ABSTRACT

In the new era of information technology ,Cloud computing brings a great challenge in storing and retrieving more number of user files in remote locations . Future file system needs exabytes of data to store trillions of user files. This scalability in storage becomes a bottleneck in storing, organizing and retrieving files in cloud system. However, providing effective search, store and organize the trillions of files using the existing algorithms is not a simple task.

In cloud storage system, more number of user files are stored on different compute nodes. To organize and maintain files in the cloud storage system, digit binate algorithm and digit compact prefix algorithms are used in the backend of the compute node and controller node. The algorithm time complexity is analysed and discussed. The performance evaluation done for both digit binate and digit compact prefix algorithms.

The implementation is done in openstack private cloud. The digit binate algorithm shows the minimum time in storing and retrieving compared to the digit binate algorithm with different compute node instances and files. The implementation is done in openstack private cloud.

Internet is important factor in cloud to provide services to users. Router is needed in Internet to determine the packet's next-hop router. This is achieved by checking the destination address of the incoming IP packet in a routing table. Because of higher number of nodes in the cloud scenario, there is a frequent change in the network topology. The new node information is to be updated in all routing tables available in the network. The frequent updation in routing table affects the process of finding the most matching entry and this made a bottleneck in the routers in the Internet. The thesis describes fast and efficient hybrid lookup algorithm that overcome this bottleneck by splitting the octets into four. The implementation is done in SDN Ryu controller and mininet virtual topology creator. The proposed algorithm shows minimum look up time and search time compared with the binary search tree.