as colour-based grouping. It allows expanding and collapsing of pixels to allow deeper exploration.

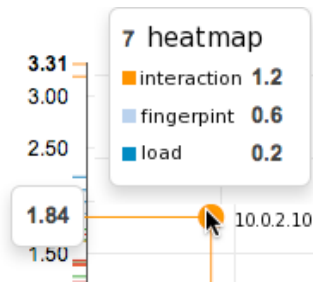


Figure 3.13. Weight score selection

A further breakdown on visitor behaviour can be generated, as illustrated in Figure 3.14 below. By using external heatmap Javascript plugin [18], this provides the current view of a visitor mouse interaction on a certain page.

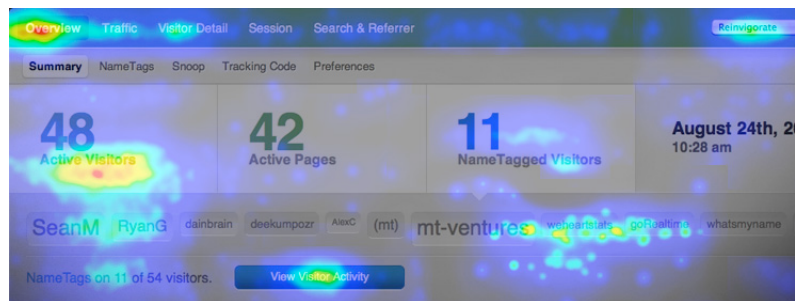


Figure 3.14. Website heatmap using plugin [59]

### 3.6.3 Extended View

Additionally, a VATRIX calendar view provides summarized traffic activities for each month, as illustrated in Figure 3.15. A lighter green colour indicates a high human presence. As opposite, a darker colour indicates a higher bot presence. This mixed colouring approach provides a better understanding of trend discovery for large amounts of data.



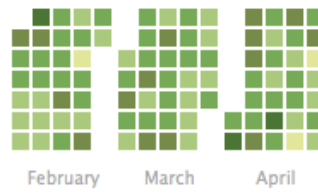


Figure 3.15. Calendar view

To explore the calendar view for further details, the system will render a histogram view to provide a graphical representation of the data distribution for a single or multiple date selection. Total requests are accumulated for each category. It calculates the total hits made by each category - human, bot and unclassified, as illustrated in Figure 3.16. For example, an analyst could select a specific date, month or year to summarize the overall connections for each category, which will be represented in a histogram view as shown below.

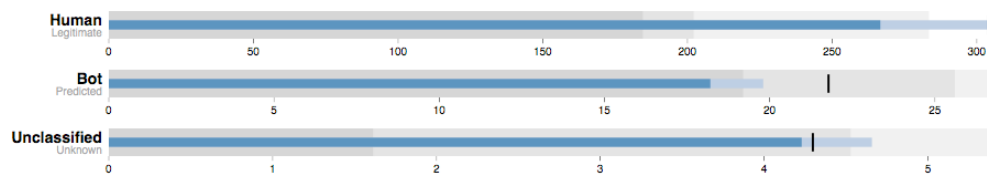


Figure 3.16. Histogram view

The histogram view is dynamically created according to the selected date range. This is to enable an analyst to view the accumulated requests made for each category. The following procedure is presented to generate the extended view:

```

procedure GenerateExtendedView
1: xmap ← XMapClassify
2: for each request_sequence of I do
3:   sum[] ← frequency(xmap)
4:   human ← sum[0]
5:   bot ← sum[1]
6:   unclassified ← sum[2]
7: end for
8: c = API_drawCalender()
9: date = API_getDateRange(c)
10: API_drawHistogram(date)

```

Figure 3.17. Generating View

### 3.7 VATRIX Response

The data collection task of the system is generated by using an external Apache module, `mod_status` module, which provides a way to monitor the internal performance of an Apache web server. However, the original development of this module does not support process states visualisation. It only renders the connection information into a tabular field. Often, important details are difficult to grasp by manually traversing the tabular field alone. VATRIX enhances this by providing a response and to monitor these process states in order to render them into a diagram. The following sub-sections will discuss about the Apache web server with the process states and how the visualisation will be produced based on a predefined matrix model.

#### 3.7.1 Apache Web Server

Apache is an open source and cross-platform web server software system. It is an industry leading web server that is widely used in production servers. As a fast and secure web server software system, Apache supports multi-threading that can handle millions requests at once [60]. It contains several compiled modules that can be extended to any functionality. According to a survey conducted by Netcraft, “*nearly two thirds of all Internet domains use Apache*” [61].

As highlighted by Li and Lu on performance optimization [62], the performance of the web server can be tuned by adjusting some important parameters dynamically. Their study examined system performance usage as the main measurement metrics and a combination of control theory and system model. They proposed a multiple input and output approach to model the web server. The result has indicated that Apache can be well fit for performance optimization.

One of the primary advantages of Apache is that it enables requests logging for information visualisation. All instances can be used to collect HTTP traffic at any given period. As default, an Apache web server records all incoming HTTP requests to a log file [63]. The format of the log file is highly configurable. During client requests, Apache also records error and diagnostic information. Apache has very comprehensive and flexible logging capabilities. It offers a wide range of options for controlling the log format. It is also capable of writing log files into a pipe of another process.

The ability to visualise HTTP traffic has always been both important and difficult. The main purpose is to detect traffic spikes, resource usage and attacks. In this case, a visualisation system must be able to discover and represent incoming requests in the network. Besides, it must be able to get more detailed information on each flow. The research conducted by Xiaojun in a remote monitoring system, presented a web-based centralised monitoring solution [64]. The study demonstrated a tool to efficiently link utilisation in flow-based networks by capturing and analysing control messages between servers. The result indicated that the proposed monitoring platform is suitable for heterogeneous network environment [64].

### 3.7.2 Process Mapping

Normally, an application server, such as Apache, is executed as a background process. It runs continuously for a long period of time. A process can increase or decrease its CPU usage, memory usage, and thread process since the startup of the application. Within this condition, a process thread may change its state, such as accepting connection, reading request, busy, idle, sending reply, waiting or closing connection. Figure 3.18 and 3.19 illustrate these states.

69 requests/sec - 62.5 MB/second - 0.9 MB/request  
 257 requests currently being processed, 383 idle workers

PID	Connections		Threads		Async connections		
	total	accepting	busy	idle	writing	keep-alive	closing
39553	1	yes	2	126	0	0	0
39554	93	yes	59	69	0	18	16
39555	25	yes	17	111	0	4	4
39556	137	yes	69	59	3	34	30
39557	174	yes	110	18	0	34	29
Sum	430		257	383	3	90	79

Figure 3.18. Apache server-status

These states provide an insight regarding the health of a process, a low traffic spikes but with a full of connection reading would indicate that resource exhaustion is happening, which may lead to a declined service.

```

                                     L
-----
W
R   R   R   RRR   R   R WWR   R W R WRRWW RWRW   R   W   WRWR
R   WRRWW   R   RRWR W WWW   R   W   W R R W RR   W WW WRRR
-----
R   R   W   R   W   W   W   W
W   R   W   W   R
W   W WRWRRRW RRRWR   W RR   R WW W RRWW   W R R   W WWWR
RRW WR   WWR   R WWR RRW W WWWW   R RWW R   W   W R WWRWR W
R RRWR WR   WWWW W RRRWR WRRLRRWRRRWRWR WRR WRWRRR WRWRRW RRRWR
W_RWWW RRWWWRRRWWWRRRWW   RWWWRRRRR R WR RWRWRWW WWRWRRRRW
.....
.....
.....

```

Scoreboard Key:  
 " \_ " Waiting for Connection, "s" Starting up, "r" Reading Request,  
 "w" Sending Reply, "κ" Keepalive (read), "d" DNS Lookup,  
 "c" Closing connection, "L" Logging, "G" Gracefully finishing,  
 "r" Idle cleanup of worker, "." Open slot with no current process

Figure 3.19. Apache scoreboard key

For a better understanding, it is useful to conceptualize these process states into a MxN matrix. A row represents a client connection, while a column represents a group of the web server instances. Each element in the matrix corresponds to a sum value of process identifier states, which shows the intensity of the requests. For example, the total of each state request for waiting connections (W) will be calculated and mapped to the MxN matrix element, which is dependent on the order of the incoming request. This concept enables better understanding of relationships between a process thread and its state, making each element to be listed in an efficient manner and represented in a very compact way. Periodically updating the value in real time for a given element in the matrix provides a possibility to estimate risk impact. Before any rendering is made, the value will be held in the matrix model as illustrated in Figure 3.20. Furthermore, the output of the rendering process will be shown as illustrated in Figure 3.21, which describes the status of the process group. The specific elements of a matrix can be denoted by a variable, in which a higher value will indicate a vast amount of requests.

$$\begin{bmatrix}
 a_{1,1} & a_{1,2} & a_{1,3} & \dots \\
 a_{2,1} & a_{2,2} & a_{2,3} & \dots \\
 a_{3,1} & a_{3,2} & a_{3,3} & \dots \\
 \vdots & \vdots & \vdots & \\
 \vdots & \vdots & \vdots & 
 \end{bmatrix}$$

Figure 3.20. MxN matrix

Many different types of process state can be rendered as illustrated in Figure 3.21. Filtering and grouping states is possible, and each plot can be adjusted effectively to any circumstance. The visibility of each group can be determined according to preference. This technique enhances awareness especially when many Apache instances are running across multiple clusters. Flooding requests could be spotted directly. An action can be taken quickly, and by selecting a given group state, an analyst can determine the final destiny of these requests.

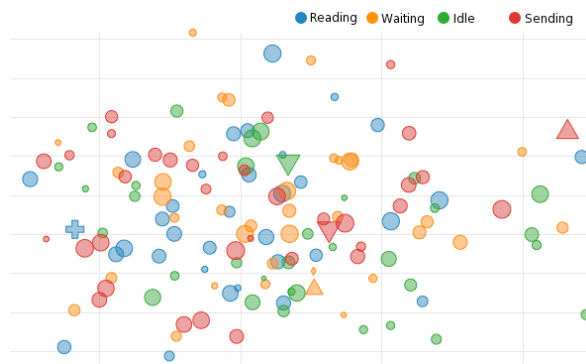


Figure 3.21. Process state visualisation using nvd3.js [65]

### 3.8 Summary

The evolution of scrapper bots, social bots and distributed denial of service bots has become an important issue. In order to secure networks from bot attacks and to discover incoming malicious bot activities, visual analytics may help to inform an analyst to gain insight from large amounts of data. Visual analytics provides better understanding, reasoning and decision making from very large and complex datasets.

In order to enable an analyst to derive insights from large amounts of data, several facilities are required. The first is a data structure to represent a large list of IP addresses. We use a Quadtree data structure that enables structuring of the data, which focuses on the data mapping that can be transformed and represented visually. In addition, a priority queuing technique is necessary in order to manage and process incoming traffic sequences. The second is a diagram that divides space into a number of regions. For this, we employ a Voronoi diagram that enables interactive visualisation to be presented into multiple regions and managed

in a more compact way. The presentation included in a single dashboard with multiple views enables automatic classification for legitimate and bot traffic.

Finally, in the absence of intense traffic spikes, locating the root cause of attacks is the key in order to respond efficiently. VATRIX enhances this by providing a response and to monitor web server process states in order to render them into a diagram. Current visual analytics systems lack the ability to respond interactively in the presence of attacks. Key features that need to be visualised, such as dynamic process states, are missing. Therefore, VATRIX, in contrast, provides clarity to allow ambiguous network traffic and dynamic process states to be clearly understood. It aims to discover unexpected orchestrated bot attacks by providing details on demand with the multiple views feature.

## Chapter 4 X-Map

New malicious bot approaches have led to an increasing rate of DDoS attacks [1, 2]. This increase has attracted many researchers to address the problem. However, one major difficulty in application layer attacks is to identify the presence of bots against legitimate flash crowds traffic. X-Map approaches this by making predictions based on visitor past observations, which uses the J48 Decision Tree C4.5 algorithm as the main classification algorithm. The result of X-Map is used for the visualisation purpose, as discussed in the previous chapter.

X-Map architecture allows client(s) to lookup single or multiple IP addresses and receives result asynchronously. X-Map attempts to overcome scalability and performance issues in large-scale network by implementing system caching with large hash table. X-Map can be implemented into shared machine or dedicated machine that are only built to be X-Map server.

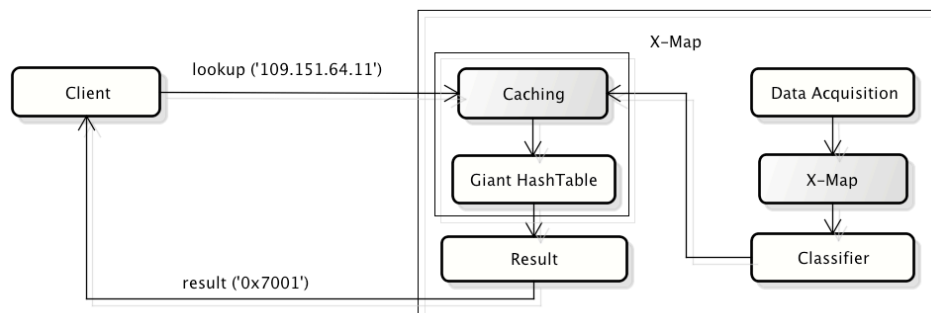


Figure 4.1 X-Map Architecture

### 4.1 Classification

Classification is the task of identifying input objects based on a training dataset containing past observations. The training dataset contains a set of supervised trained classes, each of which consists a pair of input objects with their desired class values. It holds a set of data records describing each attribute and class. The goal is to learn a classification model from the data records to predict future instances. The most commonly used classification algorithm is decision tree learning [74]. As one of the well-known supervised machine learning algorithms, the purpose is to make a classification based on a training dataset. The final result might be in one of the given categories or classes. There are two separate results associated with classification – binary classification and multiclass classification

[75]. For binary classification only two classes are recognised for the output, whereas for multiclass classification the final object result will fall into one of several classes [75].

Decision tree learning has several advantages, such as easy interpretation, fast fitting speed and low memory usage [76]. However, when using a decision tree learning algorithm, it requires a compact and accurate training dataset to be able to increase the accuracy of the classification result. One well-known issue with a decision learning tree is when significant difference occurs between the training dataset and target dataset, it results in a result inaccuracy and might prevent the algorithm to work properly [75]. Therefore, to optimise the supervised learning process, its steps are divided into two different phases, the training and testing phases. The training phase will learn a model using a training dataset, and the testing phase will test the model using unknown test data to assess the model accuracy.

One example application of a classification task, in the security domain, is to predict high-risk web traffic and discriminate them from low-risk web traffic. A decision is needed on whether to reject or allow the traffic to pass into the network. In the next following sections, X-Map will be presented along with the proposed algorithms, which use multiclass classification to make accurate predictions based on visitor past observations. Finally, the result will be discussed in the evaluation chapter.

## 4.2 Decision Tree Learning

X-Map uses decision tree learning as the main machine learning algorithm for classification. It aims to automatically learn to make accurate predictions based on visitor past observations. X-Map attempts to predict a response or class  $Y$  from observation inputs  $X_1, X_2, X_3, \dots, X_n$  by efficiently growing a binary tree. By using this technique, it emphasises on the method that can handle large datasets while reducing the computation complexity.

Table 4.1 below represents a sample predictive model used by X-Map, which maps observations based upon visitor behaviour in order to classify into the final target value. For instance, a high probability score on Heatmap – between 0 and 1 - represents a higher likelihood that the traffic belongs to human. The data



collection listed in the table is the features that will be extracted in the patterns of visitor behaviour. By using a machine-learning technique and decision tree classification model of visitor behaviour, it is possible to predict and classify a visitor from the collected data.

<b>Observation Input</b>	<b>Probability Score</b>
Heatmap	0.4
Banner reaction	0.3
Browser fingerprinting	0.2
Simultaneous request	0.1
Proof of work result	0.3
HTTP request behaviour	0.2
Authenticated session	0.6
Form submission frequency	0.4
Screen resolution	0.3
Header	0.6

Table 4.1. Predictive Detection model

This predictive model gives a valuable insight that can be generated based upon expert knowledge. It is expected to have a high accuracy rate describing current or previous visitor situations in order to determine the preference of classifier outcomes. The model was able to identify ten participants and six simulated bot attacks with an accuracy of 86.67%, which will be discussed in Section 6.6.3. The procedure starts by parsing the IP address of every incoming request. It will then observe visitor behaviour data based on the active session. A threshold calculation is executed to check whether the visitor is under a normal threshold value, the process to obtain all threshold values will be described in Section 6.2. The following procedure is presented to observe a visitor:

```

procedure ObserveVisitor
1: ip ← parseIPAddress()
2: for(i = 0; i < totalObservedData; i++)
3:   data[i] = VisitorData[ip, i];
4:   if(data[i] == null)
5:     continue
6:   else if
7:     for each data of d do
8:       if (data[d] > (threshold =
capturesThreshold))
9:         return bot
10:      else
11:        return normal
12:      end if
13:    end for
14:  end if
15: end for
16: if (!data[0] && !data[1] && !data[2] &&
!data[4])
17:   return unclassified
18: end if

```

Figure 4.2. Observe Visitor

To illustrate the algorithm, the following scenario will attempt to depict a visitor observation based upon visitor behaviour in order to classify into the final target value.

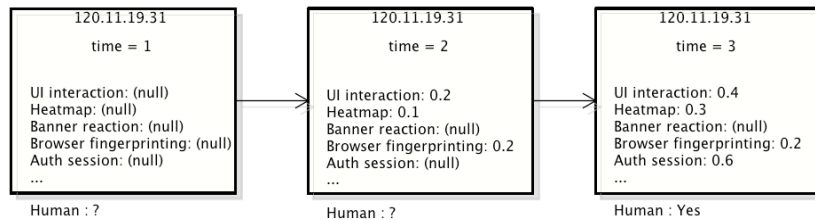


Figure 4.3. Prediction model

Given a sample data as depicted in Figure 4.3 above, the process of visitor observation is based on its IP address over several time steps until certain conditions are satisfied. It is an iterative process. First the visitor has NULL values for all observation probability scores. After the second step, only few observation data can be gathered, which is still not sufficient enough to justify – if the process stops, the classification, which uses the J48 Decision Tree C4.5 algorithm, will render the visitor as unclassified.

According to the observation, the visitor requested a page under the normal threshold limit and capable of generating sufficient user-interface interaction with

a proper heatmap and browser fingerprint value. In addition the visitor signifies a legitimate authenticated session. Therefore, on the third step, X-Map is able to identify the visitor as a valid human based on a training dataset presented in Figure 4.4 below.

IP Address	UI Interaction	Heatmap	Browser Fingerprint	Banner Reaction	Resource Load	Class
98.139.134.97	0.3	0.3	0.2	0.3	0.2	0x7000
216.39.58.78			0.1		0.2	0x7001
87.248.122.142	0.2	0.2	0.2	0.2	0.1	0x7000
98.136.63.35			0.2		0.1	0x7003
94.228.44.113	0.2	0.2	0.2	0.3	0.3	0x7000
120.84.238.58			0.2		0.3	0x7001
95.159.105.2			0.2		0.3	0x7001
62.183.105.233			0.2		0.3	0x7001
216.39.58.17			0.2		0.3	0x7001
87.248.125.48	0.4	0.4	0.2	0.3	0.3	0x7000

Figure 4.4. A possible training dataset

A possible classification tree is given below in Figure 4.5. It represents a top-down induction of the decision tree. First, it assigns the best decision attribute for the next node. It then assigns a probability score as a decision attribute for the node. For each highest score, create a new descendant of the node. It will sort training examples to leaf nodes. Finally, if training examples are perfectly classified, then stop, and otherwise iterate over new leaf nodes.

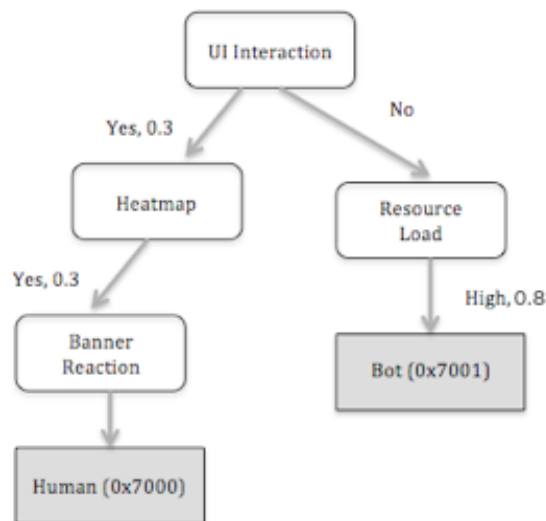


Figure 4.5. A possible classifier

Here are the assumptions used: each internal node tests one attribute of the observation input; each branch from a node extracts one value for observation input; finally, each leaf node makes a prediction where the accuracy rate will depend on the tree size and the size of the training dataset.

Since not all of the Internet bots will perform malicious tasks, therefore, in addition to the previous diagram, if the final classification states that it is a bot (0x7001), then the next decision will be to determine the malicious level. Figure 4.6 illustrates the concept.

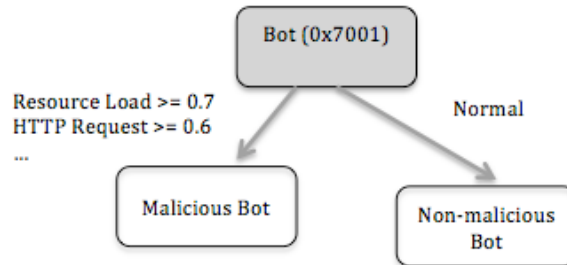


Figure 4.6. Classifying bot

This classification process will also be iterated if the final classification result is half human and half bot (0x7002). The iteration will rely on several metrics such as the browser string header, source of address, proxy address and request load intensity level. Therefore the decision can be made precisely, assuming that those values remain intact.

#### 4.3 Proposed Algorithm X-Map

The proposed algorithm attempts to discover relationship between the observation input attributes and the target attributes. This algorithm is used for predicting the value of a target attribute, e.g. to distinguish machines behind a single IP address and to observe the behaviour of visitors based on their heatmap signatures. The algorithm uses a general tree data structure and a populated training dataset for classification. The unique idea of the algorithm is to combine key-value hashing caching into the decision tree process. It enables faster processing without requiring unnecessary redundant lookup. In particular, it can be adjust into map reduce job to support faster processing for massive dataset.

As the classification tree is a supervised learning structure, the algorithm requires different attributes for each field of the training dataset. These attributes are denoted by the following V sets.

$$\begin{aligned}
 P_1 &= \{V_1, \dots, V_{s1}\} \\
 &\quad \dots \\
 P_n &= \{V_1, \dots, V_{sn}\}
 \end{aligned}$$

Figure 4.7. Attribute set

An observation step consists of an n-dimensional vector  $(V_1, V_2, \dots, V_n)$ , where  $x$  represents features being observed. The training data is a set of  $P = (P_1, P_2, \dots, P_n)$  [79]; which denotes an observation input as described in Section 5.3 (e.g., heatmap and fingerprint). The observation input has a probability value from the uniform distribution on the interval  $(0, 1]$ . For example, if  $P_1$  equals to HTTP Request Behaviour, then a high value would indicate a strong abnormal request.

