Received Date: 2nd November, 2022
Accepted Date: 10th April, 2023

Decentralized, On Demand Server System Using Smartphones

Anish Sah[1*], Manita Dahal[2], Pratik Sharma[3], Sukirti Bhattarai[4] & Sanjivan Satyal[5]

[1]*Dept. of Electronics and Computer Engineering of Tribhuvan University, Kathmandu, Nepal. E-Mail: anishsah1205@gmail.com*
[2]*Dept. of Electronics and Computer Engineering of Tribhuvan University, Kathmandu, Nepal. E-Mail: manudahal2@gmail.com*
[3]*Dept. of Electronics and Computer Engineering of Tribhuvan University, Kathmandu, Nepal. E-Mail: spra99.tech@gmail.com*
[4]*Dept. of Electronics and Computer Engineering of Tribhuvan University, Kathmandu, Nepal. E-Mail: sukibhattarai@gmail.com*
[5]*Dept. of Electronics and Computer Engineering of Tribhuvan University, Kathmandu, Nepal. E-Mail: sanziwans@pcampus.edu.np*

Abstract— **The advancement in technology has impelled entities to use websites for communication. Some websites such as result publication sites usually have low traffic flow except on days like result publication. The sudden spike in traffic flow makes the websites respond slow or even crash thus requiring a dynamic allocation of servers. This project, "Decentralized, On Demand Server System Using Smartphones", is designed and implemented to use smartphones as on-demand servers for hosting websites. The system consists of three components: 'System Website' controls the overall system and 'Smartphones' act as servers for the 'Client Websites' that are to be hosted. Smartphones can be registered and become servers/hosts using a mobile application. A client website is first uploaded to the system and distributed to the registered smartphones. When an end-user requests a website, the request is redirected by the System Website to the smartphone which holds the requested site. A connection is established where the smartphone hosts the website to the end-user. After the hosting is completed, the host becomes free. In case of network failure, a network is established using different host that makes the system reliable.**

Keywords — Decentralized Networking, Reverse Proxy, Server, Socket Programming Introduction

Networking has fascinated the world, as it has allowed billions of people throughout the globe to communicate with each other. With the advancement of networking technologies, cloud computing has become a major buzzword in the field of computing: carrying out big data processing, handling a large number of requests in e-commerce, or simply handling a server for a small shop in a corner of the street, cloud computing is everywhere. Every sector is moving towards the online world where they provide services to their customers through online tools which require websites, and servers, so we wanted to work in this field.

In carrying out the research relating to this field, we got to know about the limitations of the current technology, and its inefficiency. The demand for servers is ever rising and only a few companies (Google, Amazon) are providing these services making the market centralized and the crash of one server may affect many people and cause losses of billions. So, this project creates a more efficient and robust system that can handle the growing demand for digital tools by properly utilizing the resources that we already have and with high reliability. The available resources here mean the smartphone and its processing capabilities.

** Corresponding Author*

According to a report on Mobile User Statistics in May 2021, there are 3.8 billion smartphones in the world. Most of the time, a maximum of these devices remain inactive whereas huge servers are being run to support cloud computing resulting in the expense of a huge number of resources. By using smartphones as servers, the project makes the most out of the computing resources provided by modern-day smartphones. The project targets to provide rental service to meet the dynamic requirements on any particular day or occasion. Through the decentralized network of smartphones behaving as servers, the users get access to the required data from the nearest available device reliably. Content-based addressing makes sure that the user gets the desired data most efficiently and location-based addressing ensures that the resources are made available from the nearest server. Compared to the HTTP (Hypertext Transfer Protocol) which is location-based only, IPFS is better as it is decentralized and content-based, and the collapse of a single node will not impact the whole system. This project has been able to gain the advantages of both the above-mentioned protocol.

I. Related work

A server system is an essential component of any networking system and with the advancement and the necessities, the server system has evolved from a centralized, decentralized to a distributed system with each having its own merits and demerits. A decentralized network provides a way to avoid a single point of failure while also keeping track of the overall functioning of the system to keep track of the network flow. It also provides a way to divide the workload among the various devices within a network.

