International Journal of Innovative Research in Computer Science and Engineering (IJIRCSE)
www.ioirp.com ISSN: 2394-6364, Volume – 1, Issue – 7. June 2015

An Optimized Workload Aware Scheduler in Heterogeneous Cloud Environment

[1]P.Suganthi
Department of Computer Science Engineering,
UCE, BIT Campus, Tiruchirappalli, India
suganthi.palani@gmail.com

[2]K.Ambika
Department of Computer Science Engineering,
UCE, BIT Campus, Tiruchirappalli, India
ambikame.81@gmail.com

***Abstract*— In Cloud environs, I/O Workloads are processed in different VMs of a physical machine. Earlier scheduling techniques are assign resources to the incoming tasks without the knowledge of the tasks. Hence the high-speed network resources are not utilized efficiently. So an analysis is required to schedule the jobs in cloud environment. Based on task analysis, the available resources are scheduled by the techniques like shared credit scheduling and dynamic scheduling slice adjustment. In this paper a performance analysis of network I/O workloads hosted in different VMs of a single physical machine. The workloads are either CPU bound or network I/O bound. The I/O bound workloads needs more processing power hence they executed on high configured virtual machines. The main objective of this paper is to improve I/O performance and CPU performance based on credit sharing process and MapReduce algorithm, hence overcome poor performance in resource allocation system. First, study the impact of running idle guest domains on the overall system performance, addressing the cost, and benefit of managing idle VM instances in virtualized data-centers. The dynamic VM reconfiguration technique for data-intensive computing on clouds, called Dynamic Resource Reconfiguration. Resource reconfiguration can adjust the computing capability of individual VMs to maximize the utilization of resources. Among several factors causing resource imbalance in the VM platforms, this paper focuses on data locality. Although assigning tasks on the nodes containing their input data can improve the overall performance of a job significantly, the fixed computing capability of each node may not allow such locality-aware scheduling. This concept in real time virtualized cloud environment to improve the performance of the system.**

Keywords*— Cloud Scheduling; Virtualization; Workload aware scheduler; I/O intensive domain; CPU intensive domain; credit sharing; MapReduce

I. INTRODUCTION

A Cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers [15]. Cloud Computing is an *approach to computing* that leverages the efficient pooling of on-demand, self-managed virtual infrastructure, consumed as a service. The different types of Services provided by Cloud computing is as follows: (1) Infrastructure as a Service (IaaS) such as CPU, Storage: Amazon.com, Nirvanic, GoGrid (2) Platform as a Service (PaaS) such as Google App Engine, Microsoft Azure, Manjrasoft Aneka (3) Software as a Service (SaaS) such as SalesForce.Com

In general, the cloud is a large group of interconnected computers. These computers can be personal computers or network servers; they can be public or private. This cloud of computers extends beyond a single company or enterprise. The applications and data served by the cloud are available to broad group of users, cross-enterprise and cross-platform. Access is via the Internet. Any authorized user can access these docs and apps from any computer over any Internet connection. And, to the user, the technology and infrastructure behind the cloud is invisible. The developers cloud services only know the technologies.

Virtualization technology separates hardware and software management and offers many useful features including functional isolation, server consolidation and live migration. In addition to these capabilities, it also provides a strategy to more fully utilize the vast compute cycles available on commodity multi-core platforms. For these reasons, virtualization technology is rapidly gaining popularity in cloud computing and have been key enabling technologies in cloud infrastructures like Amazon EC2. This trend toward server consolidation where multiple virtual machines (VM) share a machine's resources naturally includes sharing of network interface cards (NIC). In such an environment, the virtual machine monitor (VMM) virtualizes the machine's NIC to give the illusion of ownership to multiple operating systems running concurrently in different VMs. Thus, server consolidation promises not only full utilization of compute resources, but also of available network bandwidth.

This paper conducts detailed performance analysis of virtualized network processing over virtualization based on Xen to fully understand its performance bottlenecks. It also quantifies performance due to architectural events and overheads in the processor by using hardware performance counters to instrument the functions along the packet processing path. The virtualized network processing only achieves 3.5Gbps of network bandwidth. The data movement in virtualized environment is more complicated than native environment and needs extra consideration. Motivated by this performance analysis, developed two VMM scheduler

optimizations to improve data movement performance by co-scheduling backend driver and frontend driver into the same cache domain and stealing credits from idling VCPUs to favor I/O VCPUs. In order to avoid cache misses on packets propose integrating a small hardware data movement engine. The engine considers VMM scheduling information and can inject packets into cores where corresponding guest domains are running.

