International Journal of Engineering Research-Online
A Peer Reviewed International Journal
Articles available online http://www.ijoer.in

Vol.3., Issue.1, 2015

RESEARCH ARTICLE

ISSN: 2321-7758

UPGRADING DIGITAL FORENSICS CAPABILITY BASED ON PARALLEL PROCESSING AND MULTILEVEL FILE BINNING

NEHA RAZDAN[1], SACHIN MAJITHIA[2]
[1]Student,[2]Assistant Professor
[1]Chandigarh Engineering College, Landran, 140307

Article Received: 19/01/2015 | Article Revised on: 29/01/2015 | Article Accepted on:02/02/2015

NEHA RAZDAN

ABSTRACT

Data forensic comes from forensic science. Forensic science is the scientific method of gathering and examining information about the past. This is especially important in law enforcement where forensics is done in relation to criminal or civil law, but forensics are also carried out in other fields, such as astronomy, archaeology, biology and geology to investigate ancient times. Data forensics, often used interchangeably with computer forensics, is essentially the study of digital data and how it is created and used for the purpose of an investigation. Data forensics is part of the greater discipline of forensics, in which various types of evidence are studied to investigate an alleged crime .In this paper the main focus on optimizing overall data size using binning and de-duplication or filtering of files. All the files are arranged with their constant size irrespective of their size. Our main work is using parallel processing because it helps to take less time to process, also response the time over the network

Keywords: Data forensics, parallel processing, sequential processing, binning, filtering, kmp algorithm

INTRODUCTION

Data Forensics: The aim of the Digital (computer) forensics is defined as “analytical and investigative techniques used for the preservation, identification, extraction, documentation, analysis and interpretation of computer media (digital data) which is stored or encoded for evidentiary and/or root cause analysis”. Figure 1 shows the process flow of digital or data forensics.

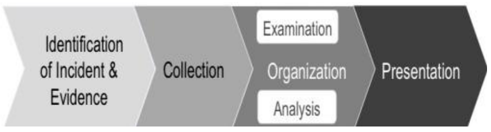


Fig 1 Process Flow of Digital Or Data Forensics

Computer forensics is also important because it can save your organization money. Many managers are allocating a greater portion of their information technology budgets for computer and network security.

Two basic types of data are collected in data forensics.

- **Persistent data:** The data that is stored on a local hard drive (or another medium) and is preserved when the computer is turned off is called persistent data.
- **Volatile data:** Any type of data that is stored in memory, or exists in transit, that will be lost.

De-duplication: Data deduplication is a technique for reducing the amount of storage space an organization needs to save its data. In most organizations, the storage systems contain duplicate copies of many pieces of data. In its simplest form, deduplication takes place on the file level; that is, it eliminates duplicate copies of the same file. This kind of deduplication is sometimes called file-level deduplication or single instance storage (SIS). Deduplication can also take place on the block level, eliminating duplicated blocks of data that occur in non-identical files. Block-level deduplication frees up more space than SIS, and a particular type known as variable block or variable length deduplication has become very popular.

The two major categories of data de-duplication:

- **Offline Data de-duplication:** In an offline deduplication state, first data is written to the storage disk and deduplication process take place at a later time.
- **Online Data de-duplication:** In an online deduplication state, replicate data is deleted before being written to the storage disk.

Binning; Binning is a way to group a number of more or less continuous values into a smaller number of "bins”. Binning is also a combination of two or more ccd images sensor pixels to form a new “super-pixel” prior to readout and digitizing. It is a preprocessing technique used to reduce the effects of minor observation errors

Knuth Morris Pratt Algorithm: In computer science, the Knuth–Morris–Pratt string searching algorithm (or KMP algorithm) searches for occurrences of a "word" W within a main "text string" S by employing the observation that when a mismatch occurs, the word itself embodies sufficient information to determine where the next match could begin, thus bypassing re-examination of previously matched characters. The algorithm was conceived in 1974 by Donald Knuth and Vaughan Pratt, and independently by James H. Morris. The three published it jointly in 1977.In studying the worst-case performance of the brute-force and boyer-moore pattern matching algorithms on specific instances of the problem; we should notice a major inefficiency. Specifically, we may perform many comparisons while testing a potential placement of the pattern against the text, yet if we discover a pattern character that does not match in the text, then we throwaway all the information gained by these comparisons and start over again from scratch with the next incremental placement of the pattern. The Knuth-Morris-Pratt (or "KMP") algorithm avoids this waste of information and, in so doing, it achieves a running time of O(n + m), which is optimal in the worst case. That is, in the worst case any pattern matching algorithm will have to examine all the characters of the text and all the characters of the pattern at least once.Figure2 represents the implementation of KMP algorithm.

