IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 1, Ver. I (Jan – Feb. 2015), PP 38-42
www.iosrjournals.org

An in-building multi-server cloud system based on shortest Path algorithm depending on the distance and measured Signal strength

[1]Abhirup Bhawal, [2]Anindita Nath, [3]Suchismita Mitra, [4]Puspendu Roy, [5]Pradip Kumar Das

Abstract: *In this paper we are proposing a system that will work as a virtual cloud computing environment. Our system will be suitable for an office for example software development firm. Cloud Computing is an emerging technology now a days. In cloud computing, the resources are shared within a network using web server. We have designed a cloud computing environment having multiple upload stations and numerous client nodes. Most commonly, the cloud computing environment has one server and multiple nodes connected with it. It server multiple purposes at a time. We have designed a cloud computing environment with decentralized multiple upload stations. In our system, we have assumed that the client nodes can be mobile and it can change its location according to time. We have also assumed that, the users will not have a fixed working desk and they can do their work from any desk they want. They need to log in to the system using their pre-defined user-id and password and he will get access to the system with all features. Now, once a user will try to upload some data in the upload stations, our system will intelligently find out the nearest server from the location of the client. The data will be uploaded in that upload station. As all the upload stations are inter connected, if he or she wants to retrieve that information later, he or she will be able to do that easily. This concept can also be implemented in multi-floor office area. We are using Bluetooth as the medium of transmission as it is low power consuming and less costly than other available standard options.*
Keywords: *Cloud Computing, Upload Stations, Bluetooth.*

I. Introduction

The technologies we are using to design the system are cloud computing, Bluetooth, Ad-Hoc Network. A brief description of those technologies follow.

Cloud Computing

Cloud Computing is computing in which large groups of remote servers are networked to allow centralized data storage and online access to computer services and resources. Clouds can be classified as public, private or hybrid. Cloud computing relies on sharing resources to achieve coherence and economies of scale, similar to a utility over a network. At the foundation of cloud computing is the broader concept of converged infrastructure and shared services. The primitive goal of cloud computing environment is to maximize the use of computing power thus reducing environmental damages as well since less power, air conditioning, Rackspace etc. The present availability of high-capacity networks, low cost computers and storage devices as well as the widespread adaptation of hardware virtualization, service-oriented architecture and autonomic and utility computing have led to growth in cloud computing.

Bluetooth

Bluetooth is a wireless technology standard for exchanging data over short distances and operates in the 2.4 GHz frequency band without a license for wireless communication. This communication protocol has primarily been designed for low-power consumption and low cost. This technology can be used for real-time data transfer usually between 10-100 meters which is reasonable within a small building. The data-transfer rate of the Bluetooth Technology is quite acceptable i.e. 3-4 Mbps. Moreover, the Bluetooth devices require low power to work. Thus this is low power consuming and it is less costly. As a result Bluetooth is a low power consuming and cost efficient wireless medium of transmission.

Ad-Hoc Network

The wireless ad-hoc network is used in our project. It's basically a decentralized, infrastructure-less network because it doesn't rely on a pre-existing infrastructure like the routers or access points. The nodes in this kind of network is completely dynamic. In this system, each node that participates in data transmission, dynamically determines the node to whom the data needs to be forwarded.

II. Motivation

Cloud computing is an emerging technology of today's era. It not only makes any system more efficient and easy to use and access, but also it reduces the requirements of the client machines. Cloud computing environment serves two purposes. Firstly, the client nodes don't need to be highly configured to do the high computing jobs. If we configure the server highly, it will serve the purpose of processing all the requests from the client nodes. Secondly, the data storage becomes centralized. As a result the chances of losing the important data due to accidents or server failures become less. But major disadvantage of this kind of systems is network limitation. The number of channels in a server is limited this is why, the number of users who can use the system resources becomes limited and it increases the waiting time eventually. As a solution to this problem we thought of a cloud computing system where there will be multiple upload stations. Thus, the number of users who can use the system at a time increases. Implementing multiple upload stations we actually get another important benefit. In existing cloud computing systems the client nodes needs to be fixed, but in our case the users can also be movable which will make the work environment more comfortable.

III. Methodology

In our system we have considered that we have multiple upload stations working as servers. Those upload station has fixed location on the floor and those are interconnected. In the following figure a top-view of the system is shown. In that figure, the blue circles represent the mobile devices and the green circles represent the static devices.