The major contributions in this paper can be summarized as follows: a) a detailed performance analysis of virtualized network processing performance and revealed its major performance bottlenecks. b) developed two VMM scheduler optimizations and proposed a small hardware data movement engine to address the data movement challenge in virtualized environment. c) implemented a simplified bridge to reduce packet switching overheads.

In this paper section 1 briefly discusses about the scheduling mechanism of Cloud Computing. Section 2 briefly reviews the existing scheduling strategies in Cloud computing. Section 3 provides the detailed process of workload aware scheduler and Section 4 explains the implemented hybrid MapReduce algorithm. Section 5 gives the performance evaluation and results obtained. Finally, section 6 gives conclusion.

II. LITERATURE SURVEY

A. Gordon [2] proposed I/O activity is a dominant factor in the performance of virtualized environments, motivating direct device assignment where the host assigns physical I/O devices directly to guest virtual machines. Examples of such devices include disk controllers, network cards, and GPUs. Direct device assignment provides superior performance relative to alternative I/O virtualization approaches, because it almost entirely removes the host from the guest's I/O path. Without direct device assignment, I/O-intensive workloads might suffer unacceptable performance degradation.

M. Kesavan [3] implemented to recognize when designing I/O proportional fairness for virtualized systems that the entities sharing resources and services are entire operating systems with their own or even proprietary internal resource management. Interestingly, such management is often adaptive with respect to perceived service latencies, with well-known examples including TCP congestion avoidance for communication devices and disk access scheduling for storage.

John R. Lange [1] analyzes the performance of VNET/P, providing insight into the fundamental challenges of overlay support for high-speed network devices. And follows with a description of new optimizations for virtual overlay networks on high-speed interconnects and briefly describes an implementation of the proposed optimizations and micro benchmark results, and also follows with an extensive evaluation of the impact of these optimizations using more complex benchmarks.

Myeongjae Jeon [4] targets consolidated underutilized servers. In order to provide sufficient memory for each virtual machine, the demand for a large amount of memory has increased. This leads to high energy consumption in the memory system. In server systems with a large amount of memory, the energy consumption in memory may exceed that in processors. Thus, one of key steps in building energy-efficient data centers is to reduce memory energy consumption.

Since the birth of VM, research in improving virtualized I/O performance never faded away. For system optimizations, sorted the domains with the same states in the run queue based on their remaining credits rather than arbitrarily insert the new domain at the end of each state section. However, they focused on the fairness of I/O performance with 1GE network and did not consider the VMM scheduler on mainstream multi-core systems where behaves significantly different from single core systems. With the same optimizations on experiment environment under various databases, find that the blocking of scheduler tickle adversely glooms the I/O performance by a factor of 100 and the run queue sort does not make any difference for I/O performance.

In addition to VMM scheduler optimizations, lots of engineering optimizations have also been implemented to improve network I/O performance in virtualization environment and analyzed virtualization performance overhead and then implemented numerous optimizations such as reusing grant table, using large page size, moving data copy to guest etc to bridge the gap between software and hardware techniques for I/O virtualization and designed cache-aware scheduling for virtualization to improve web server performance. Then adopted virtualization technology for HPC and allowed each domain to directly access the high performance network. However, they targeted to the high performance network InfiniBand rather than Ethernet Network.

In Ethernet Network, some researches including Crossbow [6] tried to address the performance issues by taking advantage of the new Ethernet NIC features like multiple TX/RX queues to allow domains to directly access the hardware. They heavily rely on hardware and hence sacrifice the features of portability and live migration, two major incentives for deploying virtualization in high end servers. Numerous studies have been done in server architectures to efficiently tackle the network I/O virtualization challenge. In industry, Intel offloads virtual switch (or packet de-multiplexing) from the driver domain to hardware NIC and deploys multiple queues to allow guest OS to directly access hardware queues.

Recently, PCI-E standard group proposes single root IO virtualization (SR-IOV) to self-virtualized a physical device into multiple lightweight PCI-E devices, significantly avoiding I/O virtualization overheads. However, they require extensive hardware platform support ranging from PCI-E, chipsets and PCI-E devices and also sacrifice virtual machine migration.