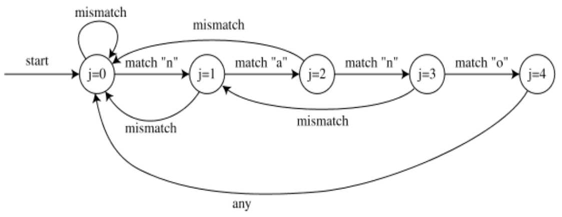


Figure2: implementation of KMP algorithm

Example Of Knuth Morris Pratt Algorithm: To illustrate the ideas of the algorithm, we consider the following example:

T = xyxxyxyxyyxyxyxyyxyxyxxy

and

P = xyxyyxyxyxx

At a high level, the KMP algorithm is similar to the naive algorithm: it considers shifts inorderfrom1ton−m, and determines if the pattern matches at that shift. The difference is that the KMP algorithm uses information gleaned from partial matches of the pattern and text to skip over shifts that are guaranteed not to result in a match. Suppose that, starting with the pattern aligned underneath the text at the leftmost end, we repeatedly “slide” the pattern to the right and attempt to match it with the text. Let’s look at some examples of how sliding can be done.

1. Consider the situation when P[1,...,3] is successfully matched with T[1,...,3]. We then and a mismatch: P[4] = T[4]. Based on our knowledge that P [1,...,3] = T[1,...,3], and ignoring symbols of the pattern and text after position 3, what can we deduce about where a potential match might be? In this case, the algorithm slides the pattern 2 positions to the right so that P[1] is lined up with T[3]. The next comparison is between P [2] and T [4].
2. Since P [2] = T [4], the pattern slides to the right again, so that the next comparison is between P [1] and T [4].

3. At a later point, P [1,...,10] is matched with T[6,...,15]. Then a mismatch is discovered: P [11]=T[16]. Based on the fact that we know T[6,...,15] =P[1,...,10] (and ignoring symbols of the pattern after position 10 and symbols of the text after position 15), we can tell that the first possible shift that might result in a match is 12. Therefore, we will slide the pattern right, and next ask whether P [1,...,11] = T[13,...,23]. Thus, the next comparisons done are P [4] == T [16], P [5] == T[17], P[6] ==T[18] and so on, as long as matches are found. In studying the worst-case performance of the brute-force and boyer-moore pattern matching algorithms on specific instances of the problem, we should notice a major inefficiency. Specifically, we may perform many comparisons while testing a potential placement of the pattern against the text, yet if we discover a pattern character that does not match in the text, then we throwaway all the information gained by these comparisons and start over again from scratch with the next incremental placement of the pattern. The Knuth-Morris-Pratt (or "KMP") algorithm avoids this waste of information and, in so doing, it achieves a running time of O(n + m),which is optimal in the worst case. That is, in the worst case any pattern matching algorithm will have to examine all the characters of the text and all the characters of the pattern at least once. The KMP pattern matching algorithm incrementally processes the text string T comparing it to the pattern string P. Each time there is a match, we increment the current indices. On the other hand, if there is a mismatch and we have previously made progress in P, then we consult the failure function to determine the new index in P where we need to continue checking P against T. Otherwise (there was a mismatch and we are at the beginning of P), we simply increment the index for T (and keep the index variable for P at its beginning). We repeat this process until we find a match of P in T or the index for T reaches n, the length of T (indicating that we did not find the pattern P in T). The main part of the KMP algorithm is the while-loop, which performs a comparison between a character in T and a character in P for each iteration. Depending upon the outcome of this comparison, the algorithm either moves on to the next characters in T and P, consults the failure function for a new candidate character in P, or starts over with the next index in T. The correctness of this algorithm follows from the definition of the failure function. The skipped comparisons are actually unnecessary, for the failure function guarantees that all the ignored comparisons are redundant-they would involve comparing characters that are already known to match.

LITERATURE REVIEW

The current research in life science area is producing large amount of genetic data. DNA related pattern matching in a given sequence is one of the fundamental problems in computational biology. Biologists often search DNA sequences to identify use full information available from protein and genes for the functional and structural behavior. Many algorithms have been proposed but more efficient and robust methods are needed for the exact pattern matching algorithms. In the proposed work an index based sequential multiple pattern matching using pair indexing algorithm gives very good performance when compared with some of the existing algorithms. The current technique avoids unnecessary DNA comparisons as a result the number of comparisons and CPC ratio gradually decreases and overall performance increases (Raju Bhukya, DVLN Somayajulu).