The following algorithm is proposed for growing the smallest possible tree in order to classify a visitor.

```

procedure Initialize (I, V)
  1: T = API_g_tree_new_full (I, V)
  2: return (T)

procedure LearnFromDataset (I, V)
  1: n ← |Set|
  2: training-dataset [1 ... n]
  3: split given training-dataset
  4: if best splitting metric ≤ threshold then
  5:   for each training-subset of V do
  6:     API_g_tree_grow (V)
  7:     find best value
  8:     R = API_g_tree_traverse (I)
  9:   end for
  10: endif
  11: return R

procedure ClassifyTraffic (I, V)
  1: P ← LearnFromDataset (I, V)
  2: if result found then
  3:   MapToHashtable (I, P)
  4: endif
  5: return

procedure ObserveBehavior (I, V)
  1: for each observation-input do
  2:   (+thread) store visitor data behavior to
  BigData
  3: end for
  4: return

```

```

procedure MapToHashtable (I, P)
  1: connect to memcached server
  2: parse traffic classifier table
  3: store key-value pair (I, P)
  4: disconnect
  5: return

procedure Destroy (T)
  1: if T != NULL then
  2:   API_g_tree_destroy(T)
  3: endif
  4: return

```

Figure 4.8. Traffic classification

The tree complexity is measured by these following parameters: the attributes, observation input and training dataset. It will determine the total number of nodes and leaves, and tree depth. Therefore, it must be explicitly controlled by stopping criteria, and otherwise, the tree growing phase will enter an infinite loop.

The tree data structure and operations, such as creation, traversal, and growing, are handled by third-party libraries, which provide the external functions. The main training dataset is divided into several subsets. Each subset will be checked against a visitor's behaviour, according to the data collection, to determine the best value. It will start to grow the tree structure once it has been found. The process iterates until all observations are done. Once the result has been obtained, it will be stored into the hashtable. Figure 4.9 below illustrates the flowchart diagram.

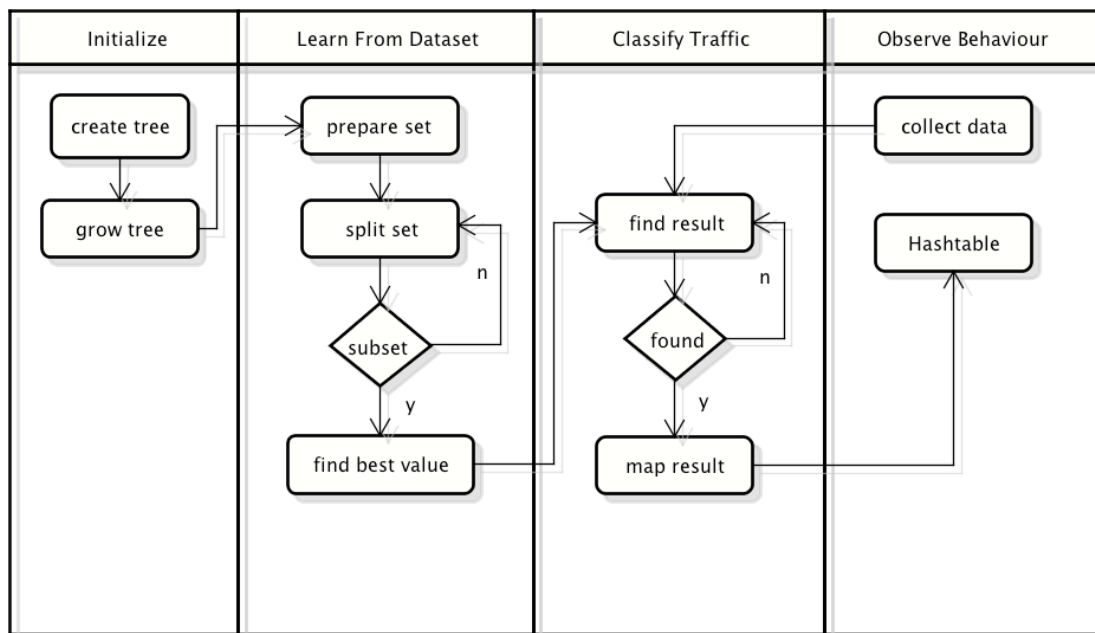


Figure 4.9. System Flowchart

#### 4.4 Proposed Algorithm for Machine Identification

Methods that use CAPTCHA such as [37] will not be able to provide puzzles to their visitors when they are using a Command Line Interface (CLI) browser. X-Map overcomes this by using machine identification that uses a browser fingerprinting method. The approach is based on research study that at least one in 286,777 browsers shares the same fingerprint [20]. The demonstration estimates that under current active sessions only one in 3,987,090 browsers will have the same fingerprint [23]. This method is effective for identifying CLI, mobile and TV browsers based on their screen resolutions via JavaScript AJAX posts. It provides a method to distinguish machines behind a single IP address.

The following algorithm is proposed for identifying machines. It stores data in a key-value pair of an IP address (I) and a browser fingerprint signature (B).

```
procedure InitHashTable ()
1: H = g_hash_table_new ()
2: for each table of H do
3:   H ← g_init_hash()
4:   H ← g_str_hash()
5: end for
6: return (H)

procedure ParseIPDataset (I, B)
1: parse-dataset = |IPSet|
2: B ← fingerprint_browser(I)
3: for each parse-dataset of IP Set do
4:   g_map_element (H, IP Set)
5:   H ← g_hash_table_insert (I)
6:   H ← g_hash_table_insert (B)
7: end for
8: return

procedure LookupFP (I)
1: if ip-dataset != NULL then
2:   H ← g_set_active_hash
3:   R ← g_hashtable_lookup(I)
4:   return R
5: endif
```

Figure 4.10. Machine identification

Initially the hashtable is created, it is produced by using third-party libraries. Each visitor will be tracked by their unique browser fingerprint, which will be described in Section 5.3.9 and Appendix A –Browser Fingerprint. The visitor IP address will

be parsed and stored in the hashtable. Finally, the system is able to identify all visitors fingerprint based on their IP address.

Figure 4.11 below illustrates the flowchart diagram. The key-value pair of IP addresses is mapped into one hashtable, consisting of an IP address as the key and a fingerprint signature as the value, which is created when the system runs for the first time.

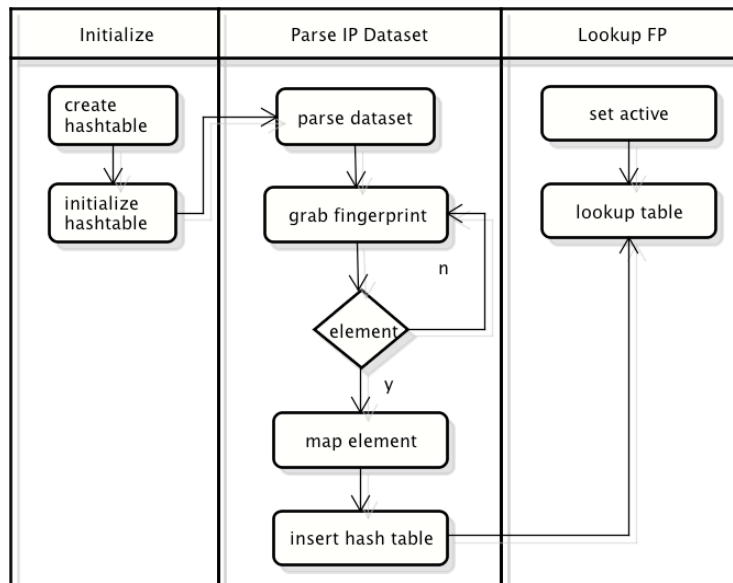


Figure 4.11 Machine identification flowchart

#### 4.5 Proposed Algorithm for Heatmap Tracker

Methods that use CAPTCHA such as the one in [37] requires active visitor response and waste times. On the other hand, a website heatmap is very effective at uncovering usability issues throughout the web to improve page design. This solution combines security protection with the opportunity of usability study. It provides a transparent method to detect the presence of human interaction.

Heatmap representation uses three-dimensional data. The two dimensions represent  $x$  and  $y$  Cartesian coordinates, and the third dimension represents the intensity of a data point, which is usually presented as a minimum or maximal integer value. Thus, the heatmap model can be presented in the following matrix.



$$\begin{bmatrix} 0 & 0 & 0 & 0 & 21 \\ 0 & 0 & 2 & 14 & 16 \\ 0 & 0 & 2 & 12 & 24 \\ 0 & 0 & 15 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 4.12. Matrix model of heatmap

In the above example, the values of 2, 2, 15, 14, 12, 21, 16 and 24 represent the sums of intensity levels for the mouse click event activities in the page region. Every mouse activity is recorded for each page. The zero value indicates that there is no activity within the region of the page. Each page is divided into four regions, which will be used to record and observe visitor behaviour based on a matrix model.

The following algorithm is proposed for identifying human presence based on their heatmap dataset.

```

procedure ObserveHeatMap (I)
1: parse-dataset = |IPSet|
2: D ← map-data
3: D ← matrix-conversion(D)
4: if (D.value > 50)
5:   inform enough activities
6:   threshold = getThreshold
7: else
8:   insufficient activities
9:   return
10: endif
11: for(i = 0; i < totalMatrixRow; i++)
12:   for(j = 0; j < totalMatrixCol; j++)
13:     if D.sum ≤ threshold then
14:       inform insufficiency of dataset
15:       return
16:     endif
17:   end for
18: end for
19: for each matrix-conversion of IP do
20:   parse data map
21:   CSV conversion
22:   give each element a rating map
23:   convert to (0,1] uniform distribution
   interval
24: end for
25: J48.train(D)
26: P ← model.predictAll(D)
27: return P

```

Figure 4.13. Heatmap tracker

In order to check the visitor activities, all regions inside the page are mapped into a matrix model. The region is a <div> tag, which defines a division in an HTML document with a unique identifier. The values are extracted by mapping all of these elements together. The conversion process is used to calculate mouse activities into a matrix model. If there are sufficient activities, it is assumed that the visitor has a legitimate browsing behaviour. Figure 4.14 below illustrates the flowchart diagram.

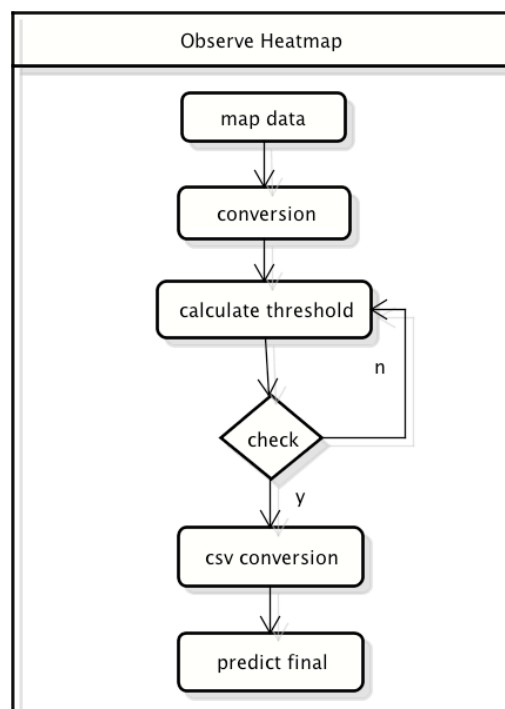


Figure 4.14 Heatmap tracker flowchart

#### 4.6 Proposed Technique for Banner Reaction

X-Map exploits this feature to combine security protection with the opportunity of company advertisements. It provides advertising support, which can lead revenue behind them.

The proposed technique, which was developed during this research, uses active and passive banner reaction observations. As opposed to Kill-Bots [37], under periods of heavy load, which is detected by VATRIX, X-Map redirects visitors to a different secure server and presents them with a page full of advertisements. X-Map then analyses how the visitors close banners, and observes their interaction heatmaps. A successful reaction will be granted with a valid website access. This

condition is only applied to visitors who have not been classified by X-Map in order to grant them services. Therefore, redirecting all visitors is not necessary.

Under passive observation, at a random time interval, X-Map presents visitors with a small portion of advertisements. Successful reaction will be regarded as a valid visitor. This technique is used along with other observation methods in order to accurately classify human.

#### **4.7 Summary**

X-Map opens new opportunities for small and medium enterprises to combat application layer HTTP DDoS attacks and spam bots. X-Map combines security protection with the opportunity of usability study. X-Map also combines security protection with the opportunity of company advertisements, which can lead beneficial revenues behind them. Combined with other methods, X-Map provides a transparent method to detect the presence of human interaction. To summarize, in this chapter the problems with the existing solutions have been discussed. The method of X-Map has been introduced. The proposed algorithms of X-Map have been covered thoroughly. Finally, the next chapter will cover the experiment settings of our work presented in this and previous chapters.

## Chapter 5 Experiment Settings

### 5.1 Introduction

The use of statistic-based techniques to mitigate bot attacks remains a debatable subject. Therefore, the increasing rate of malicious bot attacks has motivated this research to provide fair allocation across all the visitors in the presence of attacks.

The novel contributions, as outlined in Chapter 3 and Chapter 4, demonstrated how legitimate and bots traffic can be discriminated. Chapter 3 focused on the visual analytics system, which provides a monitoring mechanism in the presence of attacks. The system provides a dashboard view to represent legitimate and bot traffic by adopting a Quadtree data structure and Voronoi diagrams. Chapter 4 described the classification technique in order to support the visualisation process. The classification technique, which uses the J48 Decision Tree C4.5 algorithm, requires a training dataset – with current total size of 2010 instances - so that the features can be extracted. In this instance, a sample of raw data contains a heatmap value, banner respond, intensity requests and request behaviour. This process resulted in the creation of the visualisation diagram.

In realising the aims and objective of this research, as presented in Chapter 1, this chapter discusses the experiment settings, in which our research was conducted by utilising experiment and simulation approaches. This research study was taken in a fully controlled and structured environment. This is to enable the causal relationships of legitimate participants (n=10) and simulated bot attacks to be identified and analysed. Due to the limitation of time and no compensation available to participants, only 10 participants with the age range from 28 to 35 were employed to represent legitimate visitors.

The remaining sections of this chapter describe the experiment method with the test environment. It also provides the description about the participants. In addition, the tools used for simulating the attacks are presented. Finally, the data to be collected are also explained.

### 5.2 Experiment Method

The experiment was conducted by capturing website traffic, identifying browser fingerprints, simulating bot attacks and analysing mouse dynamics such as

movements and events of the participants. Data were captured as the participants performed a list of tasks, such as responding to the banner. The data collection is transparent to the participants and only requires JavaScript to be activated on the client side. The 10 participants are familiar with the Internet. To analyse the data, Weka 3.6.10 was used to perform classification based on a training dataset.

The experiment process can be outlined as follows:

1. Developing the initial website capable of tracking visitor data.
2. Inviting participants to visit the website.
3. Simulating bot attacks.
4. Collecting and evaluating the data.
5. Building a machine learning model.
6. Testing the accuracy of the machine learning model.
7. Performance evaluation.
8. Formulating visual analytics design.

All the legitimate participant behaviour data were captured. The characteristics of the simulated bot attacks were analysed. These data were then collected and evaluated using J48 classification for a decision tree. The initial training dataset was generated by comparing the activity patterns of the participant legitimate behaviour against simulated bot attacks. Finally, the proposed visual analytics design was rendered.

### **5.2.1 Test Environment**

The attack simulation and experiment were performed under a Virtual Private Server (VPS) environment, running the Linux 64 Bit operating system with its own static Internet Protocol (IP) address. The chosen Linux distribution was CentOS 6.5 equipped with 2 GB of Random Access Memory (RAM) and 50 GB of hard-disk space. A new instance of Apache 2.2.15 was deployed along with PHP 5.3. The connection is capable of handling an unlimited bandwidth up to 100 Mbit traffic without forced throttling.

To demonstrate, a new dummy website imitating an online shop was implemented by using an open-source e-commerce solution, PrestaShop 1.6.0 [48], and Leo Sport Shoes Theme 1.0, a free responsive theme [49].

### 5.2.2 Participants

The experiment was launched with the 10 participants to observe their browsing behaviour, as listed in Table 5.1. Participants were invited to join by posting a message on Online Social Network (OSN) asking for their contribution to visit the dummy website. All of the participants are familiar with the Internet. Each participant spent approximately 10 – 15 minutes with the experiment. The data were collected transparently from four different pages over multiple sessions. Each session generated data files with each ranging in size from 700 KB to 900 KB. For all participants a unique browser fingerprint was generated as the page loads. This is to ensure their unique identity when combined with their IP addresses.

<b>Characteristics</b>	<b>Amount</b>
<b>Sample size</b>	<b>10</b>
Male	7
Female	3
Age (28 – 35)	10
Education (BSc – MSc)	10
Safari 7.0.2	4
Firefox 28	3
Chrome 33	3
Screen resolution 1280x800	10
IT background	6
Non-IT background	4

Table 5.1. Participant characteristics

### 5.2.3 Attack Simulation

To study the behaviour of malicious bot attacks, this research project used a list of low-bandwidth attacking tools to represent the key characteristics of such attacks. These tools pose serious threats especially if some of them are integrated into a single command running on large botnets [13]. This attack simulation model is meant to show the eventual real effects of bot attacks by using alternative conditions. Table 5.2 summarizes the types of attacks used in this simulation.

<b>Attack Tool</b>	<b>Detail</b>
slowloris	Partial HTTP requests attack
slowhttptest	Slow HTTP attack
r-u-d-y	HTTP POST/GET flood
THC-SSL-DOS	SSL renegotiation flood

ApacheKiller	Multiple byte ranges attack
PhantomJS	Interface workload generator

Table 5.2. Attack simulation tools

*Slowloris* exploits HTTP vulnerabilities by sending partial HTTP requests. It has the ability to perform an attack with minimal bandwidth. *Slowloris* aims to abuse the server by keeping as many active connections as possible [50].

*Slowhttptest* provides low bandwidth application layer DoS attacks. It can perform a slow attack that exploits the flaw of the TCP persist timer. It aims to send legitimate HTTP requests and abuse the server by reading the response slowly [51].

*R-U-Dead-Yet* is a HTTP DoS attack. It exploits the long form field submissions vulnerability. It can automatically detect forms within given URL [52].

*THC-SSL-DOS* exploits the vulnerability of SSL renegotiation. It aims to trigger thousands of renegotiations to overwhelm servers with multiple requests. This tool can be launched via single TCP connection [53].

*ApacheKiller* exploits HTTP range header vulnerability by sending multiple overlapping byte ranges. It works by stacking an HTTP header to the server with multiple ranges request [54].

*PhantomJS* is a tool that can run a script to mimic human interaction on a given URL. It is suitable for generating user interaction workloads to test and monitor specific website [55].

### 5.3 Data Collection

Data collection was written using C, PHP and JQuery. It transparently collects visitor data, as listed in Table 5.3. It aims to extract certain signature characteristics in the patterns of visitor behaviour. By using a machine-learning technique and decision tree classification model of visitor behaviour, it is possible to predict and classify a visitor from the collected data, based on their weight value. There are several steps to calculate the weights: 1) calculate initial score for each observation, 2) sort the weight data, 3) Assign a weight value to each observation within each corresponding group of visitor behaviour. Appendix A provides a detailed account of the data collection script.

Observation Input	Implementation
IP and TCP header	libpcap, C
HTTP header and request behaviour	C, PHP
Proof of work	PHP, JQuery
Form submission frequency	PHP
Authenticated session	PHP
Website heatmap	PHP, JQuery
Banner reaction	PHP, JQuery
Screen resolution	JQuery
Browser fingerprint	JQuery

Table 5.3. Data collection

### 5.3.1 IP and TCP Header

Data such as IP addresses and port numbers were extracted from IP and TCP headers. These data were collected to provide information of a visitor's originating address. The implementation was written using C and libpcap. These data will not be mapped since they are only used for visualisation and request checking purposes. Therefore, the constant weight value will remain flat at 0.5. This value is to ensure that the decision making process will not be made based on this data. Therefore, this record will be ignored by the system.

### 5.3.2 HTTP Header and Request Behaviour

Data such as the HTTP method and content length used were extracted from HTTP headers in order to provide details on demand visualisation. A counter was implemented to calculate the number of simultaneous connections made. These data were collected as a supplementary mechanism to discriminate between legitimate visitors against bot attacks. The implementation was written using C and PHP. If the value reaches a certain threshold limit, the weight value will be greater than 0.6, a customised constant where a high value would indicate a stronger abnormal request. The assumption is based on the fact that bots usually generate abnormal HTTP requests.

### 5.3.3 Proof-of-work

Each visitor is provided with a task to compute a MD5 hash of a given randomised string. The results of proof-of-work data were collected as a supplementary mechanism to ensure unique identification of a visitor. The implementation was



written using PHP and JQuery. If the timing indicates a low response and a wrong answer was given, the weight value will be less than 0.2 where a low value would indicate a failed response.

#### **5.3.4 Form Submission Frequency**

A counter was implemented to calculate HTTP POST requests. These data were collected as a supplementary mechanism to discriminate between legitimate visitors against bot attacks. The implementation was written using PHP. If the value reaches a certain threshold limit, the weight value will be greater than 0.6, where a high value indicates a stronger submission frequency.

#### **3.3.5 Authenticated Session**

Each visitor session was checked from the authenticated session variable. These data were collected as a supplementary mechanism to discriminate between legitimate visitors against bot attacks. The implementation was written using PHP. If a visitor is successfully authenticated, the weight value will be greater than 0.6. A high value indicates that the visitor is legitimate.

#### **5.3.6 Website Heatmap**

A website heatmap is a graphical image that represents visitor interactions with each section of a certain page. These data were collected as a supplementary mechanism to discriminate between legitimate visitors against bot attacks. The implementation was written using PHP and JQuery. If a visitor is generating a sufficient website heatmap data, the weight value will be greater than 0.6. High value would indicate that the visitor is producing a sufficient quantity of interaction.

#### **5.3.7 Banner Reaction**

The banner reaction data were captured on the client side, and the analysis was conducted on the server side. These data were collected as a supplementary mechanism to discriminate between legitimate visitors against bot attacks. The implementation was written using PHP and JQuery. If a visitor responds to the banner advertisement correctly, the weight value will be greater than 0.7. High value would indicate that the visitor has a legitimate respond.

### 5.3.8 Screen Resolution

Screen resolution or display resolution is “*the number of distinct pixels in each dimension that can be displayed*” [77]. This data along with browser window resolution was collected to ensure the integrity of a website heatmap. The implementation was written using JQuery. If the value indicates normal screen resolution, the weight value will be greater than 0.6. High value would indicate legitimate screen resolution setting.

