

# Time Stamp Server Design Documentation

Samika Kashyap : 33151952

Kinnri Sinha : 32600488

October 1, 2021

## 1 Introduction

### 1.1 Purpose

Create a distributed synchronization time stamp server that can handle multiple clients request for synchronization.

### 1.2 High Level Design

We have build a time stamp service in java, that connects to multiple clients. The server's time is taken as the correct time that all the other client's should synchronize their clocks with. We have simulated the clients by creating multiple java programs that send request to the server to perform clock synchronization.

## 2 System Architecture

### 2.1 Architectural Design

We have created one time stamp server that will be hosted on AWS and which can connect to multiple clients as shown in the diagram. The messages passed between the server and client are in the form of UDP packets, containing the data structure Message Class. Refer Figure 1.

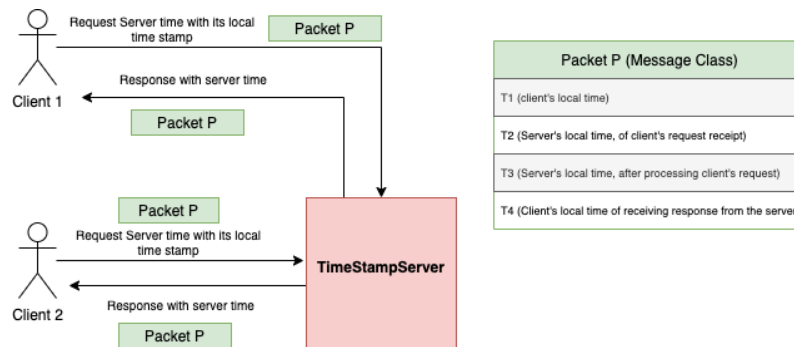


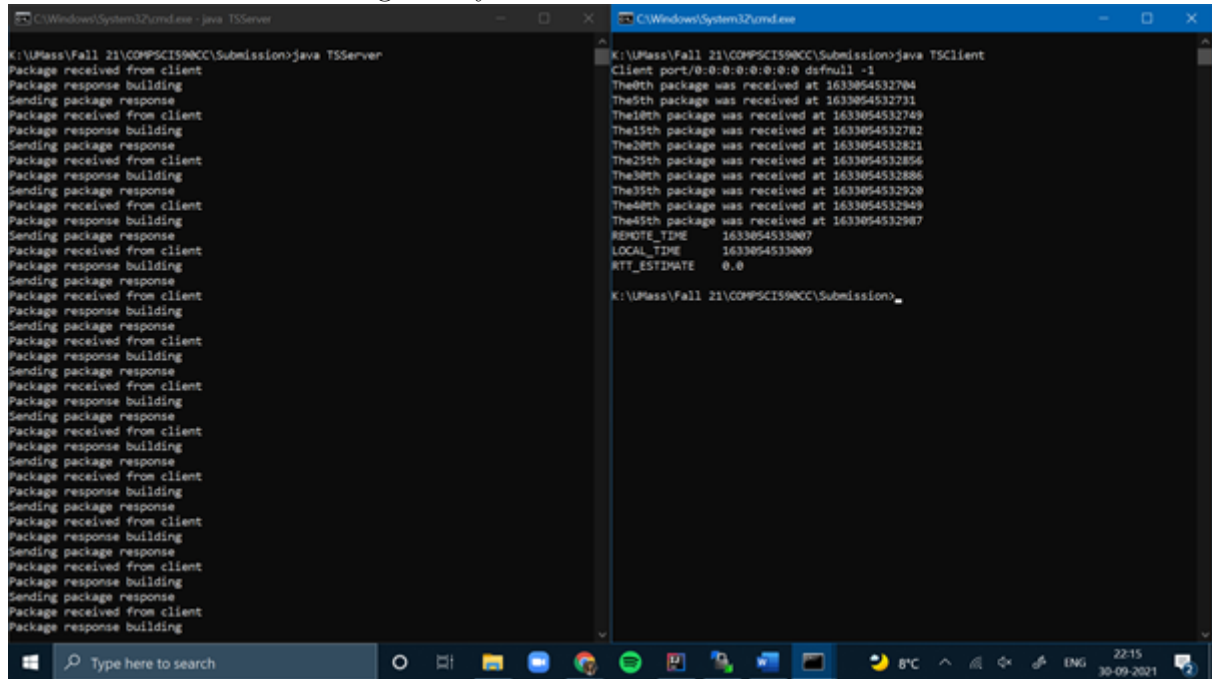
Figure 1: Architectural Design

## 2.2 Synchronization and Connection protocol description

We have used the UDP connection less protocols for the delivery of messages between client and server. This protocol allows the server to handle multiple interleaving client messages. We have implemented the NTP synchronization protocol for the clients to synchronize their clocks with the time stamp server's clock.

