Volume 4, No. 4, March-April 2013

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A Survey on Signature Generation Methods for Network Traffic Classification

Vinoth George C
Department of Computer Science and Engineering
Karunya University Coimbatore, India
vinothgeorge@karunya.edu.in

Vinodh Ewards
Department of Computer Science and Engineering
Karunya University Coimbatore, India
ewards@karunya.edu

Abstract: Traffic classification provides security to the network systems and provides basics for trusted network management. Network traffic classification is extensively required mainly for many network management tasks such as traffic policing and flow prioritization. To overcome the inaccuracy of prior protocol based traffic classification method payload based method is introduced which uses signatures for traffic classification. Signatures are generated by the session and application information of the packet. Various techniques have been developed to produce accurate and reliable signatures for traffic classification. This survey paper looks at those techniques of payload based signature generation and compares their accuracy with other schemes. Issues related to signature generation technique also discussed.

Keywords: traffic classification; signature; payload based method; application traffic; network

I. INTRODUCTION

Network traffic classification provides basic security for network systems A wide range of issues related to trusted network management depends on network traffic classification and application identification, which shows the process to discover what kind of applications are run by the users in a particular network flows. Based on the classified result of the network traffic, Quality of Service (QoS) can be deployed by the Internet service providers (ISPs). Through which network administrators can get the precise mappings on the application in the network traffic, in which trusted services will get higher priority and other services will be restricted. Earlier, Traffic classification has been done using port-based method for identifying internet applications. It inspects the TCP or UDP port numbers and identifies the application protocols according to the Internet Assigned Port Numbers authority (IANA) list of well-known ports and registered ports [1]. Due to violation of port number assignment by newly emerging applications this method is proved to be highly inaccurate.

For the accurate traffic classification payload based method is most widely applied in industry. Payload based method [9] uses signature for traffic classification which utilize stateful reconstruction of session and application information from each packets content and it is reliable and consumes moderate computational power. However, deriving accurate and efficient protocol signatures for various applications is a challenging task. Various techniques have been used to generate automated application signature and worm signature for efficient network traffic classification using payload based method. In this survey we are discussing about various port based techniques that generates automated signatures.

II. APPLICATION SIGNATURE GENERATION USING PROTOCOLS

Network traffic associated with different P2P applications has the broad range of network operations in which port based classification is highly inaccurate for some P2P applications. An efficient approach has been developed for identifying the P2P application traffic through application level signatures [5]. Application level signatures have been developed for number of popular P2P applications by designing a real-time classification system which operates on individual packets in the middle of the network. Signatures are identified by packet level traces and some available documentation. The generated application signatures can track the P2P traffic even on high speed network links.

Various P2P protocols such as Gnutella, eDonkey, BitTorrent, Kazaa which allow a random host to act as both a client and server to its peers from which HTTP response headers are downloaded. In the signature implementation the downloaded headers undergoes fixed and variable offset match with the payload of the P2P traffic [10]. To evaluate the signature based P2P classification Gigascope framework monitor is used which can perform variety of traffic measurement by automatically breaking the complex computation into multiple tasks for identifying byte count and number of packets in each direction. The experimental setup would be made by tracing the data sets using the Gigascope monitor and the data sets will be traced based on internet access and VPN. The generated traces will be allowing the classifier to be evaluated for its accuracy, robustness and scalability.

This signature generation method classifies the traffic three times as much traffic when compared to the port based method which have been used earlier and this scheme exhibits low false positive and false negative. Though this method produce accurate signature using protocol knowledge later, as more and more new applications use proprietary

protocols, this method will also become difficult to produce signature of high accuracy and reliability and thus the signature produced will be manually extracted by the network administrator thus time consuming and error prone.

III. AUTOMATED APPLICATION SIGNATURE GENERATION USING LASER

The approach that uses the protocol analysis for generating signatures are limited to identifying the security threatening traffic only and it will not consider the innocuous traffic. Signatures generated have been manually extracted so that such process cause a slow response time. A systematic LASER algorithm [2] has been proposed to generate automated application level signature for traffic classification. LASER is the LCS based Application Signature ExtRaction algorithm which automatically determine the pattern using the packet payload without any prior knowledge of protocol. This method evaluates the closeness of the signature with the pre-discovered signatures.