### 5.3.9 Browser Fingerprint

Browser fingerprinting is a technique to identify a web browser. These data were collected as a supplementary mechanism to ensure unique identification of a visitor. The implementation was written using the JQuery browser fingerprint plugin. If the value is not NULL, the weight value will be greater than 0.6. This value would indicate that the visitor is not producing a sufficient browser fingerprint.

## 5.4 Data Pre-processing

Data pre-processing was implemented using PHP 5. It reads data generated by data collection and performs Attribute-Relation File Format (ARFF) conversion. ARFF “*is an ASCII text file that describes a list of instances sharing a set of attributes*” [56]. “*ARFF files have two distinct sections. The first section is the Header information, which is followed by the Data information*” [56]. Each of the observation inputs was segmented and mapped into a probability value, as previously described. The variable represents a weight value from the uniform distribution on the interval of 0 and 1.

An example of the ARFF Header looks like the following:

```
@RELATION name
@ATTRIBUTE req    NUMERIC
@ATTRIBUTE pow    NUMERIC
@ATTRIBUTE freq   NUMERIC
@ATTRIBUTE auth   NUMERIC
@ATTRIBUTE class  {A,B}
```

An example of the ARFF Data records look like the following:

```
0.4,0.6,0.4,0.4,A
0.6,0.4,0.2,0.2,B
0.2,0.3,0.3,0.2,B
0.5,0.6,0.1,0.4,A
```

0.8,0.5,0.2,0.1,B  
0.7,0.1,0.6,0.5,B  
0.9,0.7,0.3,0.2,B  
0.3,0.2,0.2,0.3,B  
0.1,0.9,0.1,0.4,B

The subsequent columns for each row represents data collection as described in Section 5.3, and finally the last column would represent the class, which the traffic belongs to. The subsequent rows are the full training dataset records for the machine learning model.

## 5.5 Data Analysis

Data analysis was performed in Weka 3.6.10 by utilising decision tree classification - decision tree C4.5 algorithm using J48 classifier. The result will be discussed in Chapter 6.

The performance of the classification algorithm, Decision Tree J48, was evaluated using:

- **Training set**, *“The classifier is evaluated on how well it predicts the class of the instances it was trained on”* [57].
- **Cross-validation**, *“The classifier is evaluated by cross-validation, using the number of folds that are entered in the Folds text field”* [57].
- **Percentage split**, *“The classifier is evaluated on how well it predicts a certain percentage of the data which is held out for testing. The amount of data held out depends on the value entered in the % field”* [57].

## 5.6 Training Dataset

The banner puzzle training dataset was generated using Safari 7.0.2 by simulating correct and incorrect scenarios. The main classification training dataset was generated by comparing the activity patterns of the participants’ legitimate behaviour against simulated bot attacks. Appendix B provides all the dataset. This predictive model gives valuable insight that can be generated based upon expert knowledge. It is supposed to have a high accuracy rate describing current or previous visitor situations in order to determine the preference of classifier outcomes.

### 5.6.1 Dependent and Independent Variable

The result of the classification will be a dependent variable. This value relies upon the observation inputs, as described in Section 5.3. The observation inputs, which influence the value of the dependent variable, are independent variables.

### 5.6.2 Data Cleaning

The data cleaning process includes: (1) Removing unnecessary records that are not related to the experiment, and (2) Checking inaccurate records from a test dataset. This is to ensure the integrity and consistency of the test dataset format with the main training dataset.

## 5.7 Visual Analytics Requirements

Monitoring network traffic involves huge amounts of data to be collected by packet capture throughout the network. The packet capture is a mechanism to collect raw traffic data from a network. Visualisation was produced using these raw traffic data. The classification result was obtained using J48 Decision Tree. The prototype of the visualisation was rendered using D3.js, “*a JavaScript library for manipulating documents based on data*” [58].

In order to satisfy the requirements of visual analytics, the following conditions must be met [11, 12]:

1. **Interactive visualisation**, the design should support interactive visualisation.
2. **Large datasets**, the design should support in understanding, reasoning and decision making from very large datasets.
3. **Knowledge discovery**, the design should be able to assist analysts in order to gain new knowledge.
4. **Information overload**, the design should be able to turn information overload into an opportunity.

The next chapter will provide a detailed account of the visual analytics method used.

## 5.8 Limitations

Key issues in this experiment include generating flash crowds phenomenon. The simulation in this research project is only designed to work effectively for client-

server models. HTTP itself is a protocol that uses a client-server model. This research project focuses on Internet Protocol version 4 (IPv4). Currently, there is only one banner advertisement puzzle and largely depends on the screen resolution setting. In practice, more banner puzzles could be generated on various screen resolutions along with the training dataset. For the visual analytics design, it is assumed that the analyst has a foundation understanding regarding network traffic and web servers, especially with Apache. Currently, the experiment conducted in this research was using only one instance of Apache. The HTTP methods that were tested include GET and POST methods. Furthermore, the attack simulation tools' capabilities described in Section 5.2.3 are quite limited. The simulation study lacks the tools for testing the system with adaptive persistent attacks.

## **5.9 Summary**

This chapter has described the research methodology used in this study. The experiment approach has been covered along with the test environment, participant characteristics and attack simulation. A comprehensive explanation of the data collection, data pre-processing, training dataset and data analysis was given. This chapter has explained the processes necessary in preparing the data for analysis using Weka. In addition, the research limitations have been discussed. Finally, the next chapter will discuss the evaluation used in this research.

## Chapter 6 Evaluation

### 6.1 Introduction

In the previous chapter, the experiment settings have been discussed, which detailed the features needed to be extracted in order to observe visitor behaviour. The experiment was conducted by capturing raw data from website traffic, identifying browser fingerprints, simulating bot attacks and analysing mouse dynamics such as movements and events of the participants. Also in the previous Chapter 4, it discussed the classification technique to discriminate between legitimate against bot traffic in order to completely render the visualisation. The result, which will be discussed in Section 6.6, uses the J48 Decision Tree C4.5 algorithm that requires a training dataset where the processing format was already defined in Section 5.4.

This chapter describes a detailed account of the evaluation, in which the research was conducted by utilising the experiment and simulation approaches described earlier. The main purpose of the evaluation was to validate the system and to demonstrate how the system is able to discriminate between legitimate visitors (10 participants) against simulated bot attacks. All the legitimate participants' behaviour data were captured. The characteristics of the simulated bot attacks were analysed. These data were then collected and evaluated using J48 classification for a decision tree. This is to enable the causal relationships between legitimate participants and simulated bot attacks to be identified and analysed. Finally, our visual analytics evaluation is presented along with the discussion. This chapter aims to demonstrate the results that have been obtained.

To support the evaluation, the analysis results were obtained by using these following tools:

1. **Weka 3.6.10**, "*Weka is a collection of machine learning algorithms for data mining tasks*" [66].
2. **YSlow 3.1.8**, "*YSlow analyzes web pages and suggests ways to improve their performance based on a set of rules for high performance web pages*" [67].

## 6.2 HTTP Requests

The experiments below were performed by loading landing, product detail, category and login pages. The objective is to find the total number of HTTP requests made for each page. The pages were hosted under a dummy website. The results and threshold values used were generated by the YSlow 3.1.8 plugin for Firefox 28 in order to signify abnormal behaviour. The results were obtained manually by checking YSlow console outputs.

The following sections will show the results of HTTP requests from four different pages of a dummy website. A probability metric then was built for the training dataset according to these threshold values. Therefore, these threshold values could be used as a preliminary comparison between participants and simulated bot attacks. However, there can be several occasions when legitimate visitors might reach beyond these threshold values, such as reloading a page many times or opening multiple tabs in the browser within the same URL. Another possible reason is when multiple visitors are behind a proxy server.

### 6.2.1 Landing Page

The diagram in Figure 6.1 illustrates the total number of HTTP requests made to load the landing page. The screenshot for the landing page can be found in Appendix C – Landing Page. Based on this result, a specific threshold value,  $90 \leq w \leq 99$ , was adjusted to indicate a normal legitimate landing page request. However this value will not be used as the main classification criterion, as there are other metrics (listed in Section 5.3) to be used for final decision.

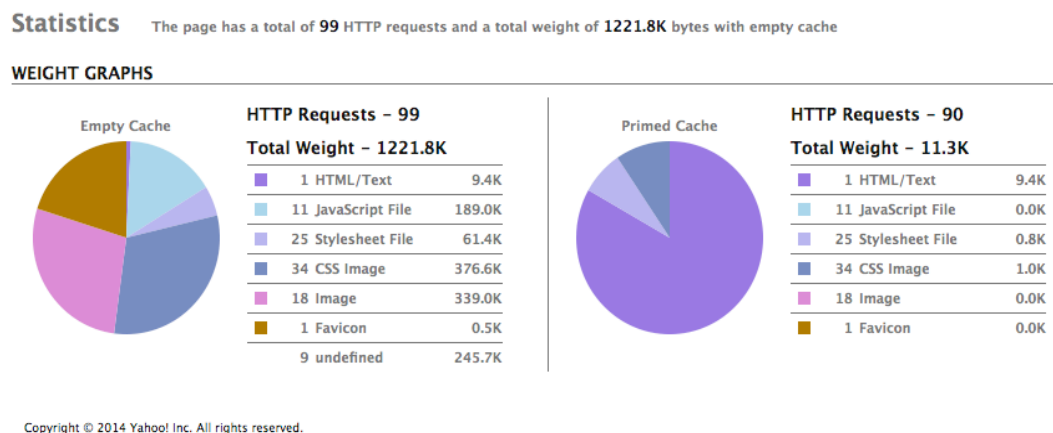


Figure 6.1. Landing page

### 6.2.2 Product Detail Page

The diagram in Figure 6.2 illustrates the total number of HTTP requests made to load the product detail page. The screenshot for the product detail page can be found in Appendix C – Product Detail Page. Based on this result, a specific threshold value,  $87 \leq x \leq 96$ , was adjusted to indicate a normal legitimate product detail page request. However this value will not be used as the main classification criterion, as there are other metrics (listed in Section 5.3) to be used for final decision.

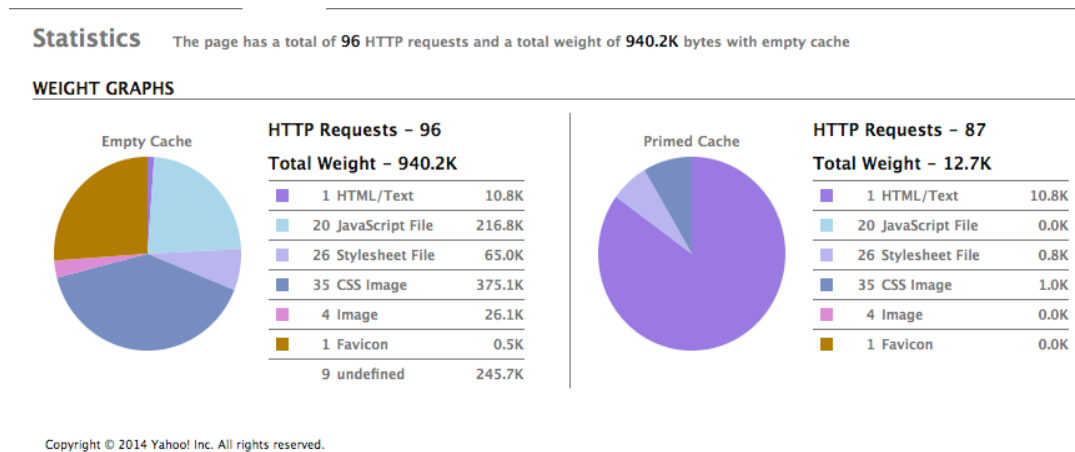


Figure 6.2. Product detail page

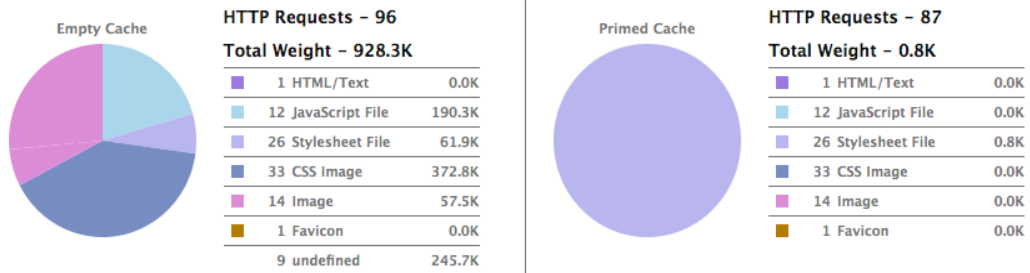
### 6.2.3 Category Page

The diagram in Figure 6.3 illustrates the total number of HTTP requests made to load the category page. The screenshot for the category page can be found in Appendix C – Category Page. Based on this result, a specific threshold value,  $87 \leq y \leq 96$ , was adjusted to indicate a normal legitimate category page request. However this value will not be used as the main classification criterion, as there are other metrics (listed in Section 5.3) to be used for final decision.



**Statistics** The page has a total of 96 HTTP requests and a total weight of 928.3K bytes with empty cache

**WEIGHT GRAPHS**



Copyright © 2014 Yahoo! Inc. All rights reserved.

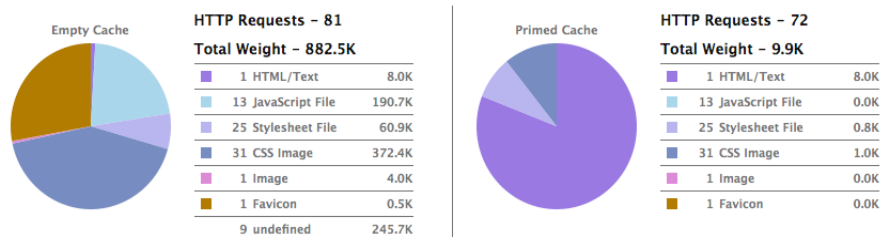
Figure 6.3. Category page

### 6.2.3 Login Page

The diagram in Figure 6.4 illustrates the total number of HTTP requests made to load the login page. The screenshot for the login page can be found in Appendix C – Login Page. Based on this result, a specific threshold value,  $72 \leq z \leq 81$ , was adjusted to indicate a normal legitimate login page request. However this value will not be used as the main classification criterion, as there are other metrics (listed in Section 5.3) to be used for final decision.

**Statistics** The page has a total of 81 HTTP requests and a total weight of 882.5K bytes with empty cache

**WEIGHT GRAPHS**



Copyright © 2014 Yahoo! Inc. All rights reserved.

Figure 6.4. Login page

### 6.3 Overall Performance Score

The overall performance score was obtained by running a YSlow test on each page, both with and without mouse tracking installed. YSlow is a browser add-on plugin that calculates the total HTTP requests made by the client. The score was computed depending on the performance load of each page, which results into a final grade, as illustrated in Figure 6.5. All pages were loaded individually while

running the YSlow add-on to obtain the final score. A higher score indicates a good grade of page performance. The results indicated a degradation of performance when the mouse tracking feature is enabled. However, their differences are very low.

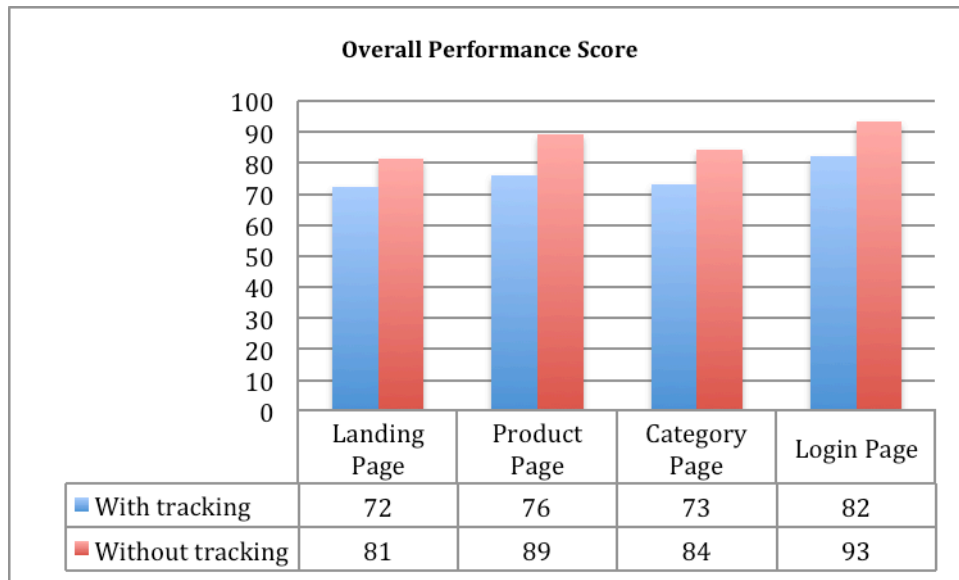


Figure 6.5. Overall performance score

#### 6.4 AJAX Request Timing

Figure 6.6 below illustrates the value of AJAX request timing made by participants (n=10). These values were obtained by using data a collection script defined in Appendix A – AJAX Request.

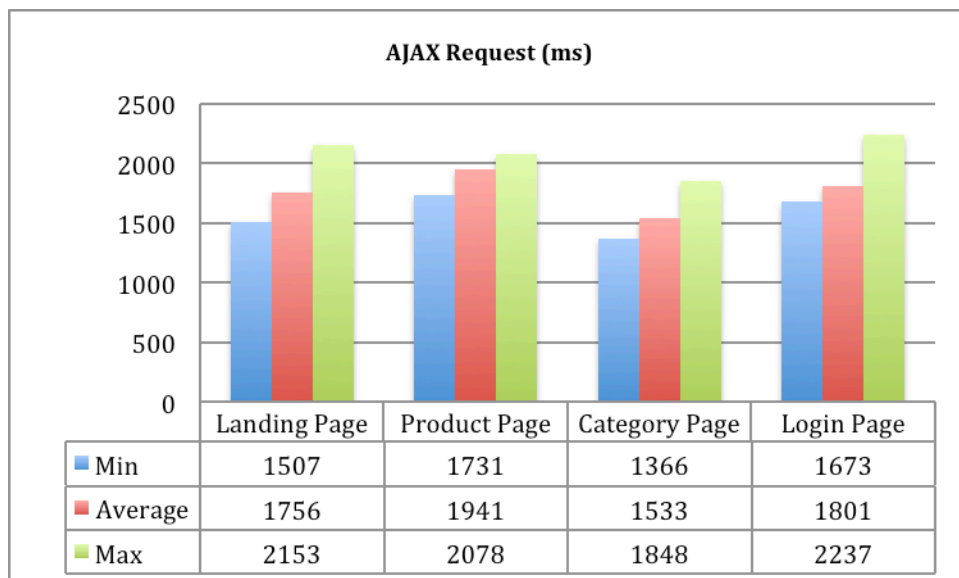


Figure 6.6. AJAX request timing

## 6.5 Simultaneous Requests

In order to gain a certainty level of maximum simultaneous requests per second, the server should appropriately handle *Apache Benchmark* (see Appendix D), which was launched ten times to obtain an average value. The average value represents a high number of successful responses received and minimum failed requests.

Test #	Test time (sec)	Requests /sec	Time per request (ms)	Transfer rate (Kbytes/sec)
1	61	16.39	3904.428	3.55
2	22	44.73	1430.965	9.71
3	85	11.69	5473.194	2.54
4	139	7.15	8949.181	1.56
5	22	44.32	1443.975	9.61
6	34	29.13	2196.904	6.32
7	45	21.86	2927.084	4.74
8	27	36.51	1752.786	7.92
9	26	38.34	1669.335	8.31
10	151	6.61	9692.308	1.43
<b>AVERAGE</b>	<b>61.2</b>	<b>25.673</b>	<b>3944.016</b>	<b>5.569</b>

Table 6.1. Apache benchmark results

As presented in Table 6.1, the average value of requests per second was 25.673. This value indicates significant positive correlation and still below the allowed threshold value of maximum simultaneous requests per second made by participants, which is 99 as described in Section 6.2.1.

Participant #	Requests /sec
1	8
2	5
3	7
4	6
5	4
6	6
7	5
8	8
9	8
10	6

Table 6.2. Participants requests /sec

As presented in Table 6.2, it was found that, on average, legitimate participants requested no more than 8 simultaneous connections per second. Therefore, a preliminary training dataset was built according to these results.

### 6.5.1 Attack Simulation

In order to disrupt the web server, it was found that the attack simulation tools were generating requests between 400 – 1000 of simultaneous connections from a single connection. The following are the results for attack simulation as specified in Appendix D. These values were obtained by using the *netstat* command in Appendix A – Simultaneous Request.

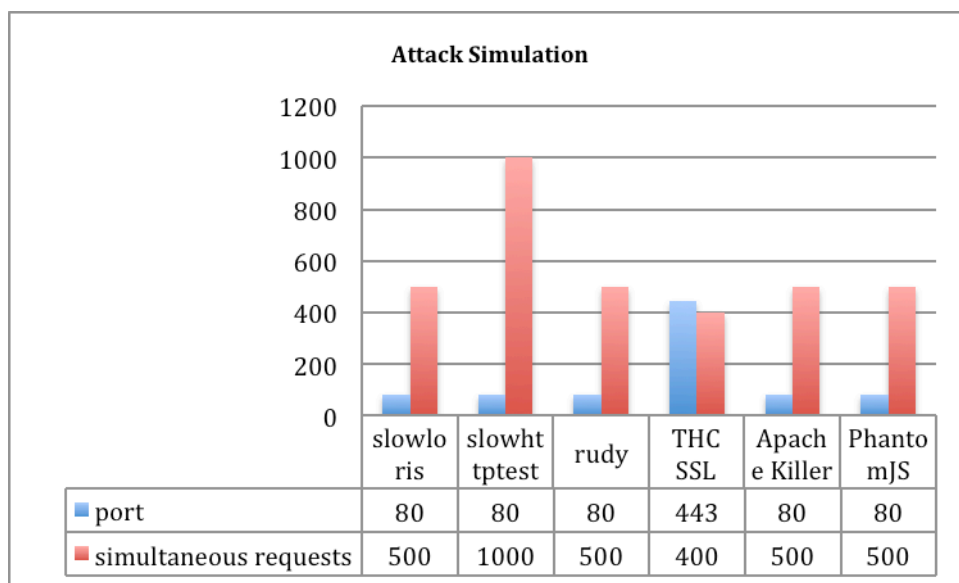


Figure 6.7. Attack simulation

*R-u-d-y* was tested on the search page, while the other tools were tested on the landing page. *THC-SSL-DOS* was tested on the secure connection on port 443. All of these attacking tool sessions were not authenticated, because it is assumed that the adversary does not have a valid credential. From all these attacking tools, only *PhantomJS* was capable of generating user interface interaction, proof-of-work, screen resolution and browser fingerprint data. However, *PhantomJS* failed to respond correctly on the banner advertisement. By using *slowhttptest* alone, it can be seen that the server stops responding after 6 seconds (see Appendix D – Slowhttptest Output). Therefore, to automate the decision making process, the visitor page requests will be checked to see whether it is beyond the normal threshold limit. Then, if a visitor has a high rate of concurrent connections and is

not capable of generating proof-of-work, it will be marked for the next decision process. Additional checking of user interaction, screen resolution, browser fingerprint data and banner reaction will also be performed. Finally if a visitor signifies a workload of form submission frequency, based on this observation by using J48 classification for a decision tree, the traffic will be classified as a bot.