III. WORKLOAD AWARE SCHEDULER

Virtualization is an abstraction of physical computer resources. A typical virtualized platform consists of a software virtual machine monitor that "virtualizes/abstracts" the physical resource of the platform and provides a simulated environment that appears to the operating system as hardware. VMM functions as an abstraction layer of the real physical devices. As a result, scheduling in virtualization is based on Virtual CPUs (VCPU) because Physical CPUs (PCPU) is transparent to domains. Each domain can be arbitrarily allocated with multiple VCPUs. Besides the default credit scheduler, VMM also keeps its legacy scheduler Simple Earliest Deadline First (SEDF). SEDF provides weighted CPU sharing in an intuitive way and uses real-time algorithms to ensure real time guarantees. However, it lacks global load-balancing on multiprocessors and is becoming obsolete. This paper focus on the default credit scheduler, a proportional fair share CPU scheduler built to achieve load balance on SMP hosts. Its overall objective is to allocate the processor resources fairly.

The scheduler organizes a local run queue of online runnable VCPUs for each PCPU and always picks a workload (VCPU) from the head of the queue to run. This queue is sorted by VCPU priority. A VCPU's priority can be one of three values: OVER, UNDER and BOOST. OVER, UNDER represents whether or not this VCPU has used up its fair share of CPU resource in the ongoing accounting period. The BOOST state provides a mechanism for domains to achieve low I/O response latency. All the VCPUs in BOOST state are placed in front of those in UNDER state in the run queue, while those with OVER state are kept in the tail portion. Based on the predefined weight, each domain is initially allocated a corresponding credit which is fairly shared among all the VCPUs that are affinitized to the domain. As a VCPU runs, it consumes credits. Every so often, a system-wide accounting thread re-computes how many credits eac active domain has earned and bumps the credits.

When it comes to multi-core architecture, there are a few twists while the scheduler functions. First of all, when there is not a VCPU of priority UNDER on a PCPU's local run queue, the scheduler will search other PCPUs for one. This load balancing ensures each domain receives its fair share of PCPU resources system-wide. Before a PCPU goes idle, the scheduler will look on other PCPUs to find any runnable VCPU. This guarantees that no PCPU idles when there is runnable work in the system.

Secondly, VCPU migration might happen based on priority difference for event notification. Whenever an event is notified to a target VCPU while it is idle, the scheduler tickles the designated PCPU and re-evaluates to see if the target VCPU preempts the current running VCPU. If there are at least two runnable VCPUs in that PCPU, the scheduler would migrate some of them to the idlers in the system to achieve load balance. Last but not the least, the scheduler checks the state of the current running VCPU during each timer interrupt and redistributes the PCPU if necessary. The running VCPU will be migrated to the online neighbor PCPU with the most idling neighbors PCPU. This policy distributes work across distinct sockets first and then distinct cores in the same socket.

IV. HYBRID MAPREDUCE SCHEDULING PROCESS

A. Virtualization Environment

A guest OS is an operating system that is installed in a virtual machine or disk partition in addition to the host or main OS. In virtualization, a single computer can run more than one OS at the same time. In disk partitioning, a guest OS must be the same as the host OS. In a virtualization solution, a guest OS can be different from the host OS. This phase can create virtualization environment with various guest operating systems. In Virtual machine all resources are allocated as per user request.

B. Workload-aware Credit Scheduler

The credit scheduler regards a time quantum as credit, which is determined by the defined weight for each domain. The credit of a running VCPU is debited by 100 every tick period (10ms); all active VCPUs are given the credit calculated by the weight of their domain every credit period (30ms). The credit of a VCPU is used to determine its priority once a credit period. If a VCPU has remaining credit (credit > 0), its priority is UNDER (−1). Otherwise a VCPU is given OVER (−2) priority, which means the VCPU has consumed more than its allocated credit. All VCPUs with the priority of UNDER are scheduled before those with the priority of OVER; a run queue maintains VCPUs with UNDER followed by those with OVER, and the scheduler picks a VCPU at the head of the run queue as a next VCPU. Once a VCPU is scheduled, it receives the time slice of 30ms and runs consuming its credit. When the time slice of a running VCPU expires, the VCPU is descheduled and is put into the tail of a list that contains VCPUs with the same priority as the descheduled VCPU. The principle to steal credits is formalized in the following equation:

$$\text{Steal} = \text{Credit}(\text{Idle}_{\text{VCPUs}})/(2 * \text{Num}(\text{IO}_{\text{VCPUs}}))$$

where Steal means the stolen credit for each I/O VCPU, Credit(Idle _VCPUs) is for the credit of all idling VCPUS. Num(IO_VCPUs) represents the number of I/O VCPUs.