Signature generation process in this automated method consists of two parts:

A. Sanitized packet collection:

Raw packets from the network traffic have been dumped using Winpcap or Libpcap to collect the packet traces for every running process in the OS and the collected packets are divided according to each flow. The collected data set is a mixture of many different applications which produce too many garbage values and the uncertainty of the traffic can be removed before fed to the signature extraction algorithm.

B. Signature extraction:

For signature extraction LCS (Longest Common Subsequence) algorithm is used for finding common subsequences and then common substrings are extracted as signatures through LASER algorithm. LCS is normally used for DNA sequence matching. For finding subsequence in packets some constraints is modified in the LCS algorithm. LCS uses constraints like number of packets per flow, Minimum substring length and packet size comparison to find common strings from the packet trace. Signatures are refined by eliminating the trivial strings.

Popular P2P applications like Limewire, BitTorrent and Fileguri validate this LASER algorithm and it shows the generated signatures were close to the signature generated by previous method. The accuracy evaluation metrics of the LASER algorithm uses the false positive, false negative and the overall accuracy of the traffic that helps in finding the misclassification ratio and signature accuracy. To measure the accuracy of data sets Traffic Measurement Agents (TMA) is deployed on the selected hosts in the network to obtain the absolute ground truth traffic and its information about applications. TMA monitors the host's network interface and finds which process is responsible for transmission of traffic and generates traffic summary log for the interface which have been used later to measure the accuracy of LASER signatures.

The signature generated exhibit low false positive and low false negative and the overall accuracy will be high. This method focus on the string or hex patterns of the traffic payload and the LASER algorithm generates reliable signature with a small training data set which minimizing human intervention.

IV. AUTOMATED APPLICATION SIGNATURE GENERATION USING AUTOSIG

While comparing with the traditional method which classifies traffic using predefined well known port numbers, the method using application signatures are more accurate. Unfortunately signatures generated are derived manually and increasing in new applications that use proprietary protocol makes difficult to maintain up to date signatures for applications. To solve this issue AutoSig [15] method is proposed which is an automatic application signature generation system. AutoSig extracts common substring sequence from the traffic flow as signature.

AutoSig algorithm first divides content of flows into small fixed size blocks called shingles.

A. Common shingle selection:

AutoSig algorithm first divides content of flows into small fixed size blocks called shingles. The length of each shingle is K bytes. The i-th shingle corresponds to the content block containing payload bytes from offset i to offset i+K-1. For a byte sequence with n bytes, n-K+1 shingles are generated. But common substrings of applications are usually very short, so K should be small in application signature generation. It is easy to generate noise shingles with small K. To filter the noises, shingles are divided into different groups according to their offsets in AutoSig. The windows are overlapped and with fixed width 2W. The i-th window covers the payload which ranges from byte i*W to byte (i+2)*W, so the size of overlapping area between two windows is size W. A shingle appearing in the same window frequently is selected as the common shingle.

B. Shingle merging:

Extracted common shingles are redundant. A common substring can produce a lot of common shingles. These redundant common shingles can be further merged into common substrings. Merging can be done by simple greedy merging algorithm. If two common shingles are overlapped or adjacent, they are merged into one substring. Substrings can be further merged with other shingles or substrings. The merge process is iterated until there are no common shingles which can be merged left. For each flow, merged substrings are generated. After merging all the samples, the substrings appearing in more than R*N flows are selected as the common substrings.

C. Substring sequence generation:

To generate signatures, reassembled common substrings are further organized as substring sequences by using a special data structure called substring tree.

a. First a tree with only one root node is constructed.
b. Second, the common substrings are sorted according to the flow numbers in descending order.

c. Third, the substrings are inserted into the tree one by one.

Each path from a signature node to the root node corresponds to a substring sequence. Depth-first searching algorithm with a stack to record the path is used to generate substring sequences. After extracting substring sequences in the tree, each substring sequence is reordered according to the flow offset of each substring. Thus the signature generated by AutoSig is accurate which saves a lot of time on manual analysis and updates signatures in time. AutoSig can tolerate noises in samples and generate effective signatures for the applications which have complex protocol and exhibit high true positive and less false positive.