Many modern storage systems use deduplication in order to compress data by avoiding storing the same data twice. Deduplication needs to use data stored in the past, but accessing information about all data stored can cause a severe bottleneck. Similarity based deduplication only accesses information on past data that is likely to be similar and thus more likely to yield good deduplication. We present an adaptive deduplication strategy that extends Extreme Binning and investigate theoretically and experimentally the effects of the additional bin accesses (Zhike Zhang, Deepavali Bhagwat, WitoldLitwin, Darrell Long, Thomas Schwarz, S.J).

Digital evidence is increasingly used in juridical proceedings. In some recent legal cases, the verdict has been strongly influenced by the digital evidence proffered by the defense. Digital traces can be left on computers, phones, digital cameras, and also on remote machines belonging to ISPs, telephone providers, companies that provide services via Internet such as YouTube, Face book, Gmail, and so on. This paper presents a methodology for the automated production of predetermined digital evidence, which can be leveraged to forge a digital alibi. It is based on the use of automation, a program meant to simulate any common user activity. In addition to wanted traces, the automation may produce a number of unwanted traces, which may be disclosed upon a digital forensic analysis. These include data reminisce of suspicious _les, as well as any kind of logs generated by the operating system modules and services. The proposed methodology describes a process to design, implement, and execute the automation on a target system, and to properly handle both wanted and unwanted evidence. Many experiments with different combinations of automation tools and operating systems are conducted. This paper presents an implementation of the methodology through VBScript on Windows 7. A forensic analysis on the target system is not sufficient to reveal that the alibi is forged by automation. These considerations emphasize the difference between digital and traditional evidence. Digital evidence is always circumstantial, and therefore it should be considered relevant only if supported by stronger evidence collected through traditional investigation techniques. Thus, a Court verdict should not be based solely on digital evidence (Aniellocastiglione (member, ieee), giuseppecattaneo, giancarlo de maio, and alfredo de santis (member, ieee)).

Parallel computing operates on the principle that large problems can often be divided into smaller ones, which are then solved concurrently to save time (wall clock time) by taking advantage of non-local resources and overcoming memory constraints. The main aim is to form common single node architecture for both MPI and PVM, which demonstrates the performance gain and losses achieved through parallel processing using MPI and PVM as separate cases. We also demonstrates the performance dependency of parallel applications on RAM of the nodes (desktop PCs) used in parallel computing. This can be realized by implementing the parallel applications like solving matrix multiplication problem, using MPI and PVM separately. The single node architecture consists of a client, a master, capable of handling requests from the client, and a slave, capable of accepting problems from the master and sending the solution back. The master and the slave communicate with each other using MPICH-2 and PVM3.4.6 when computation is under MPI and PVM respectively. The master will monitor the progress and be able to compute and report the time taken to solve the problem, taking into account the time spent in assigning the problem into slave and sending the results along with the communication delays. . We aim to evaluate and compare these statistics of both the cases to decide which among MPI and PVM gives faster performance and also compare with the time taken to solve the same problem in serial execution to demonstrate communication overhead involved in parallel computation. We aim to compare and evaluate the statistics obtained for different sizes of RAM under parallel execution in a single node involving only two cores, where one acts as master and other as slave. We also show the dependency of serial execution on RAM for the same problem by executing its serial version under different sizes of RAM (Sampath S1,Bharat Bhushan Sagar2,Nanjesh B R3).

PROPOSED TECHNIQUE

From the literature survey, it has been concluded that there is issue of investigation of phishing data from the user access pattern it handle the situation when incoming files are of dynamic file sizes. Their separation needs to done on the bases of file sizes just in two categories i.e. heavy weight and light weight files with the help of binning. In the process of binning the heavy weight file are handle and are manage more systematically more than 3 to 4 categories of different file sizes. The amount of data at the data centre is very huge it needs to be processed keeping the time parameter in mind so the data is to be processed parallel using some string matching algorithm.

