

## Utilization Data Mining to Detect Spyware

<sup>1</sup>Parisa Bahraminikoo, <sup>2</sup>Mehdi Samiei yeganeh, <sup>3</sup>G.Praveen Babu  
<sup>1,2</sup>(M.Tech.(S/w. Eng.), <sup>3</sup>(Associate Professor School of Information Technology, Jawaharlal Nehru  
Technological University, Iran)

---

**Abstract:** Malicious software (malware) is any software that gives partial to full control of your computer to do whatever the malware creator wants. Malware can be a virus, worm, Trojan, adware, spyware, root kit, etc. Spyware is a type of malware (malicious software) installed on computers that collects information about users without their knowledge. In the year 1956, Artificial Intelligence (AI) was established at Dartmouth College during a conference. The technology developed so much that it started involving many other branches of engineering such as electronics, robotics etc. This eventually led to much more complex and smart machinery involving Artificial Intelligence. With the development of malware detection systems and Artificial Intelligence, as a new technology for them, Artificial Intelligence has been applied in anti-virus engines. There are several AI approaches that applied in spyware detection systems such as Artificial Neural Networks, Heuristic Technology and Data Mining (DM) Technique. Heuristic-based Detection performs well against known Spyware but has not been proven to be successful at detecting new spyware. In this paper we focus on DM-based malicious code detectors using Breadth-First Search (BFS) approach, which are known to work well for detecting viruses and similar software. BFS is a strategy for searching in a tree when search is limited to essentially two operations: (a) visit and inspect a node of a tree; (b) gain access to visit the nodes that are neighbor to currently visited node. The BFS begins at a root node and inspect all the neighboring nodes. Then for each of those neighbor nodes in turn, it inspects their neighbor nodes which were unvisited, and so on.

**Keywords:** Spyware, Artificial intelligence, Data mining, Breadth-First Search.

---

### I. Introduction

As the application of computer and Internet is more popular, it provides a convenient way to share the information among different people; however it also gives chances to malware activities, such as propagating malicious programs, including computer viruses [1]. Programs that have the potential to break the privacy and security of a system can be labeled as Privacy Invasive Software [2]. These programs include: spyware, adware, Trojans, greyware and backdoors [3]. Spyware is a type of malware (malicious software) installed on computers that collects information about users without their knowledge. The presence of spyware is typically hidden from the user and can be difficult to detect. Some spyware, such as keyloggers, may be installed by the owner of a shared, corporate, or public computer intentionally in order to monitor users. While the term spyware suggests software that monitors a user's computing, the functions of spyware can extend beyond simple monitoring. Spyware can collect almost any type of data, including personal information like Internet surfing habits, user logins, and bank or credit account information. Spyware can also interfere with user control of a computer by installing additional software or redirecting Web browsers. Some spyware can change computer settings, which can result in slow Internet connection speeds, un-authorized changes in browser settings, or changes to software settings.

The goal of spyware is generally not to cause damage or to spread to other systems. Instead, spyware programs monitor the behavior of users and steal private information, such as keystrokes and browsing patterns. This information is then sent back to the spyware distributors and used as a basis for targeted advertisement (e.g., pop-up ads) or marketing analysis [4]. AI is the science and engineering of making intelligent machines, especially intelligent computer programs [1]. AI is set to play an important role in our lives. Researchers produce new products which duplicate intelligence, understand speech, beat the opponent chess player, and acting in complex conditions. The major problems of Artificial Intelligence include qualities such as knowledge, planning, learning, reasoning, communication, perception and capability to move and control the objects [6]. The aim of Artificial Intelligence is to develop the machines to perform the tasks in a better way than the humans [5]. As following we will describe the main application of artificial intelligence that is applied in spyware detection systems. The rest of this paper is organized as follows: section 2 briefly describes the Heuristic Technology, Section 3 explains Data mining Technique and section 4 briefly describes the Neural Network Technology.

## II. Heuristic Technology

At the present, the first and main application for spam filtering technique based on artificial intelligence is Heuristic Technology. Current anti-spyware tools make use of signature-based methods by using specific features or unique strings extracted from binaries or heuristic-based methods by using on the basis of rules written by experts who define behavioral patterns as approaches against spyware. These approaches are often considered ineffective against new malicious code [7, 8].

Heuristic Technology means "the ability of self-discovery" or "the knowledge and skills that use some methods to determine", and intelligently analyze codes to detect the unknown virus by some rules while scanning [9]. Heuristics are used quite often in Artificial Intelligence based research. They are new and constantly being refined by most antivirus companies over the last five years or so. Computational they are much faster than signature based techniques. Heuristics look for a set of characteristics within a file in order to determine whether or not it may be a potential threat. In a sense, heuristic anti-malware attempts to apply the processes of human analysis to an object. In the same way that a human malware analyst would try to determine the process of a given program and its actions, heuristic analysis performs the same intelligent decision-making process, effectively acting as a virtual malware researcher. As the human malware analyst learns more from and about emerging threats he or she can apply that knowledge to the heuristic analyzer through programming, and improve future detection rates. Antivirus software may use one or several techniques to proactively detect malware. The main essence of each method is to analyze the suspicious file's characteristics and behavior to determine if it is indeed malware.

The main concern with heuristic detection is that it often increases false positives. False positives are when the antivirus software determines a file is malicious (and quarantines or deletes it) when in reality it is perfectly fine and/or desired. Because some files may look like viruses but really aren't, they are restricted and stopped from working on your computer. In Heuristics based detection we can use Generic Signature, This technique is particularly designed to locate variations of viruses. Several viruses are re-created and make themselves known by a variety of names, but essentially come from the same family (or classification). Genetic detection uses previous antivirus definitions to locate these similar "cousins" even if they use a slightly different name or include some unusual characters. The best way to illustrate this idea is with identical twins. They may have slightly different fingerprints, but their DNA is identical.

## III. Neural Network Technology

An Artificial Neural Network (ANN) (Bishop, 1995) is an information processing paradigm that is inspired by the way biological nervous systems (i.e., the brain) are modeled with regard to information processing [13]. A neural network is designed to simulate a set of neurons, usually connected by synapses. In the nervous system, a synapse is a structure that permits a neuron to pass an electrical or chemical signal to another cell. Just as in biological systems, learning involves adjustments to the synaptic connections that exist between the neurons. Neural networks can differ on: the way their neurons are connected; the specific kinds of computations their neurons do; the way they transmit patterns of activity throughout the network; and the way they learn including their learning rate. Neural networks are being applied to an increasing large number of real world problems [14].

The neural network is configured for a specific application, such as data classification or pattern recognition, through a learning process called training [15]. In [16] has introduced how to use Single layer neural classifier to detection boot viruses, and the generic virus detector was incorporated into IBM Antivirus in May, 1994. Its structure has been shown in Fig 1.



Figure 1: Single layer neural classifier

William Arnold and Gerald Tesauro constructed multiple neural network classifiers which can detect unknown Win32 viruses by combining the individual classifier outputs using a voting procedure, following a technique described in previous work (Kephart et al, 1995) on boot virus heuristics [17]. And the system has 508 achieved effectively. Authors of [18] presented a new rule generation method from neural networks formed using a genetic algorithm (GA) with virus infection and deterministic mutation. This method can extract rules (regularities) for a pattern classification and chaotic system identification by using the same system [1].

#### IV. Data mining Technique

With the rapid development of Information Technology, the rapid growth of data has exceeded the ability of the manual processing of data. So how to help people to extract the general knowledge from the mass of data has become more and more important. In order to implement it, data mining technique is put forward and soon becomes an active research direction. Data mining analyzes the observed sets to discover the unknown relation and sum up the results of data analysis to make the owner of data to understand. Data Mining Algorithm that is from Statistics, Pattern Identification, Machine Learning [5, 11], and Database and so on, has developed comprehensive [1].

In [10] presented a data-mining framework that detects new, previously unseen malicious executables accurately and automatically. The 2001 data mining study of malicious code [8] used. Three types of features, i.e., Dynamic-link Library resource information, consecutive printable characters (strings) and byte sequences. In [12] presented a network virus precaution system based on data mining shown in Fig 2. It can detect the abnormal connecting behavior of network in real-time to discover the trace of worm virus, especially the precaution action to the new worm virus to make administrator to adopt corresponding measure to avoid tremendous loss.

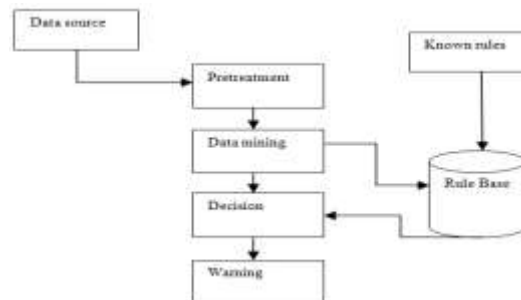


Figure 2: The structure of warning system

Data mining techniques perform better than traditional techniques such as signature-base detection and Heuristic-based detection. The focus of our analysis is executable files for the Windows platform. We use the Waikato Environment for Knowledge Analysis (Weka) to perform the experiments. Weka is a suite of machine learning algorithms and analysis tools, which is used in practice for solving data mining problems, first, we extract features from the binary files. We extract the features by using the Common Feature-based Extraction (CFBE). The purpose of employing this approach is to evaluate two different techniques that use different types of data representation, i.e., the occurrence of a feature and the frequency of a feature. CFBE method are used to obtain Reduced Feature Sets (RFSs) which are then used to generate the ARFF files and includes instances from the frequency range of 50-80. And we then apply a feature reduction method in order to reduce data set complexity. In experiments for the detection of malware, sequences of bytes extracted from the hexadecimal dump of the binary files have been represented by n-grams [3]. Finally, we convert the reduced feature set into the Attribute-Relation File Format (ARFF). ARFF files are ASCII text files that include a set of data instances, each described by a set of features [3].

Data mining base malicious approach have proven to be successful in detecting viruses and worms. Overall accuracy of 90.5% is achieved with the BF-tree algorithms.

We evaluate each learning algorithm by performing cross-validation tests to ensure that the generated classifiers are not tested on the training data. From the response of the classifiers the relevant confusion matrices were created. Four metrics define the elements of the matrix: True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN).

Metric	Abbreviation	Meaning
True Positives	TP	Number of correctly identified benign programs.
False Positives	FP	Number of wrongly identified Spyware programs.
True Negatives	TN	Number of correctly identified Spyware programs.
False Negatives	FN	Number of wrongly identified benign programs.

Table 1 Evaluation metrics

We shall now demonstrate how multi-criteria metrics can be used as an approach to trade-off some of the important aspects of EULA (End User License Agreement) classification [19]. The performance of each classifier was evaluated using the true positive rate, false positive rate and overall accuracy which are defined as follows:

**True Positive Rate (TPR):** Percentage of correctly identified benign programs  $(TP / TP+FN)$

**False Positive Rate (FPR):** Percentage of wrongly identified Spyware programs  $(FP / TN+FP)$

**Overall Accuracy (ACC):** Percentage of correctly identified programs  $(TP+TN / TP+TN+FP+FN)$

Table 2 show that Using the feature set produced by the CFBE feature selection method for  $n = 4$ , the BFT decision tree classifier achieves the highest accuracy results in Frequency Range 50-80.

Algorithm	Type	TPR	FPR	ACC
BFT Frequency Range 50-80	Trees	0.992 0.977	0.731 0.730	89.896 (5.104) 88.5222 (5.899)
Random Forest Frequency Range 50-80	Trees	0.979 0.960	0.665 0.720	89.489 (5.520) 87.077 (6.477)
Naive Bayes Frequency Range 50-80	Bayes	0.973 0.916	0.730 0.705	88.209 (6.174) 83.703 (8.202)
SMO Frequency Range 50-80	Function	0.935 0.946	0.665 0.515	86.566 (8.130) 88.5222 (7.530)

Table 2 Comparison of Algorithms for N-gram size = 4

## V. Conclusion

Spyware technique has become the most important Prevention technique. With this technologies, the system can detect virus invasion in real-time, and enlarge security management capacity of system administrators to enhance the integrity of the infrastructure of information security.

With development of Artificial Intelligence technology has provided new methods and ideas for spyware detection system. Intergraded spyware detection with AI will greatly improve the performance of the existing spyware detection system, promote more effective artificial intelligence algorithms to be proposed, and be applied in the popular detection field .The main objective of the present work is to establish a method in Spyware detection research using data mining techniques. These techniques are used for information retrieval and classification. Data mining-based malicious code detectors have been proven to be successful in detecting clearly malicious code, e.g., like viruses and worms. Results from different studies have indicated that data mining techniques perform better than traditional techniques against malicious code. However, spyware has not received the same attention from researchers but it is spreading rapidly on both home and business computers. Overall accuracy of 90.5% is achieved with the BF-tree algorithms.

## References

- [1]. Review on the application of Artificial Intelligence in Antivirus Detection System", Xiao-bin Wang Guang-yuan Yang Yi-chao Li Dan Liu.
- [2]. M. Boldt and B. Carlsson, "Privacy-invasive software and preventive mechanisms," 2nd International Conference on Systems and Networks Communications, (ICSNC 2006), Oct. 28- Nov.2, IEEE Computer Society.
- [3]. Detection of Spyware by Mining Executable Files" Raja Khurram Shazhad, Syed Imran Haider, Niklas Lavesson, 2010International Conference onAvailability,Reliability and Security.
- [4]. Behavior-based Spyware Detection" Engin Kirda and Christopher Kruegel, Greg Banks, Giovanni Vigna, and Richard A. Kemmerer.
- [5]. Review on use of Reinforcement Learning in Artificial Intelligence" Mehdi Samieiyeganeh , Parisa Bahraminikoo ,G.Praveen Babu ,12<sup>th</sup> International Conference of Science and Technology Impact on Development and Justice held at Maulana Azad National Urdu University, Hyderabad, India, on 7<sup>th</sup> & 8<sup>th</sup> February, 2012.
- [6]. Chuck Williams. "A BRIEF INTRODUCTION TO ARTIFICIAL INTELLIGENCE", 10.0 109 83 IEEE.
- [7]. C. D. Bozagac, "Application of Data Mining based Malicious Code Detection Techniques for Detecting new Spyware", White paper, Bilkent University, 2005.
- [8]. M. Schultz, E. Eskin, F. Zadok, and S. Stolfo, "Data mining methods for detection of new malicious executables," Proceedings of IEEE Symposium on Security and Privacy, 14-16 May 2001, Los Alamitos,
- [9]. Xianwei Zeng, Zhijun Zhang, and Zhi Zhang, "Heuristic skill of computer virus analysis based on virtual machine," Computer Applications and Software, Vol. 22(9), 2005, pp. 125-126.
- [10]. Matthew G. Schultz, Eleazar Eskin, Erez Zadok, and Salvatore I. Stolfo, "Data Mining Methods for Detection of New Malicious Executables," The 2001 IEEE Symposium on Security and Privacy, Oakland, CA, 2001, pp.38-49.
- [11]. I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, 2nd ed. Morgan.
- [12]. Yufeng Yang, "The Network Virus Precaution System Based on Data Mining," Journal of Shaoguan University, Vol. 26(12), 2005, pp. 31-33.
- [13]. Detection of Unknown Computer Worms based on Behavioral Classification of the Host", Robert Moskovitch, Yuval Elovici, Lior Rokach.
- [14]. Review on Artificial Intelligence and Artificial neural networks" Mehdi Samieiyeganeh , ParisaBahraminikoo,G.PraveenBabu, International Conference on Computing,Communications, Systems & Aeronautics (ICCCSA-12) , Organized by Malla Reddy College of Engineering & Technology From March 30-31,2012. Hyderabad, India.
- [15]. Artificial Intelligence: Neural Networks Simplified "Indranarain RamlallI University of Technology, Mauritius.
- [16]. Kephart, J.O., "Biologically inspired defenses against computer viruses," Proceedings of International Joint Conference on Artificial Intelligence, 1995, pp. 985-96.
- [17]. R. Moskovitch, C. Feher, N. Tzachar, E. Berger, M. Gitelman, S. Dolev, and Y. Elovici, "Unknown malcode detection using OPCODE representation," 1st European Conference on Intelligence and Security Informatics, (EuroISI 2008), 3-5 Dec., Berlin, Germany: Springer-Verlag, pp. 204-215.
- [18]. Y. Elovici, A. Shabtai, R. Moskovitch, G. Tahan, and C. Glezer., "Applying Machine Learning Techniques for Detection of Malicious Code in Network Traffic", Proceedings of the 30th annual German conference on Advances in Artificial Intelligence, KI 2007, 10-13.
- [19]. Niklas Lavesson , Martin Boldt , Paul Davidsson ,Andreas Jacobsson, "Learning to detect spyware using end user license agreements", Springer-Verlag London Limited 2009.

## Multicloud Deployment of Computing Clusters for Loosely Coupled Multi Task Computing (MTC) Applications

<sup>1</sup>A. Karthik, <sup>2</sup>P. Abdul Wahid, <sup>3</sup>D. Nagarani

<sup>1, 2, 3</sup>(M Tech(Software Engineering) Dept.of Computer Science & Technology,(Student) Sreenidhi Institute of Science & Technology/An Autonomous Institution,Hyderabad,AP, India)

---

**Abstract:** Cloud computing is gaining acceptance in many IT organizations, as an elastic, flexible, and variable-cost way to deploy their service platforms using outsourced resources. Unlike traditional utilities where a single provider scheme is a common practice, the ubiquitous access to cloud resources easily enables the simultaneous use of different clouds. In this project, we explore this scenario to deploy a computing cluster on the top of a multicloud infrastructure, for solving loosely coupled Multi-Task Computing (MTC) applications. In this way, the cluster nodes can be provisioned with resources from different clouds to improve the cost effectiveness of the deployment, or to implement high-availability strategies. We prove the viability of this kind of solutions by evaluating the scalability, performance, and cost of different configurations of a Sun Grid Engine cluster, deployed on a multicloud infrastructure spanning a local data center and three different cloud sites: Amazon EC2 Europe, Amazon EC2 US, and Elastic Hosts. Although the test bed deployed in this work is limited to a reduced number of computing resources (due to hardware and budget limitations), we have complemented our analysis with a simulated infrastructure model, which includes a larger number of resources, and runs larger problem sizes. Data obtained by simulation show that performance and cost results can be extrapolated to large-scale problems and cluster infrastructures.

**Index Terms**—Cloud computing, computing cluster, multicloud infrastructure, loosely coupled applications.

---

### I. Introduction

MULTI-TASK COMPUTING (MTC) paradigm embraces different types of high-performance applications involving many different tasks, and requiring large number of computational resources over short periods of time. These tasks can be of very different nature, with sizes from small to large, loosely coupled or tightly coupled, compute-intensive or data-intensive.

Cloud computing technologies can offer important benefits for IT organizations and data centers running MTC applications: elasticity and rapid provisioning, enabling the organization to increase or decrease its infrastructure capacity within minutes, according to the computing necessities; pay as-you-go model, allowing organizations to purchase and pay for the exact amount of infrastructure they require at any specific time; reduced capital costs, since organizations can reduce or even eliminate their in-house infrastructures, resulting on a reduction in capital investment and personnel costs; access to potentially “unlimited” resources, as most cloud providers allow to deploy hundreds or even thousands of server instances simultaneously; and flexibility, because the user can deploy cloud instances with different hardware configurations, operating systems, and software packages.

Computing Clusters have been one of the most popular platforms for solving MTC problems, especially in the case of loosely coupled tasks (e.g., high-throughput computing applications). However, building and managing physical clusters exhibits several drawbacks:

- 1) major investments in hardware, specialized installations (cooling, power, etc.), and qualified personal;
- 2) long periods of cluster underutilization; and
- 3) cluster overloading and insufficient computational resources during peak demand periods.

Regarding these limitations, cloud computing technology has been proposed as a viable solution to deploy elastic computing clusters, or to complement the in-house data center infrastructure to satisfy peak workloads. For example, the BioTeam has deployed the Univa UD UniCluster Express in an hybrid setup, which combines local physical nodes with virtual nodes deployed in the Amazon EC2. we extend this hybrid solution by including virtualization in the local site, so providing a flexible and agile management of the whole infrastructure, that may include resources from remote providers. However, all these cluster proposals are deployed using a single cloud, while multicloud cluster deployments are yet to be studied. The simultaneous use of different cloud providers to deploy a computing cluster spanning different clouds can provide several benefits:

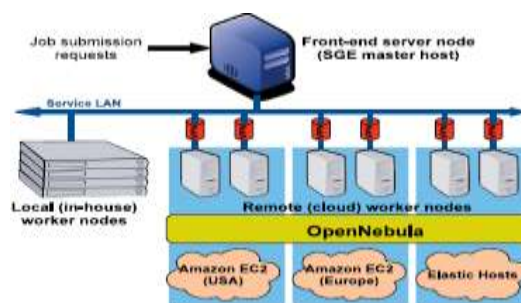
- **High availability and fault tolerance:** the cluster worker nodes can be spread on different cloud sites, so in the case of cloud downtime or failure, the cluster operation will not be disrupted. Furthermore, in this

situation, we can dynamically deploy new cluster nodes in a different cloud to avoid the degradation of the cluster performance.

- **Infrastructure cost reduction:** since different cloud providers can follow different pricing strategies, and even variable pricing models (based on the level of demand of a particular resource type, daytime versus nighttime, weekdays versus weekends, spot prices, and so forth), the different cluster nodes can change dynamically their locations, from one cloud provider to another one, in order to reduce the overall infrastructure cost.

The main goal of this work is to analyze the viability, from the point of view of scalability, performance, and cost of deploying large virtual cluster infrastructures distributed over different cloud providers for solving loosely coupled MTC applications. This work is conducted in a real experimental test bed that comprises resources from our in-house infrastructure, and external resources from three different cloud sites: Amazon EC2 (Europe and US zones) and Elastic Hosts. On top of this distributed cloud infrastructure, we have implemented a Sun Grid Engine (SGE) cluster, consisting of a front end and a variable number of worker nodes, which can be deployed on different sites (either locally or in different remote clouds). We analyze the performance of different cluster configurations, using the cluster throughput (i.e., completed jobs per second) as performance metric, proving that multicloud cluster implementations do not incur in performance slowdowns, compared to single-site implementations, and showing that the cluster performance (i.e., throughput) scales linearly when the local cluster infrastructure is complemented with external cloud nodes. In addition, we quantify the cost of these cluster configurations, measured as the cost of the infrastructure per time unit, and we also analyze the performance/cost ratio, showing that some cloud-based configurations exhibit similar performance/cost ratio than local clusters. Due to hardware limitations of our local infrastructure, and the high cost of renting many cloud resources for long periods, the tested cluster configurations are limited to a reduced number of computing resources (up to 16 worker nodes), running a reduced number of tasks (up to 128 tasks). However, as typical MTC applications can involve much more tasks, we have implemented a simulated infrastructure model, that includes a larger number of computing resources (up to 256 worker nodes), and runs a larger number of tasks (up to 5,000). The simulation of different cluster configurations shows that the performance and cost results can be extrapolated to large-scale problems and cluster infrastructures. More specifically, the contributions of this work are the following:

1. Deployment of a multicloud virtual infrastructure spanning four different sites: our local data center, Amazon EC2 Europe, Amazon EC2 US, and Elastic Hosts; and implementation of a real computing cluster test bed on top of this multicloud infrastructure.
2. Performance analysis of the cluster test bed for solving loosely coupled MTC applications (in particular, an embarrassingly parallel problem), proving the scalability of the multicloud solution for this kind of workloads.
3. Cost and cost-performance ratio analysis of the experimental setup, to compare the different cluster configurations, and proving the viability of the multicloud solution also from a cost perspective.
4. Implementation of a simulated infrastructure model to test larger sized clusters and workloads, proving that results obtained in the real testbed can be extrapolated to large-scale multicloud infrastructures..