## 6.6 Classifier Performance

This section presents the results of classifier performance using several supervised machine learning algorithms. These results were produced using Weka on a training dataset as described in Appendix B – Traffic Training Dataset. Decision tree classification (J48) obtained an accuracy of 85.2381%, as indicated in Figure 6.8.

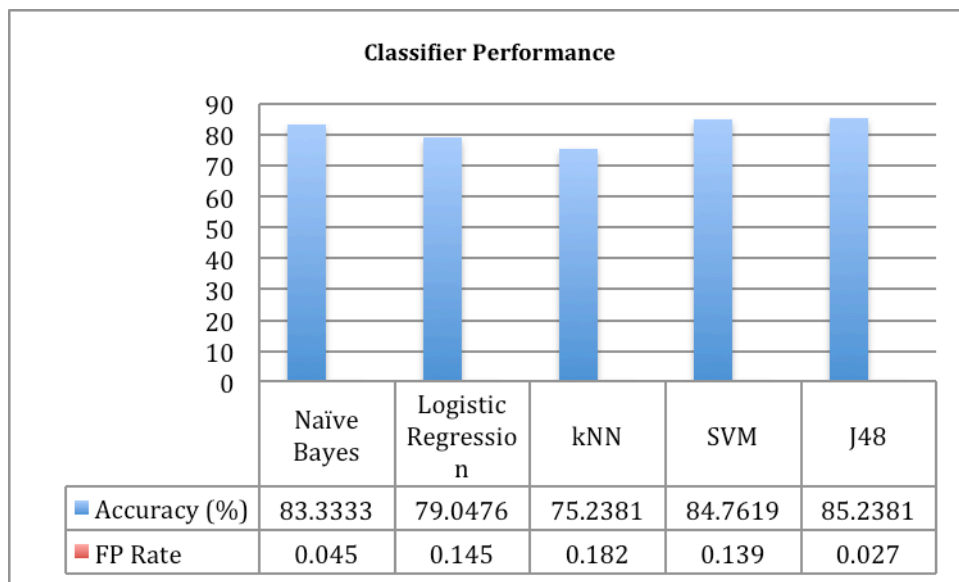


Figure 6.8. Classifier performance

According to these results, Decision Tree (J48) was finally chosen as the main machine learning algorithm due to its high accuracy value and low false-positive rate. The performance of the classification algorithm, Decision Tree J48, was evaluated using cross-validation with 10 folds. The main reason is that this setting has a lower variance, which is very important when the amount of data available is limited. In this case, the decision tree has outperformed the other machine learning algorithms. By gradually growing the training dataset size, the quality of accuracy increases, as illustrated in Figure 6.9.

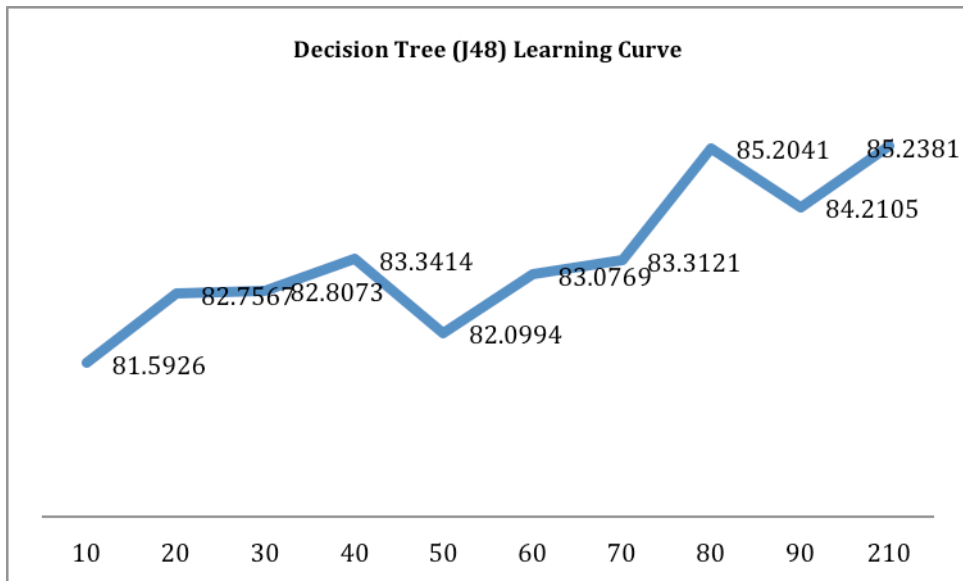


Figure 6.9. Decision tree learning curve

### 6.6.1 Testing on a Training Dataset

This section presents the results on a training dataset for decision tree classification using cross-validation with 10 folds. The testing correctly classified 179 data from a total of 210 instances with an accuracy of 85.2381%. The incorrectly classified instances were 14.7619 % (31 instances). These instances were incorrectly classified because of banner reaction inaccuracy and mismatch parsing on HTTP request load. This is a false positive error - a false detection result that indicates a given legitimate user was identified as malicious bot.

=== Run information ===

```
Scheme:weka.classifiers.trees.J48 -C 0.25 -M 2
Relation: discriminate
Instances: 210
Attributes: 9
```

```
request
pow
freq
auth
heatmap
banner
screen
fp
class
```

Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree

-----

```
request <= 0.1: 0x7003 (10.0)
request > 0.1
| banner <= 0.19: 0x7001 (128.0/28.0)
```

```
| banner > 0.19: 0x7000 (72.0)
```

```
Number of Leaves :      3
```

```
Size of the tree :      5
```

```
Time taken to build model: 0.05 seconds
```

```
=== Stratified cross-validation ===
```

```
=== Summary ===
```

```
Correctly Classified Instances      179           85.2381 %
Incorrectly Classified Instances     31           14.7619 %
Kappa statistic                     0.7288
Mean absolute error                  0.1438
Root mean squared error              0.2761
Relative absolute error              39.4958 %
Root relative squared error          64.8327 %
Total Number of Instances           210
```

```
=== Detailed Accuracy By Class ===
```

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area
0x7000	0.72	0.027	0.96	0.72	0.823	0.842
0x7001	0.97	0.255	0.776	0.97	0.862	0.842
0x7003	1	0	1	1	1	1
Weighted Avg.	0.852	0.134	0.874	0.852	0.85	0.85

### 6.6.2 Banner Reaction

This section presents the results for classifying legitimate visitors by analysing their reaction to a banner. The primary focus is to demonstrate how the system is able to recognise such activities. Mouse movement data were saved into 100 temporary variables that correspond to each section of the page and then mapped into 10 ARFF variables (see Appendix A – Banner Reaction). One banner advertisement puzzle was generated along with the training dataset. The training dataset and the participant test set can be found in Appendix B – Banner Training Dataset. The screenshot for the advertisement page can be found in Appendix C – Advertisement Page. Finally, the test was conducted on Weka for the collected test data of participants (n=10), as listed below.

```
=== Run information ===
```

```
Scheme:weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:      swing
Instances:     123
Attributes:    11
              xy1
              xy2
              xy3
              xy4
```

```

xy5
xy6
xy7
xy8
xy9
xy10
class
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===
J48 pruned tree
-----
xy2 <= 33584
| xy2 <= 30382: false (12.0)
| xy2 > 30382: true (72.0/2.0)
xy2 > 33584: false (39.0)

Number of Leaves :    3
Size of the tree :    5

Time taken to build model: 0.03 seconds
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      117           95.122 %
Incorrectly Classified Instances     6             4.878 %
Kappa statistic                     0.8996
Mean absolute error                  0.0613
Root mean squared error              0.2189
Relative absolute error              12.4971 %
Root relative squared error          44.1853 %
Total Number of Instances           123

=== Detailed Accuracy By Class ===
          TP Rate  FP Rate  Precision  Recall  F-Measure  ROC Area
Class
true          0.986   0.094    0.932    0.986    0.958    0.926
false         0.906   0.014    0.98     0.906    0.941    0.926
Weighted Avg. 0.951   0.06     0.953    0.951    0.951    0.926

=== Confusion Matrix ===

  a  b  <-- classified as
69  1  |  a = true
 5 48  |  b = false

=== Re-evaluation on test set ===

User supplied test set
Relation: swing
Instances: unknown (yet). Reading incrementally
Attributes: 11

=== Predictions on test set ===

inst#,    actual, predicted, error, probability distribution
  1      ?      1:true      + *0.972 0.028
  2      ?      1:true      + *0.972 0.028
  3      ?      1:true      + *0.972 0.028
  4      ?      1:true      + *0.972 0.028
  5      ?      1:true      + *0.972 0.028
  6      ?      1:true      + *0.972 0.028
  7      ?      1:true      + *0.972 0.028
  8      ?      1:true      + *0.972 0.028
  9      ?      1:true      + *0.972 0.028

```



10            ?            1:true            +   \*0.972 0.028

The test dataset of the participants (n=10) was evaluated using the built-in decision tree algorithm. The system has to perform a binary classification, where the true value indicates a successful attempt on closing the banner. It was found that all the participants were able to respond correctly to the banner advertisement. The model reached an accuracy of 95.122% in detecting these responses. Figure 6.10 illustrates the Receiver Operating Characteristic (ROC) plot for the training dataset. The result shows that the area under ROC = 0.9263 indicates a good value.

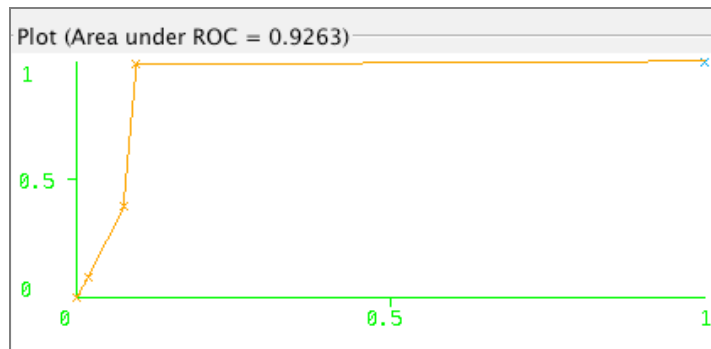


Figure 6.10. Receiver operating characteristic for the training dataset

### 6.6.3 Discriminating Traffic

The following are the results for discriminating the traffic on a test dataset of the ten legitimate participants and simulation of six bot attacks. All of the 16 test cases were evaluated using observation input data as described in Chapter 5 for the experiment settings. The results of classifying the test dataset were highly accurate, and the model was able to correctly identify ten participants and six simulated bot attacks. In practice, the accuracy of the classification could be manually checked by comparing the website heatmap data generated by the participants and the data produced by the bots. The evaluation used the main Traffic Training Dataset – Appendix B - to discriminate the traffic, as listed below.

=== Predictions on test split ===

inst#,	actual,	predicted,	error,	probability	distribution
1	?	1:0x7000	+ *1	0	0
2	?	1:0x7000	+ *1	0	0
3	?	1:0x7000	+ *1	0	0
4	?	1:0x7000	+ *1	0	0
5	?	1:0x7000	+ *1	0	0
6	?	1:0x7000	+ *1	0	0
7	?	1:0x7000	+ *1	0	0
8	?	1:0x7000	+ *1	0	0
9	?	1:0x7000	+ *1	0	0

10	?	1:0x7000	+	*1	0	0
11	?	2:0x7001	+	0.219	*0.781	0
12	?	2:0x7001	+	0.219	*0.781	0
13	?	2:0x7001	+	0.219	*0.781	0
14	?	2:0x7001	+	0.219	*0.781	0
15	?	2:0x7001	+	0.219	*0.781	0
16	?	2:0x7001	+	0.219	*0.781	0

The performance of the classification algorithm, Decision Tree J48, was evaluated using:

- **Training set**, reached an accuracy of 86.6667%.
- **Cross-validation with 10 folds**, reached an accuracy of 85.2381%.
- **Percentage split of 66%**, reached an accuracy of 83.0986%.

Figure 6.11 displays the generated tree using Weka Tree Visualizer.

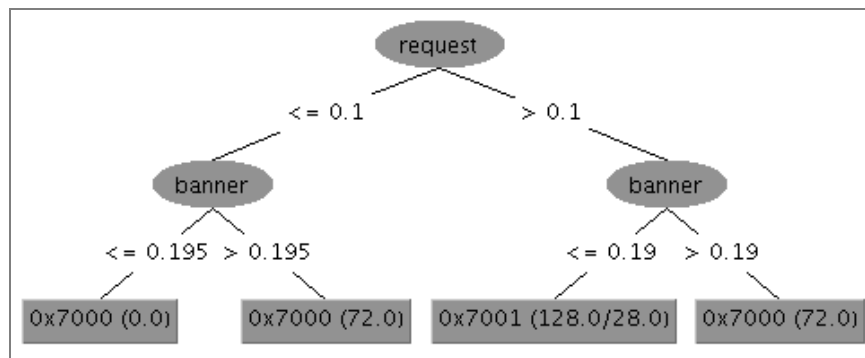


Figure 6.11. Tree view

## 6.7 Visual Analytics Design

This section presents the results for the prototype of the visual analytics design by using D3.js. A comparison with current state-of-the-art visualisation is also given.

### 6.7.1 Interactive Visualisation

The main dashboard uses the Quadtree data structure. The structure is used to represent IP addresses in the visualisation. Each dot inside the graph represents the value of an incoming connection. Similar connections with similar activities can be identified and selected for further details on demand analysis.

In practice, the classification result will be mapped accordingly into two separate canvas and represents current network traffic, as illustrated in Figure 6.12. The view can be calibrated to show differences over time. It aims to automatically classify between legitimate flash crowds against incoming bot traffic. The small,

8x8 or adjustable, quad represents a single client connection. Each quad contains a maximum of 32 bits in size, which is the IP address.

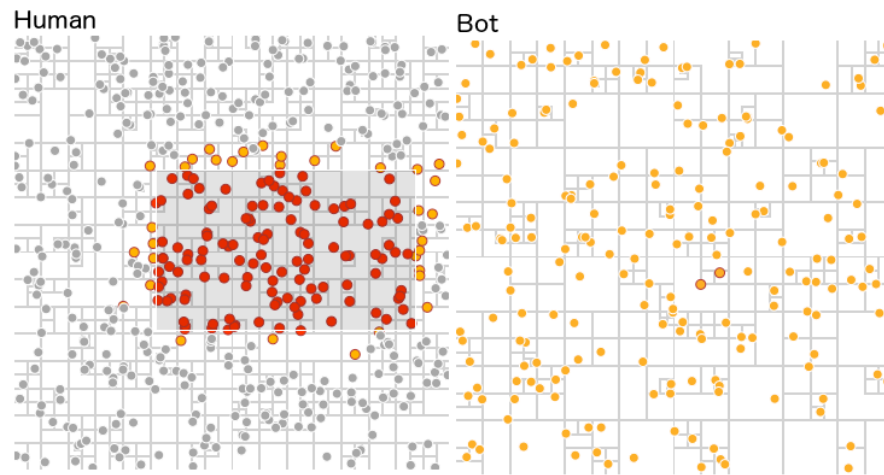


Figure 6.12. Quadtree representation using d3.js

A selection can be made using a rectangular selection. A further zooming will result into a Voronoi diagram. This is to simplify the visualisation and provide fair visualisation allocation, especially when flash crowds might occur in the network. The other reason is to group IP addresses according to their first octet. Another option such as blocking or exporting the IP address list is also given later.

By using a Voronoi diagram, it enables interactive visualisation to be presented into multiple regions, as depicted in Figure 6.13. A selection of multiple regions is possible. Thus, it provides an analyst with filtering, searching and details on demand capability. The region label indicates the first octet of an IP address.

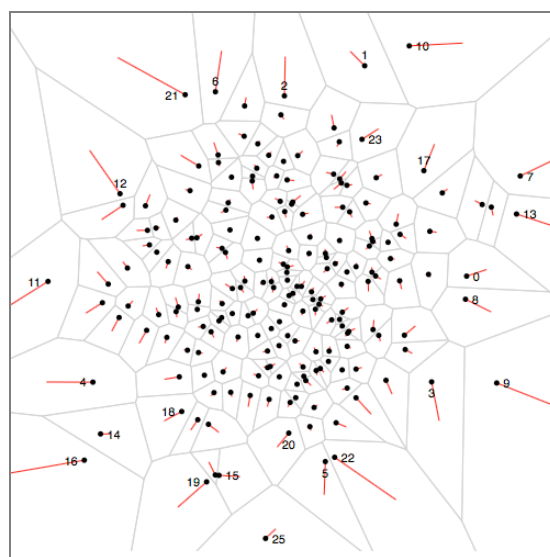


Figure 6.13. Voronoi diagram using d3.js

To understand the historical trend, the analyst may select another type – cumulative line chart - of presentation in time series format, as illustrated in Figure 6.14.

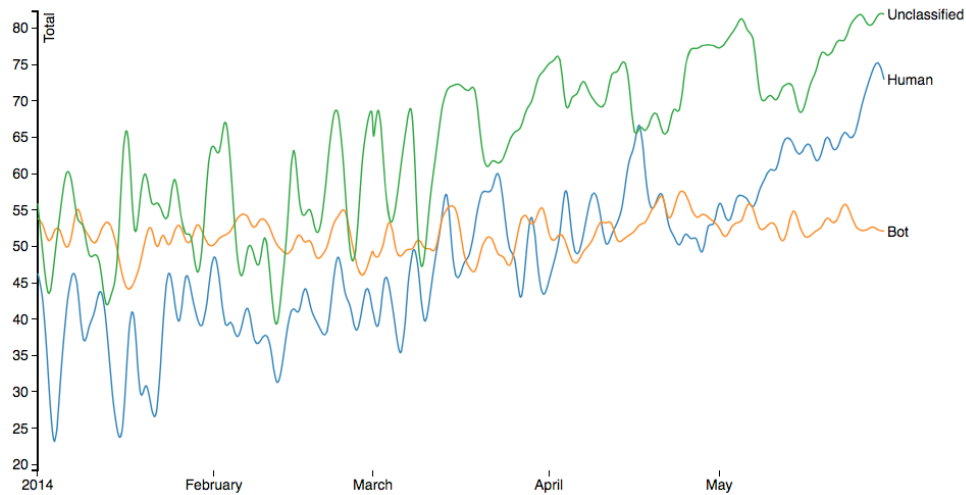


Figure 6.14. Time series format using d3.js

### 6.7.2 Large Dataset

A further expansion from the Voronoi diagram will result into a heatmap graph. This is to show which activities are intense. It enables better understanding, reasoning and decision making from very large datasets. The analyst becomes aware of the current situation by examining this heatmap graph, as illustrated in Figure 6.15. The X-axis represents client connection, while the Y-axis represents timestamp. The top row of the heatmap indicates latest situation. The green colour indicates legitimate visitors, and the violet colour indicates bot traffic. Darker colours indicate intense requests made by the client. This enables the analyst to gain an insight and discover patterns from very large datasets. However, it is assumed that bots will have constant or continually request patterns.

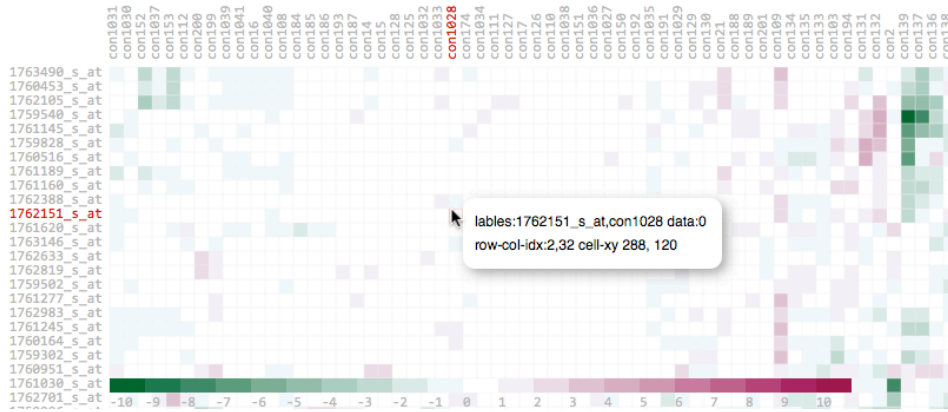


Figure 6.15. Heatmap graph example using d3.js

More information is given in Figure 6.16. Details on demand will provide more information regarding the hit rate and source of IP addresses.

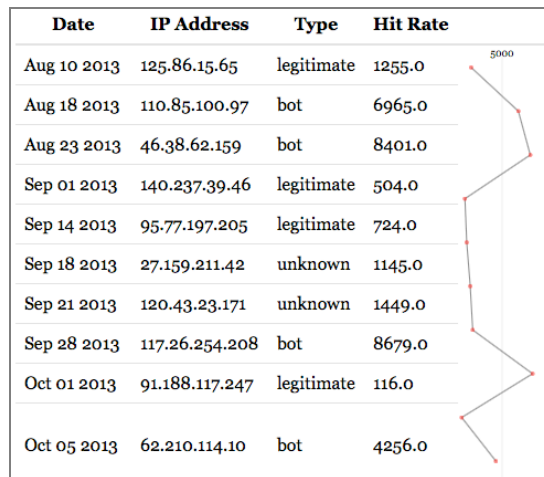


Figure 6.16. Latest hits

### 6.7.3 Knowledge Discovery

As illustrated in Figure 56, the visualisation enables the analyst to gain new knowledge to discover process states (*reading, waiting, idle and sending*) and to understand how they relate to the system health. The X-axis represents CPU usage, while the Y-axis represents memory usage. The size of the bubble indicates the total number of process instances currently running. The analyst can filter out any process group and make a particular selection, as illustrated in Figure 6.17.

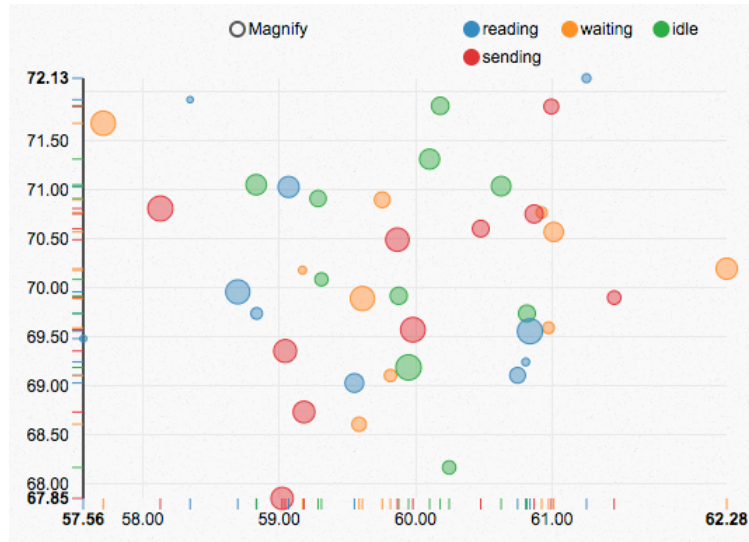


Figure 6.17. Process states visualisation using d3.js

Under certain circumstances when the web server becomes unresponsive, the analyst can quickly understand why and which process is causing the problem. This technique enhances awareness, especially when multiple instances are running.

