— Verslag P&O CW 2019–2020 Taak ST1.1 —

Department Computerwetenschappen – KU Leuven **26 februari 2020**

Team 12

Frédéric Blondeel
Dirk Vanbeveren
Bert Van den Bosch

Inhoudsopgave

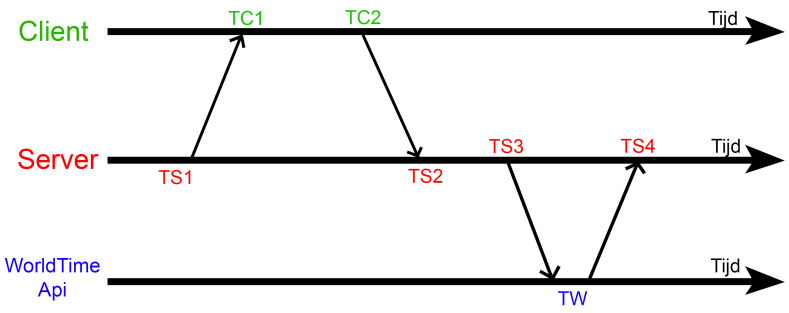
1 Introductie

Het synchroniseren van de slave devices is essentieel om video of animatie op een correcte manier over meerdere schermen weer te geven. Het eerste deel van de opgave kijkt naar het absolute verschil tussen cliënt- en referentietijd. Doordat de gebruikte server gesynchroniseerd is met een atomische wereldklok, kan het absolute verschil in tijd worden gemeten. De testen gebruiken verscheidene besturingssystemen, aparaten en browsers. Het verslag licht ook het verschil tussen UDP en TCP, twee verbindingprotocols, toe.

2 Testen

2.1 Opstelling

In de opstelling om de testen uit te voeren is er een server gebruikt waar alle clients zich op aansluiten aan de hand van een socket protocol. Met een knop op de pagina worden alle klokken gesynchroniseerd met de server door het tijdsverschil en de vertraging (ping) tussen de laatste twee te berekenen. Het tijdsverschil wordt namelijk 10 keer herberekend om daaruit mogelijke verbindingsfluctuaties uit te kunnen filteren (meer hierover in sectie 2).



Figuur 1: NTP Protocol

Voor de synchronisatie is het NTP protocol geïmplementeerd. [1] Het schema over het NTP protocol is te vinden in figuur 1. TSx en TCx zijn de tijden (in milliseconden) gemeten door de server en respectievelijk de client. Het tijdsverschil wordt gemeten door.

$$timeDiff = \frac{(TC1 - TS1) + (TS2 - TC2)}{2}$$

De vertraging voor de server om een bericht te verzenden en terug te ontvangen is:

$$ping = (TS2 - TS2) - (TC2 - TC1)$$

Hiermee is het makkelijk de servertijd te berekenen door de vergelijking:

$$timeServer = timeClient - timeDiff$$

Om vervolgens de tijdsverschil tussen de server- en de wereldtijd te berekenen is er gebruikt gemaakt van een externe API, worldtimeapi.org. Het tijdsverschil is als volgt berekend:

$$serverWorldOffset = (TW - TS4) + \frac{TS4 - TS3}{2}$$

Uiteindelijk is de volgende informatie opgeslagen:

- lijst van ping tussen client en server
- standaardafwijking van de ping
- tijdsverschil tussen client en server tijd
- tijdsverschil server en wereldtijd
- tijdsverschil client en wereldtijd