**FIG 1: EXPERIMENTAL FRAMEWORK**

## **II. Deployment Of A Multicloud Virtual Cluster**

Fig. 1 shows the distributed cluster testbed used in this work deployed on top of a multicloud infrastructure. This kind of multicloud deployment involves several challenges, related to the lack of a cloud interface standard; the distribution and management of the service master images; and the interconnection links between the service components. A brief discussion of these issues, and the main design decisions adopted in this work to face up these challenges are included in Appendix A of the supplemental material, which can be found on the Computer Society Digital Library at <http://doi.ieeecomputersociety.org/10.1109/TPDS.2010.186>. Our experimental testbed starts from a virtual cluster deployed in our local data center, with a queuing system managed by SGE software, and consisting of a cluster front end (SGE master) and a fixed number of virtual

worker nodes (four nodes in this setup). This cluster can be scaled out by deploying new virtual worker nodes on remote clouds. The cloud providers considered in this work are Amazon EC2 (Europe and US zones) and ElasticHosts. Table 1 shows the main characteristics of in-house nodes and cloud nodes used in our experimental testbed. Besides the hardware characteristics of different testbed nodes, Table 1 also displays the cost per time unit of these resources. In the case of cloud resources, this cost represents the hourly cost charged by the cloud provider for the use of its resources. Appendix B.1 of the supplemental material, which can be found on the Computer Society Digital Library at <http://doi.ieeecomputersociety.org/10.1109/TPDS.2010.186>, gives more details about the cost model used for cloud resources. On the other hand, the cost of the local resources is an estimation based on the model proposed by Walker that takes into account the cost of the computer hardware, the cooling and power expenses, and support personnel. For more information about the application of this cost model to our local data center, see Appendix B.2, which can be found on the Computer Society Digital Library at <http://doi.ieeecomputersociety.org/10.1109/TPDS.2010.186>.

TABLE 1  
Characteristics of Different Cluster Nodes

Site	Arch.	Processor (single core)	Mem. (GB)	Cost (USD/hour)
Local data center (L)	i686 32-bits	Xeon 2.0GHz	1.0	0.04
Amazon EC2 Europe (AE)	i686 32-bits	Xeon 1.2GHz	1.7	0.11
Amazon EC2 USA (AU)	i686 32-bits	Xeon 1.2GHz	1.7	0.10
ElasticHosts (EH)	AMD 64-bits	Opteron 2.1GHz	1.0	0.12

## 2.1 PERFORMANCE ANALYSIS

In this section, we analyze and compare the performance offered by different configurations of the computing cluster, focused in the execution of loosely coupled applications. In particular, we have chosen nine different cluster configurations (with different number of worker nodes from the three cloud providers), and different number of jobs (depending on the cluster size), as shown in Table 2. In the definition of the different cluster configurations, we use the following acronyms: L: local infrastructure; AE: Amazon EC2 Europe cloud; AU: Amazon EC2 US cloud; and EH: Elastic Hosts cloud. The number preceding the site acronym represents the number of worker nodes. For example, 4L is a cluster with four worker nodes deployed in the local infrastructure; and 4L+ 4AE is a eight-node cluster, four deployed in the local infrastructure and four in Amazon EC2 Europe. To represent the execution profile of loosely coupled applications, we will use the Embarrassingly Distributed (ED) benchmark from the Numerical Aerodynamic Simulation(NAS) Grid (NGB) suite. The ED benchmark consists of multiple independent runs of a flow solver, each one with a different initialization constant for the flow field.

NGB defines several problem sizes (in terms of mesh size, iterations, and number of jobs) as classes S, W, A, B, C, D, and E. We have chosen a problem size of class B, since it is appropriate (in terms of computing time) for middle-class resources used as cluster worker nodes. However, instead of submitting 18 jobs, as ED class B defines, we have submitted a higher number of jobs (depending on the cluster configuration, see Table 1 below) in order to saturate the cluster and obtain realistic throughput measures.

Cluster Configurations

Exp. ID.	Worker Nodes	Cluster configuration	No. jobs
1	4	4L	32
2	4	4AE	32
3	4	4AU	32
4	4	4EH	32
5	8	4L+4AE	64
6	8	4L+4AU	64
7	8	4L+4EH	64
8	12	4L+4AE+4AU	128
9	16	4L+4AE+4AU+4EH	128

Table 2: Cluster Configurations

As we have proven in a previous work, when executing loosely coupled high-throughput computing applications, the cluster performance (in jobs completed per second) can be easily modeled using the following equation:

$$r(n) = \frac{r_{\max}}{1 + n_{1/2}/n},$$

Equation 1: Cluster Performance

where  $n$  is the number of jobs completed,  $r_{\max}$  is the asymptotic performance (maximum rate of performance of the cluster in jobs executed per second), and  $n_{1/2}$  is the half-performance length.

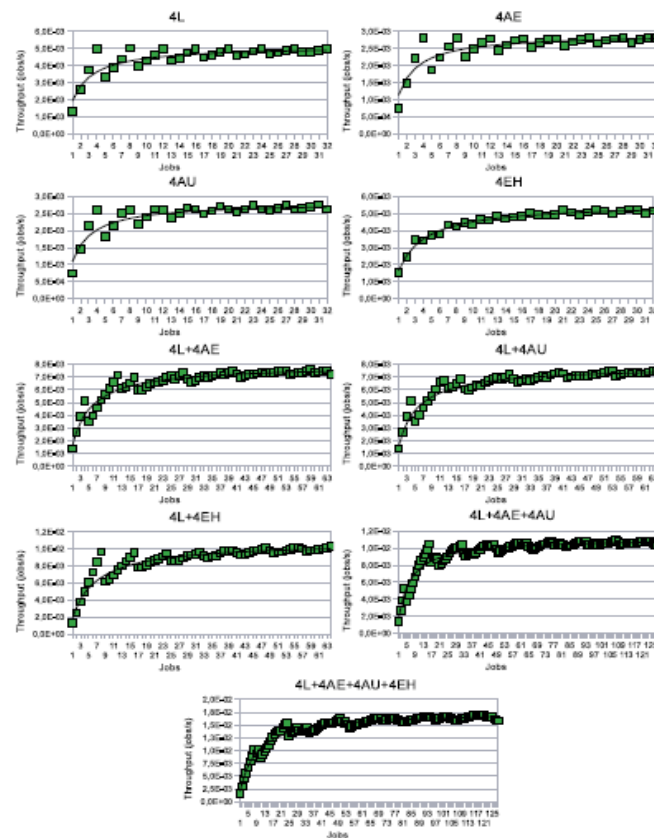


Figure2: Throughput for Different Cluster Configurations

Exp. ID	Cluster configuration	$r_{\infty}$	$n_{1/2}$
1	4L	$5.16 \times 10^{-3}$	1.7
2	4AE	$2.92 \times 10^{-3}$	1.6
3	4AU	$2.83 \times 10^{-3}$	1.6
4	4EH	$5.58 \times 10^{-3}$	2.4
5	4L+4AE	$7.90 \times 10^{-3}$	3.9
6	4L+4AU	$7.81 \times 10^{-3}$	3.9
7	4L+4EH	$1.08 \times 10^{-2}$	5.0
8	4L+4AE+4AU	$1.12 \times 10^{-2}$	5.4
9	4L+4AE+4AU+4EH	$1.76 \times 10^{-2}$	7.6

Table3: Performance Model Parameters

The above table shows the  $r_{\max}$  and  $n_{1/2}$  parameters of the performance model for each cluster configuration. The parameter  $r_{\max}$  can be used as a measure of cluster throughput, in order to compare the different cluster configurations, since it is an accurate approximation for the maximum performance (in jobs per second) of the cluster in saturation. Please, note that we have achieved five runs of each experiment, so data in above table represents the mean values of  $r_{\max}$  and  $n_{1/2}$ .

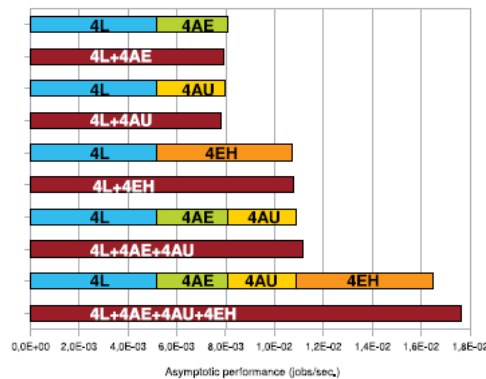
Table 3 shows the  $r_1$  and  $n_1=2$  parameters of the performance model for each cluster configuration. The parameter  $r_1$  can be used as a measure of cluster throughput, in order to compare the different cluster configurations, since it is an accurate approximation for the maximum performance (in jobs per second) of the cluster in saturation. Please, note that we have achieved five runs of each experiment, so data in Table 3 represent the mean values of  $r_1$  and  $n_1=2$  (standard deviations can be found in Appendix C, which can be found on the Computer Society Digital Library at <http://doi.ieeecomputersociety.org/10.1109/TPDS.2010.186>). If we compare 4L and 4EH configurations, we observe that they exhibit very similar performance. This is because of two main reasons: first, worker nodes from both sites have similar CPU capacity (see Table 1); and second, communication latencies for this kind of loosely coupled applications do not cause significant performance degradation, since data transfer delays are negligible compared to execution times, mainly thanks to the NFS file data caching implemented on the NFS clients (worker nodes), which notably reduces the latency of NFS read

operations. On the other hand, the lower performance of 4AE and 4AU configurations is mainly due to the lower CPU capacity of the Amazon EC2 worker nodes (see Table 1).

An important observation is that cluster performance for hybrid configurations scales linearly. For example, if we observe the performance of 4L+4AE configuration, and we compare it with the performance of 4L and 4AE configurations separately, we find that the sum of performances of these two individual configurations is almost similar to the performance of 4L + 4AE configuration. This observation is applicable to all of the hybrid configurations, as shown in Fig.3. This fact proves that, for the particular workload considered in this work, the use of a multicloud infrastructure spanning different cloud providers is totally viable from the point of view of performance and scalability, and does not introduce important overheads, which could cause significant performance degradation.

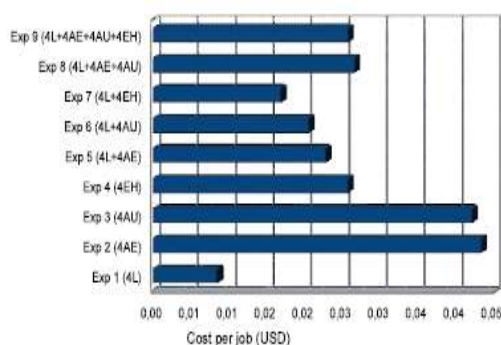
## 2.2 COST ANALYSIS

Besides the performance analysis, the cost of cloud resources also has an important impact on the viability of the multicloud solution. From this point of view, it is important to analyze, not only the total cost of the infrastructure, but also the ratio between performance and cost, in order to find the most optimal configurations.

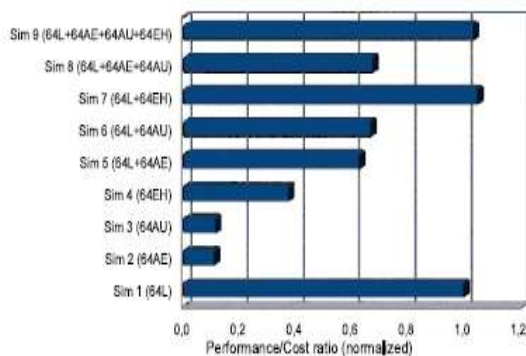


**FIGURE 3: ASYMPTOTIC PERFORMANCE COMPARISON**

The average cost of each instance per time unit is gathered in Table 1. Based on these costs, and using the cost model detailed in Appendix B of the supplemental material, which can be found on the Computer Society Digital Library at <http://doi.ieeecomputersociety.org/10.1109/TPDS.2010.186>, we can estimate the cost of every experiment. However, this cost is not suitable to compare the different cluster configurations, since we are running different number of jobs for every configuration. So, in order to normalize the cost of different configurations, we have computed the cost per job, shown in Fig. 4, by dividing the cost of each experiment by the number of jobs in the experiment. As the cost of local resources is lower than the cloud resources, it is obvious that 4L configuration results in the experiment with the lowest cost per job. Similarly, those experiments including only cloud nodes (e.g., 4AE, 4AU, and 4EH) exhibit higher price per job than those hybrid configurations including local and cloud nodes (e.g., 4L + 4AE, 4L + 4AU, and 4L + 4EH). We also observe that, for the particular workload used in this experiment, configurations including EH nodes result in a lower cost per job than those configurations including Amazon nodes (e.g., 4EH compared to 4AE and 4AU, or 4L + 4EH compared to 4L + 4AE and 4L + 4AU). Regarding these comparative cost results, it is obvious that, for large organizations that make intensive use of Computational resources, the investment on a right-sized local data center can be more cost



**Figure 4: Cost per job for different configurations**



**Figure 5: Performance/cost ratio for different configurations**

effective than leasing resources to external cloud providers. However, for small companies or small research communities, the use of cloudbased computing infrastructures can be much more useful, from a practical point of view, than setting up their own computing infrastructure. Finally, to decide what configurations are optimal from the point of view of performance and cost, we have computed the ratio between cluster performance ( $r_1$ ) and cost per job, as shown in Fig. 5 (we show the normalized values with respect to the 4L case). We can observe that, in spite of the higher cost of cloud resources with respect to the local nodes, there are some hybrid configurations (4L + 4EH, and 4L + 4AE + 4AU + 4EH) that exhibit better performance-cost ratio than the local setup (4L configuration). This fact proves that the proposed multicloud implementation of a computing cluster can be an interesting solution, not only from the performance point of view, but also from a cost perspective.

### III. Simulation Results

The previous section shows the performance and cost analysis for different configurations of a real implementation of a multicloud cluster infrastructure running a real workload. However, due to hardware limitations in our local infrastructure, and the high cost of renting many cloud resources for long periods, the tested cluster configurations are limited to a reduced number of computing resources (up to 16 worker nodes in the cluster), running a reduced number of tasks (up to 128 tasks). However, as the goal of this paper is to show the viability of these multicloud cluster solutions for solving MTC problems, which can involve much more tasks, we have

Sim. ID	Cluster configuration	$r_{\infty}$
1	64L	0.083
2	64AE	0.046
3	64AU	0.045
4	64EH	0.089
5	64L+64AE	0.128
6	64L+64AU	0.128
7	64L+64EH	0.171
8	64L+64AE+64AU	0.172
9	64L+64AE+64AU+64EH	0.259

TABLE 4: Simulation Results for Different Cluster Configurations

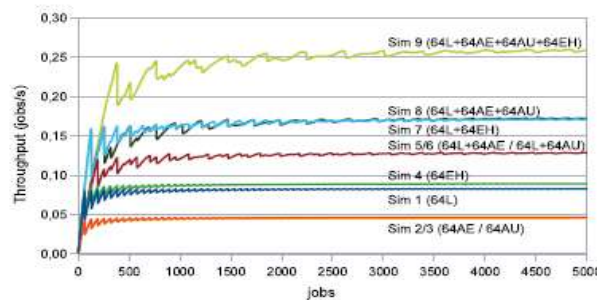


Figure 6: Throughput for simulated cluster configurations.

implemented a simulated infrastructure model, conducted by the real results obtained with the experimental testbed, which includes a larger number of computing resources (up to 256 worker nodes in the simulated infrastructure), and runs a larger number of tasks (up to 5,000). The details and validation of the simulator used in this section can be found in Appendix D of the supplemental material, which can be found on the Computer Society Digital Library at <http://doi.ieeecomputersociety.org/10.1109/TPDS.2010.186>. In addition, Appendix E, analyzes the NFS server performance for the cluster sizes used in our simulation experiments, and proves that, for loosely coupled workloads, the NFS server is not a significant bottleneck, and it has little influence on cluster performance degradation.

We have simulated different cluster configurations (see Table 4), each one running 5,000 independent tasks belonging to the NGB ED class B benchmark. The throughput results of these simulations are displayed in Fig. 6. Table 4 summarizes the asymptotic performance ( $r_1$ , expressed in jobs/s) obtained for each configuration.

Looking at throughput results, we can conclude that the experimental results obtained for the real testbed can be extrapolated to larger sized clusters and higher number of tasks. In fact, we observe in Table 4 that the asymptotic performance for the simulated model scales linearly when we add cloud nodes to the local cluster configuration. For example, we find that the asymptotic performance of 64L+64AE configuration is almost similar to sum of the performances of 64L and 64AE configurations separately. Similar observations can

be made for the rest of cluster configurations. Regarding the cost analysis for simulated infrastructures, summarized in Fig. 7, we observe that cost per job results for the real testbed can also be extrapolated to larger clusters, namely, hybrid configurations including local and cloud nodes (e.g., 64L + 64AE, 64L + 64AU, and 64L + 64EH) exhibit lower cost per job than those configurations with only cloud nodes (e.g., 64AE, 64AU, and 64EH); and configurations including EH nodes result in a lower cost per job than configurations including Amazon nodes (e.g., 64EH compared to 64AE and 64AU, or 64L + 64EH compared to 64L + 64AE and 64L + 64AU). Finally, analyzing the performance-cost ratio of the simulated infrastructures in Fig. 8, we see again a similar behavior to that observed for the real testbed, with some hybrid configurations (64L + 64EH, and 64L + 64AE + 64AU + 64EH) exhibiting better performance-cost ratio than the local setup (64L).

#### **IV. Related Work**

Efficient management of large-scale cluster infrastructures has been explored for years, and different techniques for on-demand provisioning, dynamic partitioning, or cluster virtualization have been proposed. Traditional methods for the on-demand provision of computational services consist in overlaying a custom software stack on top of an existing middleware layer. For example, the MyCluster Project creates a Condor or a SGE cluster on top of TeraGrid services. The Falkon system provides a light highthroughput execution environment on top of the Globus GRAM service. Finally, the GridWay metascheduler has been used to deploy BOINC networks on top of the EGEE middleware. The dynamic partitioning of the capacity of a computational cluster has also been addressed by several projects. For example, the Cluster On Demand software enables rapid, automated, on-the-fly partitioning of a physical cluster into multiple independent virtual clusters. Similarly, the VIOcluster project enables to dynamically adjust the capacity of a computing cluster by sharing resources between peer domains.

Several studies have explored the use of virtual machines to provide custom cluster environments. In this case, the clusters are usually completely build up of virtualized resources, as in the Globus Nimbus project, or the Virtual Organization Clusters (VOC) proposed in [10]. Some recent works, have explored the use of cloud resources to deploy hybrid computing clusters, so the cluster combines physical, virtualized, and cloud resources. There are many other different experiences on deploying different kind of utility services on cloud infrastructures, such as web servers, database appliances, or web service platforms, among others. However, all these deployments only consider a single cloud, and they do not take advantage of the potential benefits of multicloud deployments. Regarding the use of multiple clouds, Keahey et al. introduce in [11] the concept of "Sky Computing," which enables the dynamic provisioning of distributed domains over several clouds, and discusses the current shortcomings of this approach, such as image compatibility among providers, need of standards at API level, need of trusted networking environments, etc. This work also compares the performance of two virtual cluster deployed in two settings: a single-site deployment and a three-site deployment, and concludes that the performance of a single-site cluster can be sustained using a cluster across three sites. However, this work lacks a cost analysis and the performance analysis is limited to small size infrastructures (up to 15 computer instances, equivalent to 30 processors).

#### **V. Conclusions And Future Work**

In this paper, we have analyzed the challenges and viability of deploying a computing cluster on top of a multicloud infrastructure spanning four different sites for solving loosely coupled MTC applications. We have implemented a real testbed cluster (based on a SGE queuing system) that comprises computing resources from our in-house infrastructure, and external resources from three different clouds: Amazon EC2 (Europe and US zones) and ElasticHosts.

Performance results prove that, for the MTC workload under consideration (loosely coupled parameter sweep applications), cluster throughput scales linearly when the cluster includes a growing number of nodes from cloud providers. This fact proves that the multicloud implementation of a computing cluster is viable from the point of view of scalability, and does not introduce important overheads, which could cause significant performance degradation. On

the other hand, the cost analysis shows that, for the workload considered, some hybrid configurations (including local and cloud nodes) exhibit better performance-cost ratio than the local setup, so proving that the multicloud solution is also appealing from a cost perspective.

In addition, we have also implemented a model for simulating larger cluster infrastructures. The simulation of different cluster configurations shows that performance and cost results can be extrapolated to large-scale problems and clusters. It is important to point out that, although the results obtained are very promising, they can differ greatly for other MTC applications with a different data pattern, synchronization requirement, or computational profile.

The different cluster configurations considered in this work have been selected manually, without considering any scheduling policy or optimization criteria, with the main goal of analyzing the viability of the multicloud solution from the points of view of performance and cost. Although a detailed analysis and

comparison of different scheduling strategies is out of the scope of this paper, and it is planned for further research, for the sake of completeness, Appendix F of the supplemental material, which can be found on the Computer Society Digital Library at <http://doi.ieeecomputersociety.org/10.1109/TPDS.2010.186>, presents some preliminary results on dynamic resource provisioning, in order to highlight the main benefits of multicloud deployments, such as infrastructure cost reduction, high availability and fault tolerance capabilities.

### References

- [1] I. Raicu, I. Foster, and Y. Zhao, "Many-Task Computing for Grids and Supercomputers," Proc. Workshop Many-Task Computing on Grids and Supercomputers, pp. 1-11, 2008.
- [2] BioTeam "How to Uniclust and Amazon EC2," technical report, BioTeam Lab Summary, 2008.
- [3] I. Llorente, R. Moreno-Vozmediano, and R. Montero, "Cloud Computing for On-Demand Grid Resource Provisioning," Advances in Parallel Computing, vol. 18, pp. 177-191, IOS Press, 2009.
- [4] Amazon Elastic Compute Cloud, <http://aws.amazon.com/ec2>, 2010.
- [5] Elastic Hosts, <http://www.elastichosts.com/>, 2010.
- [6] E. Walker, "The Real Cost of a CPU Hour," Computer, vol. 42, no. 4, pp. 35-41, Apr. 2009.
- [7] M.A. Frumkin and R.F. Van der Wijngaart, "NAS Grid Benchmarks: A Tool for Grid Space Exploration," J. Cluster Computing, vol. 5, no. 3, pp. 247-255, 2002.
- [8] E. Walker, J. Gardner, V. Litvin, and E. Turner, "Creating Personal Adaptive Clusters for Managing Scientific Jobs in a Distributed Computing Environment," Proc. IEEE Second Int'l Workshop Challenges of Large Applications in Distributed Environments (CLADE '06).
- [9] I. Raicu, Y. Zhao, C. Dumitrescu, I. Foster, and M. Wilde, "Falkon: A Fast and Light-Weight Task Execution Framework," Proc. IEEE/ACM Conf. SuperComputing, 2007.
- [10] E. Huedo, R.S. Montero, and I.M. Llorente, "The GridWay Framework for Adaptive Scheduling and Execution on Grids," Scalable Computing—Practice and Experience, vol. 6, pp. 1-8, 2006.
- [11] J. Chase, D. Irwin, L. Grit, J. Moore, and S. Sprenkle, "Dynamic Virtual Clusters in a Grid Site Manager," Proc. 12th IEEE Symp. High Performance Distributed Computing, 2003.
- [12] P. Ruth, P. McGachey, and D. Xu, "VioCluster: Virtualization for Dynamic Computational Domains," Proc. IEEE Int'l Conf. Cluster Computing, 2005.