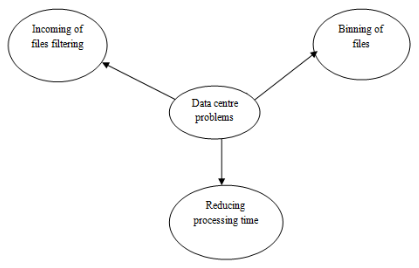


Figure3: Data centre problems

STEPS OF PROPOSEDWORK

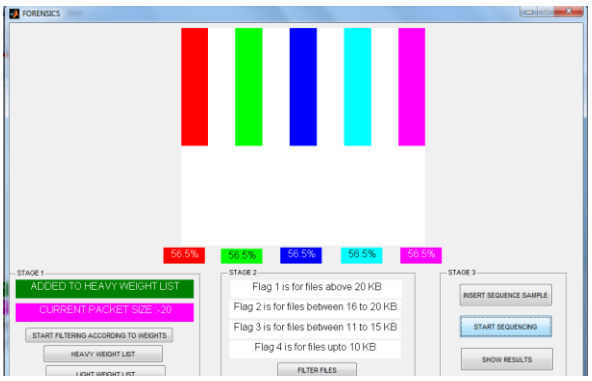
The steps of a proposed work is shown in the flowchart 1

STEP 1: Data arrives at the data centre

STEP 2: Segmentation has been done for the data because it reduces data files into smaller chunks which are easier to transfer over Internet.

STEP 3: After segmentation data compression has done using de-duplication of files which focuses on optimizing overall data files size. This technique reduces cost and increased storage efficiency, while also improving the user experience.

STEP 4: The files are then segregated into different bins depending on their size. When binning is done with constant size for all files irrespective of their size, it either leads to a large number of segments or very few segments. The files are also processed by using parallel processing because it take less time to process and save the segments, also response the time over the
network.



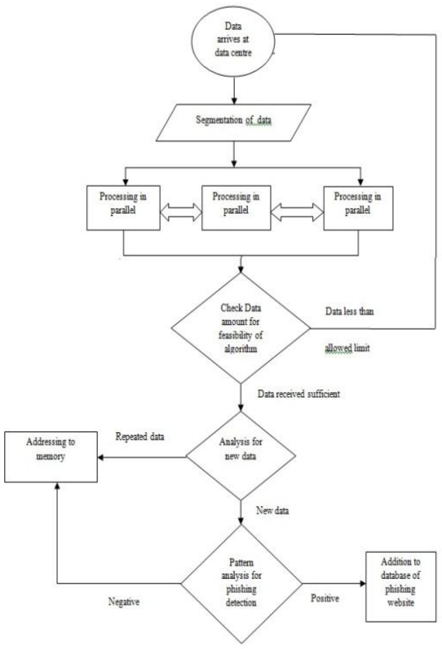
Flag 1 is for files above 20 KB
Flag 2 is for files between 16 to 20 KB
Flag 3 is for files between 11 to 15 KB
Flag 4 is for files upto 10 KB

Figure 4(Screen shot): Shows the parallel processing of files.

STEP 5: Now check the amount of data whether it is feasible for algorithm if the given data is not feasible or less than for the algorithm then again the new data will enter at the data centre till the amount of data is feasible for the algorithm.

STEP 6: If the data which we received is sufficient, then we analyze a new data. If this new data was also repeated then this data will addressing to the memory.

STEP 7: Then the new data will analyzing for the phishing detection. If this data is negative it will addressing to the memory and if it is positive it will goes in the database of phishing websites in which other phishing websites is present. By using the KMP (Knuth Morris Pratt) algorithm the string of data is matched whether it is phishing data or not.



Flowchart 1: Shows the execution of data

METHODOLOGY

Implementation Setup: The implementation of our work is done in the MATLAB. The software used to implement our proposed algorithm will be MATLAB. MATLAB is a high-level language and interactive environment for numerical computation, visualization, and programming. Using MATLAB, we can analyze data, develop algorithms, and create models and applications. The language, tools, and built-in math functions enable us to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java™.

MATLAB can be used for a range of applications, including signal processing and communications, image and video processing, control systems, test and measurement, computational finance, and computational biology.

We chose MATLAB to implement our algorithm because of the following reasons:

- High-level language for numerical computation, visualization, and application development
- Interactive environment for iterative exploration, design, and problem solving
- Built-in graphics for visualizing data and tools for creating custom plots
- Development tools for improving code quality and maintainability and maximizing performance

- Tools for building applications with custom graphical interfaces

Functions for integrating MATLAB based algorithms with external applications and languages such as C, Java,.N Various toolboxes used in MATLAB are as follows:

- Communications System Toolbox
- Data Acquisition Toolbox
- Database Toolbox

Methodology Used: This section describes a methodology and its related parameters, experiment factors.