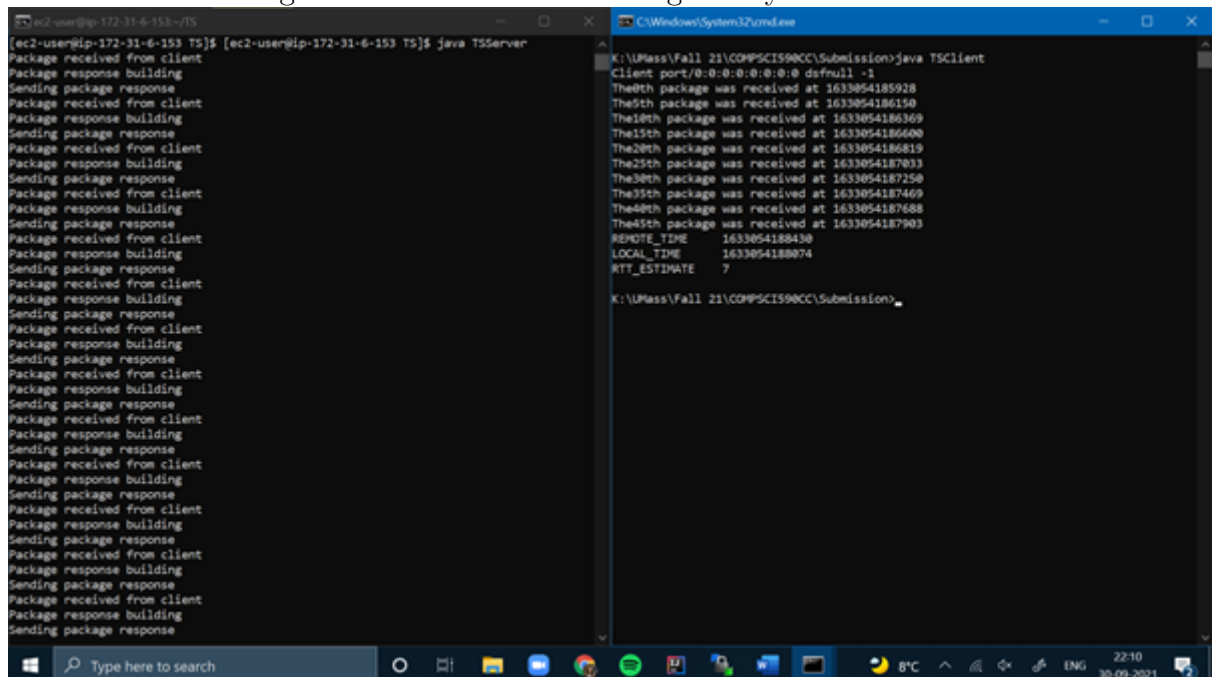
## 2.3 Screenshots

### 1. Server and client running locally



The screenshot shows two Windows command prompts side-by-side. The left prompt is titled 'C:\Windows\System32\cmd.exe - java TSServer' and displays a continuous stream of messages: 'Package received from client', 'Package response building', and 'Sending package response'. The right prompt is titled 'C:\Windows\System32\cmd.exe' and shows the execution of 'java TSCClient'. It displays the client port 'dsfull -1' and a list of received packages with their timestamps, ranging from 1633054532704 to 1633054532987. At the bottom, it shows 'REMOTE\_TIME 1633054533007', 'LOCAL\_TIME 1633054533009', and 'RTT\_ESTIMATE 0.0'.

### 2. Server running on cloud and client running locally



The screenshot shows a cloud terminal window on the left and a local Windows command prompt on the right. The cloud terminal, titled 'ec2-user@ip-172-31-6-153:~/TS', shows the execution of 'java TSServer' and a stream of 'Package received from client' and 'Package response building' messages. The local command prompt, titled 'C:\Windows\System32\cmd.exe', shows the execution of 'java TSCClient'. It displays the client port 'dsfull -1' and a list of received packages with timestamps, ranging from 1633054185928 to 1633054187903. At the bottom, it shows 'REMOTE\_TIME 1633054188430', 'LOCAL\_TIME 1633054188074', and 'RTT\_ESTIMATE 7'.