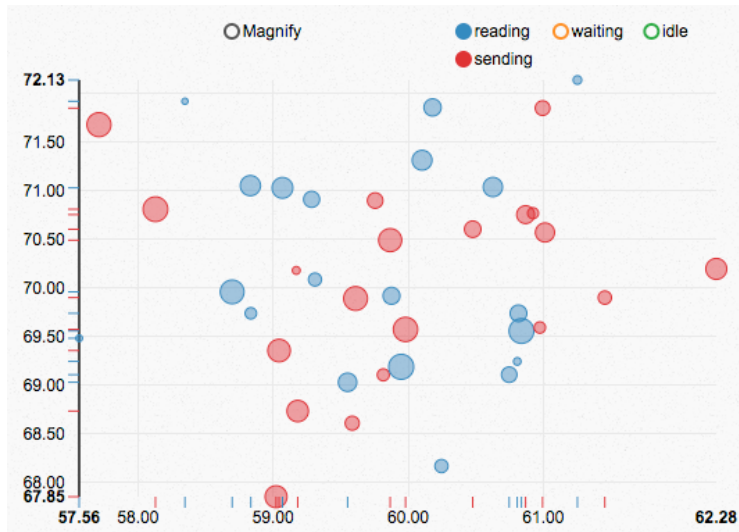


Figure 6.18. Group selection using d3.js

Furthermore, based on the selection, the analyst can explore the visualisation to obtain further information, as illustrated in Figure 6.18. The visualisation is presented in one single dashboard - this brings deeper exploration without changing the representation of the overall visualisation.

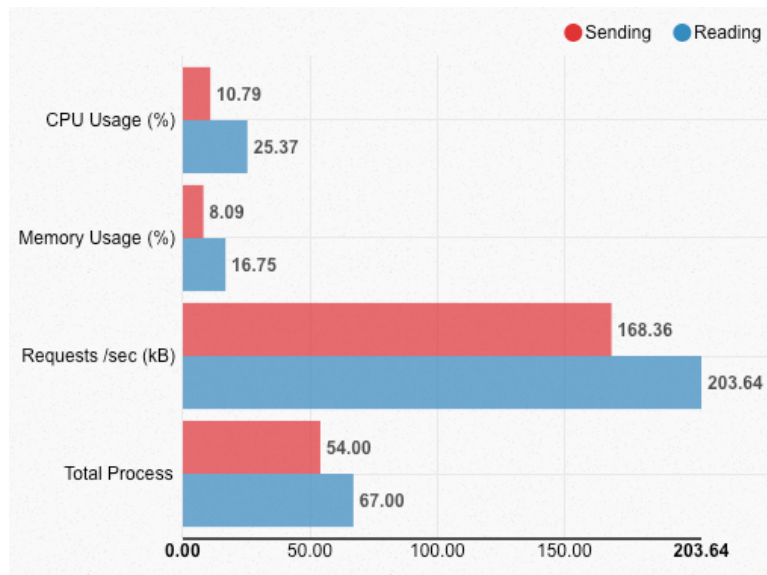


Figure 6.19. Further exploration

As an alternative, the analyst may select another type of visualisation presentation in a time series format, as illustrated in Figure 6.19. This enables historical trend to be analysed and provides an understanding regarding the current system health. Therefore, the analyst can understand the impact of a certain process state more effectively.

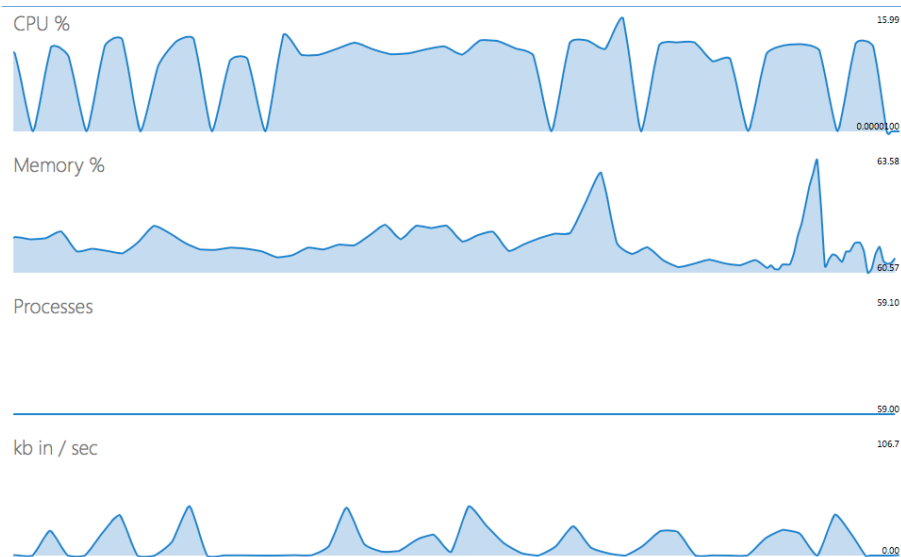


Figure 6.20. Time series format using d3.js

Compared to the traditional Apache *server-status* information and Visual Status, as depicted in Figure 6.21 and Figure 6.22, the proposed visualisation system provides a better interactivity that enables an analyst to discover knowledge regarding to Apache process states and system health.

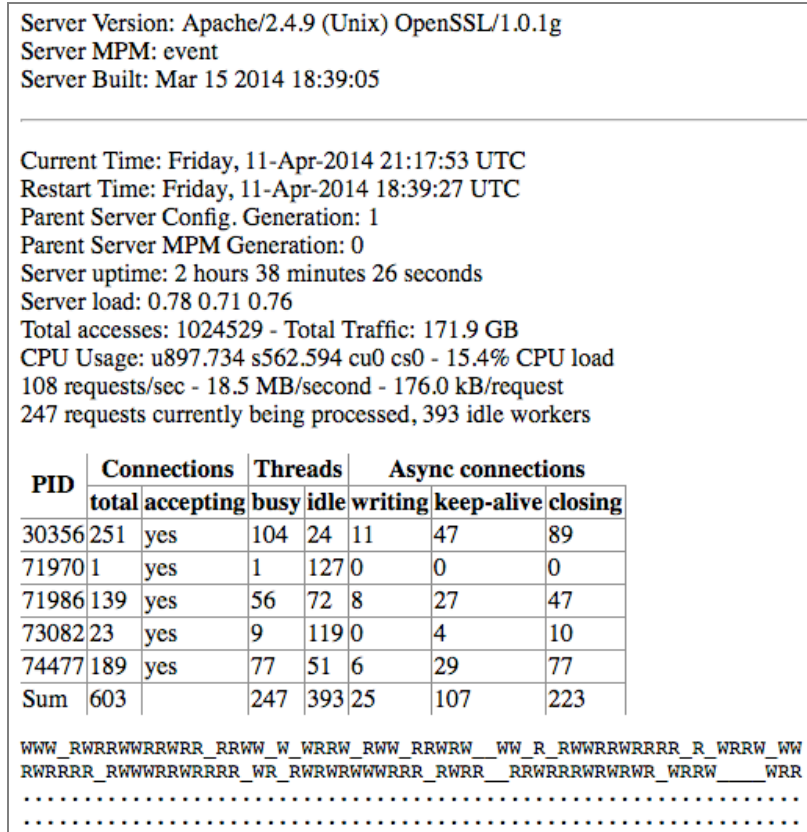


Figure 6.21. Apache process states

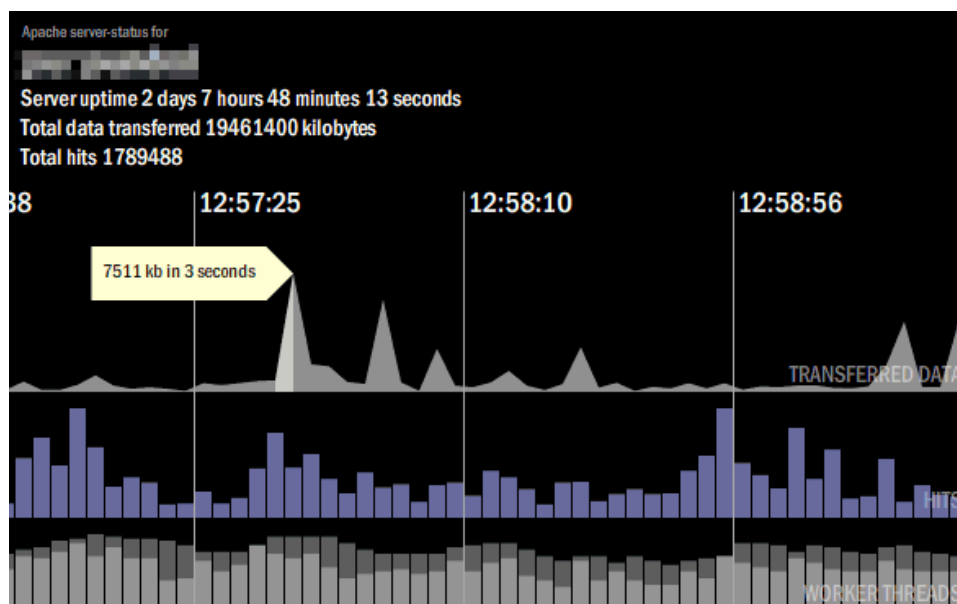


Figure 6.22. Visual status [70]

#### 6.7.4 Information Overload

As previously described, by using a heatmap graph, an analyst could gain an insight and discover patterns from very large datasets. Thus, decision making can



be made quickly to allow or block a certain connection. A further examination of the traffic is also possible, such as examining the pcap file or website heatmap data. This is to provide a measure of certainty that the current traffic belongs to a particular class. There might be several occasions behind the occurrences of such unwanted cases, for example incorrect classification results, corrupted dataset and advanced bot attacks that can perfectly mimic human behaviour, in which they can perfectly evade detection.

Thus, the proposed visualisation system aims to turn information overload into opportunity by discovering adaptive bot attacks – understanding how future bots will attack the system and need to know what will be impacted by them - and to enhance usability study by using website heatmap data. This is to “*detect the expected and discover the unexpected*” [11, 12]. Furthermore, X-Map exploits banner features to combine security protection with the opportunity of company advertisements. It provides advertising support, which can lead revenues behind them. According to a study by Shi et al. and Treesinthuros, Flash crowds and visitor volumes are often associated with banner advertisement revenues [71, 72].

## 6.8 Discussion

Reviewing the previous research questions as stated in Chapter 1:

**Question 1.** *Can we find a new solution to transparently classify bots presence against legitimate traffic?*

**To answer research question #1**, a banner advertisement was given to all the participants. In order to continue browsing the website, each participant must respond correctly to the banner advertisement. The results have shown that all the participants were able to respond correctly to the banner advertisement. The model reached an accuracy of 95.122% in detecting these responds. These results suggest that the reaction against banner advertisement could be used as a strong indicator of a human presence. However, there are several additional observation inputs that need to be considered in order to correctly classify traffic, e.g. the total of HTTP requests made and form submission frequency rate. Section 6.3.3 has presented these results, and the classification was highly accurate.

An adaptive persistent bot might have randomly clicked on every area of the page by using a brute force mechanism. However, this technique could be effectively

identified by calculating the total number of mouse click events. Further studies are therefore recommended in order to identify this threshold limit. In future investigations, it might be possible to use different input devices, such as Touchpad and Leap Motion. Furthermore, the banner advertisement should be randomly generated with a different level of complexity in order to deter the presence of bots.

**Question 2.** *By using visual analytics, how can the result of classification be presented in a decision-oriented way?*

**To answer research question #2,** the design of the visual analytics system has been given. The purpose of this design is to provide an analyst with a user interface, from which decision making can be made efficiently. Given vast amounts of network traffic, the analyst could interactively select parts of the data to determine their destiny or visualise it in detail. The analyst could grasp more comprehensive details without being constrained by large data. This brings simple action without changing the representation of the overall visualisation.

## 6.9 Summary

In this chapter the evaluation of the research has been presented. The experiment and simulation approaches have been covered along with the discussion. The results and performance of the classification algorithm, using Decision Tree J48, have been discussed. The design of the visual analytics system has been given. Finally, the research questions have been answered in the discussion section.

As it can be seen from the results in Section 6.2, the threshold values present positive results to signify abnormal behaviour. In addition, these values were used for augmenting the initial training dataset. The system has more data to analyse about the visitor behaviour and predict this against the training dataset. The results of the visitor behaviour tracking indicate a good page performance load, as indicated in Section 6.3. These results show that there is no significant performance difference compared to the pages without tracking enabled. The results of AJAX timing request, as described in Section 6.4, provide a better illustration of the entire tracking. As presented in Section 6.5, the average value of requests per second was 25.673. This value indicates significant positive correlation and still above the threshold value of maximum simultaneous requests

per second made by participants. The total requests of simultaneous connections were used to give a strong indicator of abnormal behaviour. Overall, the classifier performance obtained an accuracy of 85.2381%, as shown in Section 6.6. The decision tree (J48) algorithm was chosen due to its high accuracy value and low false-positive rate. The results for classifying banner reaction reached an accuracy of 95.122% with ROC = 0.9263. The results of classifying the test dataset were highly accurate, and the model was able to correctly identify the ten participants and six simulated bot attacks with an accuracy of 86.6667%. Finally, the visual analytics design was formulated in order to assist an analyst to discover bots presence.

Along with these results, two comparisons between VATRIX and X-Map with other current approaches have been made as illustrated in Table 6.3 and Table 6.4 below. Table 6.3 provides a comparison between existing solutions and X-Map.

	<b>Application Layer</b>	<b>CAPTCHA</b>	<b>Banner</b>	<b>Browsing Behaviour</b>
X-Map	Yes	No	Yes	Yes
Kill-Bots	Yes	Yes	No	No
Yu et al.	No	No	No	No
Xie and Yu	Yes	No	No	Yes
Oikonomou	Yes	No	No	Yes

Table 6.3 X-Map Comparison Table

X-Map gives flexible solution, which embraces the use of decision tree learning for classification and to make predictions based on visitor past observation. X-Map attempts to gather all visitors browsing behaviour before making any predictive response. X-Map effectively emphasises on the method that can handle large datasets while reducing computation complexity. This enables X-Map to have many advantages over other systems, which lacks the use of banner reaction technique. While current approaches worked well at the application layer, however, the disadvantage of these approaches is that CAPTCHA is often needed. On the other hand X-Map offers a different solution in order to discriminate traffic by employing website heatmap and banner reaction.

Table 6.4 provides a comparison between existing solutions and VATRIX. VATRIX offers an effective solution for visual analytics interaction, which embraces the use of X-Map along with the Voronoi diagram and Quadtree data structure.

	<b>Method</b>	<b>Visualisation</b>	<b>Interaction</b>
VATRIX	X-Map	Voronoi	No
Zhang et al.	Density-workload	Graph model	No
NetSecRadar	IDS dependent	Radial graph	No
VisTracer	Anomaly detection	Glyph	Yes
VACS	Elastic Search	Hierarchical	Yes
Yassem	Network abstraction	Hilbert curve	No
Shurkhovetsky	InfoVis Toolkit	Multiple graphs	Yes
Mansmann et al.	Dynamic event	Modular graphs	No

Table 6.4 VATRIX Comparison Table

VATRIX has advantages over other current approaches, which automatically classifies between legitimate flash crowds against incoming bot traffic. Current approaches are not specifically designed for bot attacks that are capable of mimicking genuine flash crowds. The disadvantage of the current approaches is that they are mainly built for detecting flood attacks and lacking the interactivity response in order to mitigate low and slow bandwidth attacks. Finally, the next chapter will review the major conclusion and future work of this thesis.

## Chapter 7 Conclusions and Future Work

This final chapter concludes this thesis by highlighting the main contributions and discussing directions for future research.

### 7.1 Conclusion

Internet services are continually subject to new threats. The presence of bot attacks has increased greatly. Although these attacks consume less bandwidth compared to volumetric attacks, application layer bot attacks are stealthier in nature, which often lead to a disruptive impact. As covered in Chapter 2, there have been several vulnerability cases that provide attackers with a high potential of extortion to bring HTTP servers down. Mitigating and differentiating these bot activities from flash crowds traffic remains an open challenge to date. In this dissertation, the bot attack problem has been investigated. In Chapter 3, the architecture of VATRIX has been presented. It provides an insight in how the system works. In order to successfully show the visualisation system, the user interface was also presented in detail. In Chapter 4, the method X-MAP has been discussed. It covers the solution for traffic classification. In Chapter 5, the experiment settings have been given. It covers the experimental simulation study consisting of two phases, the benchmark assessment and attack simulation using external tools. Finally, in Chapter 6, the evaluation has been presented along with the significant results produced from our research. A comparison with related work has been covered.

The contributions of this thesis were presented in Chapter 3 and Chapter 4. They include two primary contributions. The first is the development of the visual analytics system, VATRIX, that can interactively respond and discover bot presence. The second is the scalable technique, X-MAP, that transparently separates legitimate traffic against bot traffic. These findings have shown the advantages of website heatmaps for predicting the presence of a human with the opportunity of usability study. It is demonstrated that by analysing how visitors respond to advertisements, better traffic classification with the opportunity of company advertisements can be provided. This condition can lead to beneficial revenues behind them. These techniques provide a measure of certainty, which is missing in the current methods.

## 7.2 Future Work

A defence system relying on machine learning and artificial intelligence alone is not sufficient enough in defeating an adaptive adversary. In a world of persistent threats, this may leave an open door to other attacks. There are several future directions that can be pursued following this dissertation.

### 1. Optimizing the data acquisition and storage process

Collecting visitor behaviour on a website produces vast amounts of data. Ensuring data quality remains a challenging problem for almost every large organisation. Currently, the process of acquiring and storing visitor behaviour data is not optimised, both in terms of capacity and reliability. Incorrect or inconsistent data can significantly influence the result of classification. Regarding this issue, additional research work is required in order to optimise this process. Our evaluation shows that the current methods offer viable solutions despite the vast amounts of data they produce. Therefore, integration with a reliable storage will be the next focus.

### 2. Single point of failure

X-Map largely depends on the existence of a giant hash table produced by memcached. In this way, X-Map is able to handle asynchronous requests. It allows a client(s) to lookup single or multiple IP addresses. However, the issue of a single point of failure will need to be addressed as the amount of traffic increases. An automatic failover mechanism will need to be established in order to provide high availability. Future research work should look further into the issues regarding the single point of failure so that upon abnormal termination, key-value data can be safely collected and stored.

### 3. Sonification

The current visual analytics system is capable of discovering bot traffic. However, the future research work would consider expanding the use of sonification by turning information overload into a meaningful sound, in order to better assist an analyst.

## References

- [1] Incapsula Inc (2013), Bot Traffic Report 2013, Available at: <http://www.incapsula.com/blog/bot-traffic-report-2013.html>. [Accessed 2<sup>nd</sup> April, 2014]
- [2] Arbor Networks (2012), The Growing Threat of Application-Layer DDoS Attacks, Available at: [https://www.arbornetworks.com/component/docman/doc\\_download/467-the-growing-threat-of-application-layer-ddos-attacks?itemid=442](https://www.arbornetworks.com/component/docman/doc_download/467-the-growing-threat-of-application-layer-ddos-attacks?itemid=442). [Accessed 2<sup>nd</sup> April, 2014]
- [3] Cloud Security Alliance (2013), The Notorious Nine: Cloud Computing Top Threats in 2013, Available at: <https://cloudsecurityalliance.org/download/the-notorious-nine-cloud-computing-top-threats-in-2013/>. [Accessed 2<sup>nd</sup> April, 2014]
- [4] Yu, S., Zhou, G., Guo, S., Vasilakos, A., and Xiang, Y., *Browsing behaviour mimicking attacks on popular web sites for large botnets*, IEEE Conference on Computer Communications Workshops, pp. 947-951, 2011.
- [5] Yu, S., Zhao, G., Guo, S. and Stojmenovic, I., *Fool Me If You Can: Mimicking Attacks and Anti-attacks in Cyberspace*, IEEE Transactions on Computers, Vol PP, No 99, pp. 1-13, 2013.
- [6] Lee, M., Kompella, R. and Singh, S., *AjaxTracker: Active Measurement System for High-Fidelity Characterization of AJAX Applications*, USENIX Annual Technical Conference, pp. 1-12, 2010.
- [7] Jin, J., Zheng, N., Mao, F., Koehl, A. and Wang, H., *Evasive Bots Masquerading as Human Beings on the Web*, International Conference on Dependable Systems and Networks, Vol 1, No 12, pp. 1-12, 2013.
- [8] Yu, S., Zhao, G., Guo, S. and Stojmenovic, I., *Can We Beat Legitimate Cyber Behavior Mimicking Attacks from Botnets?*, IEEE International Conference on Computer Communications, pp. 3133-3137, 2012.
- [9] Motoyama, M., Levchenko, K., Kanich, C. and McCoy, D., *Re:CAPTCHA – Understanding CAPTCHA-Solving Services in an Economic Context*, USENIX Conference on Security, pp. 28-38, 2010.
- [10] Boshmaf, Y., Musluhkhov, I., Beznosov, K. and Ripeanu, M., *The Socialbot Network: When Bots Socialize for Fame and Money*, Proceedings of the 27<sup>th</sup> Annual Computer Security, pp. 93-102, 2011.
- [11] Keim, D., Andrienko, G., Fekete, J. and Gorg, C. (2008) *Visual Analytics: Definition, Process, and Challenges*, Berlin: Springer-Verlag.
- [12] Keim, D., Kohlhammer, J., Ellis, G. and Mansmann, F. (2010) *Mastering the Information Age Solving Problems with Visual Analytics*, Bad Langensalza: Eurographics.
- [13] Moustis, D. and Kotzanikolaou, P., *Evaluating security controls against HTTP-based DDoS attacks*, International Conference on Information, Intelligence, Systems and Applications, pp. 1-6, 2013.
- [14] Zargar, S., Joshi, J. and Tipper, D., *A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks*, IEEE Communications Surveys and Tutorials, Vol 15, No 4, pp. 1-24, 2013.
- [15] Durcekova, V., Schwartz, L. and Shahmehri, N., *Sophisticated Denial of Service Attacks Aimed at Application Layer*, International Conference on Electronics, pp. 55-60, 2012.