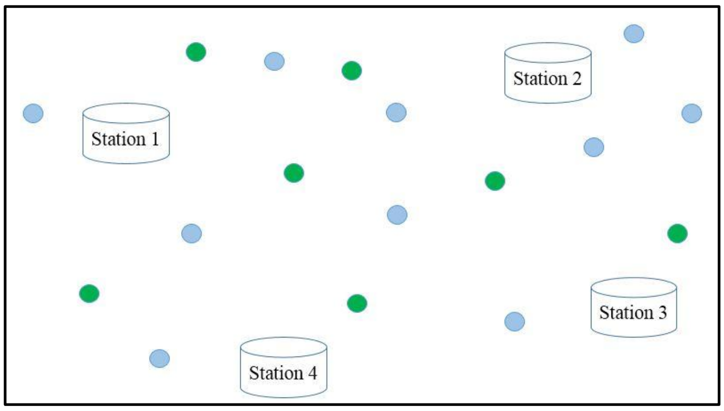


Fig. 1 – The top-view of our system

This project can be divided basically in two parts. Firstly, we have designed an application that will be running in each and every user's machine. This application will make the user able to upload their data to one of the upload stations. Secondly, during any upload that will take place, we need to determine the nearest server from the user's location. Our area of thrust in this project was to handle multiple upload stations and how those stations can run independently as well as syncing with each other.

A. User application running in every device

We have designed a platform for an office of software development firm. In that application, each user has to log in using their user-id and password. After successfully logging into the system, they can access various features like writing notes, saving them in the upload station, running and compiling source codes. For implementing those features, we used data structures that will be stored in binary files as we tried to avoid the burdens in the client nodes as well as the upload stations. For each feature, a particular response is being transmitted to the upload stations or the servers. The upload station is processing the request and sending back the proper response to the client node. When a user is being registered, a particular block of memory is allocated to that user, and all the documents or files he or she will be creating will be stored in that place. Each and every file in the upload stations are stored in an encrypted way to reduce of chance of data misplacement. We are DES encryption standard to achieve that. Moreover there are some files those are being used for general purpose, like storing the user-id and passwords of the users. In DES cryptography system, each file is encrypted with a randomly generated string of characters. In our system to provide the information those are stored in the upload stations an extra layer of protection, we are changing the randomly generated key string in a timely manner and encrypting again. Thus if anyone steals any data, it will be tough for him or her to decipher the actual content of the file. Apart from that whenever we are transmitting any data we are sending the data after encryption. After receiving the data in the upload station, the data is decrypted and processed after that. Thus, if someone taps the data on the fly, he will not be able to understand the actual data. There are two parts in this application, i.e. the server and the client. Fig. 2 depicts the flow diagram that is happening in the client node.

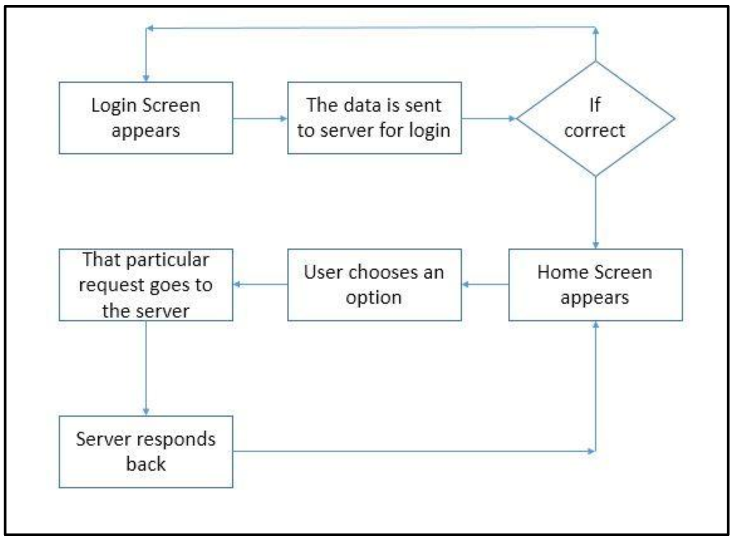


Fig. 2 the flow diagram of the client part of the application

Step algorithm for client:

Step 1 – The client has to login to the system for using the entire system and facilities thus the system prompts the user to login first.