V. APPLICATION SIGNATURE CONSTRUCTION FOR UNKNOWN TRAFFIC

Identifying applications and classifying network traffic is based on some sort of priori knowledge, which means some intensive preprocessing would be done before identifying those applications and these methods cannot deal with previously unknown applications. A new approach has been proposed to fully automate the process of deriving signatures from unidentified traffic. This method integrates the statics based flow clustering with the payload based method to eliminate the requirement of pre-labeled data set [13].

The proposed application system consists of two parts: an online classification module and an offline training module that comprises machine learning steps.

Online classifier tries to identify the know application by predicting the class of all traffic flows by matching signatures of known applications and the unidentified traffic are marked as class others. Samples in the others class are traced and used as an input for offline training.

In offline training unknown flows of the new applications undergoes flow clustering [3]. For clustering we apply X- means algorithm, which is able to estimate number of clusters. Every generated cluster consists of flow instances that are dominated by single application and new signature s for each flow can be derived. The task for the administrator is to gain information about emerging application by analyzing the clustered traffic and the newly constructed signatures. This analysis process falls into payload based classification which also takes the advantage of statistical traffic characteristics.

Before running clustering on the data set, feature selection is done by Chi-square Ranking algorithm (CHI) and two subset search algorithms CFS and CON are used. Top five ranked features are choose in CHI. And for CFS and CON the candidate subsets are generated from the original feature space using best first search. To measure the classification result various metrics such as true positive, false positive, false negative and true negative are used for calculating the overall flow accuracy. And the overall clustering accuracy is the number of correctly labeled flows in all clusters as a fraction of the total number of flows in the data set. The application signatures are automatically constructed offline and perform accurate classification online are supervised machine learning classifiers built on byte feature vectors that encode initial raw data of flow payload content.

VI. AUTOMATIC SIGNATURE GENERATION FOR WORMS USING AUTOGRAPH

To prevent the spreading of internet worms through the network traffic, worm signatures are required by the IDS. Initially network operators to generate worm signatures they study the packet traces manually. Autograph [6] is a method that automatically generates signature for novel Internet worms that propagate using TCP transport. Autograph method is extended to share port scan report among distributed monitor instances, and using trace driven simulation, demonstrate the value of this technique in speeding the generation of signature for novel worms.

Autograph monitor uses the traffic crossing the edge network as input and it outputs a list of worm signatures. There are two main stages in Autograph monitor.

A. Suspicious flow selection:

This stage uses heuristics to classify inbound TCP flows as either suspicious or non-suspicious. Tattler has been used over the incoming traffic over the Autograph monitor. It shares suspicious source address among all monitors, towards the goal of accelerating the accumulation of worm payloads. After classification, packets for these inbound flow pools are stored on disk in a suspicious flow pool and non-suspicious flow pool, respectively. Autograph can adopt any anomaly detection technique that classifies worm flows as suspicious with high probability. Port scanning flow classifier is used to demonstrate that Autograph generates highly selective and specific signatures. Autograph Performs TCP flow reassembly for inbound payloads in the suspicious flow pool. The resulting reassembled suspicious packets are analyzed in the second stage. Packets held in the suspicious flow pool are dropped from storage after a configurable interval.

B. Signature generation:

Worm signatures are generated by analyzing the content of payload from the suspicious flow pool. Payload will be divided into variable-length content blocks using Content-based Payload Partitioning (COPP). Autograph divides payload into fixed-size, non-overlapping blocks and computes the prevalence of those blocks across all suspicious pool. Blocks generated by COPP might have little changes under byte insertion or deletion. COPP decides content block boundaries probabilistically, there may be case where COPP generate very short content blocks which are unspecific and and generate many false positives. Then Autograph measures the prevalence of each content block. Then the content block with greatest prevalence is chosen as the signature Autograph classifies an inbound SYN destined for an unpopulated IP address or port with no listening process as a port scan [12]. To identify TCP port scans from spoofed IP source addresses, an Autograph monitor could respond to such inbound SYNs with a SYN/ACK, provided the router and/or firewall on the monitored network can be

configured not to respond with an ICMP host or port unreachable. If the originator of the connection responds with an ACK with the appropriate sequence number, the source address on the SYN could not have been spoofed. Autograph is designed to produce signatures with that exhibit high sensitivity and high specificity.