The approach of GFS allows replacing the expensive high-capacity servers with a large number of cheap, low-power devices. It provides fault tolerance and attends to the vast storage needs of the modern era. While sharing many of the design concepts of GFS, we accommodate smartphones and focus on providing a server platform rather than executing computational tasks[1].

The network is established among many computers where each is either server or client. The server offers services that the client has asked for. Clients rely on servers for various kinds of services, resources such as files, devices, and processing power. When the client program asks for any of the resources, the communication between client and server is established via the sockets. Sockets are the programming interfaces provided by international communication protocols such as TCP and UDP for the transport layer of the TCP/IP stack. With the reach of the internet expanding, the need for communication among the various devices has increased as well.

Socket programming can be done in Java programming language and allows the communication between the client and server through sockets which allows the client and server to write or read data to or from the other party. Java has provided a package called java.net that can be used to establish a socket connection among the various devices connected in the network. This concept has been implemented in our project using the java.net package to establish the communication between host devices and end-user [2].

A proxy server acts as a buffer between the client and server interconnected in a network. Any data that is interchanged in the network is passed to and from the server to the proxy server and eventually to the client. The main objective of the proxy server is to maintain anonymity at both ends and ensure security. A reverse proxy is a form of proxy which is used to pass requests from the internet, through a firewall to private networks. In our project, we implement this concept to redirect the end-users to the server device while also preventing the users from having direct, unmonitored access to private data residing on servers of the network[3].

IPFS is a distributed web that is an alternative to all present distributed systems and HTTP. The report "Offline but still connected with IPFS based system"[4] focuses on decentralized, data distribution and storage in offline conditions. The present system focuses more on the server-client approach or request-response approach. This system has been in use by some of the elite servers but is not generalized to the general public. We use this concept to make maximum use of its benefit and eradicate the problems that come with the web server approach. Offline condition, here, means the device involved in sharing the file can connect to each other without connecting to the global internet ensuring the no point of failure and the nodes involved does not need to trust each other as stated in the report "IPFS - Content Addressed, Versioned, P2P File System"[4]

II. Methodologies

A. System Description

It is proposed to develop a decentralized, on-demand server system using smartphones. The core objective of this project is to make smartphones to be used as servers on-demand to host websites in a network. This project forms networks of smartphones. Anyone who wishes to have their website hosted could upload their website to our system website and it would be distributed onto the predetermined certain number of registered smartphones: the smartphones which are registered to the system website. When the end-users request for those hosted websites, the system website would redirect the requests to the smartphones, which eventually would host the website.

Basic Working

The working of the system is in chronological order as shown in the block diagram. The system operates in three phases and a total of six steps as described below:

Prerequisite phase: In this phase, all the prerequisite tasks are done.

1. For any smartphone to be able to act as a server, it needs to first register itself to the System. The registration is handled by the System Website. First