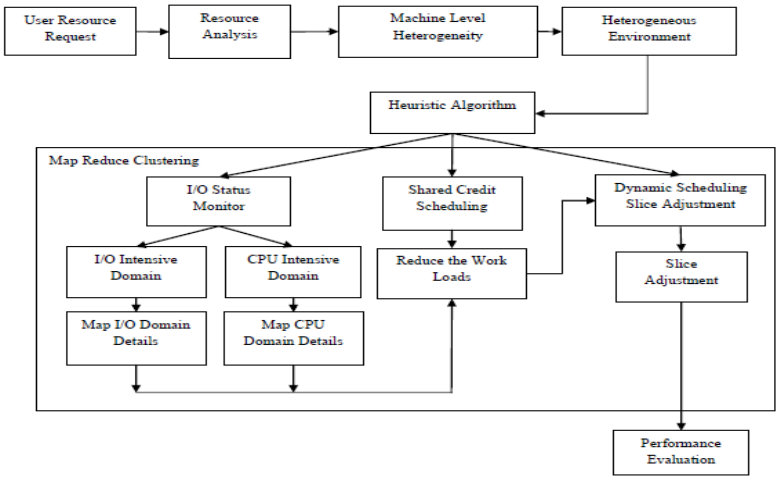

Fig. 1. Hybrid Scheduler Process

C. Optimization Strategies

The scheduler organizes a local run queue of online runnable VCPUs for each PCPU and always picks a workload (VCPU) from the head of the queue to run. The queue is sorted by VCPU priority. A VCPU's priority can be one of three values: OVER, UNDER and BOOST. OVER, UNDER represents whether or not VCPU has used up its fair share of CPU resource in the ongoing accounting period. The BOOST state provides a mechanism for domains to achieve low I/O response latency. All the VCPUs in BOOST state are placed in front of those in UNDER state in the run queue, while those with OVER state are kept in the tail portion. Based on the predefined weight, each domain is initially allocated a corresponding credit which is fairly shared among all the VCPUs that are affinities to the domain. As a VCPU runs, it consumes credits. Every so often, a system-wide accounting thread re-computes how many credits each active domain has earned and bumps the credits.

D. MapReduce Process

Map Reduce works in two phases; the Map phase and the Reduce phase. In the Map phase a dedicated node called the master node takes the input, divides it into smaller shared data splits and assigns them to worker nodes. The worker nodes may perform the same splitting operation, leading to a hierarchal tree structure. The worker node processes the assigned splits and sends the results back to the master node. The Reduce phase then begins with sorting and shuffling the partitions that are produced by the Map phase. The master node confirms the reduce workers to get their intermediate pairs through remote procedure calls. After acquiring the data for each reducer, the data is grouped and sorted by the resources. Once all reducers get sorted, the reducer calls reduce function for pairs of data. The output of reduce function writes to the corresponding output files leading to get all the needed processing achieved. This phase implements configuration manager and resource predictor to assign the jobs in queues. Then update the queues to improve the efficiency.

E. Performance Evaluation

This phase evaluates the performance of the system using throughput, failure rate and response time metrics. In general terms, throughput is the rate of production or the rate at which something can be processed. When used in the context of communication networks, such as Ethernet or packet radio, throughput or network throughput is the rate of successful message delivery over a communication domain server. Failure rate refers to the number of jobs allocated to the VM successfully completed. Run time is referred to allocate the jobs in VM as per user requirements. The proposed system provides improved throughput and response time for allocate workloads.

International Journal of Innovative Research in Computer Science and Engineering (IJIRCSE)
ISSN: 2394-6364, Volume – 1, Issue – 7. June 2015

V. IMPLEMENTATION RESULTS

The conducted experiments evaluate the proposed approach against existing approaches. This section studies performance benefits of the system optimizations by credit scheduler optimizations and MapReduce process in terms of throughput, failure rate and response time in core utilization per gigabit. In Figure, Existing represents the original system with credit scheduler optimization. The following graph represents the throughput, failure rate and response time stand for core utilization per gigabit. By this observation shows in Fig. 4 that workload-aware scheduler increases throughput by 18%, and also saves 15% in core utilization per gigabit. The credit stealing policy for favoring I/O VCPUs further improves the network performance by 12% and saves 7% in core utilization per gigabit. It is observed that both optimizations can increase the network bandwidth by 96% to 4.5 Gbps, and also save 36% in core utilization per gigabit. In this experiment, notice that the total core utilization consumed by Dom0 is reduced from 105% to 84% by using all the optimizations.