Step 2 – The login request is generated and sent to the server.

Step 3 – Server responses to the login request.

Step 4 – If the server grants the login for the user, control is passed to step 5 else it is sent back to the step 1.

Step 5 – The user in the client node gets a home screen containing several options to start the work.

In the same manner the server works in our system. The flow diagram and the step algorithm are described below.

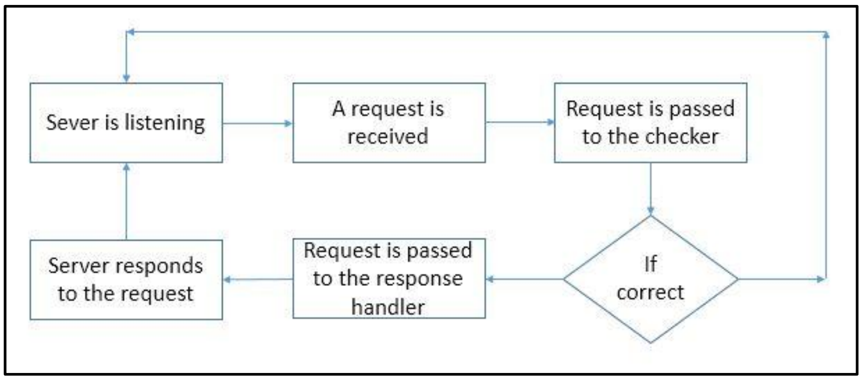

Sever is listening
A request is received
Request is passed to the checker
If correct
Request is passed to the response handler
Server responds to the request

Fig. 3 – The flow diagram for the server part

Step algorithm for server:

Step 1 – The server is always in listening mode.

Step 2 – Request from the client node is received.

Step 3 – the request is checked whether it is a valid request or not.

Step 4 – If the request is accepted and correct in the specific protocol then continue to step 4 else back to step 1 for receiving the next request.

Step 5 – The received request is passed to the request handler.

Step 6 – The request handler finishes the specified task, and then the processed result is sent back to the client node from where the request has been received.

Step 7 – The control is again sent back to Step 1.

B. Determine the nearest server based on the user's location

To determine the nearest upload station node, a response request is sent to all the upload stations. Then, if the upload station response back within a threshold time, the time is stored. According to the times required for responding back, we are approximately calculating the distance from the client node to the upload station. Then those distances are sorted in ascending order to find out the nearest upload station. Once we have selected the nearest upload station, our system is sending the data to that upload station. Fig. 4 depicts the flow diagram for selecting the nearest upload station.

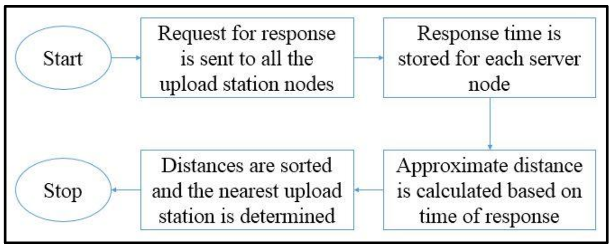
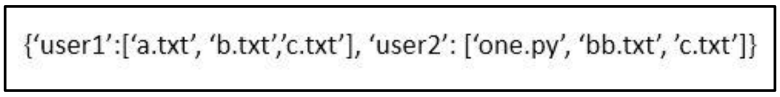

Start
Request for response is sent to all the upload station nodes
Response time is stored for each server node
Approximate distance is calculated based on time of response
Distances are sorted and the nearest upload station is determined
Stop