## Distributed Path Computation Using DIV Algorithm

<sup>1</sup>P.Vinay Bhushan, <sup>2</sup>R.Sumathi

**Abstract:** Distributed routing algorithms, which can pose significant stability problems in high-speed networks. We present a new algorithm, Distributed Path Computation with Intermediate Variables (DIV), which can be combined with any distributed routing algorithm to guarantee that the directed graph induced by the routing decisions remains acyclic at all times. The key contribution of DIV, besides its ability to operate with any routing algorithm, is an update mechanism using simple message exchanges between neighboring nodes that guarantees loop-freedom at all times. DIV provably outperforms existing loop-prevention algorithms in several key metrics such as frequency of synchronous updates and the ability to maintain paths during transitions. Simulation results quantifying these gains in the context of shortest path routing are presented. In addition, DIV's universal applicability is illustrated by studying its use with a routing that operates according to a non shortest path objective. Specifically, the routing seeks robustness against failures by maximizing the number of next-hops available at each node for each destination.

**Index Terms:** Distance-vector routing, Loop-free routing, DIV Algorithm, Dual Algorithm.

---

### I. INTRODUCTION

NETWORK Links That To Provide The Distributed Path Computation, Is A Core Functionality Of Modern Communication Networks And Is Expected To Remain So, Even Though Some Recent Proposals Contemplate The Use Of More Centralized Solutions. Depending On The Mode Of Information Dissemination And Subsequent Computation Using The Disseminated Information, There Are Two Broad Classes Of Algorithms: (I) Link-State Algorithms (Also Known As Topology Broadcast) And (ii) Distance-Vector Algorithms [1]. In Both Approaches, Nodes Choose Successor (Next-Hop) Nodes For Each Destination Based Only On Local Information, With The Objective That The Chosen Paths To The Destination Be Efficient In An Appropriate Sense—E.G., Having The Minimum Cost. Because End-To-End Paths Are Formed By Concatenating Computational Results At Individual Nodes, Achieving A Global Objective Implies Consistency Across Nodes Both In Computation And In The Information On Which Those Computations Are Based.

Inconsistent information at different nodes can have dire consequences that extend beyond not achieving the desired efficiency. Of particular significance is the possible formation of transient routing loops, which can severely impact network performance, especially in networks with no or limited loop mitigation mechanisms, e.g., no Time-to-Live (TTL) field in packet headers or a TTL set to a large value. In the presence of a routing loop, a packet caught in the loop comes back to the same nodes repeatedly, thereby artificially increasing the traffic load many folds on the affected links and nodes. The problem, a significant issue even with unicast packets, is further aggravated by broadcast packets, which not only are always caught in any loop present in the network, but also generate replicated packets on all network links. The emergence of a routing loop then often triggers network-wide congestion, which can lead to the dropping or delaying of the very same control (update) packets that are needed to terminate the loop; thereby creating a situation where a transient problem has a lasting effect. Avoiding transient routing loops remains a key requirement for path computation in both existing and emerging network technologies, e.g., see [2] for recent discussions.

Link-state algorithms, of which the OSPF [3] protocol is a well-known embodiment, disseminate the state of each node's local links (their status and the node(s) they connect to) to all other nodes in the network by means of reliable flooding. After receiving link-state updates from the rest of the nodes, each node independently computes a path to every destination. The period of potential information inconsistency across nodes is small (a few tens of milliseconds per node for typical present day networks), so that routing loops, if any, are very short-lived. On the flip side, link-state algorithms can have quite high overhead in terms of communication (broadcasting updates), storage (maintaining a full network map), and computation (a change anywhere in the network triggers computations at all nodes). These are some of the reasons for investigating alternatives as embodied in distance-vector algorithms. Distance-vector algorithms couple information dissemination and computation. Information disseminated by a node now consists of the results of its own partial path computations (e.g., its current estimate of its cost to a given destination) that it distributes to its neighbors, which in turn perform their own partial path computations (e.g., its current estimate of its cost to a given destination) that it distributes to its neighbors, which in turn perform their own Computations before further propagating any updated results to their own neighbors. The Distributed Bellman-Ford (DBF) algorithm is a well-known example of a widely used distance-vector algorithm (cf. RIP, EIGRP [4]) that computes a

shortest path tree from a given node to all other nodes. Coupling information dissemination and computation can reduce storage requirements (only routing information is stored), communication overhead (no relaying of flooded packets), and computations (a local change needs not propagate beyond the affected neighborhood). Thus, distance-vector algorithms avoid several of the disadvantages of link-state algorithms, which can make them attractive, especially in situations of frequent local topology changes and/or when high control overhead is undesirable. The research was triggered by a renewed interest in devising light-weight, loop-free path computation solutions for large-scale Ethernet networks.

Distributed path computation with intermediate variables (DIV) algorithm that enables our goals of distributed, light-weight, loop-free path computation. DIV is not by itself a routing algorithm; rather, it can run on top of any routing algorithm to provide loop-freedom. DIV generalizes the *Loop Free Invariant* (LFI) based algorithms, [6] and outperforms previous solutions including known LFI and Diffusing Computation based algorithms, such as the *Diffusing Update Algorithm*. The rules and update mechanism of DIV and their correctness proofs are rather simple, which hopefully will also facilitate correct and efficient implementations.

## II. RELATED WORK

### 2.1. ROUTING LOOPS AND COUNTING-TO-INFINITY

We begin our discussion with a simple classical example of a routing loop and counting-to-infinity which illustrates that these problems can occur quite frequently as they neither require complex topologies nor an unlikely sequence of events. Consider the network shown in Fig. 1(a). In this figure, the nodes compute a shortest path to the destination D. The cost of each link is shown next to the link and the cost-to-destination of the nodes are shown in parenthesis next to the node. We assume that nodes use *poison reverse*; i.e., each node reports an infinite cost-to-destination to its successor node. Thus, node C believes that node A can reach the destination at a cost of 3 whereas node B cannot reach the destination since node B reported a distance of infinity to node C.

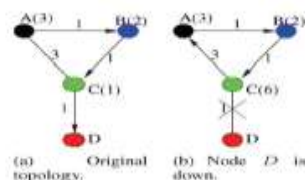


Fig. 1. A simple example of counting-to-infinity problem

Now suppose that the link between nodes C and D goes down, as shown in Fig. 1(b). Node C detects this change and attempts to find a new successor. According to the information node C has at that moment, node A is its best successor. So node C chooses node A as its successor, reports a distance of infinity to node A and distance of 6 to node B. As Fig. 1(b) shows, a routing loop has been created due to node C's choice of successor. To see how counting-to-infinity takes place in this example, note that due to poison reverse, node B believes that the destination is unreachable through node A. Thus, when it receives the update from C containing C's new cost-to-destination as 6, node B simply changes its own cost-to-destination to 7 keeping node C as its successor, reports unreachability to node C and its new cost, 7, to node A. This way, each node increases its cost to D by a finite amount each time. So, unless a maximum diameter of the graph is assumed (e.g., it is 6 in RIP) and the destination declared unreachable once the cost reaches that value, the computation never ends.

### 2.2. DIFFUSING UPDATE ALGORITHM (DUAL)

DUAL, a part of CISCO's widely used EIGRP protocol, is perhaps the best known algorithm. In DUAL, each node maintains, for each destination, a set of neighbors called the *feasible successor set*. The feasible successor set is computed using a *feasibility condition* involving *feasible distances* at a node. Several feasibility conditions are proposed in [5] that are all tightly coupled to the computation of a shortest path. For example, *Source Node Condition* (SNC) uses the feasible successor set to be the set of all neighbors whose current cost-to-destination is *less* than the minimum cost-to-destination seen so far by the node. Note that the definition of a feasible successor set depends on an origin of time, which is defined as the time when the node freshly computes the feasible successor set after it contains no preferred successor.

### 2.3. LOOP FREE INVARIANCE (LFI) ALGORITHMS

A pair of invariants, based on the cost-to-destination of a node and its neighbors, called *Loop Free Invariances* (LFI) is introduced in [7] and it is shown that if nodes maintain these invariances, then no transient loops can form (cf. Section III). Update mechanisms are required to maintain the LFI conditions: [7] introduces *Multiple-path Partial-topology Dissemination Algorithm* (MPDA) that uses a link-state type approach whereas [6] introduces *Multipath Distance Vector Algorithm* (MDVA) that uses a distance vector type approach. Similar

to DUAL, MDVA uses a diffusing update approach to increase its cost-to-destination, thus it also handles multiple overlapping cost-changes sequentially.

DIV combines advantages of both DUAL and LFI. DIV generalizes the LFI conditions, is not restricted to shortest path computations and, as LFI-based algorithms, allows for multipath routing. In addition, DIV allows for using a feasibility condition that is strictly more relaxed than that of DUAL, hence triggering synchronous updates less frequently than DUAL (and consequently, than MPDA or MDVA) as well as limiting the propagation of any triggered synchronous updates. The update mechanism of DIV is simple and substantially different from that of previous algorithms, and allows arbitrary packet reordering/ losses. Moreover, unlike DUAL or LFI algorithms, DIV handles multiple overlapping cost-changes *simultaneously* without additional efforts resulting in potentially faster convergence. Finally, DIV allows an alternate synchronous update mode (in distance vector computations) where a synchronous update goes only one hop, similar to MPDA (note though that MPDA is link-state based), which allows nodes to switch to a new successor faster without creating loops.

### III. DIV

#### 3.1. OVERVIEW

DIV lays down a set of rules on existing routing algorithms to ensure their loop-free operation at each instant. This rule-set is not predicated on shortest path computation, so DIV can be used with other path computation algorithms as well. For each destination, DIV assigns a *value* to each node in the network. To simplify our discussion and notation, we fix a particular destination and from now on, speak of the value of a node. The values can be arbitrary—hence, the independence of DIV from any underlying path computation algorithm. However, usually the value of a node will be related to the underlying objective function that the routing algorithm attempts to optimize and the network topology. Some typical value assignments are as follows: (i) in shortest path computations, the value of a node could be its cost-to-destination; (ii) as done in DUAL, the value could be the minimum cost-to-destination seen by the node from time  $t = 0$ ;

The value could be related to the number of next-hop neighbors for the destination, etc. We, however, impose one restriction on the value assignment: a node that does not have a path to a destination must assign a value of “infinity” (the maximum possible value) to itself. Intuitively, this restriction prevents other nodes from using it as a successor which is sensible since it does not have a path to the destination in the first place. This restriction turns out to be crucial for avoiding counting-to-infinity problems in shortest path environments. The basic idea of DIV is that it allows a node to choose one of its neighbors as a successor only if the value of that neighbor is less than its own value: this is called the *decreasing value property* of DIV.

DIV lays down specific update rules that guarantee that loops are not formed at any time even if the information at different nodes is inconsistent. DIV accomplishes this task by maintaining several intermediate variables that hold a replica of the value of a node at its neighbors and vice versa, and exchanging messages between neighboring nodes. Similar to (but not identical with) DUAL, the update mechanism sends update messages and for some of them, requires an acknowledgment from the neighbor. Depending on the rules for sending acknowledgments, DIV can be operated in one of the following two modes: (i) the *normal mode*, and (ii) the *alternate mode*. In the normal mode, a neighbor can hold on to sending an acknowledgment until its own value is adjusted appropriately. In the alternate mode, on the other hand, the neighbor immediately sends the acknowledgment, but could temporarily lose all paths

The main advantages of DIV are as follows:

- 1) *Separation of Routing and Loop prevention*: DIV separates routing algorithms from the task of transient loop prevention. Emancipating routing decisions from the task of loop-prevention simplifies routing algorithms. In addition, DIV is not restricted to shortest path computations; it can be integrated with other distributed path computation algorithms. where we explore a routing algorithm that attempts to increase the *robustness* of the network in terms of being able to reroute packets *immediately* (i.e., without the need for any route update) without causing a loop after a link or node failure.
- 2) *Reduced Overhead*: When applied to shortest path computations, DIV triggers synchronous updates less frequently as well as reduces the propagation radius of synchronous updates where synchronous updates are time and resource consuming updates that might need to propagate to all upstream nodes before the originator is in a position to update its path. In fact, synchronous updates may altogether be removed if counting-to-infinity is not a significant issue (e.g., mitigated using a TTL); alternate mode.
- 3) *Maintaining a path*: A node can potentially switch to a new successor more quickly, while provably still guaranteeing loop prevention (alternate mode). This is particularly useful in situations where the original path is lost due to a link failure.
- 4) *Convergence Time*: When a node receives multiple overlapping cost updates<sup>3</sup> from its neighbor, DIV allows the node to process and respond to the updates in an arbitrary manner, thus enabling an additional dimension for optimization

5) *Robustness*: DIV can tolerate arbitrary packet reordering and losses without sacrificing correctness.

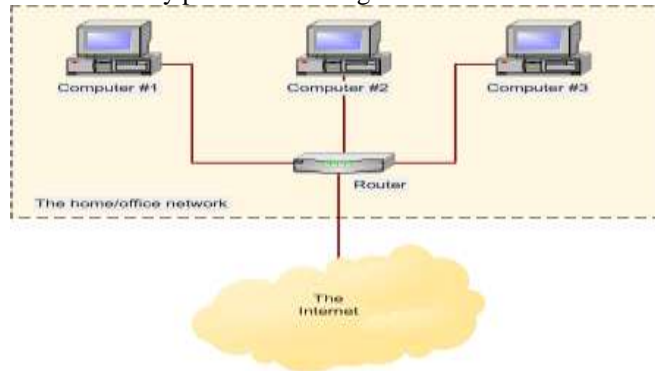


Fig 2. Broadband Router

### 3.2. DESCRIPTION OF DIV

There are four aspects to DIV: (i) the variables stored at the nodes; (ii) two ordering invariances that each node maintains; (iii) the rules for updating the variables; and (iv) two semantics for handling nonideal message deliveries (such as packet loss or reordering). A separate instance of DIV is run for each destination, and we focus on a particular destination, which at a node is, therefore, associated with a given value.

1) *The Intermediate Variables*: Suppose that a node is a neighbor of node. These two nodes maintain intermediate

variables to track each other's values. There are three aspects to each of these variables: whose value is this? who believes in that value? and where is it stored? Accordingly, we define  $V(x; y|x)$  to be the value of node  $x$  as known (believed) by node  $y$  stored in node  $x$ ; similarly  $V(y; x|x)$  denotes value of node  $y$  as known by node  $x$  stored in node  $x$ .

Thus, assuming node  $x$  has neighbors  $\{y_1, y_2, \dots, y_n\}$ , it stores, for each destination:

1) its own value,  $V(x; x|x)$ ;

2) the values of its neighbors as known to itself,

$$V(y_i; x|x) [y_i \in \{y_1, y_2, \dots, y_n\}];$$

3) and the value of itself as known to its neighbors  $V(x; y_i|x) [y_i \in \{y_1, y_2, \dots, y_n\}]$ .

That is,  $2n+1$  values for each destination. This is  $O(N \cdot d)$  storage complexity in a network with  $N$  destination nodes and average node degree  $d$ . The variables  $V(y_i; x|x)$  and  $V(x; y_i|x)$  are called intermediate variables since they endeavor to reflect the values  $V(y_i; y_i|x)$  and  $V(x; x|x)$ , respectively. In steady state, DIV ensures that  $V(x; x|x) = V(x; y_i|x) = V(y_i; y_i|x)$ .

2) *The Invariances*: As stated in the overview of DIV (cf. Section III-A), the fundamental idea of DIV is to ensure that a successor node always has a smaller value. However, a node may not know what the most recent value of one of its neighbors is due to inconsistency in information. Thus, DIV requires each node to maintain at all times the following two invariances based on its set of locally stored variables.

3) *Invariance 1*: The value of a node is not allowed to be more than the value the node thinks is known to its neighbors.

$$V(x; x|x) \leq V(x; y_i|x) \text{ for each neighbor } y_i \quad (1)$$

4) *Invariance 2*: A node can choose one of its neighbors as a successor only if the value of is less than the value of as known by node; i.e., if node is the successor of node, then

$$V(x; x|x) > V(y; x|x) \quad (2)$$

Thus, due to Invariance 2, a node can choose a successor only from its *feasible successor set*  $\{y_i | V(x; x|x) > V(y_i; x|x)\}$ . The two invariances reduce to the LFI conditions if the value of a node is chosen to be its current cost-to-destination.

5) *Update Messages and Corresponding Rules*: There are two operations that a node needs to perform in response to network changes: (i) decreasing its value and (ii) increasing its value. Both operations need notifying neighboring nodes about the new value of the node. DIV uses two corresponding update messages, *Update::Dec* and *Update::Inc*, and acknowledgment (ACK) messages in response to *Update::Inc* (no ACKs are needed for *Update::Dec*). Both *Update::Dec* and *Update::Inc* contain the new value, (the destination), and a sequence number. The ACKs contain the sequence number and the value (and the destination) of the corresponding *Update::Inc* message. DIV lays down precise rules for exchanging and handling these messages which we now describe.

*Decreasing Value*: Decreasing value is the simpler operation among the two. The following rules are used to decrease the value of a node  $x$  to a new value  $V_0$ :

- Node  $x$  first simultaneously decreases the variables  $V(x; x|x)$  and the values  $V(x; y_i|x) \forall i = 1, 2, \dots, n$  to  $V_0$ ;
- Node  $x$  then sends an *Update::Dec* message to all its neighbors that contains the new value  $V_0$ .

- Each neighbor  $y_i$  of  $x$  that receives an Update::Dec message containing  $V_0$  as the new value updates  $V(x; y_i|y_i)$  to  $V_0$ .

*Increasing Value:* Increasing value is potentially a more complex operation, however, conceptually it is simply an inverse operation: in the decrease operation a node first decreases its value and then notifies its neighbors; in the increase operation, a node first notifies its neighbors (and waits for their acknowledgments) and then increases its value. In particular, a node  $x$  uses the following rules to increase its value to  $V_1$ :

- Node  $x$  first sends an Update::Inc message to all its neighbors.
- Each neighbor  $y_i$  of  $x$  that receives an Update::Inc message sends an acknowledgment message (ACK) when it is able to do so according to the rules explained in details below (Section III). When  $y_i$  is ready to send the ACK, it first modifies  $V(x; y_i | y_i)$ , changes successor, if necessary (since the feasible successor set may change), and then sends the ACK to  $x$ ; the ACK contains the sequence number of the corresponding Update::Inc message and the new value of  $V(x; y_i | y_i)$ . Note that in this case it is essential that node  $y_i$  changes successor, if necessary, *before* sending the ACK.
- When node  $x$  receives an ACK from its neighbor  $y_i$ , it modifies  $V(x; y_i | x)$  to  $V_1$ . At any time, node  $x$  can choose any value  $V(x; x | x) \leq V(x; y_i | x) \forall i=1, 2, \dots, n$ .

*Rules for Sending Acknowledgment: The Two Modes:* We now describe how a node decides whether it can send an ACK in response to an Update::Inc message. There are two possibilities: each possibility leads to a distinct behavior of the algorithm, which we refer to as modes.

Suppose that node  $y_i$  received an Update::Inc message from node  $x$ . Recall that node  $y_i$  must increase  $V(x; y_i | y_i)$  before sending an ACK. However, increasing  $V(x; y_i | y_i)$  may remove node  $x$  from the feasible successor set at node  $y_i$ . If node  $x$  is the only preferred node in the feasible successor set of node  $y_i$ , then node  $y_i$  may lose its path if  $V(x; y_i | y_i)$  is increased without first increasing  $V(y_i; y_i | y_i)$ . In such a case node  $y_i$  has two options: 1) first increase  $V(y_i; y_i | y_i)$  and then increase  $V(x; y_i | y_i)$  and send the ACK to node  $x$ ; or 2) increase  $V(x; y_i | y_i)$ , send ACK to node  $x$  and then increase  $V(y_i; y_i | y_i)$ . If a node uses option 1), we say that DIV is operating in its *normal mode*; if a node uses option 2), we say that DIV is operating in *alternate mode*. In the normal mode [i.e., using option 1)], update requests propagate to upstream nodes in the same manner as in DUAL and other previous works. If node  $x$  is not the sole desirable successor of node  $y_i$ , then node  $y_i$  will immediately respond to node  $x$ 's Update::Inc message. Otherwise, node  $y_i$  wants to increase its own value before sending an ACK. Node  $y_i$  issues its own Update::Inc message to *all* its neighbors, including node  $x$ . The set of neighbors of node  $y_i$  that do not depend on node  $y_i$  for reaching the destination *based on their current values* (which includes node  $x$ ), would immediately respond to node  $y_i$  with an ACK. When node  $y_i$  receives ACKs (in response to node  $y_i$ 's Update::Inc message) from all its neighbors, it will send ACK to node  $x$  (as a response to node  $x$ 's Update::Inc message). This process terminates due to acyclicity of the successor graph; when the node  $y_i$  is a "leaf" node (i.e., a node that is not a downstream node for any node), all its neighbors will immediately respond with an ACK.

At this point, we pause to briefly discuss a few basic aspects of "protocol machinery" associated with waiting for ACKs at a node. We assume the existence of a separate "liveness" protocol operating between neighbors and used to detect link/node failures. When waiting for ACKs from neighbors, node maintains a list of pending ACKs for each Update::Inc message it is keeping track of. Nodes are removed from the list either upon receipt of the intended ACK or upon notification of the failure of the liveness protocol to the node. A timer is associated with pending ACKs and retransmission of the Update::Inc message is performed when the timer expires. After a given number of unsuccessful retransmission attempts, the node declares its session to the neighbor failed irrespective of the current status of the liveness protocol, and proceeds to re-initialize it. By following these simple rules, we can ensure that the transmission of ACKs proceeds unimpeded even in the presence of losses and node/link failures.