In this performance chart analyse the matrix that includes throughput status to improve the efficiency of the system. Hybrid scheduler provides improved performance than the credit scheduler.

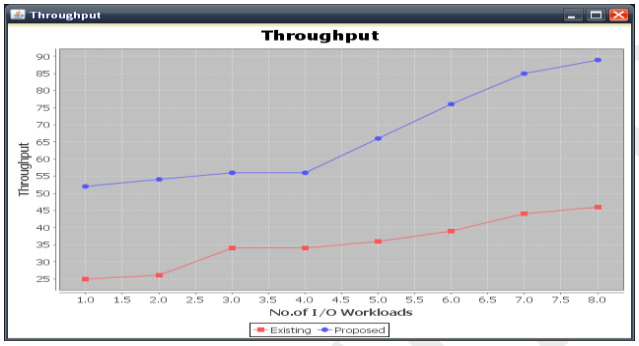

Fig. 2. Throughput

This performance chart shows the failure rate of the system. Hybrid scheduler provides reduced failure rate than the credit scheduler.

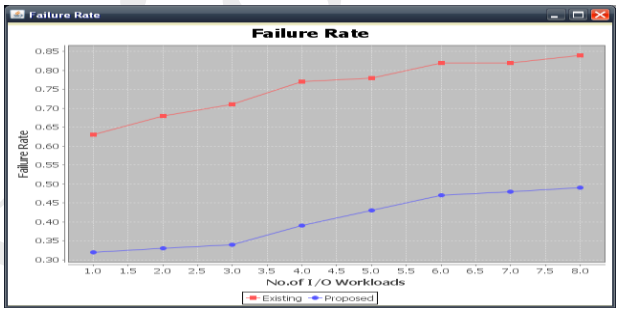

Fig. 3. Failure Rate

In this performance chart analyse the matrix that includes response time status to improve the efficiency of the system. Hybrid scheduler provides improved performance than the credit scheduler.

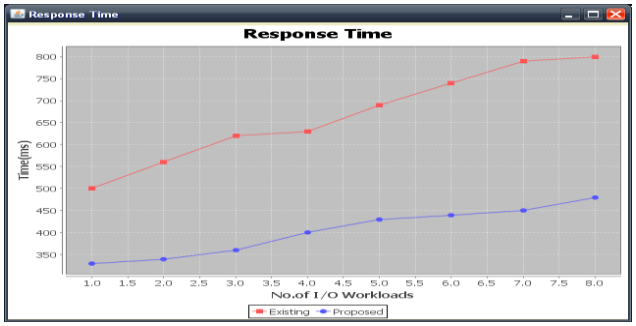

Fig. 4. Response Time

VI. CONCLUSIONS

Many of the applications on the cloud need to achieve the performance goals such as deadlines. This work proposed and evaluated a novel scheduler that extended the real time cloud scheduling approach to derive minimum map and reduce task slot count for performing task scheduling with deadline constraints in java environment. The current job ordering is inspired by the existing scheduling approaches. The proposed scheduler improves the input data locality of a virtual Map Reduce cluster, by temporarily increasing cores to VMs to run local tasks. With VM reconfiguration, each node can be adjusted to provide only the necessary amount of resource demanded for that node. Addressing resource requirements of different types of workloads (CPU/IO) which may cause load imbalance and underutilization of the cluster will be future work. And also extend this approach to implement this approach in hadoop environment.

References

[1] M. F. Mergen, V. Uhlig, O. Krieger, and J. Xenidis, "Virtualization for high-performance computing," SIGOPS Oper. Syst. Rev., vol. 40, no. 2, pp. 8–11, Apr. 2006.

[2] J.W. Jang, M. Jeon, H.-S. Kim, H. Jo, J.-S. Kim, and S. Maeng, "Energy reduction in consolidated servers through memory aware virtual machine scheduling," Computers, IEEE Transactions on, vol. 60, no. 4, pp. 552 –564, april 2011.