- [16] Juniper Networks (2013), Defending Against Application-Layer DDoS Attacks, Available at: <http://www.juniper.net/us/en/local/pdf/whitepapers/2000550-en.pdf>. [Accessed 2<sup>nd</sup> April, 2014]
- [17] Prolexic (2014), Prolexic Quarterly Global DDoS Attack Report, Available at: <http://www.prolexic.com/knowledge-center-ddos-attack-report-2014-q1.html>. [Accessed 2<sup>nd</sup> April, 2014]
- [18] JS Heatmaps (2013), Create HTML5 Heatmaps with Canvas and JavaScript, Available at: <http://www.patrick-wied.at/static/heatmapsjs/>. [Accessed 2<sup>nd</sup> April, 2014]
- [19] Huang, J. and White, R., *Improving Searcher Models Using Mouse Cursor Activity*, Proceedings of the 35<sup>th</sup> International ACM SIGIR, pp. 195-204, 2012.
- [20] Eckersley, P., *How Unique is Your Web Browser*, Proceedings of the 10<sup>th</sup> International Conference on Privacy Enhancing Technologies, pp. 1-18, 2010.
- [21] Mulazzani, M., Reschl, P. and Huber, M., *Fast and Reliable Browser Identification with JavaScript Engine Fingerprinting*, Web Security and Privacy Conference, pp. 1-10, 2013.
- [22] Mowery, K., Bogenreif, D., Yilek, S. and Shacham, H., *Fingerprinting Information in JavaScript Implementations*, In Proceedings of W2SP, pp. 1-11, 2011.
- [23] Panopticlick (2013), How Unique and Trackable is Your Browser?, Available at: <http://panopticlick.eff.org>. [Accessed 2<sup>nd</sup> April, 2014]
- [24] Forbes (2014), Information for the World's Business Leaders, Available at: <http://www.forbes.com>. [Accessed 2<sup>nd</sup> April, 2014]
- [25] InformationWeek (2014), News Connects The Business Technology Community, Available at: <http://www.informationweek.com>. [Accessed 2<sup>nd</sup> April, 2014]
- [26] Verma, S., *Impact of Repetitive and Contextual Advertisements on Consumer Behavior: An exploratory Study*, International Association of Computer Science and Information Technology, pp. 221-225, 2009.
- [27] Muda, M., Musa, R., Mohamed, R. and Hamzah, H., *The Influence of Perceived Celebrity Endorser Credibility on Urban Women's Responses to Skincare Product Advertisement*, IEEE Colloquium on Humanities, Science and Engineering, pp. 620-625, 2011.
- [28] Kitts, B., Zhang, J., Roux, A. and Mills, R., *Click Fraud Detection with Bot Signatures*, IEEE International Conference on Intelligence and Security Informatics, pp. 146-150, 2013.
- [29] Kitts, B., Zhang, J., Wu, G. and Mahato, R., *Click Fraud Botnet Detection by Calculating Mix Adjusted Traffic Value*, IEEE International Conference on Intelligence and Security Informatics, pp. 151-153, 2013.
- [30] Kaiser, E. and Feng, W., *Helping TicketMaster: Changing the Economics of Ticket Robots with Geographic Proof-of-Work*, IEEE Conference on Computer Communications Workshops, pp. 1-6, 2010.
- [31] Pandey, A. and Rangan, P., *Mitigating Denial of Service Attack using Proof of Work and Token Bucket Algorithm*, Students Technology Symposium, pp. 43-47, 2011.
- [32] Laurie, B. and Clayton, R., *Proof-of-Work Proves Not to Work*, In Proceeding WEAS, pp. 1-9, 2004.



- [33] Wikipedia (2013), Proof of Work System, Available at: [http://en.wikipedia.org/wiki/Proof-of-work\\_system](http://en.wikipedia.org/wiki/Proof-of-work_system). [Accessed 2<sup>nd</sup> April, 2014]
- [34] Ari, I., Hong B., Miller, E., Brandt, S. and Long D., *Managing Flash Crowds on the Internet*, International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems, pp. 246-249, 2003.
- [35] Yu, S., Zhou, W., Guo, S., Xiang, Y. and Jia, W., *Discriminating DDoS Attacks from Flash Crowds Using Flow Correlation Coefficient*, IEEE Transactions on Parallel and Distributed Systems, Vol. 23, No. 6, pp. 1073-1080, 2012.
- [36] Thapngam, T., Yu, S. and Zhou, W., *DDoS discrimination by Linear Discriminant Analysis (LDA)*, International Conference on Networking and Communications (ICNC), pp. 532-536, 2012.
- [37] Kandula, S., Katabi, D., Jacob, M. and Berger, A., *Botz-4-Sale: Surviving Organized DDoS Attacks That Mimic Flash Crowds*, Symposium on Networked Systems Design & Implementation, pp. 287-300, 2005.
- [38] Xie, Y. and Yu, S., *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, 2009.
- [39] Xie, Y. and Yu, S., *Monitoring the Application-Layer DDoS Attacks for Popular Websites*, IEEE/ACM Transactions on Networking, Vol. 17, No. 1, pp. 15-25, 2009.
- [40] Oikonomou, G. and Mirkovic, J., *Modeling Human Behavior for Defense Against Flash-Crowd Attacks*, IEEE International Conference on Communications, pp. 1-6, 2009.
- [41] Zhang, J. and Huang, M., *Visual Analytics Model for Intrusion Detection in Flood Attack*, IEEE International Conference on Trust Security and Privacy in Computing and Communications, pp. 277-284, 2013.
- [42] Zhao, Y., Zhou, F. and Shi, R., *NetSecRadar: A Real-Time Visualization System for Network Security*, IEEE Symposium on Visual Analytics Science and Technology, pp. 281-282, 2012.
- [43] Fischer, F., Fuchs, J. and Vervier, P., *VisTracer: A Visual Analytics Tool to Investigate Routing Anomalies in Traceroutes*, International Symposium on Visualization for Cyber Security, pp. 1-8, 2012.
- [44] Fischer, F. and Keim, D., *VACS: Visual Analytics Suite for Cyber Security*, VAST Challenge 2013, pp. 1-2, 2013.
- [45] Shurkhovetsky, G., Bahey, A. and Ghoniem, M., *Visual Analytics for Network Security*, IEEE Symposium on Visual Analytics Science and Technology, pp. 1-2, 2012.
- [46] Yassem, T., *A visual analytics approach to network security hygiene*, System and Network Engineering, pp. 1-16, 2013.
- [47] Mansmann, F., Fischer, F. and Keim, D., *Real-Time Visual Analytics for Event Data Streams*, Proceedings of the 27<sup>th</sup> Annual ACM Symposium on Applied Computing, pp. 801-806, 2012.
- [48] PrestaShop (2014), Ecommerce Software to create your Online Store, Available at: <http://www.prestashop.com>. [Accessed 2<sup>nd</sup> April, 2014]
- [49] Leo Theme (2013), Leo Sport Shoes Free Prestashop Theme, Available at: <http://www.leotheme.com/prestashop/themes/216-leo-sportshoes.html>. [Accessed 2<sup>nd</sup> April, 2014]

- [50] Slowloris (2011), Slowloris HTTP DoS, Available at: <http://ha.ckers.org/slowloris>. [Accessed 2<sup>nd</sup> April, 2014]
- [51] Slowhttptest – Google Code (2011), Application Layer DoS attack simulator, Available at: <http://code.google.com/p/slowhttptest>. [Accessed 22<sup>nd</sup> February, 2014]
- [52] R-u-dead-yet – Google Code (2011), HTTP POST Denial of Service tool, Available at: <http://code.google.com/p/r-u-dead-yet>. [Accessed 22<sup>nd</sup> February, 2014]
- [53] THC SSL DOS (2011), A tool to verify the performance of SSL, Available at: <http://www.thc.org/thc-ssl-dos>. [Accessed 22<sup>nd</sup> February, 2014]
- [54] Radware DDoSPedia (2011), Apache Killer Attacks, Available at: <http://security.radware.com/knowledge-center/DDoSPedia/Apache-Killer>. [Accessed 22<sup>nd</sup> February, 2014]
- [55] PhantomJS (2014), Full web stack No Browser required, Available at: <http://phantomjs.org>. [Accessed 22<sup>nd</sup> February, 2014]
- [56] The University of Waikato (2008), Attribute-Relation File Format, Available at: <http://www.cs.waikato.ac.nz/ml/weka/arff.html>. [Accessed 22<sup>nd</sup> February, 2014]
- [57] Weka 3 Manual (2013), Data Mining with Open Source Machine Learning Software in Java, Available at: <http://www.cs.waikato.ac.nz/ml/weka/documentation.html>. [Accessed 22<sup>nd</sup> February, 2014]
- [58] D3.js (2013), Data-Driven Documents, Available at: <http://d3js.org>. [Accessed 2<sup>nd</sup> April, 2014]
- [59] Reinvigorate (2012), Drupal Plugins, Available at: <http://drupal.org/project/reinvigorate>. [Accessed 2<sup>nd</sup> April, 2014]
- [60] Laurie, B. and Peter, L. (2002) *Apache: The Definitive Guide*, 3<sup>rd</sup>. Ed. Sebastopol: O’reilly.
- [61] Netcraft (2014), March 2014 Web Server Survey, Available at: <http://news.netcraft.com/archives/2014/03/03/march-2014-web-server-survey.html>. [Accessed 2<sup>nd</sup> April, 2014]
- [62] Li, J. and Lu, M., *The Performance Optimization and Modeling Analysis based on the Apache Web Server*, International Control Conference, Vol 2, No 1, pp. 1712-1716, 2013.
- [63] Tantithamthavorn, C. and Rungsawang, A., *Knowledge Discovery in Web Traffic Log: A Case Study of Facebook usage in Kasetsart University*, International Joint Conference on Computer Science and Software Engineering, Vol 3, No 9, pp. 247-252, 2012.
- [64] Xiaojun, B., *A Remote Monitoring System Based on Web and SNMP*, Industrial Control and Electronics Engineering, Vol 1, No 1, pp. 452-455, 2012.
- [65] NVD3.js examples (2014), Scatter Chart, Available at: <http://nvd3.org/examples/scatter.html>. [Accessed 2<sup>nd</sup> April, 2014]
- [66] Weka The University of Waikato (2014), Data Mining with Open Source Machine Learning Software in Java, Available at: <http://www.cs.waikato.ac.nz/ml/weka/>. [Accessed 2<sup>nd</sup> April, 214]
- [67] YSlow (2014), Yahoo YSlow for Safari, Available at: <http://developer.yahoo.com/yslow/>. [Accessed 2<sup>nd</sup> April, 214]
- [68] Wikipedia (2013), Quadtree, Available at: <http://en.wikipedia.org/wiki/Quadtree>. [Accessed 2<sup>nd</sup> April, 2014]

- [69] Wikipedia (2013), Voronoi Diagram, Available at: <http://en.wikipedia.org/wiki/Voronoi>. [Accessed 2<sup>nd</sup> April, 2014]
- [70] Visual Status (2012), A tool to visualize Apache module mod\_status output, Available at: <http://kyyhkynen.net/stuff/visualstatus/>. [Accessed 2<sup>nd</sup> April, 2014]
- [71] Shi, H., Tsai, J., Ho, W. and Lee, K., *Autoregressive Integrated Moving Average Model for Long-Term Prediction of Emergency Department Revenue and Visitor Volume*, International Conference on Machine Learning and Cybernetics, Vol 3, No 1, pp. 979-982, 2011.
- [72] Treesinthuros, W., *An Empirical Study of E-commerce Marketing Success*, International Conference on ICT and Knowledge Engineering, Vol 1, No 10, pp. 10-13, 2012.
- [73] Wikipedia (2013), FIFO, Available at: <http://en.wikipedia.org/wiki/fifo>. [Accessed 2<sup>nd</sup> May, 2014]
- [74] Kurematsu, M. and Fujita, H., *A Framework for Integrating a Decision Tree Learning Algorithm and Cluster Analysis*, IEEE International Conference on Intelligent Software Methodologies, Tools and Techniques, pp. 225-228, 2013.
- [75] Wikipedia (2013), Statistical Classification, Available at: [http://en.wikipedia.org/wiki/statistical\\_classification](http://en.wikipedia.org/wiki/statistical_classification). [Accessed 2<sup>nd</sup> May, 2014]
- [76] Iqbal, S., Shaheen, M. and Basit, F., *A Machine Learning Based Method for Optimal Journal Classification*, International Conference for Internet Technology and Secured Transactions, pp. 259-264, 2013.
- [77] Wikipedia (2014), Display Resolution, Available at: [http://en.wikipedia.org/wiki/Display\\_resolution](http://en.wikipedia.org/wiki/Display_resolution). [Accessed 2<sup>nd</sup> May, 2014]
- [78] Voronoi Handout (2005), Computational Geomerty, Available at: [http://www.cs.tufts.edu/comp/163/notes05/voronoi\\_handout.pdf](http://www.cs.tufts.edu/comp/163/notes05/voronoi_handout.pdf). [Accessed 2<sup>nd</sup> May, 2014]
- [79] Wikipedia (2014), C4.5 Algorithm, Available at: [http://en.wikipedia.org/wiki/C4.5\\_algorithm](http://en.wikipedia.org/wiki/C4.5_algorithm). [Accessed 2<sup>nd</sup> May, 2014]

## Appendix A – Data Collection Scripts

### A.1 IP and TCP Header

```
void print_ip_header(unsigned char* Buffer, int Size)
{
    unsigned short iphdrlen;

    struct iphdr *iph = (struct iphdr *)Buffer;
    iphdrlen =iph->ihl*4;

    memset(&source, 0, sizeof(source));
    source.sin_addr.s_addr = iph->saddr;

    memset(&dest, 0, sizeof(dest));
    dest.sin_addr.s_addr = iph->daddr;

    fprintf(logfile,"    |-Source IP      : %s\n",inet_ntoa(source.sin_addr));
    fprintf(logfile,"    |-Destination IP  : %s\n",inet_ntoa(dest.sin_addr));
}

void print_tcp_packet(unsigned char* Buffer, int Size)
{
    unsigned short iphdrlen;

    struct iphdr *iph = (struct iphdr *)Buffer;
    iphdrlen = iph->ihl*4;

    struct tcphdr *tcph=(struct tcphdr*)(Buffer + iphdrlen);

    print_ip_header(Buffer,Size);

    fprintf(logfile,"\n");
    fprintf(logfile,"TCP Header\n");
    fprintf(logfile,"    |-Source Port      : %u\n",ntohs(tcph->source));
    fprintf(logfile,"    |-Destination Port : %u\n",ntohs(tcph->dest));

};
```

### A.2 Request Behaviour

```
<?php $handle = fopen("counter.txt", "r");
if(!$handle)
{
    echo "could not open the file" ;
}
else { $counter = (int )
    fread($handle,20);

fclose ($handle);
$counter++;
$handle = fopen("counter.txt", "w" ); fwrite($handle,$counter) ;
fclose ($handle) ; } ?>
```

### A.3 Simultaneous Request

```
ip=$1
export ip
netstat -n | awk '{if ($4 == ENVIRON["ip"]":80") {if ($6 ==
"ESTABLISHED"){print("-i "$5) }}}'
```

```
netstat -plan | grep :80 | wc -l
```

## A.4 Proof of Work

```
<script type="text/javascript">

$('body').onload(function(event) {
  var msg = $.md5('testing');
  $('#msg').text(msg);
});

</script>
```

## A.5 Form Submission Frequency

```
foreach($_POST["name"] as $value) {
  $var .= count($value);
}
echo $var;

$total = ""

if(isset($_POST['cb1'])){
  //count it
  $total = $total + 1;
}
if(isset($_POST['cb2'])){
  // add it to the total
  $total = $total + 1;
}
```

## A.6 Authenticated Session

```
require './includes/bootstrap.inc';
drupal_bootstrap(DRUPAL_BOOTSTRAP_FULL);
if (!user_access('administer nodes')) {
  drupal_access_denied();
  exit(0);
}
```

## A.7 Website Heatmap

```
<!DOCTYPE html>
<html lang="en">
...
<div id="heatmapArea" />
<script type="text/javascript" src="heatmap.js"></script>
<script type="text/javascript">
window.onload = function(){

  // heatmap configuration
  var config = {
    element: document.getElementById("heatmapArea"),
    radius: 30,
    opacity: 50
  };

  //creates and initializes the heatmap
  var heatmap = h337.create(config);

  // let's get some data
  var data = {
    max: 20,
    data: [
```

```

        { x: 10, y: 20, count: 18 },
        { x: 25, y: 25, count: 14 },
        { x: 50, y: 30, count: 20 }
        // ...
    ]
};

heatmap.store.setDataSet(data);
};
</script>
</html>

```

## A.8 Banner Reaction

```

<head>
<script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>

    <h1>Swing Motion Experiment</h1>
    
    <p id="msg"></p>
    <p id="arff"></p>

    <script type="text/javascript">

        var i=0;
        var x=0;
        var y=0;
        var sum = [];

        $('body').mousemove(function(event) {
            x = event.pageX;
            y = event.pageY;

            var msg = 'mousemove() position . x : ' + x + ', y : ' + y + '
[counter] : ' + i;
            $('#msg').text(msg);

            sum[i] = x + '' + y;

            i++;

        });

        $('body').dblclick(function(event) {

            var output = '';

            for(var j = sum.length - 10; j < sum.length; j++)
            {
                output = output + sum[j] + ',';
            }

            $('#arff').text(output+'?');
            console.log(output+'?');

            i = 0;
            sum.length = 0;

        });

    </script>
</body>
</html>

```

## A.9 Screen Resolution

```
<script  
src="//ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>  
<script type="text/javascript">  
width = $(window).width();  
height = $(window).height();  
</script>
```

## A.10 Browser Fingerprint

```
<script type="text/javascript"  
src="http://www.blangdon.com/js/fingerprint.js"></script>  
<script type="text/javascript">  
var fingerprint = new Fingerprint().get();  
document.getElementById("fingerprint").innerHTML = fingerprint;  
</script>
```

## Appendix B – Training and Test Dataset (ARFF)

### B.1 Traffic Training Dataset

@relation discriminate

```
@attribute 'request'    numeric
@attribute 'pow'        numeric
@attribute 'freq'       numeric
@attribute 'auth'       numeric
@attribute 'heatmap'   numeric
@attribute 'banner'     numeric
@attribute 'screen'     numeric
@attribute 'fp'         numeric
@attribute 'class'      {0x7000, 0x7001, 0x7003}
```

@data

```
0.86,0.15,0.78,0.11,0.11,0.27,0.28,0.16,0x7000
0.86,0.19,0.85,0.10,0.11,0.11,0.18,0.15,0x7000
0.84,0.18,0.77,0.10,0.11,0.13,0.18,0.10,0x7000
0.75,0.16,0.67,0.11,0.10,0.34,0.10,0.16,0x7000
0.86,0.14,0.87,0.10,0.10,0.32,0.22,0.19,0x7000
0.87,0.13,0.83,0.10,0.10,0.27,0.27,0.13,0x7000
0.78,0.14,0.73,0.11,0.10,0.25,0.12,0.19,0x7000
0.79,0.13,0.62,0.10,0.10,0.27,0.24,0.13,0x7000
0.81,0.10,0.67,0.10,0.11,0.26,0.17,0.19,0x7000
0.72,0.11,0.80,0.11,0.10,0.17,0.22,0.15,0x7000
0.73,0.10,0.76,0.11,0.11,0.19,0.27,0.19,0x7000
0.84,0.18,0.66,0.10,0.10,0.17,0.19,0.15,0x7000
0.75,0.19,0.86,0.11,0.10,0.38,0.11,0.18,0x7000
0.70,0.16,0.84,0.11,0.11,0.14,0.24,0.14,0x7000
0.81,0.14,0.74,0.10,0.10,0.35,0.16,0.17,0x7000
0.82,0.16,0.70,0.11,0.10,0.37,0.21,0.14,0x7000
0.73,0.14,0.83,0.10,0.10,0.28,0.13,0.17,0x7000
0.88,0.14,0.88,0.11,0.11,0.34,0.26,0.13,0x7000
0.79,0.12,0.78,0.10,0.10,0.25,0.11,0.19,0x7000
0.80,0.11,0.67,0.11,0.11,0.27,0.16,0.13,0x7000
0.71,0.19,0.87,0.10,0.10,0.18,0.28,0.19,0x7000
0.73,0.18,0.67,0.10,0.11,0.37,0.12,0.18,0x7000
0.84,0.16,0.87,0.11,0.10,0.35,0.17,0.11,0x7000
0.79,0.16,0.85,0.10,0.10,0.34,0.10,0.17,0x7000
0.70,0.14,0.75,0.11,0.11,0.32,0.22,0.10,0x7000
0.74,0.15,0.65,0.10,0.11,0.23,0.14,0.16,0x7000
0.85,0.13,0.78,0.11,0.10,0.21,0.26,0.19,0x7000
0.73,0.12,0.74,0.10,0.10,0.23,0.11,0.16,0x7000
0.77,0.13,0.64,0.11,0.10,0.14,0.16,0.19,0x7000
0.85,0.12,0.60,0.10,0.10,0.16,0.28,0.16,0x7000
0.76,0.10,0.80,0.10,0.11,0.37,0.13,0.19,0x7000
0.71,0.10,0.78,0.10,0.11,0.13,0.26,0.15,0x7000
0.82,0.18,0.68,0.11,0.10,0.34,0.18,0.18,0x7000
0.83,0.17,0.87,0.10,0.10,0.36,0.23,0.15,0x7000
0.74,0.15,0.77,0.11,0.10,0.34,0.15,0.18,0x7000
0.89,0.15,0.75,0.11,0.11,0.33,0.28,0.14,0x7000
0.80,0.13,0.65,0.10,0.10,0.24,0.20,0.10,0x7000
0.81,0.12,0.61,0.10,0.10,0.26,0.25,0.14,0x7000
0.72,0.13,0.81,0.11,0.10,0.24,0.17,0.10,0x7000
0.74,0.10,0.79,0.11,0.11,0.23,0.10,0.13,0x7000
0.78,0.11,0.69,0.10,0.11,0.14,0.15,0.19,0x7000
0.86,0.10,0.65,0.11,0.11,0.16,0.27,0.13,0x7000
0.70,0.18,0.78,0.10,0.10,0.14,0.12,0.19,0x7000
0.78,0.17,0.74,0.10,0.10,0.39,0.17,0.13,0x7000
0.89,0.18,0.64,0.11,0.10,0.37,0.29,0.19,0x7000
0.84,0.15,0.62,0.11,0.11,0.36,0.22,0.15,0x7000
```