Turning next to the alternate mode [i.e., using option 2)], we trade simpler and faster processing of ACKs for the risk of having node  $y_i$  without a successor for a period of time (until it is allowed to increase its value). At a first glance, this may seem unwise. However, if node  $x$  originated the value-increase request in the first place because the link to its successor was *down* (as opposed to only a finite cost change), then the old path does not exist and the normal mode has no advantage over the alternate mode in terms of maintaining a path. In fact, in the alternate mode, the downstream nodes get ACKs from their neighbors more quickly and thus can switch earlier to a new successor (which hopefully has a valid path) than in the normal mode. It is not necessary for all nodes to use the same mode (either normal or alternate); each node can make an independent selection. In particular, the following scheme can be used to choose the best mode: we include a bit in the Update::Inc message that indicates whether the request is in response to a loss of old path. Then an intuitive strategy is that nodes use the normal mode if the old path exists, and use the alternate mode if the old path does not exist. However, the normal mode does have one significant advantage over the alternate mode: the counting-to-infinity problem cannot arise in the normal mode whereas there is no such guarantee in the alternate mode. Thus, the previous strategy is perhaps useful only in the cases where counting-to-infinity is not a significant problem or a mitigation mechanism is in place. *Semantics for Handling Message Reordering:* So far, we have

been working under the implicit assumption that all update messages and ACKs come to a node in order and without any loss. In practice both loss and reordering are possible. Thus, it is important to ensure the correctness of DIV under possible packet reordering or packet losses. Towards this goal, we maintain the following two semantics that account for nonzero delays between origination of a message at the sender and its reception at the receiver and possible reordering of messages and ACKs.

6) *Semantic 1: A node ignores an update message that comes out-of-order ( i.e., after a message that was sent later).*

7) *Semantic 2: A node ignores outstanding ACKs after issuing an Update::Dec message.*

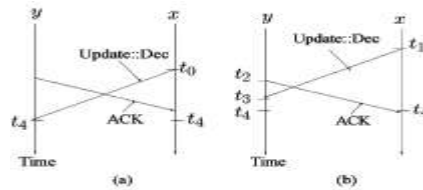


Fig. 2. Two cases of possible message exchanges between two neighboring nodes which would violate (3). Both cases are shown to be contradictory.

These semantics are enforced using the embedded sequence numbers in the update messages (recall that an ACK includes the sequence number of the Update::Inc that triggered it).

### 3.2.1 PROPERTIES OF DIV

The two main properties of DIV are:

- 1) Loop-Free Operation at Every Instant
- 2) Multiple Overlapping Updates and Packet Losses
- 3) Counting-to-Infinity
- 4) Frequency of Synchronous Updates

## IV. Routing Under General Cost Functions

One of the important advantages of DIV is that it is not tied to a particular cost function when computing a routing. We illustrate the benefits of this decoupling using a cost function that instead of the standard shortest path distance function, seeks to maximize the number of next-hops available at all nodes for each destination. The availability of multiple next-hops ensures that the failure of any one link or neighbor does not impede a node's ability to continue forwarding traffic to a destination. A failure results in the loss of at most one next hop to a destination, so that the node can continue forwarding packets on the remaining ones without waiting for new paths to be computed. In other words, the routing is *robust* to local failures. This may be an appropriate objective in settings where end-to-end latency is small and bandwidth plentiful, e.g., as in the previously mentioned large-scale Ethernet networks spanning entire metropolitan areas, which provided some of the early motivations for developing DIV. Multiple available next-hops also improve load balancing by distributing packet transmissions across the multiple associated links.

## V. Results And Performance

### 5.1. PERFORMANCE OF DIV IN SHORTEST PATH ROUTING

DBF, DUAL (using SNC as its feasibility condition), and DIV (using DBF to compute value updates), and compare their performance in terms of loop avoidance<sup>9</sup> and convergence time. The simulations are performed on random graphs with fixed average degree of 5, but in order to generate a reasonable range of configurations, a number of other parameters are varied. Networks with sizes ranging from 10 to 90 nodes are explored in increments of 10 nodes. For each network-size, 100 random graphs are generated. Link costs are drawn from a bimodal distribution: with probability 0.5 a link cost is uniformly distributed in  $[0,1]$ ; and with probability 0.5 it is uniformly distributed in  $[0,100]$ . For each graph, 100 random link-cost changes are introduced, again drawn from the same bimodal distribution. All three algorithms are run on the same graphs and sequences of changes. Processing time of each message is random: it is 2 s with probability 0.0001, 200 ms with probability 0.05, and 10 ms otherwise.

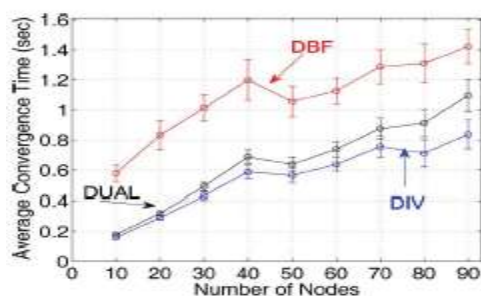


Fig. 7 shows average convergence time

The time from a link cost change until no more update messages are exchanged—of all three algorithms, as the size of the graphs are varied. The vertical bars show the standard deviations. Both DIV and DUAL converge faster than the vanilla DBF; however, DIV performs better, especially for larger graphs. This is because DIV's conditions are satisfied more easily, and hence a synchronous update can be performed faster

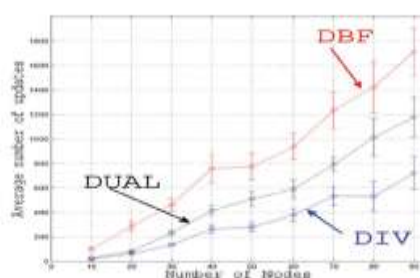


Fig. 8. Number of update messages required for convergence.

Fig. 8 shows the number of control messages used by each algorithm. The trend is very similar to that of the convergence time. DIV issues the least number of control messages before convergence, followed by DUAL and DBF. In DIV-R, node values are initialized to the minimum hop counts to the destinations. At each step, a random node executes DIV-R, that emulates the distributed operation. Iterations are stopped when no improvement is detected for any node. The Shortest-Path-First (SPF) solution is computed using a “black box optimization” approach as in, where link-weights are randomly perturbed to explore the space of possible solutions and optimize the out-degrees of nodes produced by the resulting SPF calculations. The essential difficulty is that the SPF computations for different destinations are coupled through the link-weights since each link uses only one weight for all destinations. The SPF results across different destinations must, therefore, be balanced.

## VI. CONCLUSION

*Distributed Path Computation with Intermediate Variables* (DIV), that achieves this by laying down a rule-set over existing routing algorithms (eg. RIP, IGRP, EIGRP) and defining an efficient update mechanism for enforcing those rules; both are easy to implement. When used with shortest-path computation algorithms, DIV was shown to perform better than current alternatives, such as diffusing update algorithm (DUAL) (and, consequently, the protocols based on DUAL), both analytically and by simulation along various metrics. Another significant Advantage of DIV is that it handles message losses and out-of-sequence delivery, and allows nodes to adopt arbitrary policies for handling multiple overlapping updates, opening up the possibility of various optimizations. Finally, the rule-set and proof of correctness of DIV are intuitive, which should facilitate efficient (and correct) implementations. The benefit of an operation decoupled from shortest path computations was illustrated through the DIV-R algorithm. DIV-R assigns node values with the view of optimizing the network's “local repair” ability in the event of node (or link) failures. This mainly depends on the simulation results of DIV. We believe this flexibility of DIV to have applicability in other environments.

## VII. Acknowledgement

We would like to thank the anonymous referee for helpful Comments

## REFERENCES

- [1] A. Shankar, C. Alaettinoglu, K. Dussa-Zieger, and I. Matta, "Transient and steady-state performance of routing protocols: Distance-vector versus link-state," *Internetw.: Res. Exper.*, vol. 6, no. 2, pp. 59–87, Jun. 1995.
- [2] P. Francois and O. Bonaventure, "Avoiding transient loops during IGP convergence in IP networks," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2005, vol. 1, pp. 237–247.
- [3] J. Moy, "OSPF version 2," Internet Engineering Task Force, RFC 2328, 1998 [Online]. Available: <http://www.rfc-editor.org/rfc/rfc2328.txt>
- [4] R. Albrightson, J. J. Garcia-Luna-Aceves, and J. Boyle, "EIGRP – A fast routing protocol based on distance vectors," presented at the Netw./ Interop. 1994.
- [5] J. J. Garcia-Luna-Aceves, "Loop-free routing using diffusing computations," *IEEE/ACMTrans. Netw.* vol. 1, no. 1, pp. 130–141, Feb. 1993.
- [6] S. Vutukury and J. J. Garcia-Luna-Aceves, "MDVA: A distance-vector multipath routing protocol," in *Proc. IEEE INFOCOM*, Anchorage, AK, Apr. 2001, vol. 1, pp. 557–564.
- [7] S. Vutukury and J. J. Garcia-Luna-Aceves, "A simple approximation to minimum-delay routing," in *Proc. ACM SIGCOMM*, 1999, pp.227–238.
- [8] K. Elmeleegy, A. L. Cox, and T. S. E. Ng, "On count-to-infinity induced forwarding loops in Ethernet networks," in *Proc. IEEE INFOCOM*, 2006, pp. 1–13.
- [9] W. T. Zaumen and J. J. Garcia-Luna-Aceves, "Dynamics of distributed shortest path routing algorithms," in *Proc. ACM SIGCOMM*, Zurich, Switzerland, Sep. 1991, pp. 31–42.

## Author Bibliography



**Mr. P. vinay Bhushan** received his B.Tech in Computer science and Engineering from J.B. Institute Of Engineering and Technology, JNTU, Hyderabad and Pursuing M.Tech in Computer science Engineering from Aurora's Technological And Research Institute, JNTU, Hyderabad.



**Mrs. R. Sumathi** working as Sr.Assistant Professor in the Department of Computer Science and Engineering, in Aurora's Technological And Research Institute with a teaching experience of 5 years. She has received her Master Degree in Technology in computer science From VNR VJIEET JNTU, Hyderabad.

## Adaptive SOA with Interactive Monitoring Techniques and HPS

<sup>1</sup>Kasara Venkata Ajay Reddy, <sup>2</sup>Maddineni Kiran Chowdary,

<sup>3</sup>Dr.R.V.Krishnaiah

<sup>1</sup>Department of CSE, DRK institute of science and technology Hyderabad, India

<sup>2</sup>Department of CSE, DRK college of engineering and technology Hyderabad, India

<sup>3</sup>Principal, Department of CSE, D.R.K Institute of science and technology, Hyderabad, India.

---

**Abstract:** With the advent of distributed technologies and related Integrated Development Environments (IDEs), business collaborations across the world irrespective of the platforms in which applications were built. The underlying technology is Web Services which is based on XML standard and the architecture is SOA (Service Oriented Architecture). Many SOA frameworks came in to existence. However, their primary focus is discovering, composition of services in order to make flexible service oriented distributed enterprise applications. Such systems without human expertise may be incomplete. To overcome this problem, this paper proposes techniques to collaborate services based software with human experts thus forming a knowledge-intensive and people centric web besides sharing and reusing software components in distributed environment. To achieve this, we propose a flexible and adaptive approach which facilitates software services and humans to collaborate using Web Services and its important standards like SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) and UDDI (Universal Description Discovery Integration). This kind of human centric enterprise web has far reaching implications and it can attract the human experts to participate in rendering their best possible services. Interaction monitoring techniques that are adaptive thus realizing standards based people-centric systems.

**Index Terms:** SOA, Web Services, human expertise provisioning, service adaptation

---

### I. INTRODUCTION

In the real world, systems are built on different platforms for various organizations. As the globalization led to business collaboration possibilities, there is great demand for large scale business collaborations. Fortunately, distributed technologies such as Web Services and architectural standards to foster them such as SOA are made available. This has led to an increasing interest in the research circles to focus on adaptive techniques for optimizing business collaborations. However, it is a challenging job as service based systems need the human expertise for optimization. The inherent complexity of such systems makes it difficult to adapt to new requirements, partners, expertise levels and so on. To have a seam less integration of software services and human experts web services technology is used as it can support loosely coupled business collaborations including HPS (Human Provided Services). Schall et al., in [8] proposed a framework that supports involvement of human beings in distributed environment. Shahaf and E. Horvitz [9] developed a document translation service that involves software components and also human experts. However, it proved to have inadequate quality. Afterwards, a new phenomena known as HPS came into existence. Human provided services are essential no matter how good the distributed technologies are. In other words, the complex business systems with collaborative business software components are incomplete without provisioning human experts somewhere in the system. Web services and SOA are the defacto standards that help in realizing such systems with human provisioning too. This has led to the development of socio-technical systems instead of pure technical systems having software components to collaborate each other sans humans interfere [5]. Such systems are loosely coupled in terms of technological components and but having social and human aspects that are tightly coupled. The aspects pertaining to technical include develop, deploy, discover and integrate services with even BPEL for flexible compositions. The social aspect in such systems [5] include humans, their relationships, expertise levels, styles of their behavior, attitudes and apprehensions that differ in degrees among human beings.

When distributed applications with HPS are considered, they are mostly dynamic in nature in terms of people with their skills, experiences; incentives in collaborations are ever changing or evolving in other words. This nature of HPS needs really flexible technologies and architectures. These expectations are reached and realized using Web Services and SOA respectively. They are in technical frameworks that allow flexible interactions among service components and human experts in flexible and adaptive fashion thus forming Mixed Service Oriented Systems. For these systems to work properly they depend on three essential fundamentals known as HPA, adaptive interaction models and adaptive service infrastructure. HPA does mean that certain services are provided by human experts. This is required because many applications run perfectly with input

from domain experts. As systems are to be compatible with changing requirements and also adaptive in nature, human interaction and expert input is essential wherever required. At the same time the interactions must be very flexible and adaptive in nature. The ever changing HPS and the level of quality and expertise demand such flexible software compositions provisioning human inputs. In order to realize these two fundamental aspects, the third one is required i.e. adaptive service infrastructure. This kind of infra should include service features and service behavior at runtime. The system with this infra has four phases namely monitoring phase, analysis phase, adapting phase and execution phase. All interactions in the system are annotated for better apprehension and adaptation at runtime. The monitoring phase observes service interactions and events when system is running. The analysis phase is to analyze service behavior, relations and provide ranking in terms of reliability and dependability using certain interaction metrics as discussed by Skopik et al., in [11]. Depending on the calculated ranking metrics, the relations among the clients and services are provided. Moreover the analysis is context – aware that is to make use of annotations at runtime. In the planning or adaptation phase rewards to the services are given and at the same time punishment may be given to services that do not perform as expected. This has impact on future participation of services and operations in the enterprise system. The adaptations and changes are made visible in the execution phase which actually keeps ball moving such that system provides desired services to all the stakeholders involved.

## II. PRIOR WORK

Web services and SOA are standards driven. They enable heterogeneous applications to be integrated seamlessly. However, a human provided service that is possible to integrate with enterprise applications where machine to machine communication is realized is still in its inception. This is the area where new standards or protocols are required to come into picture. Towards this end efforts have been put forth by the popular vendors in terms of products such as Bpel4People [1] and WS-HumanTask. Major vendors such as SAP, Oracle, IBM, Adobe, and Active Endpoints collaborated in 2007 and published specifications for WS-HumanTask and BPEL4People in an attempt to standardize the human interaction with BPEL processes. As the name implies WS-HumanTask is the specification that provides proper definitions for human tasks, behavior and operations pertaining to such tasks. The service enabled human tasks thus emerged are controlled by a protocol with interoperability feature introduced. BPEL4People specification provides a well defined extension to WS-BPEL which already existed. First class citizen is WS-BPEL that with the new specification resolves the problem of human interaction with BPEL. Fig. 1 shows integration of human experts with workflow of BPEL.

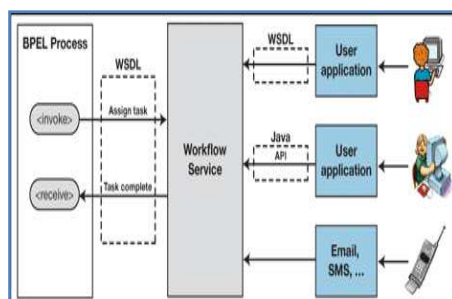


Fig.1 – Human Interaction Integration with BPEL processes [7]

As seen in fig. 1 there is workflow service that makes integration of BPEL process and human expertise in a single framework in distributed environment. In order to realize human interaction with BPEL it has to be aligned with the rich vision of web technology i.e., Web 2.0 that facilitates active contribution from people. HPS framework [8] came into existence to facilitate humans to involve and provide their expert services uniformly. In [9] a system was discussed that involves computer and humans in translation of documents without realizing true SBS (Service Based System). Improved flexibility in distributed systems is made possible with feedbacks from environmental conditions also making the enterprise application to adapt to the conditions. As discussed in [6] MAPE is a mechanism that achieves adaptability to such environmental conditions. This enables the systems to become more and more adaptable and flexible thus facilitating further preventing under-utilization of resources available. Many existing researches in the area of enterprise applications and distributed computing the human element are kept outside the loop. This prevents the rich features that human beings possess can't be realized as part of the framework of the distributed architectures. However, Croudsourcing techniques as presented in [3] that introduces social element in the computing applications. This makes the enterprise applications a provision to have collaboration with human experts as part of the business processes. MAS (Multi Agent Systems) are introduced in [4] which makes use of social techniques. Technique that addresses the challenges in realization of self-configuration in dynamically is found in [12].

### III. SOA WITH PERSONAL PROVISIONING

Human provided services as discussed in [1] can be integrated with SOA. Humans can directly interact with service oriented systems in order to share their expertise with system. However, when human beings are not willing to show their identity directly, avatars can be used in the web environments. The avatars actively participate and act on behalf of humans. Such complex SOA system has to address personal provisioning with various constraints pertaining to given context and the need based human expertise involvement can be realized. The conceptual overview is as shown in fig. 1

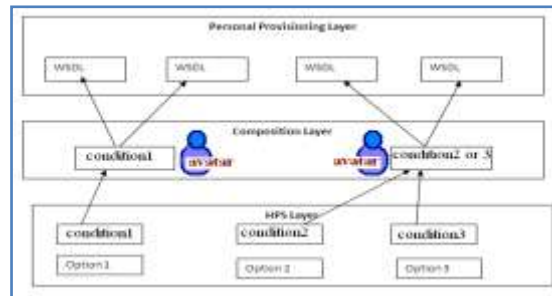


Fig. 2 – SOA with Personal Provisioning

HPS layer enables human provided services to be integrated with other layers. This layer is responsible to provide pre-defined data structures, human services, interfacing standards and other operations. However, the technologies used are very much similar to that of SBS (Software Based Services). This layer provides a web based interface through which human experts can have interaction with other layers. In other words, the HPS layer has applications used by human beings that are having standards driven interaction with rest of the enterprise application. As can be seen in fig. 1, human services are rendered and the avatars represent humans. In order to have this provision a script is required that creates corresponding avatars in the upper layer.

The composition layer is lets humans make use of avatars to represent them in the interactions. Humans have to configure the avatars to suit their requirements. Though the avatar characteristics defined by users are static in nature initially, over a period of time they are personalized and reflect the true nature of humans. This is achieved by maintaining personal profiles and updating them periodically thus enabling the avatars to reflect humans social aspects such as service quality, interaction behavior and interests. Thus the initial static characteristics and the dynamic behaviors characterize avatars. The owners' ongoing activities and preferences cause the profile associated with avatar to be changed.

The personal provisioning layer has services who discover avatars based on the functional and non functional properties and related interfaces. A single service instance gets created for each and every client. The kind of communication is one to one. Software services can serve concurrent request while the humans can serve only single requests at any given point of time. However, personalization can be made to the clients with respect to the avatars they use frequently. This layer is responsible to have web services instances provided by various vendors. These services instances are personalized and one new instance may be created for each client. Listing 1 shows sample script used for service deployment.

```
def profile = profilePlgIn.connect(); //current profile
def Language=datatype.create("tconf.xsd","langTypeA") //imports
def Status=datatype.create("tconf.xsd","statType")
def i=callinterceptor.create() //interaction logging
i.hooks=[in:"RECEIVE", out : "PRE_STREAM"] //hooks on streams
i.code={ m -> ... } //logged message

def arrSrv=webservice.build { //interface definition
// create web service
  TranslationService(binding:"doc,lit", namespace="http://...") {
    interceptors+=i //attach interceptor
    docQueue = [:] //current document queue
    repEP = "" //reporting endpoint
    // create translateDoc operation, return doc refId
    translateDoc(docref:String, fromLang:Language,
      toLang:Language, response:int) {
    def refId = genId(docQueue) //new id for doc
```

```
//active pre-processing with checks
if (profile.checkTotalLoad() < LOAD THR)
docQueue.put(refId,docref)
return refId
}
getJobStatus(refId:int, response:Status){
return report(refId)
}
cancelJob(refId:int){
docQueue.remove(refId)
}
setAsyncReportEP(wsdl:String) {
repEP = wsdl
}
}
}
def srv=arrSrv[0] // only one service declared, take it
def h=host.create("somehost:8181") // import back-end host
srv.deployAt(h) // deploy service at remote back-end host
```

Listing 1 – Service deployment script

Listing 1 shows the service description script. Listing 2 given below shows WSDL excerpt related to document translation.

```
<?xml version="1.0" encoding="utf-8"?>
<definitions ...>
<types>
<schema elementFormDefault="qualified"
targetNamespace="http://socsoa.infosys.tuwien.ac.at/">
xmlns="http://www.w3.org/2001/XMLSchema">
<element name="translateDocRequ">
<complexType>
<sequence>
<element name="document" type="xsd:anyURI" />
<element name="fromLang" type="Language" />
<element name="toLang" type="Language" />
</sequence>
</complexType>
</element>
<simpleType name="Language">
<restriction base="xsd:string">
<enumeration value="German" />
<enumeration value="English" />
</restriction>
</simpleType>
...
</schema>
</types>
<message name="translateDocRequest">
<part name="parameters" element="xsd1:translateDocRequ"/>
</message>
...
<portType name="TSPortType">
<operation name="translateDoc">
<input message="tns:translateDocRequest" Action=.../>
<output message="tns:translateDocResponse" Action=.../>
</operation>
<operation name="getJobStatus"> ... </operation>
<operation name="cancelJob"> ... </operation>
...
</portType>
<binding type="tns:TSPortType" name="..."> ... </binding>
<service name="TranslationService"> ... </service>
</definitions>
```

## Listing 2 – WSDL Excerpt on Document Translation

**IV. Description and Discovery of Services**

A distributed environment where components are to adapt new schemes must use configurable and flexible mechanisms for service discovery. The client gets dynamically compiled metadata before each request with respect to non functional and functional properties in order to update its view. The realization of this is done by using WSDL file of HPS. Discovery of a service is required for two reasons. First of discovery is required in order to find an avatar. Functional properties are used to discover avatar. Then they are clubbed with metrics pertaining to capability metrics that reveal how satisfied the clients of avatar. The search queries are carried out using SPARQL. The structures on which queries are performed are FOAF structures. Responsibility and reliability of avatars determine the project work.

**V. IMPLEMENTATION OF SYSTEM AND ITS ADAPTION**

The implementation of the proposed system is done to support service provisioning and also the expertise provided by human beings. The architectural framework is provided in fig. 3.