[3] L. Cherkasova and D. Gupta, "When virtual is harder than real: Resource allocation challenges in virtual machine based it environments," HPL-2007-25, Tech. Rep., 2007.

[4] T. C. Chieu, A. Mohindra, A. Karve, and A. Segal, "Dynamic scaling of web applications in a virtualized cloud computing environment," in ICEBE '09, 2009, pp. 281–286.

[5] F. Hao, T. V. Lakshma, S. Mukherjee, and H. Song, "Enhancing dynamic cloud-based services using network virtualization," in ICEBE '09, 2009, pp. 281–286.

[6] A. Menon, J. R. Santos, Y. Turner, G. J. Janakiraman, and W. Zwaenepoel, "Diagnosing performance overheads in the xen virtual machine environment," in Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments, ser. VEE '05. ACM, 2005, pp. 13–23.

[7] A. Menon, A. L. Cox, and W. Zwaenepoel, "Optimizing network virtualization in xen," in Proceedings of the annual conference on USENIX '06 Annual Technical Conference, ser. ATEC '06. USENIX Association, 2006, pp. 2–2.

[8] L. Cherkasova and R. Gardner, "Measuring cpu overhead for i/o processing in the xen virtual machine monitor," in Proceedings of the annual conference on USENIX Annual Technical Conference, ser. ATEC '05. USENIX Association, 2005, pp. 24–24.

[9] K. K. Ram, J. R. Santos, Y. Turner, A. L. Cox, and S. Rixner, "Achieving 10 gb/s using safe and transparent network interface virtualization," in Proceedings of the 2009 ACM SIGPLAN/ SIGOPS international conference on Virtual execution environments, ser. VEE '09. ACM, 2009, pp. 61–70.

[10] J. R. Santos, Y. Turner, G. Janakiraman, and I. Pratt, "Bridging the gap between software and hardware techniques for i/o virtualization," in USENIX 2008 Annual Technical Conference on Annual Technical Conference, ser. ATC'08. USENIX Association, 2008, pp. 29–42.

[11] C. K. Huang, H. H. Nien, S. K. Changchien, and H. W. Shieh, "Image encryption with chaotic random codes by grey relational grade and Taguchi method," Optics Communications, vol. 280, no. 2, pp. 300–310, 2007.

[12] S. Mazloom and A. M. Eftekhari-Moghadam, "Color image encryption based on CoupledNonlinear ChaoticMap," Chaos, Solitons and Fractals, vol. 42, no. 3, pp. 1745–1754, 2009.

[13] Y. Tang, Z. Wang, and J. A. Fang, "Image encryption using chaotic coupled map lattices with time-varying delays," Communications in Nonlinear Science and Numerical Simulation, vol. 15, no. 9, pp. 2456–2468, 2010.

[14] C. E. Shannon, "Communication theory of secrecy systems," Bell System Technical Journal, vol. 28, no. 4, pp. 656–715, 1949. [5] O. Edward, Chaos in Dynamical Systems, Cambridge University Press, Cambridge, UK, 2nd edition, 2003.

[15] Neha Mehrotra, "Cloud-Testing vs. Testing a cloud", 10th Annual International Software Testing Conference 2010.

[16] Wayne Jansen, Timothy Grance , "Guidelines on Security and Privacy in Public Cloud Computing", National Institute of Standards and Technology, Special publication 800 – 144

[17] "Security of Cloud Computing Providers Study" Sponsored by CA Technologies Independently conducted by Ponemon Institute LLC Publication Date: April 2011.

[18] Ramana Reddy, Munaga V.N.Prasad, D.Sreenivasa Rao: "Robust Digital Watermarking of Color Images under Noise Attacks" in International Journal of Recent Trends in Engineering, Vol.1, No. 1, May 2009.

[19] J. F. Hauer et al., "Use of the WECC WAMS in wide-area probing tests for validation of system performance and modeling," IEEE Trans. Power Syst., vol. 24, pp. 250–257, Feb. 2009.

Authors Profile:

Ms. P.Suganthi born in Ramanathapuram District, Tamilnadu, India. She received her Bachelor's degree B.Sc., Computer Science from Madurai Kamaraj University, MCA degree from Anna University, Chennai and M.Phil degree in Computer Science at Bharathiar University, Coimbatore. She is pursuing M.E., degree in Computer Science Engineering at UCE, BIT Campus, Tiruchirappalli. She is in the teaching field for more than six years.