VII. AUTOMATIC SIGNATURE GENERATION FOR WORMS USING POLYGRAPH

It is found that content-based intrusion detection systems are easily evaded by polymorphic worms. The previously used Autograph method fails to identify the polymorphic worms, which vary their payload on every infection attempt. Polygraph [8] is a method that generates signature that perfectly matches polymorphic worms [11]. Polygraph generates signatures that consist of multiple disjoint content substrings. In doing so, Polygraph leverages our insight that for a real-world exploit to function properly, multiple invariant substrings must often be present in all variants of a payload; these substrings typically correspond to protocol framing, return addresses, and in some cases, poorly obfuscated code.

The polygraph monitor consists of flow classifier which collects the monitored traffic flows from the network and classifies the reassembled flows into contiguous byte flows, and classifies reassembled flows destined for the same IP protocol number and port into a suspicious flow pool and an innocuous flow pool. The labeled flow from the classifier is stored in the signature evaluator. The flows at the pool were then passed into the polygraph signature generator that performs certain methods like token extraction and signature construction to generate the sequence of signature from the monitored traffic flows. Polygraph signature generator takes a suspicious flow pool and an innocuous flow pool as input, and produces a set of signatures as output, chosen to match the worms in the input suspicious flow pool, and to minimize false positives, based on the innocuous flow pool.

The generated signature matches labeled flow data that stored in the signature evaluator. Signatures that generated are from the flow substrings or the tokens. It significantly improve signature quality and allow Polygraph to adapt to attacks that change over time.

Evaluation of various algorithms on a range of polymorphic worms demonstrates that Polygraph produces signatures for polymorphic worms that exhibit low false negatives and false positives.

VIII.AUTOMATIC SIGNATURE GENERATION FOR WORMS USING HAMSA

Hamsa is a fast network based automated signature generation method for polymorphic worms. This method of signature generation will be considered as attack-resilient and noise-tolerant [7]. Hamsa uses the worm flow classifier which classifies the flows into suspicious and normal traffic pool. These flows will then pass into the Hamsa signature generator.

These signature generator process consist of following procedures

A. Token Extractor:

This process extracts the token from the flow using suffix array based algorithm which finds all the byte sequences. The idea of the extraction process is that the worm flows will constitute at least some fraction of the pool so that number of tokens that extracted will be reduced.

B. Core:

This process uses the Greedy signature generation algorithm. This allow the attacker full flexibility to include any content in the polymorphic worms. It helps to extract the unique token that is used for generating signatures.

C. Token Identification:

This process tests the token specificity in the normal traffic pool.

D. Signature Refiner:

It finds the common tokens from the set of sequence tokens from the flows in the suspicious pool that matches the signature from the core. The signature generated will be low false positive without affecting its sensitivity.

Signature generated by Hamsa can be deployed easily by IDSes such as Bro and Snort. This signature generation method outperforms Polygraph and Autograph interms of efficiency and accuracy.

IX. AUTOMATIC SIGNATURE GENERATION USING SIMPLIFIED REGULAR EXPRESSION

Deriving accurate and efficient application signatures for various applications is not a trivial task, however due to the rapid evolution of network applications, the signatures are also subject to change with time instead of staying fixed. Therefore the high cost signature generation process has to be repeated from time to time in order to keep the signatures up to date. To address this problem an approach that automatically learns signatures from data has been developed [14]. This method derives regular expression (regexp) signatures that not only provide sufficient flexibility and expressive power but also are widely supported by practical network intrusion detection systems (e.g., Snort and Bro) and traffic classification systems (e.g., l7-filter).

The regular expression signature generation process involves following procedures.

A. Pre-processing:

It observes raw packets from a network tap or a trace file, which are then tracked into sessions according to 5-tuple. Next is a TCP/IP normalizer that deals with IP fragmentation and TCP reconstruction. The last step is payload extraction. Each flow is represented by two byte sequences (i.e., initiator to responder and the other direction) whose lengths may vary. And we produce a signature for each direction.

B. Advanced Tokenization:

The tokens are developed based on the substrings, which are common to the input byte sequences of each application

class. K-common substring extraction can be done by using the suffix tree algorithm. The flow payload of some applications reveals position-specific information. On one hand, some of the tokens found in the last step always appear at the same offset in payload. In order to capture the position properties in the signatures, we transform each input flow sequence into a set of vectors, where the first element in each vector represents a token that occurs in the flow and the second element represents the position where the token occurs.