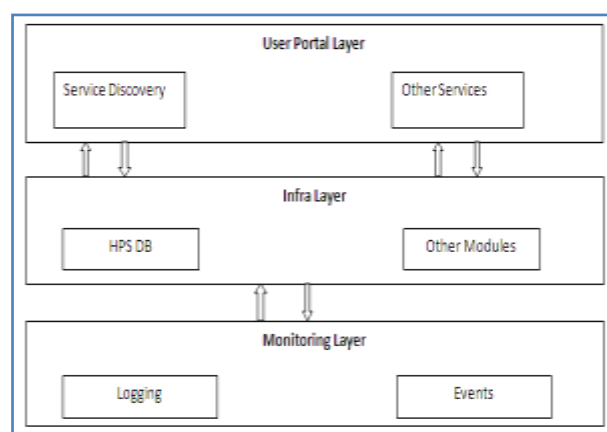


Fig. 3 – Architectural overview

As seen in fig. 3, there are three layers in the architecture. The uppermost layer has service discovery and other required services. Users can discover required services using service discovery module. All such interactions are logged for future retrieval and revisions. The infrastructure layer in the middle contains database of human provided services, other modules and hosting environment. The adaptation module verifies rules and takes necessary steps. The steps include adaptation of HPS and the registry pertaining to HPS. The monitoring layer is responsible to identify and process SOAP interactions and changes related to environment. This layer is responsible to trigger composite events and adaptation.

**VI. Adaptation Strategies**

Timely adaption of require services is an important activity. In the mixed service oriented system, it is essential. The study in this paper is on client – driven interventions, and provider driven interventions which are the reasons that tells the need for service adaptations. Client driven interactions are used to ensure that the unreliable services are not supported in client. This enables the system to replace unreliable services, services that take more time and miss deadlines, is replaced by services that are reliable. Service owners initiate provider driven interactions. This will protect them from clients with malicious intentions. For example, the requests coming from client and cause attacks like replay, denial of attack and so on. Adaptations could be intrusive and distinct. For instance they are behavior adaptations and interface adaptations. Interface adoptions indicate that certain service is updated or deleted from the distributed environment. Such adaptations may be made by system based on the ongoing experience of runtime operations. With respect to behavior adaptations, they are more intrusive and they are capable of changing the behavior of avatars. In [3] it is described how to perform modifications at run time without cause a service going offline. In listing1, a typical SOAP interaction example is given.

```

<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
  xmlns:types="http://socsoa.infosys.tuwien.ac.at/Type"
  xmlns:ts="http://socsoa.infosys.tuwien.ac.at/TS">
  
```

```

<soap:Header>
<types:timesent value="2010-09-01T15:13:21"/> <types:deadline value="2010-09-06T12:00:00"/>
<types:context tags="Web,Services,SOA,research,paper"/>
<wsa:MessageID>uuid:722B1240-...</wsa:MessageID>
<wsa:ReplyTo>http://socsoa....</wsa:ReplyTo>
<wsa:From>http://socsoa....</wsa:From>
<wsa:To>http://socsoa....</wsa:To>
<wsa:Action>http://socsoa....ac.at/Type/RFS</wsa:Action>
</soap:Header>
<soap:Body>
<!-- applied document translation request -->
<!-- schema details omitted -->
</soap:Body>
</soap:Envelope>

```

Listing 3 – SOAP interaction example

Listing 3 shows an interaction with SOAP. The SOAP is an application level communication protocol used in distributed applications. The temporal properties of SOAP calls are utilized in the system. Simple request response patterns are also implemented. This means that a request can be utilized by either avatar or software service. A request can also be rejected if necessary. In these kinds of interactions, dynamic behavior profiles are maintained in order to process request that come from various clients. Many behavioral metrics are context sensitive with respect to expertise area. The services provided by HPS through avatar are also can be ranked. However, these ranking changes over time as the expertise of human beings evolve over time.

## VII. DISCUSSION

Evaluation of the proposed system is done in terms of scalability, performance and functional properties. With respect to adaption it is cyclic with some kind of looping that includes logging and analysis of interactions. The cost of such analysis is measure using the units discussed in [10]. The adaptation performance is provided in fig. 4 and 5.

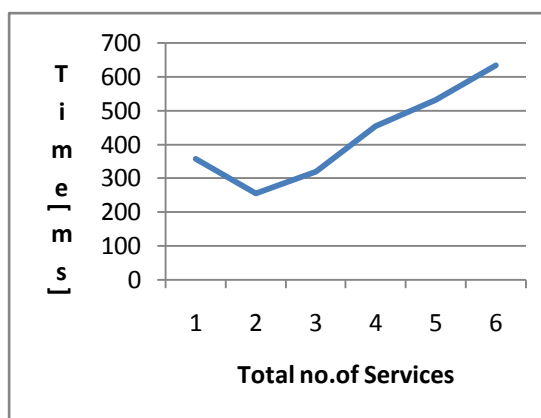


Fig. 4 – server interface update

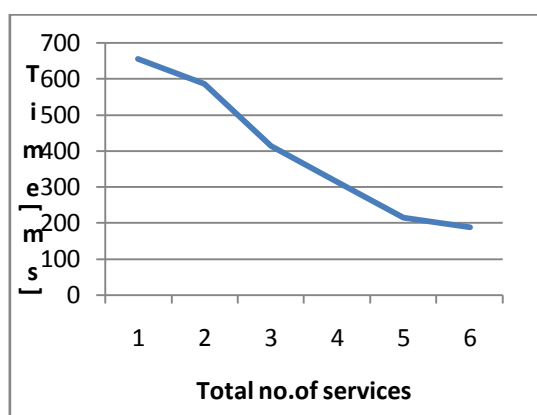


Fig. 5 – service closure update

As seen in fig. 4 and 5, service adaption details are presented. When number of services is increasing the time taken is increased. The time is taken in milliseconds indicates the average time that is required to perform redeployment of an altered WSDL and underplaying an operation of any service. When higher amount of services the time taken is increases while for less number of services the time taken is relatively less. Interface adaptations that need redeployment are 10 times slower than the closure exchanges.

### VIII. SIMULATION SCENARIO

The proposed framework is tested using simulation. The simulation toolkit used for this purpose is “Repeat Symphony”. Human behavior is simulated with respect to interest drifts, working styles, expertise evolution over a period of time, and reliability. Simulation setup is created in environments with 25 clients and 5 avatars. The clients are having certain relations with avatars that represent human provided services. Time is set in such a way that clients send request periodically in 10 rounds interval. To process such request avatar needs 1 or 2 rounds. This way it is found that an avatar can serve only seven clients in average. Simulated agents are used to represent clients and service agents while testing the application after hosting the services. The proposed adaptations are also considered while evaluating the framework.

The experiential setup considers two expertise areas that are unique. 10 different tags are used to describe those areas. Avatars make use of 5 tags for having interest profiles. 3 different tags are used to annotate client requests. Each avatar is associated with clear a profile that is located in area A or B. The area A is known as white nodes while area B is known as black nodes. Initially avatars are not overloaded as they can match with the number of client requirements and expected expertise.

Running experiments caused some problems. These problems are due to human roles’ drifting skills, varying incentives, evolving expertise, specifications related to standards and so on. Here is an assumption that is avatars can shift between interests that cause them to have different levels of expertise. However, in general the deployed services that serve clients in the long run remain same. Only human expertise is evolved over a period of time and that is reflected in the profile of human associated with the corresponding avatar.

### IX. Experimental Results

Experimental results revealed that the clients can change from one avatar to another avatar based on the service requirements and the HPS (Human Provided Service). This is because; the requirements of clients change over time. In the simple approach experiments adaptations are less while the aggressive approach has evidence to many adaptations. As the testing is done with simulation scenario, it has to be implemented in the real world in order to realize the framework in a live environment. The empirical results show that the system is promising and it can be used in real time applications by using the global standards being improved.




### X. CONCLUSION

In this paper we fostered mixed service – oriented system that involves software components and also human provided services. The integration of such services into the architecture of SOA is the new trend that we fostered in this paper. When such environment is realized with human user’s involvement, many social dynamics such as expertise of the user, interests of user, personalization and many other implications pertaining to human users are to be taken care of. This warrants a systematic approach or framework that can address such implications well. In this paper our endeavor is to have a framework which seamlessly integrates the distributed software components through web services technology and human users’ involvement orchestrated well in the framework so as to enable the enterprise applications to have human expertise involvement apart from rich machine to machine interaction among diverse applications of related businesses. We aim, in the future, to improve the framework further by inventing new strategies for monitoring and adaptation.

### Referneces

- [1] A. Agrawal et al. Ws-bpel extension for people (bpel4people), version 1.0., 2007.
- [2] M. Amend et al. Web services human task (ws-humantask), version 1.0., 2007.
- [3] D. Brabham. Crowdsourcing as a model for problem solving: An introduction and cases. *Convergence*, 14(1):75, 2008.
- [4] V. Bryl and P. Giorgini. Self-configuring socio-technical systems: Redesign at runtime. *Int’l Trans. on Syst. Science and App.*, 2(1):31–40, 2006.
- [5] A. Cherns. The principles of sociotechnical design. *Human Relations*, 29(8):783–792, August 1976.
- [6] S. Dobson et al. A survey of autonomic communications. *ACM Trans. Auton. Adapt. Syst.*, 1(2):223–259, 2006.
- [7] Matjaz Juric, Doug H. Todd. SOA Web Services Journal: BPEL Processes and Human Workflow. 12-04-2006. <http://webservices.sys-con.com/read/204417.htm>
- [8] D. Schall, H.-L. Truong, and S. Dustdar. Unifying human and software services in web-scale collaborations. *Internet Comp.*, 12(3):62–68, 2008.
- [9] D. Shahaf and E. Horvitz. Generalized task markets for human and machine computation. In *AAAI*, 2010.
- [10] F. Skopik, D. Schall, and S. Dustdar. Modeling and mining of dynamic trust in complex service-oriented systems. *Information Systems*, 35:735–757, 2010.

- [11] F. Skopik, D. Schall, and S. Dustdar. Modeling and mining of dynamic trust in complex service-oriented systems. *Information Systems*, 35:735–757, 2010.
- [12] J. Zhang and R. J. Figueiredo. Autonomic feature selection for application classification. In *ICAC*, pages 43–52. IEEE, 2006.

	Kasara Venkata Ajay Reddy is student of DRK institute of science and technology, Hyderabad, AP, INDIA. He has received B.Tech Degree computer science and engineering, M.Tech Degree in computer science and engineering. His main research interest includes data mining. Cloud computing.
	Maddineni Kiran Chowdary is student of DRK college of engineering and technology, Hyderabad, AP, INDIA. He has received B.Tech Degree computer science and engineering, M.Tech Degree in computer science and engineering. His main research interest includes data mining. Software Engineering.
	Dr. R.V. Krishnaiah is working as Principal at DRK Institute of Science & Technology, Hyderabad, AP, and India. He has received M.Tech Degree EIE and CSE. His Main Research interest includes Data Mining, Software Engineering.

## Injection of Attacks in MANETs

Konagala Pavani<sup>1</sup>, Dr. Damodaram Avula<sup>2</sup>

<sup>1</sup>(Department of Computer Science and Engineering, Vaagdevi College of Engineering, Jawarharlal Nehru Technological University, Hyderabad, Andhra Pradesh)

<sup>2</sup>(Director, Academic Audit Cell SE, JNTU, Hyderabad, Andhra Pradesh)

---

**Abstract:** Mobile ad-hoc networks are widely used in the tactical battlefield, emergency search and rescue missions. They are also well used in civilian ad-hoc situations like conferences and classrooms due to the ease and speed in setting up such networks. Unlike traditional mobile wireless networks, ad hoc networks do not rely on any fixed infrastructure. Instead, hosts rely on each other to keep the network connected. The wireless ad-hoc networks are mostly vulnerable to security attacks because of its features of open medium, dynamic topology, lack of centralized management and node mobility. In this paper we proposed a means to inject black hole attack and wormhole with a protocol Ad-hoc On Demand Distance Vector (AODV) and conducted experiments using NS2 simulator. The results show that performance of network decreased in presence of attacks.

**Keywords:** MANET, AODV, Black hole attack, Wormhole attack, security

---

### I. Introduction

A mobile ad hoc network (MANET) sometimes called as mobile mesh network consists of a collection of peer mobile nodes that are capable of communicating with each other without help of a fixed infrastructure. As MANETs provide mobile nodes with reliable routing services in the absence of a network infrastructure, they are emerging as a promising platform for a variety of applications in military and civilian domains, sensor networks, rescue operations, students on campus, free internet connection sharing.

Further, MANETs are decentralized networks and the network topology is unpredictably dynamic because of node mobility. As a result, mobile nodes in MANETs act as both hosts and routers and need to discover the dynamic topology and deliver messages by themselves. These mobile nodes establish the routing tables by exchanging routing messages with each other and then deliver the data packets for others. Therefore, developing a system to maintain routing tables reliably is the most fundamental and critical issue related to MANETs.

#### 1.1 Features of wireless open network

- Temporary meshed network formed by a collection of mobile nodes
- Fully self-organized network
- MANETs do not rely on any established infrastructure for the network initialization and operation
- Multi-Hop: due to limited transmission range
- Distributed approach: lack of infrastructure to support network operation
- Dynamic topography: MANET entities are mobile nodes
- Nodes cooperation: basic operations are performed by whole community
- Peer-to-Peer (P2P) analogies

#### 1.2 Security Issues in MANET

MANETs are much more vulnerable to attacks [1][2] than wired networks. This is because of the following reasons.

##### 1.2.1. Open Medium

Eavesdropping is much easier than in wired network because of wireless links. In wired networks an attacker must gain physical access to the network by passing through firewalls and gateways. But in wireless ad hoc networks an attacker may attack from all the directions and damage any node.

##### 1.2.2. Dynamically changing network topology

Mobile node appears and disappears from the network; thereby any malicious node joins in the network without being detected. The dynamic and cooperative nature of ad-hoc networking without a centralized authority for authentication and monitoring is susceptible to attacks that breaks down or exploit the cooperative behavior of the ad-hoc routing. The mobile nodes that are roaming independently may have

inadequate physical protection and can be captured and compromised. Attackers using these captured nodes can perform far more damaging attacks from within the network and such attacks are much harder to detect since the captured nodes will contain the private keys and passwords used within the network.

## **II. Related Work**

Despite the fact of popularity of MANET [3], these networks are very much exposed to attacks. Wireless links also makes the MANET more susceptible to attacks which make it easier for the attacker to go inside the network and get access to the ongoing communication [4]. Different kinds of attacks have been analyzed in MANET and their affect on the network.

## **III. Routing Protocols**

The primary goal of routing protocols [5] in ad-hoc network is to establish optimal path (min hops) between source and destination with minimum overhead and bandwidth consumption so that packets are delivered in a timely manner. A MANET protocol should function effectively over a wide range of networking context from small ad-hoc group to larger mobile Multihop networks. MANET routing protocols can be classified according to the protocols mechanism of route discovery and route update, into three categories: proactive (table-driven), reactive (on-demand) and hybrid.

### **3.1 Proactive routing protocols**

These are used to maintain up-to-date route information from one node to other node in the network.  
eg. DSDV (Destination Sequence Distance Vector)

### **3.2 Reactive routing protocols**

Contrary to proactive routing protocols, reactive routing, as the name may imply, establish and discover the route node only when there is a demand for that node.  
eg: Ad-hoc On-Demand Distance Vectoring [1] [2] (AODV), Dynamic Source Routing (DSR).

### **3.3 Hybrid routing protocols**

These protocols show the characteristics of both reactive and proactive routing protocols. Reactive routing protocols are used to adjust the network connectivity changes using minimal routing overhead by avoiding unnecessary periodic route information update at each node.  
Eg: Zone Routing Protocol (ZRP), ZHLS etc.

## **IV. AODV Routing Protocol**

In this paper we have taken AODV [2] [6] routing protocol. It is a pure on-demand routing protocol. For sending messages to destination, AODV discovers and establishes routes only when needed and maintains only those that remain active. The protocol consist two essential procedures: route discovery and route maintenance. It broadcasts RREQ messages to its immediate neighbors. These neighbors in turn rebroadcast them to their neighbors. This process continues unless the RREQ message reaches the destination. Upon receiving the first RREQ message from the source node, it sends a RREP to the source node following the same reverse path. All the intermediate nodes also set up forward route entries in their table. Upon detecting error in any link to a node, the neighboring nodes forward route error message to all its neighbors using the link. These again initiate a route discovery process to replace the broken link.

## **V. Types of Attacks**

Attacks can be classified as internal and external attacks based on the source of attacks. External attacks are done by unauthorized users and these attackers are not necessarily disconnected from the network, though. The targeted network might be a self-contained entity that is linked to other networks using the same infrastructure or communication technology. Whereas internal attacks are sourced from inside a particular network. A compromised node with access to all other nodes within its range poses a high threat to the functional efficiency of the whole network.

Another type of classification is active attack and passive attack. Some attacks are classified depending on the layer of occurrence.

### **5.1 Network Layer Attack**

The list of different types of attacks on network layer is:

**5.1.1 Black hole attack:** An attacker may create a routing black hole [7], in which all packets are dropped. By sending forged routing packets, the attacker could route all packets for some destination to itself and then discard them.

5.1.2 Wormhole attack: In the wormhole [8] attack, a malicious node tunnels messages received in one part of the network over a low latency link and replays them in a different part. Due to the nature of wireless transmission, the attacker can create a wormhole even for packets not addressed to it, since it can overhear them in wireless transmission and tunnel them to the colluding attacker at the opposite end of the wormhole.

5.1.3 Byzantine Attack: In this attack, a compromised intermediate node or a set of compromised intermediate nodes works in collusion and carries out attacks such as creating routing loops, forwarding packets on non-optimal paths and selectively dropping packets which result in disruption or degradation of the routing services.

5.1.4 Resource Consumption Attack: In this attack, an attacker tries to consume or waste away resources of the other nodes present in the network. The resources that are targeted are:

- Battery power
- Band width
- Computational power

5.1.5 Routing Attack:

There are several attacks[9] [10] which can be mounted on the routing protocols and may disrupt the proper operation of the network such as routing table overflow, packet replication, rout cache poisoning and rushing attack.

5.2 Transport layer attack

5.2.1 Session Hijacking: At first the attacker spoofs the IP address of target machine and determines the correct sequence number. After that he performs DOS attack on the victim. As a result the target system becomes unavailable for some time. The attacker now continues the session with the other system as a legitimate system.

5.3 Application Layer Attack

5.3.1 Repudiation: In simple terms, repudiation refers to the denial or attempted denial by a node involved in a communication of having participated in all or part of the communication.

5.4 Multi Layer Attack

5.4.1 Denial of service (DoS): In this type of attack, an attacker attempts to prevent legitimate and authorized users from the services offered by the network.

5.4.2 Jamming: In this form of attack, the attacker initially keeps monitoring the wireless medium in order to determine the frequency at which the destination node is receiving signals from the sender. It then transmits signals on that frequency so that error free reception at the receiver is hindered.

5.4.3 SYN Flooding: In this form of attack, a malicious node sends a large amount of SYN packets to a victim node, spoofing the return address of the SYN packets.

5.4.4 Distributed DOS Attack: Distributed Denial of Services is more severe form of DoS

In this paper we have implemented black hole and worm hole attacks.

## **VI. Black Hole Attack**

Black hole attack is an active attack type, which leads to dropping of messages. In this type of attack, malicious nodes never send true control messages initially. To carry out a black hole attack, malicious node waits for neighboring nodes to send RREQ messages. When the malicious node receives an RREQ message, without checking its routing table, immediately sends a false RREP message giving a route to destination over itself, assigning a high sequence number to settle in the routing table of the victim node, before other nodes send a true one. Therefore requesting nodes assume that route discovery process is completed and ignore other RREP messages and begin to send packets over malicious node. In the same manner the malicious node attacks all RREQ messages and takes over all routes. Therefore all packets are sent to a point when they are not forwarding anywhere. This is called as black hole attack which swallows all objects and matter. To succeed a black hole attack, malicious node should be positioned at the center of the wireless network.

If malicious node masquerades false RREP message as if it comes from another victim node instead of itself, all messages will be forwarded to the victim node. By doing this, victim node will have to process all incoming messages and is subjected to a sleep deprivation attack. The black hole attack affects the whole network.

## VII. Worm Hole Attack

The AODV routing protocol is vulnerable to wormhole attack since the colluding nodes involved in this attack uses a high speed channel to send messages. In wormhole attack, the attacker receives packets at one point in the network, forwards them to another point in the network through a link with much less latency than the default links used by the network. This link or tunnel between two attackers is called as wormhole. It can be established through a single long-range wireless link or a wired link between the two attackers. Hence it is simple for an attacker to make the tunneled packet arrive sooner than other packets transmitted over a normal multi-hop route.

## VIII. Simulation Environment

Simulations are often used to model natural, machine or human systems in order to gain insight into their functioning. Simulators for communication networks can provide near accurate reproductions of most features in the environment, like noise, probability of loss or alteration of data. Network Simulator ns-2 [11] [12] [13] is used to run MANET simulations. NS-2 is a simulation project developed by the University of California Berkley. NS is part of software of VINT [14] project which is supported by DARPA since 1995. It is one of the most widely used network simulators for wired and wireless networks. NS is an object – oriented, discrete event driven network simulator which is written in C++, with an OTcl interpreter as a frontend, and is available free. It follows the layered approach, and is accompanied by a rich set of protocols.

We run two simulations, one without the attacker node and other including the attacker node. The simulation parameters are shown in TABLE I

We apply the random way-point model in ns-2 to emulate node mobility patterns with a topology of 1000m by 632m. We use UDP/CBR (Constant Bit Rate) as underlying transport protocol, and AODV protocol is used in the experiments. The maximum number of connections is set to be 10 out of which 3 nodes are malicious nodes, traffic rate is 10, the pause time between movements is 10s. These settings are typical ad-hoc settings with adequate mobility and data load overhead, and are used in all our experiments. For evaluation trace files are generated with the normal nodes and attacked nodes.

## IX. Experimental Results

Fig 2 shows the designed network with 10 nodes. In this normal nodes are represented in blue color where as attack nodes are represented in red color i.e. the nodes 0 to 6 are normal nodes and nodes 7 to 9 are attacked nodes. node 7 is a black hole node where as nodes 8 and 9 are worm hole nodes. A tunnel (duplex link) is created between 8 and 9.