0.75,0.13,0.82,0.10,0.10,0.34,0.14,0.18,0x7000  
0.76,0.15,0.78,0.11,0.11,0.36,0.19,0.12,0x7000  
0.87,0.13,0.68,0.10,0.10,0.27,0.11,0.18,0x7000  
0.82,0.13,0.66,0.11,0.11,0.26,0.24,0.14,0x7000  
0.73,0.11,0.86,0.10,0.11,0.24,0.16,0.17,0x7000  
0.74,0.10,0.75,0.11,0.11,0.26,0.21,0.14,0x7000  
0.85,0.18,0.65,0.10,0.10,0.17,0.26,0.17,0x7000  
0.87,0.18,0.63,0.10,0.10,0.16,0.19,0.13,0x7000  
0.88,0.17,0.89,0.11,0.10,0.18,0.11,0.17,0x7000  
0.79,0.15,0.79,0.11,0.11,0.16,0.16,0.13,0x7000  
0.83,0.16,0.69,0.10,0.11,0.37,0.28,0.16,0x7000  
0.71,0.15,0.65,0.11,0.11,0.39,0.13,0.13,0x7000  
0.86,0.12,0.63,0.11,0.10,0.38,0.26,0.19,0x7000  
0.77,0.13,0.83,0.10,0.10,0.36,0.18,0.12,0x7000  
0.78,0.12,0.72,0.11,0.10,0.38,0.23,0.19,0x7000  
0.89,0.10,0.62,0.11,0.11,0.29,0.15,0.12,0x7000  
0.84,0.10,0.60,0.11,0.10,0.28,0.28,0.18,0x7000  
0.75,0.18,0.80,0.10,0.10,0.26,0.20,0.11,0x7000  
0.83,0.17,0.76,0.11,0.10,0.28,0.25,0.18,0x7000  
0.87,0.18,0.66,0.10,0.10,0.19,0.10,0.11,0x7000  
0.75,0.17,0.62,0.11,0.10,0.21,0.22,0.18,0x7000  
0.86,0.15,0.75,0.11,0.11,0.12,0.27,0.11,0x7000  
0.81,0.15,0.80,0.11,0.10,0.18,0.20,0.17,0x7000  
0.82,0.14,0.69,0.10,0.11,0.20,0.25,0.11,0x7000  
0.73,0.12,0.89,0.11,0.10,0.11,0.17,0.17,0x7000  
0.84,0.13,0.79,0.10,0.11,0.39,0.29,0.10,0x7000  
0.79,0.10,0.77,0.10,0.11,0.38,0.22,0.16,0x7000  
0.70,0.18,0.67,0.11,0.10,0.29,0.14,0.12,0x7000  
0.71,0.10,0.63,0.11,0.10,0.31,0.19,0.16,0x7000  
0.82,0.18,0.83,0.10,0.10,0.29,0.11,0.12,0x7000  
0.73,0.16,0.73,0.11,0.11,0.20,0.16,0.15,0x7000  
0.84,0.17,0.86,0.10,0.11,0.18,0.28,0.11,0x7000  
0.79,0.14,0.61,0.10,0.10,0.17,0.21,0.14,0x7000  
0.70,0.12,0.74,0.11,0.11,0.15,0.13,0.10,0x7000  
0.81,0.13,0.64,0.10,0.11,0.36,0.25,0.13,0x7000  
0.82,0.12,0.60,0.10,0.11,0.38,0.10,0.10,0x7000  
0.73,0.10,0.80,0.11,0.10,0.29,0.15,0.13,0x7000  
0.84,0.11,0.70,0.10,0.10,0.27,0.27,0.19,0x7000  
0.79,0.18,0.68,0.10,0.11,0.26,0.20,0.15,0x7000  
0.70,0.19,0.88,0.11,0.10,0.24,0.12,0.18,0x7000  
0.81,0.17,0.78,0.10,0.10,0.15,0.24,0.14,0x7000  
0.82,0.16,0.67,0.11,0.10,0.17,0.29,0.18,0x7000  
0.73,0.14,0.87,0.11,0.10,0.38,0.21,0.11,0x7000  
0.74,0.16,0.83,0.10,0.10,0.10,0.26,0.18,0x7000  
0.79,0.12,0.75,0.10,0.10,0.35,0.19,0.13,0x7000  
0.70,0.13,0.65,0.11,0.10,0.33,0.11,0.16,0x7000  
0.71,0.12,0.61,0.10,0.10,0.35,0.16,0.13,0x7000  
0.82,0.10,0.81,0.11,0.11,0.26,0.28,0.16,0x7000  
0.73,0.18,0.71,0.10,0.11,0.24,0.20,0.12,0x7000  
0.74,0.10,0.60,0.10,0.11,0.19,0.25,0.16,0x7000  
0.89,0.17,0.88,0.10,0.10,0.25,0.18,0.12,0x7000  
0.76,0.16,0.70,0.10,0.10,0.38,0.22,0.15,0x7000  
0.74,0.15,0.62,0.10,0.10,0.33,0.22,0.10,0x7000  
0.82,0.14,0.88,0.11,0.10,0.35,0.27,0.14,0x7000  
0.76,0.14,0.72,0.10,0.11,0.19,0.19,0.11,0x7001  
0.87,0.12,0.62,0.11,0.11,0.10,0.11,0.18,0x7001  
0.88,0.11,0.88,0.10,0.11,0.12,0.16,0.15,0x7001  
0.79,0.12,0.71,0.11,0.10,0.13,0.28,0.10,0x7001  
0.70,0.10,0.61,0.11,0.10,0.11,0.20,0.10,0x7001  
0.85,0.17,0.66,0.11,0.11,0.10,0.13,0.16,0x7001  
0.76,0.18,0.79,0.10,0.10,0.18,0.18,0.11,0x7001  
0.77,0.17,0.75,0.11,0.11,0.13,0.10,0.15,0x7001  
0.88,0.15,0.65,0.10,0.10,0.11,0.15,0.15,0x7001  
0.89,0.14,0.61,0.11,0.10,0.13,0.20,0.19,0x7001  
0.80,0.15,0.81,0.10,0.10,0.14,0.12,0.19,0x7001  
0.82,0.12,0.79,0.11,0.11,0.10,0.25,0.13,0x7001

0.86,0.13,0.69,0.10,0.10,0.11,0.17,0.20,0x7001  
0.74,0.12,0.88,0.11,0.11,0.13,0.22,0.17,0x7001  
0.85,0.10,0.78,0.10,0.10,0.14,0.14,0.12,0x7001  
0.80,0.10,0.76,0.10,0.11,0.10,0.27,0.18,0x7001  
0.71,0.18,0.66,0.11,0.11,0.11,0.19,0.13,0x7001  
0.72,0.17,0.62,0.11,0.11,0.13,0.24,0.10,0x7001  
0.83,0.18,0.82,0.10,0.10,0.14,0.29,0.17,0x7001  
0.74,0.16,0.72,0.11,0.10,0.12,0.21,0.17,0x7001  
0.89,0.13,0.70,0.11,0.11,0.11,0.14,0.11,0x7001  
0.80,0.14,0.60,0.10,0.10,0.19,0.26,0.18,0x7001  
0.71,0.12,0.80,0.11,0.10,0.10,0.18,0.18,0x7001  
0.72,0.11,0.69,0.10,0.10,0.12,0.23,0.10,0x7001  
0.83,0.12,0.89,0.10,0.11,0.10,0.15,0.10,0x7001  
0.74,0.10,0.79,0.11,0.11,0.11,0.20,0.17,0x7001  
0.75,0.19,0.75,0.10,0.11,0.13,0.25,0.21,0x7001  
0.80,0.18,0.67,0.10,0.11,0.18,0.25,0.18,0x7001  
0.71,0.16,0.87,0.11,0.11,0.19,0.17,0.18,0x7001  
0.72,0.15,0.83,0.10,0.11,0.11,0.22,0.10,0x7001  
0.83,0.16,0.66,0.11,0.11,0.19,0.14,0.10,0x7001  
0.87,0.14,0.86,0.10,0.10,0.10,0.19,0.17,0x7001  
0.75,0.13,0.82,0.10,0.10,0.12,0.24,0.14,0x7001  
0.86,0.14,0.72,0.11,0.10,0.13,0.16,0.21,0x7001  
0.81,0.11,0.70,0.11,0.11,0.19,0.29,0.15,0x7001  
0.72,0.19,0.60,0.10,0.10,0.10,0.21,0.10,0x7001  
0.73,0.11,0.86,0.11,0.11,0.12,0.26,0.19,0x7001  
0.84,0.19,0.69,0.10,0.10,0.10,0.18,0.14,0x7001  
0.75,0.17,0.89,0.10,0.11,0.11,0.10,0.14,0x7001  
0.70,0.17,0.64,0.10,0.11,0.10,0.23,0.20,0x7001  
0.81,0.15,0.77,0.11,0.10,0.18,0.28,0.15,0x7001  
0.72,0.13,0.67,0.10,0.10,0.19,0.20,0.15,0x7001  
0.76,0.14,0.87,0.11,0.11,0.17,0.12,0.10,0x7001  
0.87,0.12,0.77,0.10,0.10,0.18,0.24,0.10,0x7001  
0.75,0.11,0.73,0.11,0.11,0.10,0.29,0.14,0x7001  
0.79,0.12,0.63,0.10,0.10,0.18,0.14,0.21,0x7001  
0.70,0.10,0.76,0.10,0.10,0.19,0.26,0.21,0x7001  
0.72,0.17,0.81,0.10,0.11,0.15,0.19,0.15,0x7001  
0.76,0.18,0.64,0.11,0.10,0.16,0.11,0.10,0x7001  
0.84,0.17,0.60,0.10,0.10,0.18,0.16,0.19,0x7001  
0.75,0.15,0.80,0.11,0.10,0.19,0.28,0.14,0x7001  
0.79,0.16,0.70,0.10,0.11,0.17,0.13,0.14,0x7001  
0.70,0.14,0.60,0.11,0.11,0.18,0.25,0.21,0x7001  
0.72,0.14,0.88,0.10,0.10,0.14,0.18,0.15,0x7001  
0.76,0.12,0.78,0.11,0.11,0.15,0.10,0.10,0x7001  
0.84,0.11,0.74,0.10,0.10,0.17,0.15,0.19,0x7001  
0.88,0.19,0.87,0.11,0.11,0.18,0.27,0.14,0x7001  
0.79,0.10,0.77,0.10,0.10,0.16,0.19,0.14,0x7001  
0.81,0.17,0.75,0.10,0.10,0.15,0.12,0.20,0x7001  
0.85,0.18,0.65,0.11,0.11,0.13,0.17,0.15,0x7001  
0.73,0.17,0.61,0.11,0.11,0.15,0.22,0.12,0x7001  
0.84,0.15,0.81,0.10,0.11,0.16,0.14,0.19,0x7001  
0.88,0.16,0.71,0.11,0.10,0.14,0.26,0.19,0x7001  
0.76,0.15,0.60,0.10,0.10,0.19,0.11,0.11,0x7001  
0.81,0.11,0.89,0.10,0.11,0.14,0.11,0.20,0x7001  
0.82,0.13,0.78,0.11,0.11,0.16,0.16,0.12,0x7001  
0.73,0.11,0.68,0.10,0.10,0.14,0.28,0.12,0x7001  
0.74,0.10,0.64,0.10,0.11,0.19,0.13,0.16,0x7001  
0.85,0.18,0.84,0.11,0.10,0.17,0.18,0.16,0x7001  
0.80,0.18,0.82,0.11,0.11,0.16,0.11,0.17,0x7001  
0.71,0.16,0.72,0.10,0.11,0.14,0.23,0.17,0x7001  
0.72,0.15,0.68,0.11,0.11,0.16,0.28,0.21,0x7001  
0.83,0.16,0.88,0.10,0.10,0.17,0.20,0.21,0x7001  
0.78,0.13,0.86,0.11,0.10,0.16,0.13,0.15,0x7001  
0.89,0.14,0.76,0.10,0.11,0.14,0.25,0.10,0x7001  
0.77,0.13,0.65,0.11,0.11,0.16,0.10,0.19,0x7001  
0.81,0.11,0.85,0.10,0.11,0.17,0.22,0.14,0x7001  
0.72,0.19,0.75,0.11,0.10,0.15,0.27,0.14,0x7001

```

0.74,0.19,0.73,0.11,0.11,0.14,0.27,0.15,0x7001
0.78,0.17,0.63,0.10,0.11,0.12,0.12,0.15,0x7001
0.86,0.16,0.89,0.10,0.11,0.17,0.17,0.19,0x7001
0.87,0.18,0.78,0.11,0.11,0.19,0.29,0.16,0x7001
0.78,0.16,0.68,0.10,0.11,0.17,0.14,0.11,0x7001
0.73,0.16,0.73,0.10,0.10,0.16,0.27,0.17,0x7001
0.84,0.14,0.86,0.11,0.11,0.17,0.19,0.17,0x7001
0.75,0.12,0.76,0.10,0.11,0.15,0.11,0.12,0x7001
0.76,0.11,0.72,0.10,0.11,0.17,0.16,0.16,0x7001
0.71,0.11,0.70,0.10,0.10,0.16,0.29,0.10,0x7001
0.82,0.19,0.60,0.11,0.10,0.17,0.21,0.10,0x7001
0.73,0.10,0.80,0.10,0.11,0.15,0.26,0.17,0x7001
0.74,0.19,0.76,0.11,0.11,0.17,0.18,0.14,0x7001
0.85,0.17,0.89,0.10,0.11,0.18,0.23,0.21,0x7001
0.73,0.16,0.85,0.10,0.11,0.10,0.28,0.18,0x7001
0.88,0.16,0.83,0.10,0.10,0.19,0.21,0.19,0x7001
0.82,0.13,0.67,0.11,0.11,0.13,0.20,0.10,0x7001
0.70,0.12,0.63,0.10,0.11,0.18,0.25,0.14,0x7001
0.74,0.13,0.83,0.11,0.10,0.16,0.17,0.14,0x7001
0.82,0.12,0.79,0.10,0.11,0.18,0.22,0.18,0x7001
0.86,0.10,0.69,0.11,0.10,0.19,0.27,0.18,0x7001
0.88,0.10,0.67,0.10,0.11,0.18,0.27,0.12,0x7001
0.1,0.0,0.1,0.1,0.1,0.0,0.0,0.1,0x7003
0.1,0.1,0.0,0.0,0.0,0.0,0.0,0.1,0x7003
0.1,0.0,0.0,0.1,0.0,0.1,0.1,0.1,0x7003
0.1,0.1,0.0,0.0,0.1,0.1,0.1,0.0,0x7003
0.0,0.0,0.0,0.1,0.1,0.0,0.1,0.0,0x7003
0.0,0.0,0.0,0.1,0.0,0.0,0.0,0.0,0x7003
0.1,0.0,0.0,0.1,0.0,0.1,0.1,0.1,0x7003
0.0,0.1,0.0,0.0,0.0,0.1,0.0,0.0,0x7003
0.1,0.0,0.1,0.1,0.0,0.0,0.0,0.1,0x7003
0.0,0.1,0.0,0.1,0.1,0.0,0.1,0.1,0x7003

```

## B.2 Traffic Test Dataset

```
@relation discriminate
```

```

@attribute 'request'    numeric
@attribute 'pow'        numeric
@attribute 'freq'       numeric
@attribute 'auth'       numeric
@attribute 'heatmap'    numeric
@attribute 'banner'     numeric
@attribute 'screen'     numeric
@attribute 'fp'         numeric
@attribute 'class'      {0x7000, 0x7001, 0x7003}

```

```
@data
```

```

0.38,0.43,0.12,0.40,0.50,0.51,0.40,0.48,?
0.21,0.58,0.11,0.47,0.54,0.68,0.44,0.40,?
0.13,0.49,0.13,0.43,0.53,0.64,0.57,0.49,?
0.12,0.53,0.15,0.52,0.52,0.60,0.57,0.45,?
0.11,0.57,0.17,0.48,0.51,0.56,0.50,0.41,?
0.17,0.50,0.13,0.46,0.56,0.58,0.52,0.43,?
0.37,0.41,0.11,0.53,0.54,0.64,0.48,0.44,?
0.36,0.45,0.13,0.49,0.53,0.60,0.48,0.43,?
0.12,0.58,0.19,0.47,0.51,0.62,0.43,0.45,?
0.32,0.49,0.17,0.54,0.69,0.61,0.46,0.46,?
0.81,0.13,0.69,0.11,0.10,0.17,0.16,0.20,?
0.87,0.16,0.85,0.11,0.11,0.16,0.11,0.14,?
0.89,0.17,0.87,0.11,0.10,0.15,0.24,0.13,?
0.82,0.10,0.73,0.11,0.11,0.14,0.19,0.19,?
0.82,0.11,0.61,0.10,0.11,0.13,0.22,0.10,?
0.84,0.15,0.63,0.10,0.10,0.19,0.22,0.16,?

```

### B.3 Banner Training Dataset

@relation swing

```
@attribute 'xy1'    numeric
@attribute 'xy2'    numeric
@attribute 'xy3'    numeric
@attribute 'xy4'    numeric
@attribute 'xy5'    numeric
@attribute 'xy6'    numeric
@attribute 'xy7'    numeric
@attribute 'xy8'    numeric
@attribute 'xy9'    numeric
@attribute 'xy10'   numeric
@attribute 'class' {true, false}
```

@data

```
33166, 33166, 33066, 33066, 32966, 32966, 32967, 32967, 32867, 32867, true
4232569, 32369, 32369, 32269, 32269, 32169, 32169, 32169, 31969, 31969, true
4232071, 32071, 31971, 31971, 31771, 31771, 31671, 31671, 31571, 31571, true
4233579, 33579, 33079, 33079, 32878, 32878, 32678, 32678, 32678, 32678, true
4232878, 32878, 32778, 32778, 32678, 32678, 32278, 32278, 32078, 32078, true
4231779, 31779, 31679, 31679, 31579, 31579, 31479, 31479, 31279, 31279, true
4233584, 33584, 33384, 33384, 33284, 33284, 33184, 33184, 33084, 33084, true
4232987, 32987, 32786, 32786, 32686, 32686, 32486, 32486, 32385, 32385, true
4231684, 31684, 31784, 31784, 31884, 31884, 32084, 32084, 32084, 32084, true
4231581, 31581, 31481, 31481, 31482, 31482, 31382, 31382, 31185, 31185, true
4231472, 31472, 31473, 31473, 31475, 31475, 31476, 31476, 31377, 31377, true
4233082, 33082, 33082, 33082, 32981, 32981, 32780, 32780, 32680, 32680, true
4233077, 33077, 32977, 32977, 32976, 32976, 32775, 32775, 32575, 32575, true
4233380, 33380, 33280, 33280, 33180, 33180, 33079, 33079, 32979, 32979, true
4232889, 32889, 32789, 32789, 32788, 32788, 32687, 32687, 32686, 32686, true
4232389, 32389, 32389, 32389, 32288, 32288, 32188, 32188, 32087, 32087, true
4231790, 31790, 31488, 31488, 31087, 31087, 30785, 30785, 30784, 30784, true
4231387, 31387, 31386, 31386, 31385, 31385, 31384, 31384, 31383, 31383, true
4231782, 31782, 31681, 31681, 31581, 31581, 31580, 31580, 31479, 31479, true
4230976, 30976, 30975, 30975, 31175, 31175, 31275, 31275, 31274, 31274, true
4232277, 32277, 32276, 32276, 32075, 32075, 31974, 31974, 31973, 31973, true
4233089, 33089, 33087, 33087, 33086, 33086, 33085, 33085, 33084, 33084, true
4233480, 33379, 33379, 33278, 33278, 33277, 33277, 33077, 33077, 32977, true
4232971, 32971, 32973, 32973, 32874, 32874, 32874, 32874, 32875, 32875, true
4232682, 32682, 32681, 32681, 32581, 32581, 32581, 32581, 32481, 32481, true
4232681, 32681, 32481, 32481, 32382, 32382, 32284, 32284, 31887, 31887, true
4232584, 32584, 32583, 32583, 32483, 32483, 32383, 32383, 32183, 32183, true
4232181, 32181, 32178, 32178, 32176, 32176, 32175, 32175, 32174, 32174, true
4232479, 32479, 32478, 32478, 32477, 32477, 32376, 32376, 32375, 32375, true
4232775, 32775, 32675, 32675, 32575, 32575, 32574, 32574, 32474, 32474, true
4231485, 31485, 31385, 31385, 31285, 31285, 31184, 31184, 31184, 31184, true
4231286, 31286, 31282, 31282, 31281, 31281, 31280, 31280, 31279, 31279, true
4231480, 31480, 31479, 31479, 31477, 31477, 31376, 31376, 31375, 31375, true
4231475, 31475, 31374, 31374, 31374, 31374, 31273, 31273, 31272, 31272, true
4231572, 31572, 31671, 31671, 31770, 31770, 31870, 31870, 32070, 32070, true
4232780, 32780, 32778, 32778, 32778, 32778, 32577, 32577, 32575, 32575, true
4232391, 32391, 32491, 32491, 32490, 32490, 32690, 32690, 32689, 32689, true
4233085, 33085, 33084, 33084, 33083, 33083, 33181, 33181, 33180, 33180, true
4233177, 33177, 33077, 33077, 32977, 32977, 32876, 32876, 32875, 32875, true
4231590, 31590, 31690, 31690, 31689, 31689, 31789, 31789, 31887, 31887, true
4231486, 31486, 31783, 31783, 31882, 31882, 32082, 32082, 32081, 32081, true
4231475, 31475, 31575, 31575, 31775, 31775, 31875, 31875, 31975, 31975, true
4231674, 31674, 31773, 31773, 31873, 31873, 31972, 31972, 32072, 32072, true
4231887, 31887, 31885, 31885, 31883, 31883, 31882, 31882, 31782, 31782, true
4231775, 31775, 31576, 31576, 31376, 31376, 31377, 31377, 31278, 31278, true
4231279, 31279, 31376, 31376, 31374, 31374, 31373, 31373, 31473, 31473, true
4230968, 30968, 30970, 30970, 31070, 31170, 31270, 31270, 31370, 31370, true
```



42332181, 332181, 323180, 323180, 317180, 317180, 310179, 310179, 308178, 308178, false  
 42199182, 199182, 194177, 194177, 192173, 192173, 192168, 192168, 192167, 192167, false  
 4250174, 42172, 42172, 32171, 32171, 27170, 27170, 28170, 30170, 31170, false  
 42335254, 335254, 334251, 334251, 327249, 327249, 313245, 313245, 307242, 307242, false  
 42186203, 181199, 181199, 177197, 177197, 176195, 176195, 175195, 175195, 174195, false  
 42166251, 166251, 165250, 165250, 163248, 163248, 162247, 162247, 161246, 161246, false  
 4263252, 63252, 61245, 61245, 61241, 61241, 61239, 61239, 60239, 60239, false  
 42193111, 193111, 167100, 167100, 15592, 15592, 14786, 14786, 14483, 14483, false  
 42330124, 330124, 324123, 324123, 321123, 321123, 317121, 317121, 311119, 311119, false  
 42312188, 312188, 314181, 314181, 314173, 314173, 314167, 314167, 314164, 314164, false