C. Multiple sequence Aligment:

To find the best common subsequence, multiple sequence alignment (MSA) technique is used which is widely studied and applied in bioinformatics for finding regions of similarity in biological sequences. The simplest case in computer science is the longest common subsequence (LCS) algorithm that computes the global similarity between two strings because it guarantees a maximum similarity in terms of the number of matched characters rather than consecutive matches, which is preferred for producing meaningful and low false-positive signatures. From the pair-wise alignment results, the similarity scores are converted to distances by normalizing and subtracting from one. Thus a distance matrix is obtained. In the next step, a tree used to guide the final multiple alignment process is calculated from the distance matrix using the Neighbor Joining method.

D. Signature construction:

The final stage of this approach is to transform the alignment results into regular expression signatures. It is done by changing the special symbols back to tokens, and replacing the variable length with‘‘.*’’ and the fixed length gaps with‘‘.{n}’’.

The signatures are close to real life handcrafted signatures because they consist of the critical regexp features that are essential for identifying applications. The results show that with well-designed methods, the labor-intensive signature deriving process can be automated.

Table 1. Comparison of methods used for generating application signatures

Work	Method	Description	Merits	Demerits
Karagiannis [5]	Protocols	Generates signature for popular P2P application based on the protocol	Easy to generate signature since it is based on the protocol	It is manual and time consuming Difficult to produce signatures for new applications
Byung-Chul [2]	LASER algorithm	Automated method for generating application signature using LCS algorithm	This approach generates accurate signature and does not need any prior knowledge of protocols	Lack of publicly available documentation
Ye [15]	AutoSig	Automated method for generating application signature	Signature generated by AutoSig is accurate which saves a lot of time on manual analysis and updates signatures in time	It fails to generate signature for worms
Wang Y [13]	Integrating Clustering and Payload based method	Automated method for generating signatures for unidentified traffic	This method generates signatures of higher accuracy It does not need any priori knowledge before generating signatures	Signatures are difficult to learn and any knowledge about the unknown application is tedious
Kim [6]	Autograph	Automatically generates signature for internet worms	Signatures generated with high true positive and low false positives	It focused on single substring patterns and failed to match polymorphic worms
Newsome [8]	Polygraph	Produces signature that matches polymorphic worms	Polygraph generates high-quality signatures for matching polymorphic worms	It matches the flow only if all the elements in the set are found in its payload
Li [7]	Hamsa	Fast and automatic signature generating method for polymorphic worms	Does not need host information for generating signatures It generates signature of high accuracy and attack resilience It outperforms Polygraph in generating signatures for polymorphic worms	Signature generation using Hamsa is considered to be expensive
Wang Y [14]	Simplified Regular Expression	Automatically generates signature based on the traffic payloads	Signatures of high quality are produced and exhibit low false negatives and false positives. It is most widely used techniques for generating signatures for traffic classification	Sequence alignment algorithm which is used for generating signatures are less efficient

Table 2. Application signature generated by various methods

Method	Application	Signature
Protocol Analysis	LimeWire	'GET' or 'HTTP' followed "User-Agent: Limewire" or "UserAgent: Limewire" or "Server: Limewire"
LASER	LimeWire	"LimeWire" "Content-Type:" "Content-Length:" "X-Gnutella- Content-URN" "run:sha:1" "XAlt" "X-Falt" "X-Create-Time:" "X-Features:" "X-Thex-URI"
AutoSig	HTTP	[HTTP/1.]; [GET\0x20/][HTTP/1.];
Polygraph	Apache-Knacker	GET .* HTTP/1.1\r\n.*: .* \r\nHost: .* \r\n.*: .*\r\nHost: .*\xFF\xBF.*\r\n
Hamsa	Apache-Knacker	{'\xff\xbf': 1, 'GET ': 1, ': ': 4, '\r\n': 5, ' HTTP/1.1\r\n': 1, '\r\nHost: ': 2}
Simplified Regular Expression	HTTP	''http/(0\.9|1\.0|1\.1) [1-5][0-9][0-9][\x09-\x0d -_]*(connection:|content-type:|content-length:|date:)| post [\x09-\x0d -_]* http/[01]\.[019]''