Fig. 4 – Flow diagram for selecting the nearest upload station

C. Data structure

The entire building block of the permanent data storage system is data structures like, dictionary, list, tuples etc. We have chosen to use simple text files to store the data permanently and to use basic data structures extensively as mentioned earlier. We have used dictionary to store several data which is responsible for the user to grant access to number of services. A dictionary is a special kind of data-structure of python, which support the following features. Keys must be immutable, and this key can be number, string, tuple or anything, but, it cannot be changed after creation, because of hashing. And moreover, the keys must be unique again because of hashing. There are no restrictions of values in a dictionary, and the keys will be listed in arbitrary order1. For instance, the file responsible for the users to login to the system looks like this: Here, the username field is a string, containing the username, and there is a tuple corresponding to the key i.e. the username, and the first field of the tuple contains the password, and the second field contains the string for the generation of the cryptographic key which we use to encrypt the data stored by the user. There are several other similar kinds of files containing other related data. For example, the next screenshot shows the text file containing the list of the files, a user saved in the system. In this case, we have used a dictionary, where the username has been used as the key, and there is a list corresponding to each key, containing the name of the files saved by the user. As I have mentioned in the motivation, we are trying to build a system, which will be low power consuming as well as cost efficient, we have purposely chosen to use data-structures stored in simple text files, instead of database. As a result the whole system will not be bulky and will be very suitable for the smaller systems as well as for the larger systems. Fig. 5 depicts one of the data structure used for our system.



```
{'user1':['a.txt', 'b.txt','c.txt'], 'user2': ['one.py', 'bb.txt', 'c.txt']}
```

Fig. 5 – Data structure showing the files saved the users

IV. Results

The results and the finding of our project are portrayed in this section. The fig. 6 depicts the user interface that a user gets whenever he or she is logging into the system.

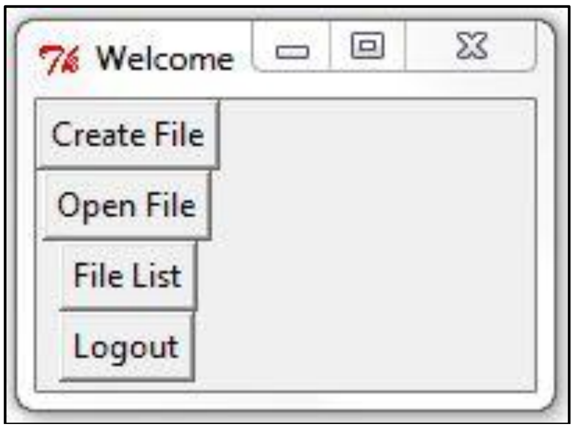
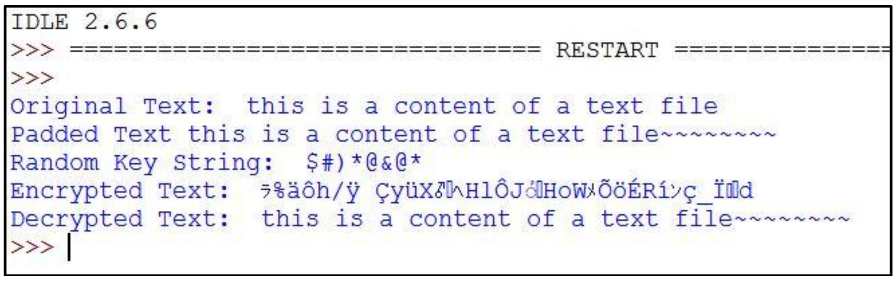


Fig. 6 – The user interface of the user after logging into the system

The next figure i.e. Fig. 6 depicts the results of the encryption and decryption we are using to protect the information in the upload stations as well as during transmission.



```
IDLE 2.6.6
>>> ================================ RESTART ================
>>>
Original Text:  this is a content of a text file
Padded Text this is a content of a text file~~~~~~~~
Random Key String:  $#)*@&@*
Encrypted Text:  ｦ%äôh/ÿ ÇyüX♂^HlÔJ♂HoW×ÕöÉRí›ç_Ïd
Decrypted Text:  this is a content of a text file~~~~~~~~
>>> |
```

Fig. 7 – Result showing the encryption that is being used in our system

V. Conclusion And Future Work

Our system is working as per our expectation. The algorithms that was initially sketched upon papers with pencils using the mathematical formulas and concepts are working in a real life system. Moreover, we are still working in the system to make it further better. In future we will try to integrate this system with body area network and wearable system so that it can be used to ease our daily life burdens and tasks that we perform every day.

Acknowledgement

We sincerely thank Mr. Avranil Tah, Founder of ESL and Mr. Arnab Pal, Techno-Marketing Consultant of ESL for guiding us during the project. Their help in this project really made our task easy. We faced many difficulties in that period of time, regarding the transmission of Bluetooth which we ultimately managed to solve by their intense guidance. Moreover, we want to thank our colleagues and friends for drawing our attention to the most important and vital overlooked points.