## B.4 Banner Test Dataset

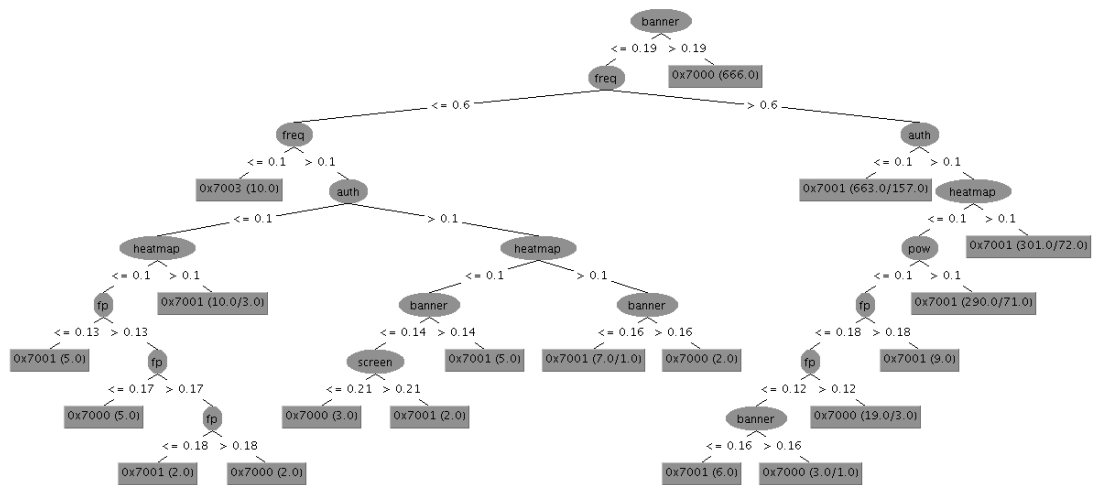
@relation swing

```
@attribute 'xy1'    numeric
@attribute 'xy2'    numeric
@attribute 'xy3'    numeric
@attribute 'xy4'    numeric
@attribute 'xy5'    numeric
@attribute 'xy6'    numeric
@attribute 'xy7'    numeric
@attribute 'xy8'    numeric
@attribute 'xy9'    numeric
@attribute 'xy10'   numeric
@attribute 'class'  {true, false}
```

@data

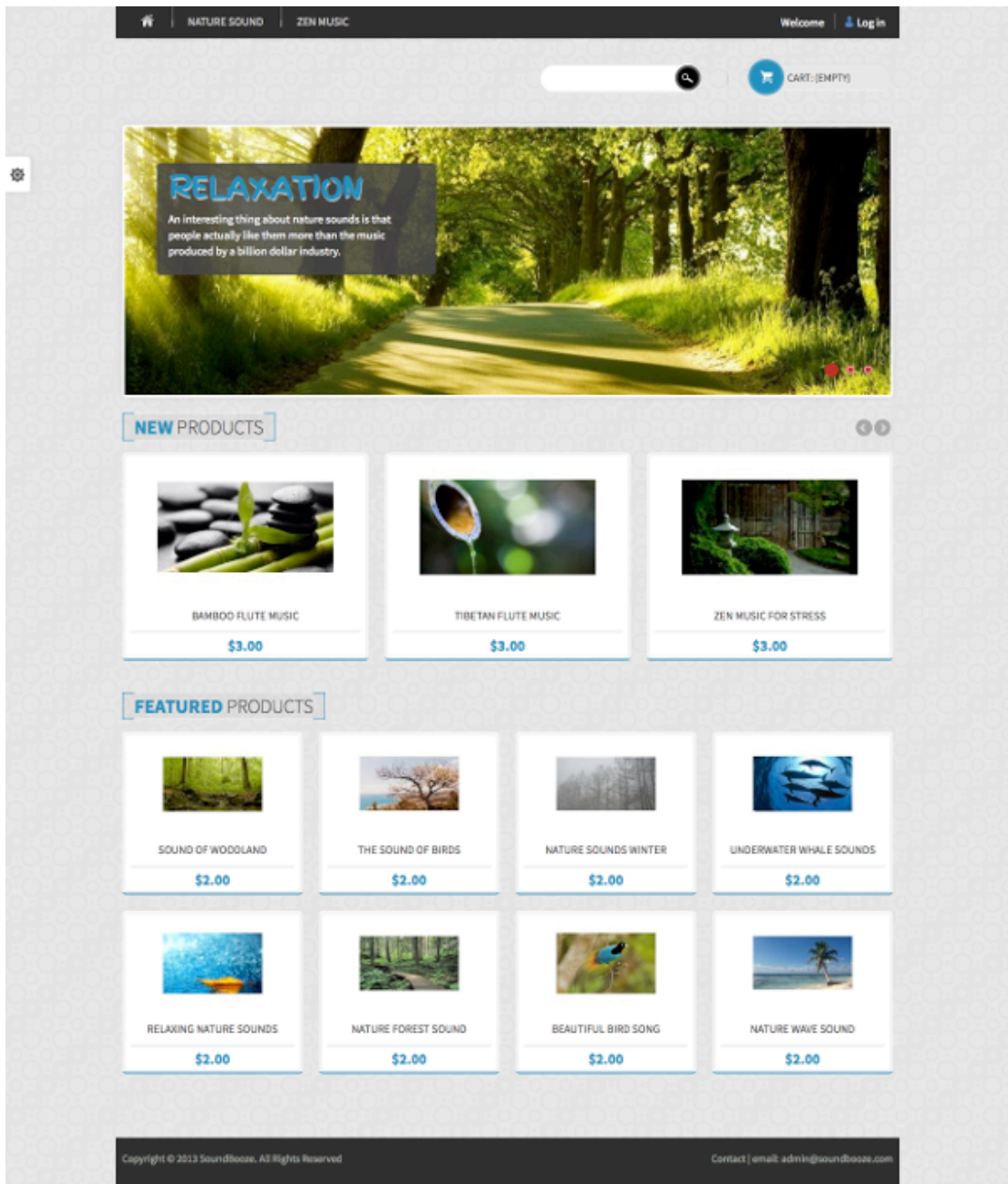
31481, 31481, 31381, 31381, 31281, 31281, 31183, 31183, 31084, 31084, ?  
 31676, 31676, 32077, 32077, 32278, 32278, 32279, 32279, 32079, 32079, ?  
 32779, 32779, 32579, 32579, 32479, 32479, 32379, 32379, 32279, 32279, ?  
 32779, 32779, 32579, 32579, 32479, 32479, 32379, 32379, 32279, 32279, ?  
 32476, 32476, 32176, 32176, 31876, 31876, 31776, 31776, 31677, 31677, ?  
 32083, 32083, 32082, 32082, 32081, 32081, 32080, 32080, 32078, 32078, ?  
 31096, 31096, 31194, 31194, 31488, 31488, 31685, 31685, 31783, 31783, ?  
 33073, 33073, 32973, 32973, 32873, 32873, 32773, 32773, 32673, 32673, ?  
 31276, 31276, 31376, 31376, 31477, 31477, 31577, 31577, 31677, 31677, ?  
 33082, 33082, 32982, 32982, 32882, 32882, 32681, 32681, 32581, 32581, ?

## B.5 Tree Visualizer for Traffic Training Dataset



# Appendix C – Website Pages

## C.1 Landing Page



## C.2 Product Detail Page

The screenshot shows a product detail page for 'Bamboo Flute Music'. The page layout includes a top navigation bar with 'NATURE SOUND' and 'ZEN MUSIC' menus, a search bar, and a shopping cart icon labeled 'CART: [EMPTY]'. A breadcrumb trail indicates the path: Home > Zen Music > Bamboo Flute Music. On the left, a 'CATEGORIES' sidebar shows 'Nature Sound' and 'Zen Music' (the active category). The main content area features the product title 'Bamboo Flute Music', a description 'Music ideal for meditation, healing and relaxation.', 'Format: MP3', and 'Duration: 60 minutes'. A note states 'This product is only available for digital download.' Below this is a 'More details' button, the price '\$3.00', and an 'ADD TO CART' button. A '1000 items in stock' indicator is present. A 'Share on Facebook!' link is located below a product image of bamboo flutes and stones. At the bottom, there are 'More info' and 'Comments' tabs, and a description: 'Music ideal for meditation, healing and relaxation.' The footer contains copyright information for SoundBooze and a contact email.

## C.3 Login Page

The screenshot shows a login page with a top navigation bar and a search bar. A breadcrumb trail indicates the path: Home > Login. On the left, a 'CATEGORIES' sidebar shows 'Nature Sound' and 'Zen Music'. The main content area is titled 'LOG IN' and contains two sections: 'CREATE AN ACCOUNT' and 'ALREADY REGISTERED?'. The 'CREATE AN ACCOUNT' section has a heading 'Enter your e-mail address to create an account.', an 'E-mail address' input field, and a 'Create an account' button. The 'ALREADY REGISTERED?' section has 'E-mail address' and 'Password' input fields, a 'Forgot your password?' link, and a 'Log In' button. The footer contains copyright information for SoundBooze and a contact email.



## C.4 Category Page

The screenshot shows a website category page for "Nature Sound". At the top, there is a navigation bar with "NATURE SOUND" and "ZEN MUSIC" links, and a "Welcome" message with a "Log in" button. Below the navigation bar is a search bar and a "CART: [EMPTY]" button. The main content area features a "CATEGORIES" sidebar with "Nature Sound" and "Zen Music" options. The "Nature Sound" category is selected, displaying a featured image of a field with a haystack and a "More" link. Below this, there is a "Nature Sound" section with a "There are 12 products" indicator and a "COMPARE" button. The products are displayed in a grid of 12 items, each with a thumbnail image, a title, and a price of \$2.00. The products are: SOUND OF WOODLAND, THE SOUND OF BIRDS, NATURE SOUNDS WINTER, UNDERWATER WHALE SOUNDS, RELAXING NATURE SOUNDS, NATURE FOREST SOUND, BEAUTIFUL BIRD SONG, NATURE WAVE SOUND, SOUND OF BUBBLES, CRICKETS NATURE SOUND, NATURE SNOWSTORM RIVER, and SOFT RELAXING RAIN. At the bottom of the page, there is a footer with copyright information and contact details.

Home » Nature Sound

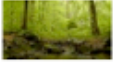

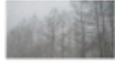









CATEGORIES

- Nature Sound
- Zen Music

Nature sound ideal for meditation, relaxation and to fight sleep disorders, anxiety and depression.

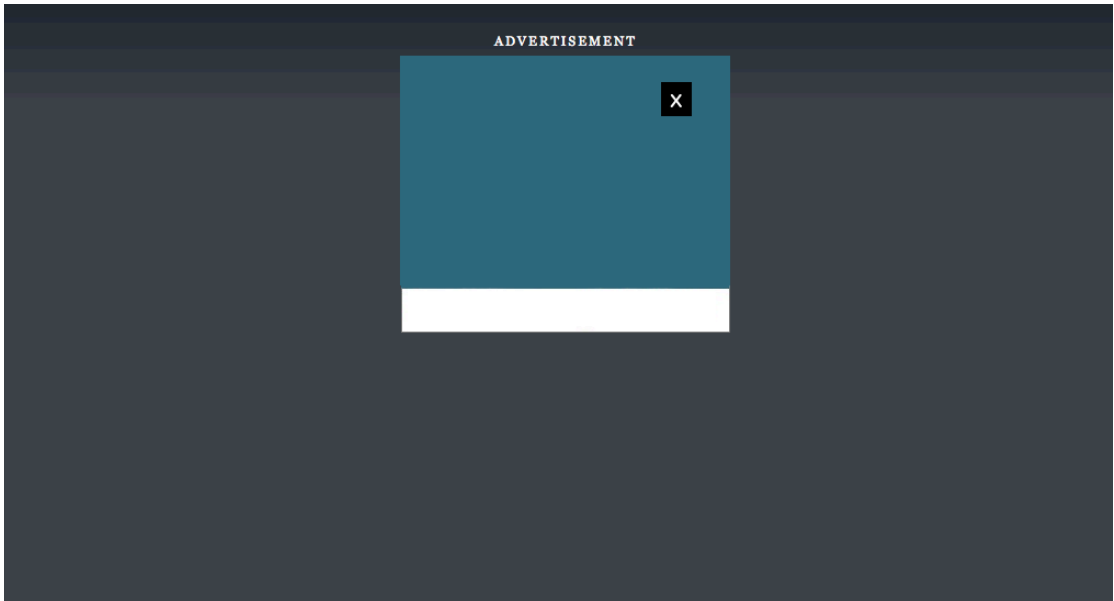
More »

**Nature Sound** There are 12 products. COMPARE

 SOUND OF WOODLAND \$2.00	 THE SOUND OF BIRDS \$2.00	 NATURE SOUNDS WINTER \$2.00
 UNDERWATER WHALE SOUNDS \$2.00	 RELAXING NATURE SOUNDS \$2.00	 NATURE FOREST SOUND \$2.00
 BEAUTIFUL BIRD SONG \$2.00	 NATURE WAVE SOUND \$2.00	 SOUND OF BUBBLES \$2.00
 CRICKETS NATURE SOUND \$2.00	 NATURE SNOWSTORM RIVER \$2.00	 SOFT RELAXING RAIN \$2.00

Copyright © 2013 SoundBooze. All Rights Reserved. Contact | email: admin@soundbooze.com

## C.5 Advertisement Page



## Appendix D – Attack Simulation CLI

### D.1 Slowloris

```
$ ./slowloris.pl -dns $url -port 80 -timeout 90 -num 500 -tcpto 5 -httpready
```

### D.2 Slowhttptest

```
$ ./slowhttptest -c 1000 -X -g -o slow_read_stats -r 1000 -w 1 -y 2 -n 5 -z 32  
-k 3 -u $url -p 3 -l 300
```

### D.3 R-u-d-y

```
$ ./r-u-dead-yet.py $url
```

[parameters]

URL: http://www.soundbooze.com/path-to-post-url.php

number\_of\_connections: 500

attack\_parameter: login

proxy\_addr: ""

proxy\_port: 0

### D.4 THC SSL DOS

```
$ ./thc-ssl-dos $ip $port
```

### D.5 ApacheKiller

```
$ ./killapache.pl $url $numforks
```

### D.6 PhantomJS

```
var page = require('webpage').create();  
  
page.onConsoleMessage = function(msg) {  
    console.log('page message: ' + msg);  
};  
  
page.open("http://soundbooze.com", function(status) {  
    if (status !== "success") {  
        console.error("error loading test page " + testHtml);  
        phantom.exit();  
    }  
    var i = 0;  
    var interval = setInterval(function() {  
        if (i === 10) {  
            clearInterval(interval);  
            phantom.exit();  
        }  
        var x, y;  
        x = y = i * 10;  
        console.log('sending mousemove event to ' + x + ', ' + y);  
        page.sendEvent('mousemove', i * 10, i * 10);  
        page.sendEvent('click', 100,100);  
        i++;  
    }, 500);  
});
```

## D.7 Slowloris Output

Welcome to Slowloris - the low bandwidth, yet greedy and poisonous HTTP client  
Multithreading enabled.

Connecting to soundbooze.com:80 every 90 seconds with 500 sockets:

```
Building sockets.  
Building sockets.  
Building sockets.  
Building sockets.  
Building sockets.  
Building sockets.  
Building sockets.  
Building sockets.  
Building sockets.  
Building sockets.  
Building sockets.  
Sending data.
```

Current stats: Slowloris has now sent 1193 packets successfully.  
This thread now sleeping for 90 seconds...

```
Sending data.
```

Current stats: Slowloris has now sent 1219 packets successfully.  
This thread now sleeping for 90 seconds...

## D.8 Slowhttptest Output

Using:

```
test type:                SLOW READ  
number of connections:    1000  
URL:                      http://www.soundbooze.com/  
verb:                     GET  
receive window range:    1 - 2  
pipeline factor:          3  
read rate from receive buffer: 32 bytes / 5 sec  
connections per seconds:  1000  
probe connection timeout: 3 seconds  
test duration:            300 seconds
```

Tue Apr 8 01:34:47 2014:slow HTTP test status on 0th second:

```
initializing: 0  
pending:      1  
connected:   0  
error:       0  
closed:      0  
service available: YES
```

Tue Apr 8 01:34:48 2014:slow HTTP test status on 30th second:

```
initializing: 0  
pending:      8  
connected:   857  
error:       0  
closed:     135  
service available: YES
```

Tue Apr 8 01:34:53 2014:slow HTTP test status on 5th second:

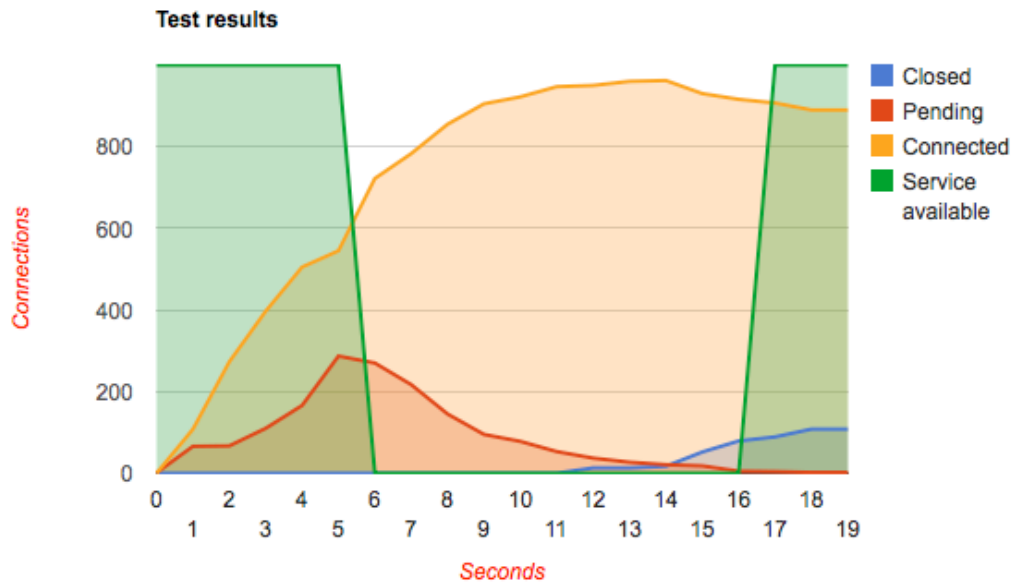
```
initializing: 0  
pending:     237  
connected:  617  
error:      0  
closed:     0  
service available: NO
```

Tue Apr 8 01:34:53 2014:slow HTTP test status on 35th second:

```
initializing: 0  
pending:      8  
connected:   857  
error:       0  
closed:     135  
service available: NO
```

### Test parameters

<b>Test type</b>	SLOW READ
<b>Number of connections</b>	1000
<b>Receive window range</b>	1 - 2
<b>Pipeline factor</b>	3
<b>Read rate from receive buffer</b>	32 bytes / 5 sec
<b>Connections per seconds</b>	200
<b>Timeout for probe connection</b>	3
<b>Target test duration</b>	300 seconds



### D.9 PhantomJS Output

```
sending mousemove event to 0, 0
page message:
sending mousemove event to 10, 10
page message:
sending mousemove event to 20, 20
page message:
sending mousemove event to 30, 30
page message:
sending mousemove event to 40, 40
page message:
sending mousemove event to 50, 50
page message:
sending mousemove event to 60, 60
page message:
sending mousemove event to 70, 70
page message:
sending mousemove event to 80, 80
page message:
sending mousemove event to 90, 90
page message:
sending mouseclick event at 90, 90
```

## D.10 THC SSL DOS Output

```
Handshakes 0 [0.00 h/s], 1 Conn, 0 Err
Handshakes 0 [0.00 h/s], 3 Conn, 0 Err
Handshakes 8 [7.71 h/s], 9 Conn, 0 Err
Handshakes 33 [26.02 h/s], 15 Conn, 0 Err
Handshakes 69 [35.78 h/s], 20 Conn, 0 Err
Handshakes 123 [54.53 h/s], 26 Conn, 0 Err
Handshakes 176 [49.37 h/s], 30 Conn, 0 Err
Handshakes 205 [31.41 h/s], 30 Conn, 0 Err
Handshakes 258 [52.97 h/s], 35 Conn, 0 Err
Handshakes 329 [70.33 h/s], 40 Conn, 0 Err
Handshakes 408 [79.60 h/s], 45 Conn, 0 Err
Handshakes 500 [92.21 h/s], 51 Conn, 0 Err
Handshakes 612 [112.06 h/s], 56 Conn, 0 Err
Handshakes 746 [132.82 h/s], 61 Conn, 0 Err
Handshakes 877 [131.44 h/s], 66 Conn, 0 Err
Handshakes 1013 [136.66 h/s], 70 Conn, 0 Err
Handshakes 1148 [134.85 h/s], 70 Conn, 0 Err
Handshakes 1280 [132.08 h/s], 73 Conn, 0 Err
Handshakes 1423 [142.14 h/s], 73 Conn, 0 Err
Handshakes 1565 [142.07 h/s], 77 Conn, 0 Err
Handshakes 1686 [121.74 h/s], 82 Conn, 0 Err
Handshakes 1820 [133.09 h/s], 84 Conn, 0 Err
Handshakes 1948 [128.68 h/s], 86 Conn, 0 Err
```

## D.11 ApacheKiller Output

```
host seems vuln
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
ATTACKING soundbooze.com [using 50 forks]
:pPpPpppPpPPppPpppPp
```

## D.12 R-u-d-y Output

```
[!] Using configuration file
[!] Attacking: http://www.soundbooze.com/
[!] With parameter: search
```

## D.13 Apache Benchmark Output

```
musicalware:~ root# ab -n 1000 -c 64 http://www.soundbooze.com/
```

```
This is ApacheBench, Version 2.3 <$Revision: 655654 $>  
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/  
Licensed to The Apache Software Foundation, http://www.apache.org/
```

```
Benchmarking www.soundbooze.com (be patient)
```

```
Completed 100 requests  
Completed 200 requests  
Completed 300 requests  
Completed 400 requests  
Completed 500 requests  
Completed 600 requests  
Completed 700 requests  
Completed 800 requests  
Completed 900 requests  
Completed 1000 requests  
Finished 1000 requests
```

```
Server Software:      Apache  
Server Hostname:     www.soundbooze.com  
Server Port:         80
```

```
Document Path:       /  
Document Length:     0 bytes
```

```
Concurrency Level:    64  
Time taken for tests: 61.007 seconds  
Complete requests:   1000  
Failed requests:     0  
Write errors:        0  
Non-2xx responses:   1000  
Total transferred:   222000 bytes  
HTML transferred:    0 bytes  
Requests per second: 16.39 [#/sec] (mean)  
Time per request:    3904.428 [ms] (mean)  
Time per request:    61.007 [ms] (mean, across all concurrent requests)  
Transfer rate:       3.55 [Kbytes/sec] received
```

### Connection Times (ms)

	min	mean[+/-sd]	median	max
Connect:	150	160 64.3	153	1343
Processing:	243	3686 2406.4	3958	8798
Waiting:	243	3683 2407.7	3958	8798
Total:	394	3846 2407.9	4115	8951

### Percentage of the requests served within a certain time (ms)

50%	4115
66%	5454
75%	6183
80%	6434
90%	6928
95%	7215
98%	7518
99%	8023
100%	8951 (longest request)