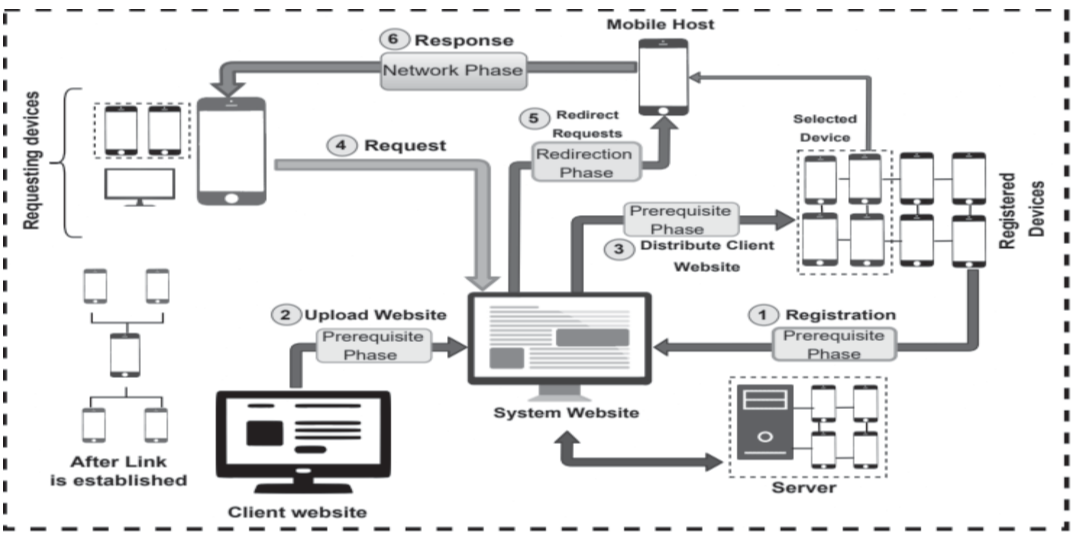

Figure: System Block Diagram

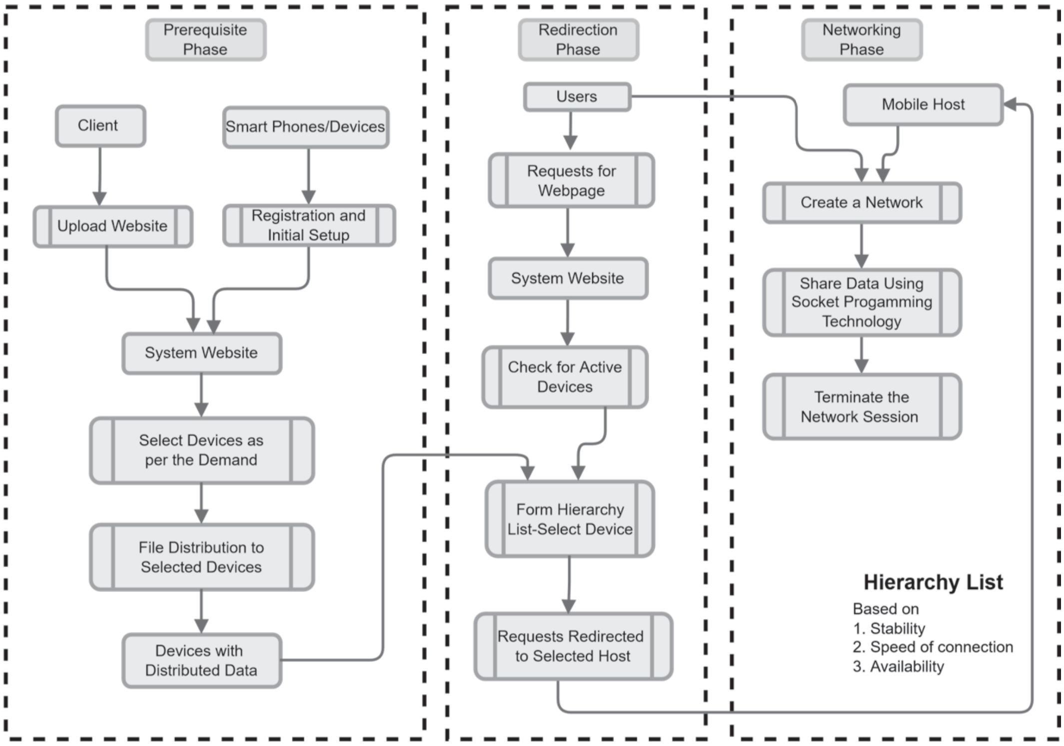

Figure: System Block Diagram

of all, smartphone's need to install a mobile application (System Application). During this installation, a part of the smartphones memory is reserved and controlled by the System Application.

2. Any website which is in need of our service is called the Client Website. Initially, the Client Website along with the data to be shared is uploaded to the System Website's Database.