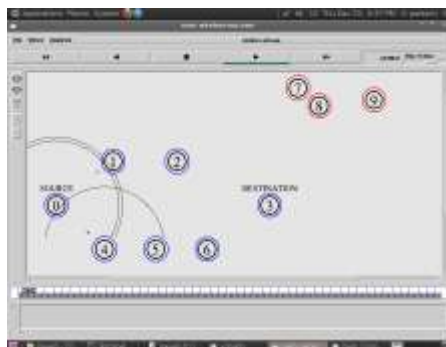


Fig 2: Network in normal condition

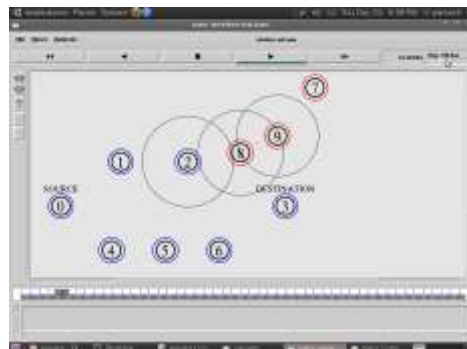
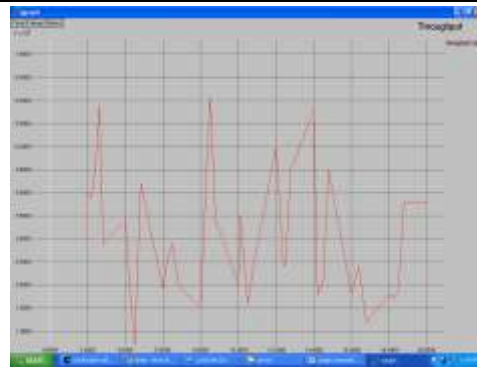
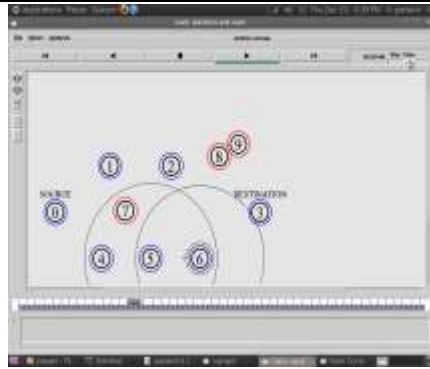


Fig 3: Network under wormhole attack

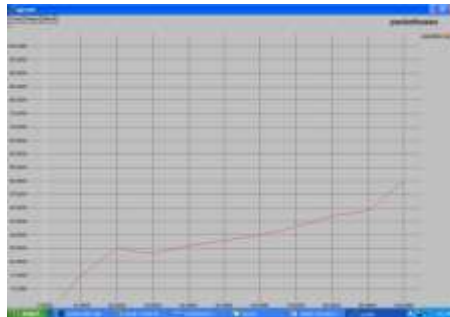
In the fig 3 a wormhole attack is created and packets are transferring from 8 to 9 instead of to 3 which is a destination node.

In fig 4 we are trying to inject a black hole attack. To implement this attack node 7 is moving nearer to source node to show the shortest distance to destination node. So the packets will transfer from source (node 0) to destination (node 3) through node 7. But node 7 as it is a black hole node drops the packets instead of sending to destination (node 3).



**Fig 4: Network under Black hole attack    Fig 5: Graph showing Throughput**

Fig 5 shows number of packets delivered i.e. throughput. In presence of attack nodes number of packets delivered gets decreased.



**Fig 6: Impact of attacks on number of packets loss**

Fig 6 shows the number of packets loss with respect to time which gradually increases as the attacks are created in network. Results also show that the network performance decreases due to the increased attacks.

## X. Conclusion

The security of the Ad Hoc network routing protocols is still an open problem and deserves more research work. In this paper, we have analyzed the security threats faced in an ad hoc network. We have implemented Black hole Attack and Worm hole Attack against AODV routing protocol using Network Simulator-2. This research defines a first fruitful effort towards the definition of an attack implementation for auditing the resilience of Ad Hoc routing protocols and discovering new vulnerabilities in such communication elements. The detection and evasion of such black holes and wormholes in an ad-hoc network is still considered as future challenging task.

## References

- [1] Bounpadith Kannhavong, Hidehisa Nakayama, Yoshiaki Nemoto, Nei Kato, Abbas Jamalipour. A Survey of Routing Attacks in Mobile Ad Hoc Networks, IEEE Wireless Communication, 14 (5), pp. 85-91, 2007
- [2] K. Biswas and Md. Liaqat Ali, Security threats in Mobile Ad-Hoc Network, Master Thesis, Blekinge Institute of Technology Sweden, 22nd March 2007.
- [3] Y. Zhang and W. Lee, Intrusion Detection in Wireless Ad Hoc Networks, Mobicom 2000
- [4] P.V.Jani, Security within Ad-Hoc Networks, Position Paper, PAMPAS Workshop, Sept. 16/17 2002.
- [5] Elizabeth M. Royer et. al. A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks, IEEE Personal Communication, April 1999.
- [6] Yih-Chun Hu, Adrian Perrig, David B. Johnson, Ariadne: A secure On-Demand Routing Protocol for Ad Hoc Networks, MobiCom 2002, September 23-28, 2002, Atlanta, Georgia, USA
- [7] LathaTamilselvan and Dr. V sankaranarayanan,, Prevention of Blackhole Attack in MANET, The 2nd InternationalConferenceonWirelessBroadbandand Ultra Wideband Communications, 2007, pp.21-26.
- [8] Zaw Tun and Aung Htein Maw ,Wormhole Attack Detection in Wireless Sensor Networks ,World Academy of science, Engineering and Technology 46 2008.
- [9] Ping Yi, Zhoulin Dai, Shiyong Zhang, Yiping Zhong A New Routing Attack in Mobile Ad Hoc Networks.In the International Journal of Information Technology Vol. 11 No. 2.
- [10] R.H. Khokhar, Md. A.Ngadi, S. Manda. A Review of Current Routing Attacks in Mobile Ad Hoc Networks, International. Journal of Computer Science and Security, 2 (3), pp. 18-29, 2008.
- [11] <http://www.isi.edu/nsnam/ns/>
- [12] K. Fall and e Varadhan. The ns Manual (formerly ns Notes and Documentation), 2000.
- [13] NS by example <http://nile.wpi.edu/NS/overview.html>, 14 May 2006.
- [14] Virtual InterNetworkTestbed, <http://www.isi.edu/nsnam/vint>, 14 May 2006.

## Navigation Cost Modeling Based On Ontology

<sup>1</sup>Madala Venkatesh, <sup>2</sup>Dr.R.V.Krishnaiah

<sup>1</sup>Department of Computer Science & Engineering, DRK institute of science and technology, Hyderabad, India.

<sup>2</sup>Principal, Department of Computer Science & Engineering, D.R.K Institute of science and technology, Hyderabad, India.

**Abstract:** Web databases when queried result in huge number of records when users of query need a portion of those results which are real interest to them. This problem can be solved using concept hierarchies. Knowledge representation in the form of concepts and the relationships among them (Ontology) allows effective navigation. This paper presents provisions for categorization and ranking in order to reduce the number of results of query and also ensure that the navigation is effective. User should not spend much time to view the actual subset of records he is interested in from the avalanche of records that have been retrieved. For experiments, PubMed database which is in the public domain is used. The PubMed data is medical in nature and organized as per the annotations provided that is instrumental in making concept hierarchies to represent the whole dataset of PubMed. The proposed technique in this paper provides a new search interface that facilitates end users to have effective navigation of query results that are presented in the form of concept hierarchies. Moreover the query results are presented in such a way that the navigation cost is minimized and thus giving rich user experience in this area. The empirical results revealed that the proposed navigation system is effective and can be adapted to real world systems where huge number of tuples is to be presented.

**Index Terms** – Concept hierarchy, effective navigation, annotated data

### I. Introduction

The amount of data provided over World Wide Web (WWW) is increasing rapidly every year. In the past decade it started growing drastically. Especially biomedical data and the literature pertaining to it that reviews the aspects of biomedical data across the globe have seen tremendous growth in terms of quantity. Biological data sources such as [2], [3], and [4] are growing in terms of lakhs of new citations every year. The queries made by people associated with healthcare domain have to search such databases by providing a search keyword. The results are very huge in number and the users are not able to view all the records when they actually need a subset of them. This has led to users to refine query with other keywords and get the desired results after many trials. Here it has to be observed that user time is wasted in refining search criteria and also the navigation of query results which are abundant and bulky. The navigation cost is more as user has to spend lot of time in finding the required subset of rows from the bulk of search results. This problem has been researched in [2], [3], [4] and the problem is identified as information overload. Figure 1 shows static navigation of MeSh hierarchy of biomedical data.



Fig. 1: Static structure of MeSh hierarchy of biomedical data [5]

The solutions are of two types namely categorization and ranking. However, these two can be combined to have more desired results. The proposed system is specially meant for presenting results in such a way that the navigation cost is reduced. For this purpose categorization techniques is used and concept

hierarchies are built. The categorization techniques are supported by simple ranking techniques. The proposed solution uses citations as described in [8] and effectively constructs a navigation tree that can reduce cost of navigation and user's experience is much better when compared with existing systems that do not use these techniques. These techniques are being used by e-Commerce systems to let their users have smooth navigation to the results returned by such systems.

The proposed system uses a cost model that lets it estimate the cost of navigation and make decisions in providing concept hierarchies. The cost of navigation is directly proportionate to the navigation subtree instead of the whole results in the tree. Earlier work on dynamic categorization of query results are in [3], [4], [10] and [6]. They made use of query dependent clusters based on the unsupervised technique. However, they neglect the process of navigation of clusters. In this aspect the proposed system is distinct and provides dynamic navigation on a pre-defined concept hierarchy. Another telling difference between existing systems and the proposed one is that the proposed system uses navigation cost model that minimizes navigation cost no matter what the bulky of search results is. Overall, our contributions are development of a framework for effective navigation of query results; a formal model for cost estimation; algorithm to optimize the results' navigation cost; experimental evidence on the effectiveness.

## II. Overview Of The Proposed Framework

MeSH (Medical Subject Headings) is the hierarchy based on which the framework is built. It is a made up of a concept hierarchy. A concept hierarchy is a labeled tree which has concept nodes and edges with a root node. Each node is identified by a unique ID and associated with a label. As discussed in [9], as per the MeSH concept hierarchy the label of parent is not specific when compared with that of child. Almost all concept hierarchies follow the same conventions. We have built a prototype application whose interface after search operation is as shown in fig. 2.



Fig. 2 – Prototype application

When a query made by user, the application returns results in terms of citations list associated with MeSH concepts. The application constructs an initial navigation tree which is nothing but a concept hierarchy. Every node in that hierarchy is known as a concept. The navigational tree presented initially may contain nodes that are empty. As MeSH is very large hierarchy, the proposed application removes empty nodes and thus reduces size of the navigation tree. The removal of empty node is done carefully by preserving node relationships. The resultant navigation tree is nothing but the initial navigation tree where the empty nodes have been removed to reduce the size of the tree.

By removing empty nodes from the navigation tree, the tree size is reduced. However, the structure of the navigation tree is very big and can't be presented to end user as it is. The proposed application automatically expands the tree at required place to ensure that user navigation cost is less and user will directly see the desired results. Towards this end the application takes care of hiding unnecessary nodes and presenting desired nodes in such a way that the navigation cost of to the user is negligible. We consider a node expansion at given place as an "EdgeCut". In case of trees, in general an EdgeCut is a subset of edges because the removal of edge causes a new component to be created. Visualization of an EdgeCut is shown in fig. 3.

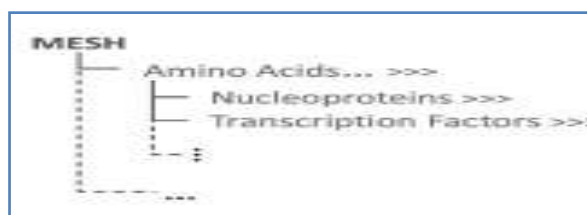


Fig. 3 – Visualization of EdgeCut [1]

Navigation tree with an EdgeCut is shown in fig. 9 on the user interface. The EdgeCut results in creation of sub trees and they are of two types namely lower and upper. Before an EdgeCut is performed, the navigation tree is converted into something known as active tree. It is achieved by annotating root node. The navigation tree is reduced both width and height wise due to the EdgeCut and presentation of the resultant tree. We do assume any user's preference on the results and every part of the tree can be reached by user as the navigation does not result in information loss in the proposed framework. We also use a convention i.e., ">>>>" used to provide hyperlinks that facilitate user to perform EdgeCut operations in recursive manner. Generally we expect EdgeCut to be triggered by user on the lower component of sub trees. However, it is also possible that such operation may occur at upper component sub tree. Fig. 4 shows the resultant tree with hyperlinks in ">>>>" convention.



Fig. 4 – Resultant Tree

### III. Architecture Of Prototype System

The architecture of the prototype application is as shown in fig. 5. The architectural operations are classified into online and offline operations.

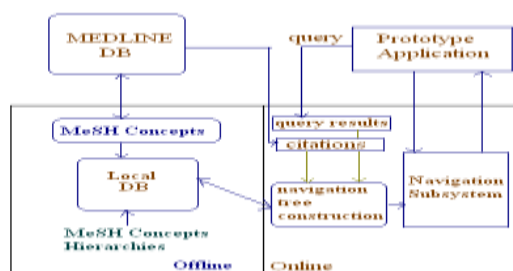


Fig. 5 – Architecture of Proposed Application

As seen in fig. 5, the proposed system operations are classified into online and offline operations. The offline operations include storing MeSH concepts hierarchies into local database, exchanging MeSH concepts between PubMed DB and Local DB. The online operations include question results generation, obtaining citations from MEDLINE database. Navigation tree construction and other navigation sub system operations such as active tree visualization are also online operations. The online operations actively start with query given by end user on biomedical database.

### IV. Navigation And Cost Model

Once user issues a query keyword, the prototype application generates an initial active tree with single component tree. Its root is the root of the MeSH hierarchy. Later on user performs one of the following operations.

**EXPAND:** It takes place when user clicks ">>>>" hyperlink that causes EdgeCut to be performed that shows a set of nodes.

**SHOWRESULTS:** User performs this operation to view results.

**IGNORE:** User can simply ignore a node after looking at its label and moves to next concept.

**BACKTRACK:** This action occurs when user performs undo on the last EdgeCut operation.

User continues these operations until he gets the intended results. We simplify this simple navigational model and call it "TOPDOWN". The TOPDOWN model has only three operations namely EXPAND, SHOWRESULTS and IGNORE.

```
EXPLORE(I(n))
If n is the root
S←EXPAND I(n) // that is S←EdgeCut(I(n))
For each ni in S
EXPLORE (I(ni))
```

```

else, if n is not a leaf –node, choose one of the following:
1.      SHOWRESULTS I(n)
2.      IGNORE I(n)
3.      S ←EXPAND I(n)
For each ni in S
EXPLORE(I(Ni))
Else, choose one of the following: // n is a leaf node
1.      SHOWRESULTS I(n)
2.      IGNORE I(n)

```

Listing 1 – TOPDOWN Navigation Model

The cost model as specified in [2] considers the following to compute navigation cost.

- Number of EXPAND actions.
- Number of concept nodes shown by a single EXPAN action.
- Number of citations presented for a single SHOWRESULTS action.

For each of these things cost of 1 is considered. The cost of exploring a component sub tree  $I(n)$  rooted at node  $n$  is  $\text{cost}(I(n)) =$

$$P_E^N(I(n)). (1 - P_c(I(n)). |L(I(n))| + P_c(I(n)). (B + |S| + \sum_{S \in S} \text{cost}(I_c(S))) ) \quad (1)$$

where normalized  $P_E(I(n))$  is represented as  $P_E^N(I(n))$  thus the sum of normalization of sub tree of the component after an EdgeCut is equal to 1.

## V. Best Edgecut Algorithms

Optimal cost can be computed by recursively listing all possible sets of EdgeCuts. This starts from the root and traverses every concept in the tree. This algorithm is expensive. To overcome this Opt EdgeCut algorithm is proposed which provides minimum expected navigation cost.

```

Algorithm. Opt-EdgeCut
Input: The navigation tree T
Output: The best EdgeCut
1 Traversing T in post order, let n be the current node
2 while n ≠ root do
3 if n is a leaf node then
4 mincost(n, ∅) ← PE (n)*L(n)
5 optcut(n, ∅) ← { ∅ }
6 else
7 C(n) ←enumerate all possible Edge Cuts
for the tree rooted at n
8 Π(n) ←enumerate all possible sub trees
for the tree rooted at n
9 foreach I(n) ∈ Π(n) do
10 compute PE (I(n)) and Pc(I(n))
11 foreach C ∈ C(n) do
12 if C is a valid EdgeCut for I(n) then
13 cost(I(n), C) =
P_E^N(I(n)). (1 - P_c(I(n)). |L(I(n))|
+ P_c(I(n)). (B + |S| + \sum_{S \in S} \text{cost}(I_c(S))) )
14 else
15 cost(I(n), C) = ∞
16 mincost(n, I(n)) ← min Ci ∈ C(n) cost(I(n), Ci)
17 optcut(n, I(n)) ← Ci
18 return optcut(root, E) // E is the set of all tree edges

```

Listing 2 – Opt-EdgeCut Algorithm

The algorithm Opt-EdgeCut is supposed to compute the minimum expected navigation cost required for navigating the tree from bottom up in post order fashion.

## VI. Heuristic-ReducedOpt Algorithm

The Opt-EdgeCut algorithm is computational more expensive and can't be practically used for most of the queries. Therefore, we proposed a new algorithm known as Heuristic-ReducedOpt as shown in listing 3.

```

Algorithm. Heuristic-ReducedOpt
Input: Component subtree I(n), number z of partitions
Output: The best EdgeCut
1  $\hat{z} \leftarrow z$ 
2 repeat
3  $k \leftarrow \lceil \text{EnCT } L(n).PE(n) / \hat{z} \rceil$ 
4 Partitions  $\leftarrow k\text{-partition}(I(n), k)$ 
// call k-partition algorithm [14]
5  $\hat{z} \leftarrow \hat{z} - 1$ 
6 until |Partitions|  $\leq z$ 
7 construct reduced subtree I' (n) from Partitions
8 EdgeCut'  $\leftarrow \text{Opt-EdgeCut}(I'(n))$ 
9 EdgeCut  $\leftarrow$  corresponding of EdgeCut' for I(n)
10 return EdgeCut
  
```

Listing 3 – Heuristic ReducedOpt Algorithm

This algorithm is based on k-partition algorithm [10]. This is adapted for our use here. The algorithm works in bottom-up fashion. For every node (n), the algorithm prunes heaviest children one by one till the weight of *n* falls below k.

## VII. Experimental Evaluation

For evaluating the proposed application, expansion time performance and average navigation cost are considered. The empirical studies are made in a PC with XP as operating system. Oracle 10 g is used as backend and Java is used to implement all algorithms. The proposed application achieves improvement in navigation cost when compared with Top1-LeaveWise. Minimum improvement is 16 percent and maximum improvement is 41 percent. Fig. 6 shows number of EXPAND actions comparison.

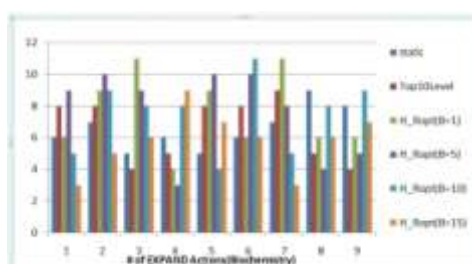


Fig. 6 – Comparison of number of expand operations

As can be seen in fig. 6, ten queries have been presented for six types of algorithms. The X axis takes question numbers while the Y axis represents value. Out of all the algorithms the proposed application and its algorithm. Fig. 7 shows the results of number of concepts shown.

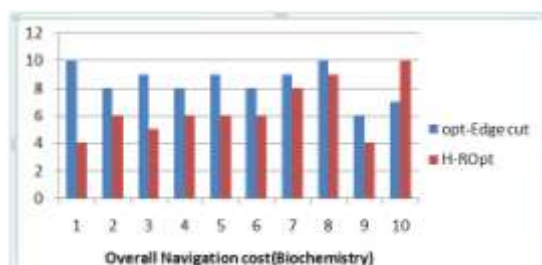


Fig. 7 – Comparison of number of concepts revealed

Fig. 7 shows the number of concepts shown when an EXAPND action takes place. The results revealed that our approach is superior to many other approaches.

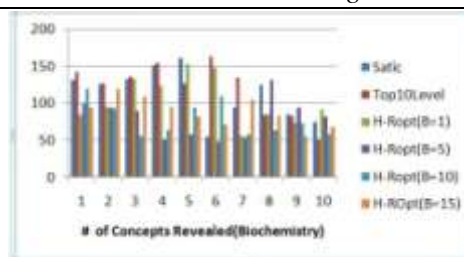


Fig. 8 – Comparison of overall navigation cost

As can be seen in fig. 8 comparison is made on proportional navigation cost of Heuristic-ReducedOpt over Opt-EdgeCut. Opt-EdgeCut algorithm has best results when executed on biochemistry database.

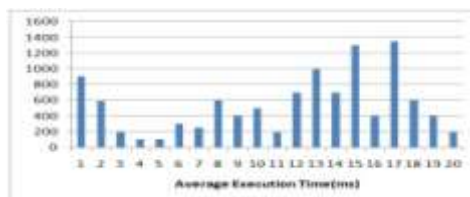


Fig. 9 - Heuristic-ReducedOpt EXPAND performance.

As seen in fig. 9, the average time of Heuristic-ReducedOpt to execute and EXPAND action with respect to each query of table 1. The average values are taken from the number of EXPAND action provided in fig. 6.

## VIII. Conclusion

This paper presents a framework for effective navigation of results of query given to biomedical databases such as PubMed. The problem with query results is that biomedical database returns millions of records and users have to spend some time to navigate to the desired records in the results. This is known as navigation cost. Such problem is also known as information overload problem. The aim of the proposed framework is to address the problem by reducing navigation cost. We achieve this by organizing the results based on the associated MeSH (Medical Subject Headings) hierarchy by proposing a method that works on the resulting navigation tree. The method is known as dynamic navigation method. After applying this method, every node when expanded reveals a subset of required rows thus reducing navigation cost. We have described the underlying cost models and also evaluated them. We developed a prototype application to test the framework's functionality. The empirical results revealed that the proposed framework is effective and can be used in the real time applications.

## References

- [1] Abhijith Kashyap, Vagelis Hristidis, Michalis Petropoulos, and Sotiria Tavoulari (2011), "Effective Navigation of Query Results Based on Concept Hierarchies". IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 23, NO. 4.
- [2] J S. Agrawal, S. Chaudhuri, G. Das and A. Gionis: Automated Ranking of Database Query Results. In Proceedings of First Biennial Conference on Innovative Data Systems Research (CIDR), 2003.
- [3] K. Chakrabarti, S. Chaudhuri and S.W. Hwang: Automatic Categorization of Query Results. SIGMOD Conference 2004: 755-766.
- [4] Z. Chen and T. Li: Addressing Diverse User Preferences in SQLQuery- Result Navigation. SIGMOD Conference 2007: 641-652.
- [5] HON (2010): Health On the Net Foundation: Medical information. Available online at: <http://www.hon.ch/cgi-bin/HONselect?cat+A> [viewed: 15 August 2012]
- [6] A. Kashyap, V. Hristidis, M. Petropoulos, and S. Tavoulari: BioNav: Effective Navigation on Query Results of Biomedical Databases. (Short Paper), ICDE 2009, to appear. Available at <http://www.cs.fiu.edu/~vagelis/publications/BioNavICDE09.pdf>
- [7] S. Kundu and J. Misra, "A Linear Tree Partitioning Algorithm," SIAM J. Computing, vol. 6, no. 1, pp. 151-154, 1977.
- [8] Medical Subject Headings (MeSH®). <http://www.nlm.nih.gov/mesh/>
- [9] Medical Subject Headings (MeSH), <http://www.nlm.nih.gov/mesh/>, 2010.
- [10] Vivísimo, Inc. -Clusty. [Online]. Available: <http://clusty.com/>, 2008



**Madala Venkatesh** has received B.Tech degree in Computer Science and Engineering and pursuing M.Tech in Computer Science and Engineering. His main research includes Data Mining and Cloud Computing.