System Parameters- The experiments are conducted using Intel Dual Core 64-bit processor with a minimum of 256 MB RAM or higher and hard disk 10 GB or higher. The implementation is done in the MATLAB. In this report, the KMP algorithm has been implemented parallel processing of the above configuration.

Experimental Parameters-In order to evaluate the effectiveness of the proposed technique in the data forensics. The two parameters are must be determined.

1. Time processing:-The time processing is defined as the two sets of functions for measuring absolute elapsed time: clock and etime, and Tic and Toc. The clock and etime functions use dates to track elapsed time, and are useful if you need accuracy to only .01second. Originally, the TIC and TOC functions relied on clock for the time and had the same accuracy. We use a tic toc function to calculate the time processing.

2. Complexity: -It is used to characterize something with many parts where those parts interact with each other in multiple ways. Complexity of an object or system is a relative property.

RESULTS ANALYSIS

In this section we evaluate the effectiveness of proposed technique by using the parameters. We have generated the graphs of our proposed method by using two parameters first is time processing, second is complexity.

In these graphs a number of files which is to be processed .The table is also given below in which number of files and the time taken to complete its processing. The number of files is in KB and the time is given in seconds.

Scenerio1 (Sequential Processing): In this scenario, files are processed in sequential manner. In sequential processing files take much more time to process and the time processing of all the files is very large.

Table 1 Shows the time taken to process the different files

No of files (in KB)	Time(in seconds)
5000	95.55
4000	80.53
3000	76.49
2000	73.63
1000	63.54
500	50.44
250	45.42
100	40.39

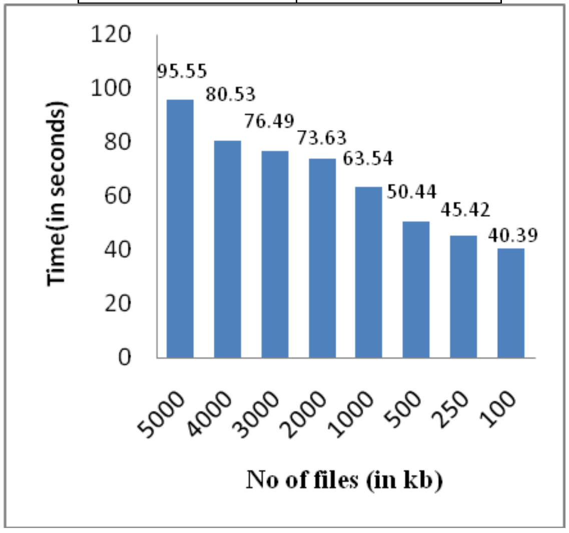


Figure 5 : Represents the graph in sequential processing

Scenerio2. (Parallel Processing)

In this scenario, files are processed in parallel manner. In parallel processing, files take small amount of time to process.

Table 2: Shows the time taken to process the different files

No of files (in KB)	Time(in seconds)
5000	75.55
4000	66.64
3000	59.54
2000	45.44
1000	39.36
500	35.32
250	30.15
100	27.22

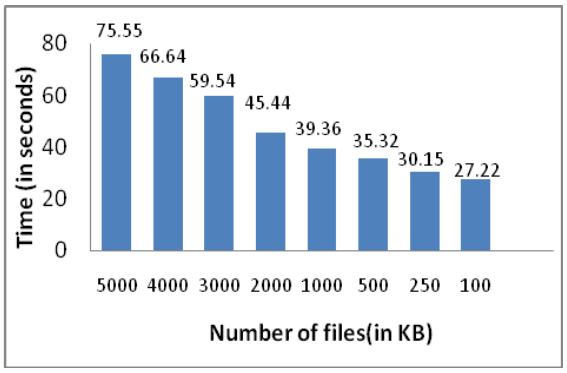


Figure 6: Represents graph in parallel processing

With these two graphs we can show that time taken in parallel is much faster than the sequential. Now days we have a number of files in the data that is to be accessed. To make our parallel processing better we use the idea of binning and filtering. With the help of this our time become more less. We have taken a same number of files for processing in parallel by use the concept of binning and filtering.

Scenerio3. (Parallel Processing with binning and filtering)

In this scenario, files are processed in parallel manner using binning and filtering. A lot of time must be saved with the help of binning and filtering.