3. After that, the Client Website is distributed by the System to the selected smartphones. The smartphones are selected through a Hierarchy List. Only the first-tier smartphones (primary distributors) will have access to the file database during the initial distribution phase. After that, other tier smartphones will receive the data to be hosted through the first-tier smartphones. This frees the system from the task of distributing every data itself and thus prevents bottleneck conditions. The smartphones would access the database for the addresses of the receivers.

Note: The Hierarchy List is maintained by the System Website. Each registered smartphone gets rewarded with points on the basis of its stability, availability, and speed of network connection. More the points, the higher on the list. The devices from the top of the list are selected as first-tier smartphones (primary distributors).

The memory spaces in the smartphones that are reserved and controlled by the System Website, cannot be accessed by the smartphones' owners as long as they are registered.

Redirection Phase: This phase starts when any end-user wants to visit a website hosted by our system.

4. The end-user requests for a website just like in normal internet through URL. The request is first transferred to the System Website.

5. Upon receiving the request, the System redirects the requests to the selected smartphone: the System Website checks for the available

smartphones/devices and selects a smartphone based on the 'Hierarchy List' and the requested data.

Networking Phase: In this phase, a virtual network is created between the selected smartphone/device and the particular requesting devices and thus forming a one-to-many topology that increases the hosting capacity of the network.

6. In response to the requested webpage, the smartphone provides the data to the end-user. When the sharing is done, the network is terminated, thus making the smartphones/devices available to the system. The available devices can again be used to serve other users or the data can be rewritten by another client website.

The working of the proposed system can further be explained using the following example:

Example: Result Publication

Let's take an example of online result publication. The website's visit count is minimal on normal days but on the result publication day, its traffic increases to the extent where the server limit is exceeded and the website is inaccessible to the users. It is economically inefficient to buy more server space which would remain idle for most of the days of the year.

This issue can be solved by our proposed system. First, the client website needs to share the result to be published on our System Website prior to the publication. The System Website will distribute it to the selected smartphones. On usual days it works normally with minimal traffic with fewer hosting smartphones. But on the result publication days, the number of hosts increased significantly to match the heavy traffic on the website. This is accomplished by distributing the data to additional smartphones/hosts beforehand. The end-users and the selected smartphone make a virtual network in response to the request generated. The virtual network is terminated after the data is done sharing. The data from the smartphone's memory will be erased after the application is uninstalled.

III. **Result and analysis**

The Client Website is registered and uploaded to the system database for hosting. This is done using the System Website. The registration and uploading phase is simple. The smartphone that is to be used as a server is registered to the System Website database. Then a unique user id is provided to identify each user. After registration the Status and IP address of the device is updated timely. Update is done on the basis of change of IP address of the smartphone. The host is able to serve multiple requests from multiple devices. It was tested and observed that the host could serve more than 20 different devices without showing signs of lags. The fetched file from the database is stored in the smartphones memory. This memory is not accessible even to the smartphone owner; hence the integrity of the websites is maintained. The website stored in this reserved memory is replaceable and is easily overwritten if another website needs to be hosted.

Table 1: App storage data

App	8.37MB
Data	Variable (During test 57.34KB)
Cache	Variable (During test 221 KB)
Total	8.65 MB (During test)

Table 2: Battery consumption data

Foreground Running	517 mA/h (tested for 4000 mAh battery)
Background Running	345 mA/h (tested for 4000 mAh battery)

IV. **Conclusion**

This project successfully designed and implemented a system where smartphones act as decentralized servers

References

[1] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," ACM SIGOPS Operating Systems Review, vol. 37, no. 5, pp. 29–43, 2003.

[2] Kalita Limi Socket Programming. Department of Science and Engineering. Assam Down Town Univerrsity.

[3] C. Yogita (2015). A Study of Recent Research Trends of Proxy Server. International Journal of Advanced Technology in Engineering and Science.

[4] Criestea, Alexandru-Gabriel. Alboaie, Lenuta. Panu, Andrei. Radulescu, Vlad. (2020) Offline but still connected with IPFS based system.