X. CONCLUSION

This paper surveys significant works in the field of application signature generation based traffic classification. It examines various payload based technique and their merits and demerits of those application signature generation methods are shown in Table I. It shows how the application signatures for each application are generated and how it outperforms the previously used methods. Since payload based method is used in real time it is necessary to consider the best signature scheme that exhibit low false positive and low false negative. Table II shows the different application signatures generated by various methods. In this survey paper, we have outlined a number of critical operational requirements for real-time classifiers and qualitatively critiqued the reviewed works against these requirements. There is still a lot of room for further research in the field. Machine learning techniques along with payload based method gets popularity [4]. While most of the approaches build their classification models based on sample data collected at certain points of the Internet, those models usability needs to be carefully evaluated. Payload based traffic classification can be still extended by generating signatures of higher accuracy and low cost.

XI. REFERENCES

[1] Bernaille L, Teixeira R, Akodkenou I, Soule A, Salamatian K. Traffic classification on the fly. SIGCOMM Comput Commun Rev 2006;36(2):23–6.

[2] Byung-Chul P, Won YJ, Myung-Sup K, Hong JW. Towards automated application signature generation for traffic identification. In: Proceedings of the Network Operations and Management Symposium, 2008. NOMS 2008. IEEE; 2008, p. 160–7.

[3] Erman J, Arlitt M, Mahanti A. Traffic classification using clustering algorithms. MineNet '06: Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data. ACM: New York, NY, U.S.A., 2006; 281–286.

[4] Haffner P, Sen S, Spatscheck O, Wang D. ACAS: automated construction of application signatures. In: Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data, Philadelphia, Pennsylvania, USA, ACM; 2005.

[5] Karagiannis T, Broido A, Brownlee N, Claffy KC, Faloutsos M. Is P2P dying or just hiding? [P2P traffic measurement]. In: Proceedings of the Global Telecommu- nications Conference, 2004. GLOBECOM '04. IEEE, vol. 3; 2004, p. 1532–8.

[6] Kim H-A, Karp B. Autograph: toward automated, distributed worm signature detection. In: Proceedings of the 13th Conference on USENIX Security Symposium, vol. 13, San Diego, CA, USENIX Association; 2004.

[7] Li Z, Sanghi M, Chen Y, Kao M-Y, Chavez B. Hamsa: fast signature generation for zero-day polymorphicworms with provable attack resilience. In: Proceedings of the 2006 IEEE Symposium on Security and Privacy, IEEE Computer Society; 2006.

[8] Newsome J, Karp B, Song D. Polygraph: automatically generating signatures for polymorphic worms. In: Proceedings of the 2005 IEEE Symposium on Security and Privacy, IEEE Computer Society; 2005.

[9] Risso. F., Baldi. M., Morandi. O., Baldini. A. and Monclus. P. (2008) "Lightweight, payload-based traffic classification: an experimental evaluation." In: Proceedings of the ICC '08 IEEE International Conference on Communications; p. 5869–75.

[10] Sen. S., Spatscheck. O. and Wang. D. (2004) "Accurate, scalable in-network identification of p2p traffic using application signatures." In: Proceedings of the 13th International Conference on World Wide Web, New York, NY, USA, ACM.

[11] Tang. Y., Xiao. B. and Lu. X. (2009) "Using a bioinformatics approach to generate accurate exploit- based signatures for polymorphic worms." Computer Security; 28(8):827–42.

[12] Wang. K, Cretu G , and S. J. Stolfo. Anomalous payload based worm detection and signature generation. In Proc. Of Recent Advances in Intrusion Detection (RAID), 2005.

[13] Wang Y, Xiang Y, Yu. SZ. An automatic application signature construction system for unknown traffic. Concurrency and Computation–Practice and Experience 2010;22(13):1927–44.

[14] Wang. Y., Xiang. Y., Zhou. W. and Yu. S. (2012) "Generating regular expression signatures for network traffic classification in trusted network management." In: Proceedings with Journal of Network and Computer Applications Volume 35, Issue 3, Pages 992–1000.

[15] Ye MJ, XuK, WuJP, PoH. AutoSig—automatically generating signatures for applications. In: Proceedings of the Ninth IEEE International Conference on Computer and Information Technology CIT' 09; 2009,p.104–9.