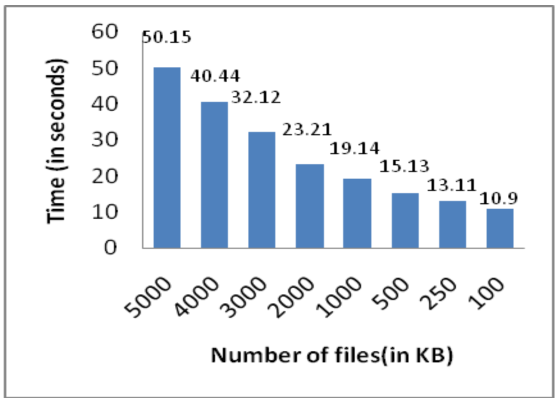


Figure 7: Represents graph in parallel by using binning and filtering

Table 3 Shows the time taken to process the different files

No of files (in KB)	Time(in seconds)
5000	50.15
4000	40.44
3000	32.12
2000	23.21
1000	19.14
500	15.13
250	13.11
100	10.9

CONCLUSION

In this paragraph, the result of the simulations that we have mentioned here we conclude that the simulations have proved to improve upon the time factor that was utterly needed for the kind of data processing ability that is needed in this task. Earlier only binning and filtering of files were done to enhance the capacity and efficiency of the system. But as an enhanced technology we have used the idea of parallel processing instead of sequential processing ultimately reducing the overall time required to process the data. Results of the parallel processing have been much better than the previous results.

FUTURE SCOPE

The field of data forensics and incident response (Computer security incident management which involves monitoring and proactive detection of security threats) is new field of research that attracts more and more scientists. It poses a lot of challenges due to exponential rise in dynamic being generated and accessed data around the world and also distributed nature of cloud. In our approach we have presented a solution to real time data access patterns on static database. In future more innovative techniques would be required for dynamic database as it spreads over multiple data centers.

ACKNOWLEDGMENT

Although I have taken lots of efforts in this project but it would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to all of them. I am highly indebted to Er. Sachin Majithia, Asst. Professor, CEC, Landran for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his extended support in completing the project. I would also like to express my gratitude towards my parents & Er. Hemant Servia for their kind co-operation and encouragement which helped me in completion of this project.

REFERENCES

[1]. Irfan Khan. “Prefix Span Algorithm Based on Multiple Constraints for Mining Sequential Patterns”, International Journal of Computer Science and Management Research, Vol. 1, Issue 5, December 2012.

[2]. B.Mallik and D.garg, "CFM Prefix Span: A pattern growth algorithm incorporating monetary and compactness", pp 4509-4555, July 2012.

[3]. S. Zawoad and R. Hasan, "Towards building proofs of past data possession in cloud forensics," ASE Science Journal, 2012

[4]. K. Ruan, J. Carthy, T. Kechadi, and M. Crosby, "Cloud forensics: An overview," in proceedings of the 7th IFIP International Conference on Digital Forensics, 2011.

[5]. S. Zawoad, A.K. Dutta and R. Hasan, "Salas: Secure Logging-as-a-Service for Cloud Forensics", Symposium on Information, Computer and Communications Security (ASIACCS), 2013

[6]. R. Marty, "Cloud Application Logging for Forensics", Proceedings of the 2011 ACM Symposium on Applied Computing, 2011

[7]. G. Sibiya, H. Venter and T. Fogwill, "Digital forensic framework for a cloud environment", Proceedings of the 2012

[8]. Fabio Marturana ,SimoneTacconi ,Gianluigi Me, "A case study on digital forensics in the cloud", International Conference on Cyber-Enabled Distributed Computing and

[9]. ShahrzadZargari ,DavidBenford ,"Cloud Forensics: Concepts, Issues, and Challenges", Third International Conference on Emerging Intelligent Data and Web Technologies,2012

[10]. Guangxuan Chen', Yanhui nu', Panke Qin', JinDu2 ,"Suggestion To Digital Forensics In Cloud Computing Era", Proceedings ofIC-NIDC2012

[11]. AmitChhabra, Gurvinder Singh "A Cluster Based Parallel Computing Framework (CBPCF) for Performance Evaluation of Parallel Applications", International Journal of Computer Theory and Engineering, Vol. 2, No. 2 April, 2010.

[12]. Rafiqul Zaman Khan, MdFiroj Ali, "A Comparative Study on Parallel Programming Tools in Parallel Distributed Computing System: MPI and PVM", Proceedings of the 5th National Conference; INDIACom-2011.

[13]. Sampath S, Sudeepa K.B, Nanjesh B R "Performance Analysis and Evaluation of Parallel Applications using a Cluster Based Parallel Computing Framework", International Journal of Computer Science and Information Technology Research Excellence (IJCSITRE), Vol.2,Issue 1,Jan-Feb 2012.