**Dr. R. V. Krishnaiah** has received Ph.D from JNTU Ananthapur, M.Tech in CSE from JNTU Hyderabad, M.Tech in EIE from NIT (former REC) Warangal and B.Tech in ECE from Bapatla Engineering College. His main research interest includes Data Mining, Software Engineering..

## Preclusion Measures for Protecting P2P Networks from Malware Spread

<sup>1</sup>P.Manasa, <sup>2</sup>Manohar Gosul

Post Graduate Student, Associate Professor  
Department of CSE, SR Engineering College, Warangal, AP, India.

---

**Abstract:** Malicious software or Malware is the software developed with malicious intentions. Hackers use it for spoiling computer programs or to get access to sensitive information. The detection of such malware can be done by writing program which can understand the dynamics of malware. Towards this end this paper presents an analytical model which can effectively characterize the true nature of malware and how it spreads in P2P networks such as Gnutella. The proposed model is compartmental model which involves derivation of network conditions and system parameters in such a way that under those parameters and conditions the underlying P2P network reaches a malware free equilibrium. The proposed model can also perform evaluation of strategies such as quarantine used to control malware spread. Afterwards the model has been enhanced and tested with networks of smart cell phones. The empirical results revealed that the proposed model is effective and useful.

**Index Terms:** Malware, peer-to-peer networks, compartmental model, Bit Torrent and Time to Live (TTL).

---

### I. Introduction

Peer-to-peer networks are networks where there is no specific designation of nodes in the networks. In case of domain network, it is required to designate something as server constantly and other nodes as clients. The P2P model is different from domain model. Peer means a node with same designation. It does mean that in P2P network there is no concept of naming server and client. All the nodes are given equal importance and that is the reason they are known as peers. The usage of P2P networks has become popular and now the usage is spread to various domains which network is possible with certain flexibility. The kind of network required by such systems is the network that has flexible nodes and they are having no much dependency among them. The use of this kind of network has resulted in the flexibility of network connections or services. The proposed network also resembles Gnutella [1] where flooding is the search process. The search process in the network starts with flooding. In this process, a peer forwards the query to its neighbors and then this is repeated until all possibilities are tried in the TTL limits. In the relevant example, the Gnutella systems were affected by the Mandragore Worm. When a node is compromised, it is possible that the compromised node can spread the malware into other systems easier than that of the same without compromised node. The whole search process begins with query in P2P networks. When a node gives query to other node, it is given to a node and gets forwarded if the node is in the TTL limits. The model presented in this paper considers flooding approach as used in Gnutella networks.

### II. Prior Work

Many researchers investigated time for achieving the measurement of P2P systems. Such measurement oriented works such as [2], [3] and [4] have good analytical models. This is meant for temporal evaluation of the information available in the network. These works focus on the regular file transformation. However, they are not applied to the malware spreads in the system rapidly. All of them are specialized in networks like Bit Torrent and take time to extend the networks Gnutella, KaZaA and so on. The research papers [5] and [6] address the issue of worms in P2P networks with the help of a simulation study with respect to worms and their neighbor possible migration mechanisms. On the malware study there are some epidemiological models in order to understand the dynamics of spread of malware in decentralized networks of P2P model. However, this assumption looks wrong as it is really invalid. The fact is that the chances of infecting a peer are limited to its TTL hops away from it and not this entire network.

In the models specified, another important behavior not considered is the incorporation of user behavior. In P2P networks, every user has two states. They are known as “on state” and “off state”. As the name implies, the on state indicates that the peers are actively connected to network while the off state indicates that the peers are not actively connected to network. The infecting probabilities are more when peers are in on state. The peers that go offline in the P2P network having less probability of getting infected. In [7] Bit Torrent is considered as an empirical model for malware spreading while the results of infected nodes in the models where dynamic hit list-based malware is considered in Bit Torrent networks and they are shown in [8] and [9]. These

models are having a known drawback. It is that they are ignoring dynamics of node like transitions offline and other such models in the real world and the models are applicable to only Bit Torrent networks.

In the researches [10] and [11], the authors used a graph model for P2P networks. From this they derive a limiting condition which limits the scope of the adjacency graph for both virus and worm that is prevalent in the network. The models ignored an important fact that if any node in the P2P network is infected that node is likely candidate to spread malware in the network. However, the compromised node can't infect all the other nodes in the network. It has to be remembered that the infected node can infect any other node which is within TTL hop ratios in the network.

In this paper, we present a good model that demonstrates the dynamic of malware in the P2P network which is modeled after Gnutella. This is capable of overcoming the drawbacks of other networks. As the model has two phases it is possible. In the first phase, the peers in the TTL range are quantified. In the second phase, neighborhood information is taken and considered in studying the final dynamics of malware spread.

### III. Malware Propagation Model For P2P Networks

In this paper, this section is an important section as it describes the aspects of the proposed propagation model for Gnutella like P2P networks. The model is assured to ignore regular files and the malware is not ignored. The block diagram of the proposed model is as shown in fig. 1.

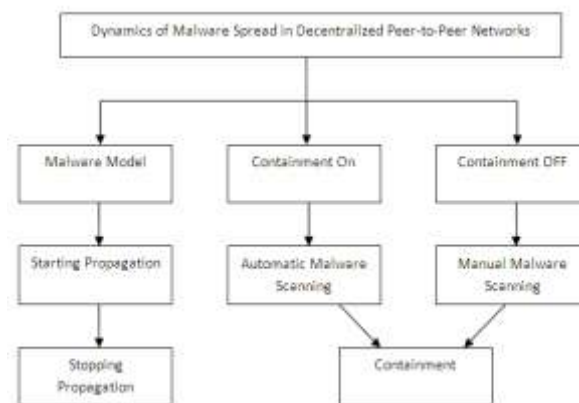


Fig. 1: Block diagram for Proposed Model

#### 3.1 Search Mechanism

In P2P networks, the sharing of information among the nodes is possible first of all by sending the search request. How this request is transmitted over network is important. As per this paper, when a query has to be made, the node is supposed to give the answer. The model used in the network is usage of TTL in the query processing. The query involved TTL bounds and passing message in the TTL bounds. There are two approaches for searching in P2P networks. The first approach is known as flooding where every node sends query to its entire neighbor. When a peer receives affirmative message, it will forward it to next node in TTL. The response of a peer is affirmative if the TTL of the query is greater than zero and then it forwards the query to its neighbors otherwise the query gets discarded. In the proposed system it is assumed that it is sufficient when a file is distinguished as a genuine file or any malware that is sufficient for the proposed system. An approach given in [12] is used now in order to qualify the neighborhood of the search. Then we define the same generating function for PMF (Probability Mass Function).

#### 3.2 Notation and Parameters of P2P Network

The model parameters of the P2P considered for experimentation and their notations are described in table 1.

$\lambda_{on}, \lambda_{off}$	Rate at which off and on peers switch on and off
$\lambda$	Rate at which a peer generates queries
$1/\mu$	Average download time for a particular file
$r_1$	Rate at which peers terminate ongoing downloads
$r_2$	Rate at which peers renew interest in downloading a file after having deleted it
$1/\delta$	Average time for which a peer stores a file

Table 1: Model parameters of P2P network

### 3.3 Compartmental Model

The proposed model is made as compartmental model where the peers are classified into compartments. And each node in the network belongs to specific compartment. The system is based on power law topologies. The compartmental model is based on the concept of node degree [13]. The four classes in the compartmental model are described below in table 2.

$P_S^{(k)}$	Number of peers wishing to download a file.
$P_E^{(k)}$	Number of peers, currently downloading the malware
$P_I^{(k)}$	Number of peers with a copy of the malware.
$P_R^{(k)}$	Number of peers who either have deleted the malware or are no longer interested downloading any file.

**Table 2: Partitioned classes in compartmental model**

### 3.4 Assumptions Made

The following assumptions are made in the proposed system based on the mean-field approach. The assumptions are significant in achieving and characterizing the spread of malware in the decentralized P2P networks and presented here.

- Differential function of time is the number of members in a compartment. It is true for small and big size members present in the compartment.
- The more emphasis is kept on the number of members in each class though differential equations are used in the compartmental model.
- The spread of files in the network is predefined and deterministic. For instance that communication among nodes is starting with search operation and that is based on the flooding approach in the network.
- The size of network is fixed for certain time while making experiments. This is required to characterize the dynamics and spread of malware in the decentralized P2P networks.

With degree “k”, the dynamics of malware in the with respect to classes in the compartmental model can be represented by using the following equations.

$$\begin{aligned} \frac{dP_{Son}^{(k)}}{dt} &= \lambda P_{Son}^{(k)} (1 - (1 - p_{inf})^{z_{av}^{(k)}}) - r_1 P_{Eon}^{(k)} - \\ &\quad + r_2 P_{Ron}^{(k)} - \lambda_{off} P_{Son}^{(k)} + \lambda_{on} P_{Soff}^{(k)} \\ \frac{dP_{Eon}^{(k)}}{dt} &= \lambda P_{Son}^{(k)} (1 - (1 - p_{inf})^{z_{av}^{(k)}}) - r_1 P_{Eon}^{(k)} \\ &\quad - \mu P_{Eon}^{(k)} - \lambda_{off} P_{Eon}^{(k)} + \lambda_{on} P_{Eoff}^{(k)} \\ \frac{dP_{Ion}^{(k)}}{dt} &= \mu P_{Eon}^{(k)} - \delta P_{Ion}^{(k)} - \lambda_{off} P_{Ion}^{(k)} + \lambda_{on} P_{Ioff}^{(k)} \\ \frac{dP_{Ron}^{(k)}}{dt} &= \delta P_{Ion}^{(k)} - r_2 P_{Ron}^{(k)} - \lambda_{off} P_{Ron}^{(k)} + \lambda_{on} P_{Roff}^{(k)} \\ \frac{dP_{Soff}^{(k)}}{dt} &= \lambda_{off} P_{Son}^{(k)} - \lambda_{on} P_{Soff}^{(k)} \end{aligned}$$

### 3.5 Malware Free Equilibrium

R0 represents the basic reproduction model which is used as a metric that is used to govern the stability of the malware free equilibrium globally. The R0 quantifies the number of vulnerable peers where their security has been compromised by some of the infected hosts in their lifetime. It is very clear in experimental results consideration that if R0 is less than 1, it ensures that the epidemic dies out fast and can't take the endemic state [14]. Stability information is considered very important as it can give guarantee that the system is always malware free even if the newly infected peers are introduced. The following formula is used to achieve MFE in the decentralized P2P network.

$$F = [\frac{\partial F_i(x_0)}{\partial x_j}], \quad V = [\frac{\partial v_i(x_0)}{\partial x_j}], \quad 1 \leq i, j \leq m,$$

### 3.6 Quarantine

Quarantine is nothing but removing infected nodes from the network. By doing so in nodes, it is possible that the limiting the malware spread is done and that is the reason the guaranteed nodes limit the spread of malware. The quantization of quarantine rate is done in this section. The basic reproduction is represented as R0. Quarantine does mean that the node is taken out of network. It is assumed that when nodes are removed from network, they P2P network remains good and does not result into disconnected components.

The following equations represent additional terms.

$$\frac{dP_{lon}^{(k)}}{dt} = \mu P_{Eon}^{(k)} - \delta P_{lon}^{(k)} - \lambda_{off} P_{lon}^{(k)} + \lambda_{on} P_{loff}^{(k)} - \eta P_{lon}^{(k)}$$

$$\frac{dP_{Ron}^{(k)}}{dt} = \delta P_{lon}^{(k)} - r_2 P_{Ron}^{(k)} - \lambda_{off} P_{Ron}^{(k)} + \lambda_{on} P_{Roff}^{(k)} + \nu P_Q^{(k)}$$

and the dynamics of  $P_Q^{(k)}$  are described by

$$\frac{dP_Q^{(k)}}{dt} = \eta P_{lon}^{(k)} - \nu P_Q^{(k)}$$

#### IV. RESULTS

This section is used to validate our system through simulation results. The purpose of simulations done is to observe the dynamics of malware spread in decentralized peer-to-peer networks. To achieve this custom simulator is built. The simulation results are analyzed. For thousands of nodes results were simulated and the topology used in power-law topology. As per the system parameters and analytical model described in prior sections, the simulation is carried out and the results were analyzed. Each experiment is performed 20 times and the results are averaged.

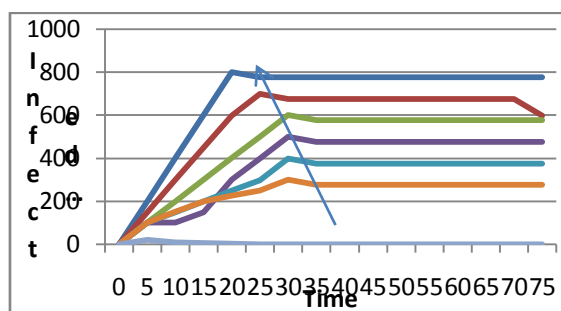


Fig.2: Effect of  $\eta_{on}$  on malware intensity

Fig. 2 shows the results which visualize the time and infected peers. When time grows, the benefits of offline users also more.

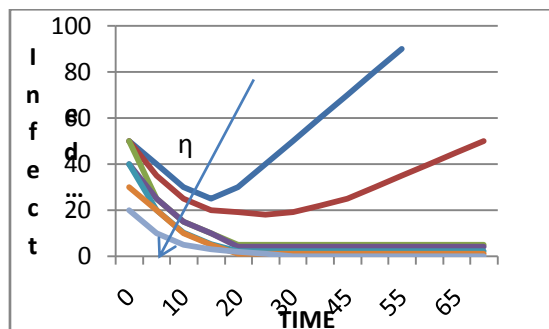


Fig. 3: Effect of quarantine on malware intensity

As can be seen in fig. 3, the effect of quarantine has been plotted. The peers infected is taken in X axis while the time taken for quarantine is given in Y axis.

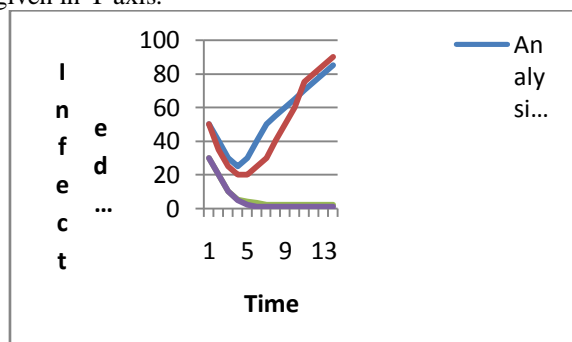
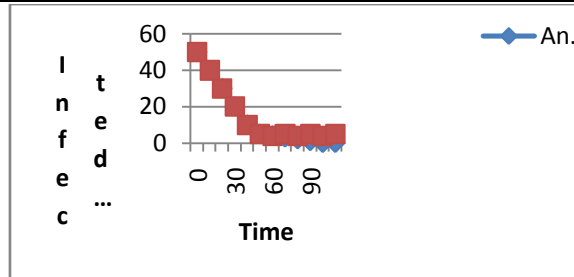


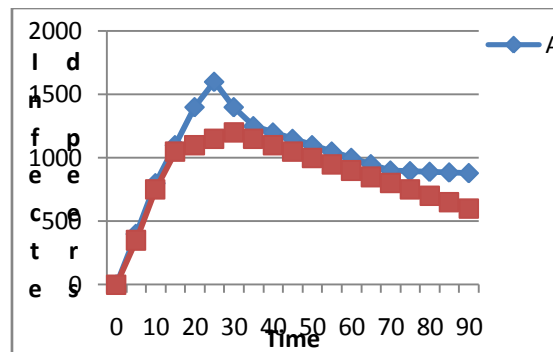
Fig. 4: Effect of quarantine on the system in (29-31) for  $\eta = 0.02$ .

As can be seen in fig. 4 analysis and simulation are shown. The infected peers and the time taken for performing quarantine are visualized in fig. 4.



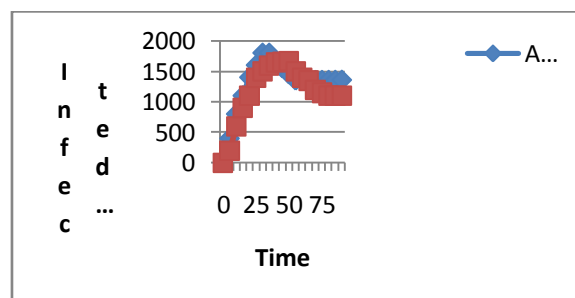
**Fig. 5: Impact of  $\square$  on malware intensity ( $\square=0.005$ )(5-12)**

As can be seen in fig. 5, it is evident that it uses the basic reproduction number to be greater than 1. This is assumed to prevail for an epidemic. When  $R_0$  is less than 1, the number of infected peers is dropping down to zero.



**Fig. 6: Impact of  $\square$  on malware intensity ( $\square=2.0$ )(5-12)**

As can be seen in fig. 6, it is evident that it uses the basic reproduction number to be greater than 1. This is assumed to prevail for an epidemic. When  $R_0$  is not less than 1, then it reaches epidemic proportions. Malware presences in the nodes that run most of the time online are likely to get infected more when compared the same with offline.



**Fig. 7: Influence of offline duration on malware intensity for the system in (5-12)**

As can be seen in fig. 7, the peer's infected and time is plotted in Y and X axes. The influence of offline duration on malware intensity could be found.

## V. Conclusion



In this paper, a model is developed to analyze the spread of malware in Peer – to – Peer networks. The characteristics of malware spreads and its dynamics are incorporated in the analytical model. The model features both offline and online transitional behavior of malware and its dynamics. The proposed model also takes communication patterns for experiments such as size of neighborhood into account. The proposed system also tries to quantify the influence of malware and their ratio in the production. In the estimating of  $R_0$  the above features can help in accurately modeling the spread of malware. The experiments reveal that the proposed analytical model with certain parameters is capable of proving the efficiency of our model.

## References

- [1] Clip2, "TheGnutellaProtocolSpecificationv0.4," [http://www.stanford.edu/class/cs244b/gnutella\\_protocol\\_0.4.pdf](http://www.stanford.edu/class/cs244b/gnutella_protocol_0.4.pdf), Mar. 2001.
- [2] X. Yang and G. de Veciana, "Service Capacity in Peer-to-Peer Networks," Proc. IEEE INFOCOM '04, pp. 1-11, Mar. 2004.
- [3] D. Qiu and R. Srikant, "Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks," Proc. ACM SIGCOMM, Aug. 2004.
- [4] J. Munding, R. Weber, and G. Weiss, "Optimal Scheduling of Peer-to-Peer File Dissemination," J. Scheduling, vol. 11, pp. 105-120, 2007.

- [5] A. Bose and K. Shin, "On Capturing Malware Dynamics in Mobile Power-Law Networks," Proc. ACM Int'l Conf. Security and Privacy in Comm. Networks (SecureComm), pp. 1-10, Sept. 2008.
- [6] L. Zhou, L. Zhang, F. McSherry, N. Immorlica, M. Costa, and S. Chien, "A First Look at Peer-to-Peer Worms: Threats and Defenses," Int'l Workshop Peer-To-Peer Systems, Feb. 2005.
- [7] J. Schafer and K. Malinka, "Security in Peer-to-Peer Networks: Empiric Model of File Diffusion in BitTorrent," Proc. IEEE Int'l Conf. Internet Monitoring and Protection (ICIMP '09), pp. 39-44, May 2009.
- [8] J. Luo, B. Xiao, G. Liu, Q. Xiao, and S. Zhou, "Modeling and Analysis of Self-Stopping BT Worms Using Dynamic Hit List in P2P Networks," Proc. IEEE Int'l Symp. Parallel and Distributed Processing (IPDPS '09), May 2009.
- [9] W. Yu, S. Chellappan, X. Wang, and D. Xuan, "Peer-to-Peer System-Based Active Worm Attacks: Modeling, Analysis and Defense," Computer Comm., vol. 31, no. 17, pp. 4005-4017, Nov. 2008.
- [10] A. Ganesh, L. Massoulie, and D. Towsley, "The Effect of Network Topology on the Spread of Epidemics," Proc. IEEE INFOCOM, 2005.
- [11] Y. Wang, D. Chakrabarti, C. Wang, and C. Faloutsos, "Epidemic Spreading in Real Networks: An Eigenvalue Viewpoint," Proc. IEEE Int'l Symp. Reliable Distributed Systems (SRDS), 2003.
- [12] M. Newman, S. Strogatz, and D. Watts, "Random Graphs with Arbitrary Degree Distribution and Their Applications," Physical Rev. E, vol. 64, no. 2, pp. 026118(1-17), July 2001.
- [13] R. Pastor-Satorras and A. Vespignani, "Epidemic Dynamics in finite size Scale-Free Networks," Physical Rev. E, vol. 65, no. 3, p. 035108(1-4), Mar. 2002.
- [14] O. Diekmann and J. Heesterbeek, "Mathematical Epidemiology of Infectious Diseases: Model Building, Analysis and Interpretation". Wiley, 1999.

### About The Authors

	Manasa Pochu received the Btech Degree in Information Technology from Vaagdevi College of Engineering, Warangal, A.P, India. Currently doing Mtech in Computer Science and Engineering at SR Engineering College, Warangal, India. Her research interests include Networking and Security.
	Manohar Gosul received the B.Tech degree in Computer Science & Engineering from PoojyaDoddappaAppa Engineering College, Gulbarga, India and M.Tech degree in Computer Science & Engineering from PoojyaDoddappaAppa Engineering College, Gulbarga, India. Currently he is an Associate Professor in the department Computer Science & Engineering, SR Engineering College, Warangal, India. His research interests include Data Mining, Advance Databases.

## C-Worm Traffic Detection using Power Spectral Density and Spectral Flatness Measure

Sushma Mergu<sup>1</sup>, G. Dileep Kumar<sup>2</sup>

Post Graduate Student, Assistant Professor

<sup>1, 2</sup>(Department of CSE, SR Engineering College, AP, India)

---

**Abstract:** As Internet and its technologies are improving with rapid pace, there are security threats growing with same pace. The malicious software such as worm is causing such threats to IT systems linked to information super highway. The worms are capable of replicating themselves and infect systems over network. Their traffic propagation can be detected by employing anti worm or virus software. However, there is a new type of worm that can camouflage itself so as to prevent anti worm software from identifying it. The difference between normal worm's traffic and C-worm's traffic can't be found when time domain is considered. However, in terms of frequency definitely it can be differentiated. Based on this hypothesis, this paper presents novel schemes such as PSD and SFM that are capable of differentiating the traffic of C-worm from background traffic. The empirical results revealed that our schemes are effecting in detecting camouflaging worms effectively besides identifying normal worms.

**Keywords** – Traffic propagation, worm, camouflaging worm, time domain, and frequency domain.

---

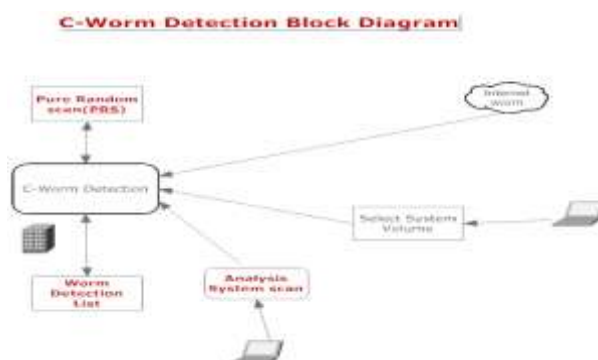
### I. Introduction

Worm is a word with broad meaning. It refers to any program which is malicious in nature. Such program could be a VIRUS, worm etc. They have common features which are also there with biological virus. The common features include that they replicate themselves and also propagate from one machine to another machine. The means of propagation is only through infected storage media and also networks of all kinds including those without wire. Active worms continuously strive to propagate themselves to other systems and make them insecure. This is a problem which has been around ever since the world came across malicious programs for the first time. Some worms include Slammer [2], Sasser [3] and Code-Red [1]. Some worms will work together by forming bonnets and cause more damage to IT systems. The attacks made by such worms include DDoS; attack to obtain sensitive information; destroying data [5] and also put forth unwanted materials such as advertisements. Many such worms are commonly known as malware (malicious software). This includes virus as well. The virus could be boot sector virus, file virus, love virus, time bomb virus, Trojan virus and so on. There is enough evidence in the history that some people have made it their business to create malware and also solutions to prevent them. This is major problem in the world of computers. This man made evil will continue posing threats to IT systems and also cause the businesses to loose confidential information and thus loosing confidence and profits in the business [4], [6].

Researchers also predicting the possibility of malicious programs such as bonnets to collaborate and cause more security threats to IT world. Such collaborated bonnets is known as super bots [7]. As there were reports of worms causing major damage to IT systems, the past few years saw significant research in the area of worms. Worm detection and prevention is an essential task required by all systems involved in IT. Thus the presence of anti-worm software is felt and the same is done through research. The process of identifying the worms by observing their scan traffic much anti-worm software succeed in detecting and also preventing any damage to IT systems. The emergence of Internet and also other networking facilities and communication systems paved way for the increase of threats caused by worms. Studying different kinds of worms and their impact on the IT systems and also prevention techniques are to be given paramount importance. When a worm infects a system, it will propagate its traffic in the system to cause damage to its data. It also strives to propagate the traffic to other systems though infected storage media and networks to other systems in the real world. They keep on identifying IP addresses of systems in the world and infect them though the ways known to them. The common way they follow is generating scan traffic in the time domain and frequency domain. Thus they make all the systems attacked by worms vulnerable to security threats. There is a possibility of loosing companies' sensitive information that leads to collapse of business or losing in revenues in large scale. The patterns of the worms [2], [8], [9] are increasing day by day. The more patterns of worm propagation is known, the more possibility to detect and prevent them. The assumption of all software in the world that is sued to combat worms is that the worms generate scan traffic and try to replicate themselves and infect systems in the same network and remote networks. The patterns are generally having same characteristics so as to enable anti-worm to detect them. However, a new class of worm has come into existence. This new worm is capable of hiding its presence.

It is more dangerous than any other worm. Its name is camouflaging worm (C-worm) as the name suggests, it is capable of hiding its scan propagation so as to let the anti worm software believe that there is no worm exists in the system. That is the reason, this special worm is known as Camouflaging Worm (C-worm).

The attackers are constantly increasing their tactics. They are trying to write malicious programs or worms that can hide and defeat the worm detection systems available in the real world. Thus they are making stealth attack successfully as the worm detection systems fail to distinguish the scan traffic of normal's worms and camouflaging worms with respect to time and frequency domains [10], [11]. The C-worm is capable of hiding its traffic or stopping scan traffic when it detects the process of detecting worm in the system. Thus it is hiding itself in such a way that the normal worm detecting systems are not capable of detecting it easily. They can't distinguish the traffic of scan with respect to time and frequency domains. They may be able to detect differences in time domain but fail in detecting in frequency domain. A new scheme is required in order to detect such worms that do not reveal any presence of it to the worm detection systems in general. As it is quite different from other worms, it hides any noticeable traffic that reflects its presence. To achieve this, it manipulate scan traffic in such a way that the traffic can't be detected by the systems where worm detection schemes are running. Therefore, the worm detecting systems are useless in case of C-worms that cause more damage to IT systems when compared with traditional worms [12], [13]. The C-worms are capable of achieving their goal of propagating the systems in the real world and cause damage to the systems without being detected by worm detecting systems.



**Fig. 1: Block Diagram for C-Worm Detection**

We propose new schemes in detecting C-worms in this paper. We achieve this based on the hypotheses that make two points clear. The first point is that C-worms traffic is different from other worms. The second important point is that it is not possible to differential scan traffic of C-worms with traditional worms in time domain. However, they can be distinguished to find differences in terms of frequency domain. These hypotheses make it easy to make experiments and prove the hypotheses with relative ease. This observation makes difference between the detection of worms and also C-worms. In the proposed system we develop two new schemes that make it possible. They are not able to differentiate normal worms and C-worms in terms of time domain while they are capable of detecting such patterns of worms in the frequency domain. PSD and SFM are the two techniques that are used to achieve the worm detection system with a difference. For every PSD the c-worm traffic shows less SFM and this is the evidence that the camouflaging worm hides itself and when reported this is known to others as well. The scan traffic of the C-worm could be based on the port number of IP address. It uses both based on the requirement. The experiments reveal that our schemes are effective when compared with many existing worm detection systems. Moreover, we also used many metrics such as DR (Detection Rate) and DT (Detection Time) and MIR (Maximal Infection Ratio) in order to evaluate the efficiency of the proposed schemes.

## **II. Related Work**

Worms are similar to biological viruses that cause damage to health of human beings and other animals. They have features like self-propagation and replication. These features are with malicious programs that cause problems to computers in the given network. Such malware is known as worms. The worm which is scanning traffic through IP address and also port number of systems and trying to propagate itself to new systems in the networking domain is known as active worm. As the worms are capable of damaging IT systems, the need for research to prevent the same has been felt. In accordance with this, researchers spend considerable time on this topic and still it needs further improvements [9], [16]. Active worms can use many ways in which they can propagate themselves from one system to another system. One such way is Pure Random Scan (PRS). This is a kind of scan in which the worms continuously and randomly find IP addresses and ports of other systems and propagate itself to those systems which IP addresses are known to the worm. Other ways in which worms can propagate include file sharing, email, network port scanning and instant messaging or chatting [17].

When some IP addresses are known to the worms, they try to propagate themselves by maintaining a hit list and following the strategies to propagate themselves into those systems whose IP addresses are scanned by worm. The worms also split IP address space in order to avoid repetition of work and thus they divide and conquer in terms of scanning and propagating themselves into new networks. Some researches also considered developing a new topology that is attack resilient with respect to worms [18], [19].

There is a special category of worm that is quite different from the other worms described above. This worm is capable of manipulating its scan traffic and thus making it possible that the traditional worm detecting systems fail to help in this regard. A new camouflaging worm thus created is causing more damage to IT world as it is not detected by conventional anti-worm programs. Essentially the worms that hide their scan traffic are polymorphic in nature [20], [21]. Such worms are known as Camouflaging worms as they are hiding their presence and making the normal worm detection systems vulnerable. With respect to stealthiest, the normal worm and C-worm are having certain similarities. Both are generating same traffic and finding the similarities such as both can detect the difference between the normal traffic and worm's scan traffic. The other main difference between them is that the traditional worm detectors can't find difference while the proposed scheme can distinguish the traffic of the C-worm in the time domain. However, it is challenging to find such result from other schemes. The proposed scheme finds the difference in scan traffic of normal worm and systematic in frequency domain though in time domain it can't differentiate the existing worms and new kind of worm known in frequency domain. The new class of worm is named "C-Worm". Due to self propagation nature of C-worm and its ability to manipulate to hide its presence in the system by camouflaging technique. The actual detection of worms is provided in the next section.

## **2.1 Worm Detection**

The detecting of worms is the research that has been around for the past many years. The reason for this kind of research is to protect IT systems by preventing malicious code from entering into our network. The detection systems of worm are of two types. They are known as host – based detection and the second one is network-based detection. Host based detection systems are to analyze the scan traffic in the hosts they are available. They identify and prevent worms whose main purpose is propagating from one system to another system and spoil the whole communication system. [23], [24]. The network based systems for detecting worms use different approach. They use IP addresses of the scanned systems and then try to propagate themselves. Many researchers worked on these kind of systems [12], [13]. The worms that spread to other machine can be prevented by employing effective worm detection systems. However, the existing worm detection systems are not adequate special type of worms such as C-worm can't be detected by them. A network based systems are widely used and the wide spreading of worms is a proven fact, it is evident that new schemes are essential to combat such special kind of worms both in terms of both time domain and also frequency domain. As traffic is not confined to a particular system and it is related to networks, it is essential that the proposed scheme must be of type network-based worm detection system. In the Internet there must be a provision to detect worms such as Cyber center [8], network telescope [25] and SANSISC [15]. The detections systems can be spread across WWW in order to successfully detect the presence of worms successfully. Each monitor passively monitor sings either IP address or using port numbering through the network based detection systems. Such network based detection systems are capable of analyzing the scan traffic so worms and recognizing them. Many proposed systems in the literature [13], [14] are able to provide statistics and analyze patterns generated by worms. These schemes are based on the global scan traffic monitoring and detection of anomalous traffic [21], [2]. A state-space feedback control model is presented by [26] in order to detect and control spread of worms and viruses. This is done by measuring the velocity of the new connections made by an infected computer. However, the approaches that analyze scan traffic of worms are mainly used in developing detection systems.

## **III. Modeling Of The C-Worm**

The C-Worm modeling is based on our observations that have been made after some research. The C-worm block diagram is shown in fig. 1. The initial research revealed that the C-Worm is not same as other worms though it has similarities with normal worms. The normal worms perform scan traffic in order to replicate themselves and also propagate from one system to another system in a network environment. The same is followed by C-Worms also. However, there are two observations made clearly. The first observation is that, the C-Worm scan traffic involves IP addresses and port numbers and scan traffic is different from normal worms. The second observation is that the detection systems can't find the difference between scan traffic of C-worms and normal worms in terms of frequency domain. In time domain they appear to be same. The second observation also reveals that it is essential to differentiate the C-Worm traffic from other worm's traffic only in frequency domain. Based on these observations, our experiments are made. Our experiments focused on the traffic analysis and frequency domain and the results revealed that our scheme is capable of detecting C-Worms. When our scheme launches, it analyzes the dynamics of C-Worm traffic in Internet. It follows a theory known as control system theory [27]. In order to demonstrate effectiveness of the proposed scheme, the overall traffic

flow of C-Worm should be slow so as to show the detection process effectively. Control parameters are introduced to this effect such as attack probability on each infected computer. This indicates the probability in which C-Worm participates in the propagation of the worm. The control parameter in our model is generic in nature and its value is 1 indicating traditional worms and other value for C-Worms. In the process of modeling camouflaging worm, the following characteristics are followed.

- The traffic of C-worm is similar to non-worm traffic in terms of time domain. This means that over a period of time the scan traffic of the normal worm and C-worm is same.
- C-Worm does not show any trends while its propagation so as to hide its presence effectively.
- The average traffic of Worm is sufficient to model the C-Worm propagation model faster in order to cause rapid damage on the Internet.

We assume that the worm attacker manipulates scan traffic and the scan traffic of C-Worm follows different random distribution means.

### 3.1 Propagation Model of C-Worm

Epidemic dynamic model is used to work with the propagation model of C-Worm [2], [9]. This model assumes any given computer should be in one of the following states. The states are vulnerable, immune and infected. Immune state computer can't be infected. The vulnerable computer is the one that can be infected by C-Worm. The infected state does mean that the system is already infected. The epidemic model for the traditional PRS is represented as:

$$\frac{dM(t)}{dt} = \beta \cdot M(t) \cdot [N - M(t)],$$

The epidemic model for the C-Worm propagation is represented as:

$$\frac{dM(t)}{dt} = \beta \cdot M(t) \cdot P(t) \cdot [N - M(t)].$$

## IV. Performance Evaluation

Performance of the proposed scheme is evaluated using some evaluation metrics known as IR, DT, and MIR. The detection time is the time taken to detect C-Worm. MIR provides ratio of number of infected computers and total number of vulnerable computers. The higher the values of these metrics, the more effective the attacks are. The lower these values are, the lower the effectiveness of attacks.

### 4.1 Simulation Setup

The experiments are made both for normal and C-Worm traffic. The total number of vulnerable computers is assumed to be around 30000. By varying parameters C-Worm attacks are simulated. The detection involved port scan traffic and also non worm traffic. Logs and traffic traces are used to observe the behavior of worms. The detection results of C-Worm are provided in Table1.

Schemes	VAR	TREND	MEAN	SPEC(W)	SPEC
<b>Detection Rate(DR)</b>	48%	0%	14%	96.4%	99.3%
<b>Maximal Infection Ratio(MIR)</b>	14.4%	100%	7.5%	4.4%	2.8%
<b>Detection Time(DT) in Minutes</b>	2367	$\infty$	1838	1707	1460

**Table 1: Detection results for C-Worm**

Table 1 shows the results of detection with various parameters and also with various evaluation schemes such as DR (Detection Rate), MIR (Maximal Infection Ratio) besides providing the detection time in minutes.

### 4.2 Detection Performance for Traditional PRS Worms

The detection performance of traditional PRS worms is presented in fig. 3 and 4. The results use evaluation metrics such as MIR and DR respectively.

#### Maximal Infection Ratio of PRS Worm

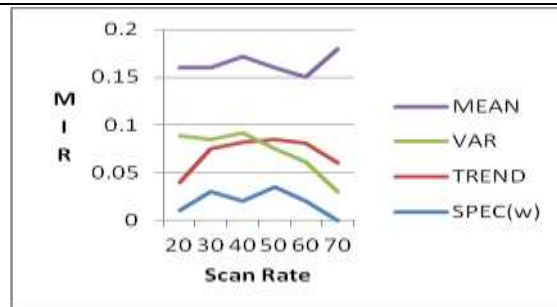


Fig. 2: Maximal Infection Ratio of PRS Worm

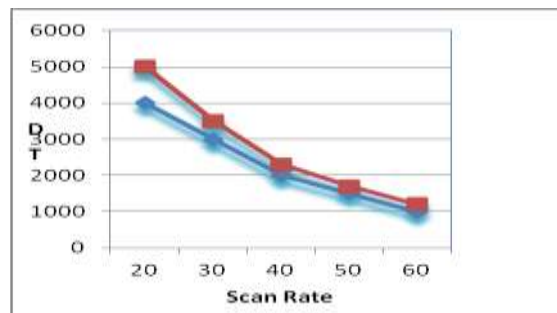


Fig. 3: Detection Time of PRS Worm

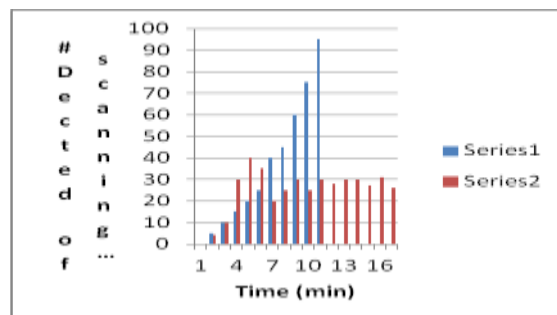


Fig. 4: Number of Detected Scanning Hosts on Camouflaging Worm

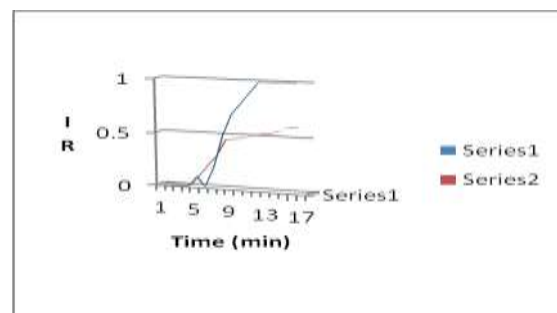


Fig. 5: Infected Ratio for the C-Worm and PRS Worm

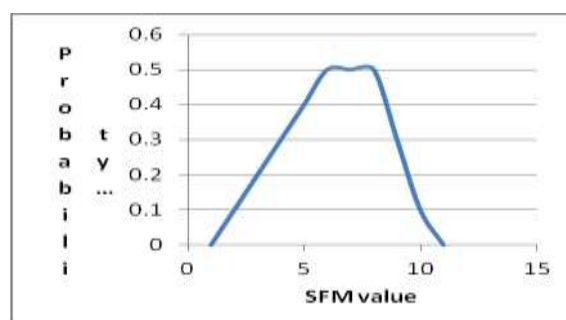


Fig. 6: PDF of SFM on normal non-worm traffic



## V. Conclusion

A new type of malware or worm is studied in this paper. The worm known as Camouflaging Worm, as the name implies, can hide its propagation and scan information from the worm detection systems and cause damage to IT systems. As the conventional detection systems can't identify the presence of such worm, we developed a scheme that can identify the C-Worm in terms of frequency domain. The modeling and detection of this worm is based on the observations we made. The observation include, the C-Worm also propagates by scanning IP addresses of the systems in the network. The second observation is that in the frequency of the scanning the C-Worm is different from other worms. These two observations are used as hypotheses in this paper and the research is made based on these hypotheses. The practical work and the results reveal that the hypotheses are fully supported or proved to be correct.

## References

- [1] D. Moore, C. Shannon, and J. Brown, "Code-Red: A Case Study on the Spread and Victims of an Internet Worm," Proc. Second Internet Measurement Workshop (IMW), Nov. 2002.
- [2] D. Moore, V. Paxson, and S. Savage, "Inside the Slammer Worm," Proc. IEEE Magazine of Security and Privacy, July 2003.
- [3] CERT, CERT/CC Advisories, <http://www.cert.org/advisories/>, 2010.12 IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, VOL. 8, NO. 3, MAY/JUNE 2011
- [4] P.R. Roberts, Zotob Arrest Breaks Credit Card Fraud Ring, <http://www.eweek.com/article2/0,1895,1854162,00.asp>, 2010.
- [5] R. Naraine, Botnet Hunters Search for Command and Control Servers, <http://www.eweek.com/article2/0,1759,1829347,00.asp>, 2010.
- [6] R. Vogt, J. Aycock, and M. Jacobson, "Quorum Sensing and Self-Stopping Worms," Proc. Fifth ACM Workshop Recurring Malcode (WORM), Oct. 2007.
- [7] S. Staniford, V. Paxson, and N. Weaver, "How to Own the Internet in Your Spare Time," Proc. 11th SENIX Security Symp. (SECURITY), Aug. 2002.
- [8] Z.S. Chen, L.X. Gao, and K. Kwiat, "Modeling the Spread of Active Worms," Proc. IEEE INFOCOM, Mar. 2003.
- [9] Zdnet, Smart Worm Lies Low to Evade Detection, <http://news.zdnet.co.uk/internet/security/0,39020375,39160285,00.htm>, 2010.
- [10] C. Wright, S. Coull, and F. Monrose, "Traffic Morphing: An Efficient Defense Against Statistical Traffic Analysis," Proc. 15<sup>th</sup> IEEE Network and Distributed System Security Symp. (NDSS), Feb. 2008.
- [11] C. Zou, W.B. Gong, D. Towsley, and L.X. Gao, "Monitoring and Early Detection for Internet Worms," Proc. 10th ACM Conf. Computer and Comm. Security (CCS), Oct. 2003.
- [12] S. Venkataraman, D. Song, P. Gibbons, and A. Blum, "New Streaming Algorithms for SuperSpreader Detection," Proc. 12<sup>th</sup> IEEE Network and Distributed Systems Security Symp. (NDSS), Feb. 2005.
- [13] J. Wu, S. Vangala, and L.X. Gao, "An Effective Architecture and Algorithm for Detecting Worms with Various Scan Techniques," Proc. 11th IEEE Network and Distributed System Security Symp. (NDSS), Feb. 2004.
- [14] SANS, Internet Storm Center, <http://isc.sans.org/>, 2010.
- [15] C.C. Zou, W. Gong, and D. Towsley, "Worm Propagation Modeling and Analysis under Dynamic Quarantine Defense," Proc. First ACM CCS Workshop Rapid Malcode (WORM), Oct. 2003.
- [16] C. Zou, D. Towsley, and W. Gong, "Email Worm Modeling and Defense," Proc. 13th Int'l Conf. Computer Comm. and Networks (ICCCN), Oct. 2004.
- [17] Y. Li, Z. Chen, and C. Chen, "Understanding Divide-Conquer-Scanning Worms," Proc. Int'l Performance Computing and Comm. Conf. (IPCCC), Dec. 2008.
- [18] D. Ha and H. Ngo, "On the Trade-Off between Speed and Resiliency of Flash Worms and Similar Malcodes," Proc. Fifth ACM Workshop Recurring Malcode (WORM), Oct. 2007.
- [19] L. Martignoni, D. Bruschi, and M. Monga, Detecting Self-Mutating Malware Using Control Flow Graph Matching," Proc. Conf. Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA), July 2006.
- [20] R. Perdisci, O. Kolesnikov, P. Fogla, M. Sharif, and W. Lee, "Polymorphic Blending Attacks," Proc. 15th USENIX Security Symp. (SECURITY), Aug. 2006.
- [21] Linux.com, Understanding StealthScans: orewarnedis Forearmed, <http://security.itworld.com/4363/LWD010321vcontrol3/page1.html>, 2010.
- [22] J.Z. Kolter and M.A. Maloof, "Learning to Detect Malicious Executables in the Wild," Proc. 10th ACM SIGKDD, Aug. 2004.
- [23] X. Wang, W. Yu, A. Champion, X. Fu, and D. Xuan, "Detecting Worms via Mining Dynamic Program Execution," Proc. IEEE Int'l Conf. Security and Privacy in Comm. Networks (SECURECOMM), Sept. 2007.
- [24] M. Bailey, E. Cooke, F. Jahanian, J. Nazario, and D. Watson, "The Internet Motion Sensor: A Distributed Blackhole Monitoring System," Proc. 12th IEEE Network and Distributed Systems Security Symp. (NDSS), Feb. 2005.
- [25] R. Dantu, J.W. Cangussu, and S. Patwardhan, "Fast Worm Containment Using Feedback Control," IEEE Trans. Dependable and Secure Computing, vol. 4, no. 2, pp. 119-136, Apr.-June 2007.
- [26] K. Ogata, Modern Control Engineering. Pearson Prentice Hall, 2002.

### About The Authors

	Sushma Mergu received the B.Tech Degree in Computer Science and Engineering from Kamala Institute of Technology and Science, Karimnagar, A.P. India. Currently doing M.tech in Computer Science and Engineering at SR Engineering College, Warangal, India. Her research interests include Networking and Security.
	G.Dileep Kumar received the B.Tech degree in Computer Science & Engineering from JSN College of Engineering & Technology, Kaghaz nagar, India and M.Tech degree in Software Engineering from Ramappa Engineering College, Warangal, India. Currently he is an Assistant Professor in the department Computer Science & Engineering, SR Engineering College, Warangal, India. His research interests include Data Mining, Network Security and Mobile Adhoc Networks.