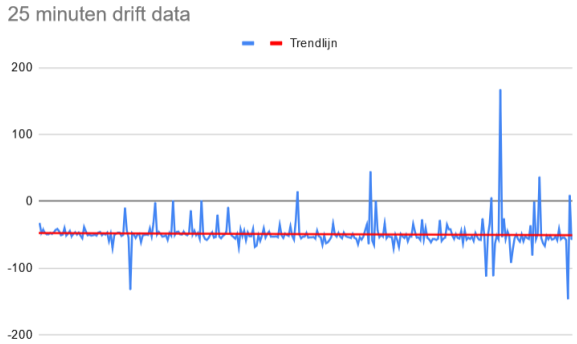
2.2 UDP vs TCP

UDP heeft in tegenstelling tot TCP geen gevestigde verbinding nodig tussen bijvoorbeeld server en client. TCP zal garanderen dat data correct aankomt door middel van foutopsporing en zal ook in de goede volgorde binnenstromen. Om geen pakketten te verliezen zal TCP deze in een *receive buffer* steken en zal de applicatie de ontvangen data pas lezen als ze er klaar voor is. Tegenover UDP waar de data continu zal binnenstromen, ontvangen of niet. Deze zal ook niet aan foutopsporing doen en de juiste volgorde niet garanderen. Het is duidelijk dat UDP veel sneller is doordat deze minder stappen en controle bevat. Nochtans is het niet mogelijk het UDP protocol te gebruiken omdat browsers deze functionaliteit niet bevatten. Daarom is er nu gebruik gemaakt van Socket.io dat met het TCP protocol werkt.

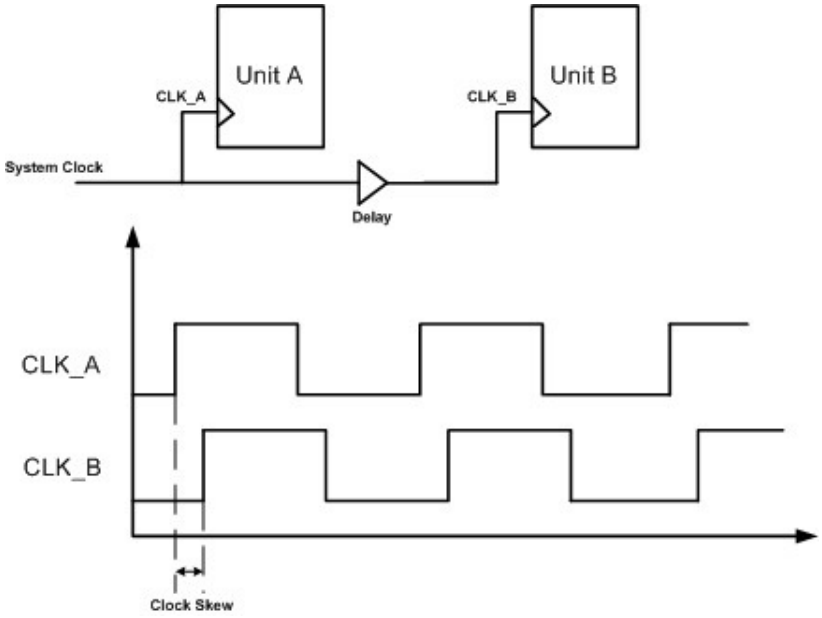
2.3 Drift en skew

Drift zal ervoor zorgen dat een klok niet meer synchroon met zijn oorspronkelijke referentie loopt. Windows lost dit op met een wekelijkse resync (het tijdsverschil zal dus op een sawtooth diagram lijken) terwijl Mac OSX rekening zal houden met de klok skew en andere hardware invloeden om zo beter synchroon te blijven. Klok skew is het verschil van tijd van een kloksignaal tussen 2 componenten binnen een systeem(zie figuur 3 [**?**]).

Het tijdsverschil van een windowscomputer en een atoomklok is gedurende 25 minuten gemeten (zie figuur 2). De trendlijn is door de grafiek getrokken en het is duidelijk dat er geen effect ervan te zien is. Over langere tijdsperiode zal de drift groter worden, maar voor de animatie zal dit verwaarloosbaar zijn aangezien het onwaarschijnlijk is dat iemand dagenlang deze zal laten afspelen.



Figuur 2: Drift

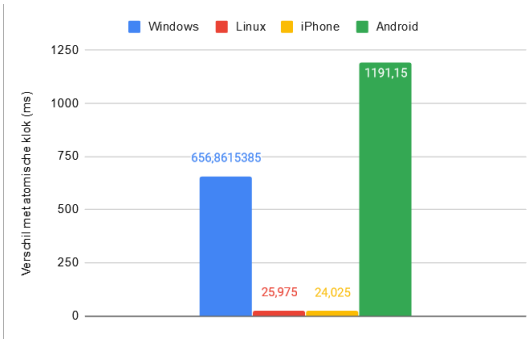


Figuur 3: Skew

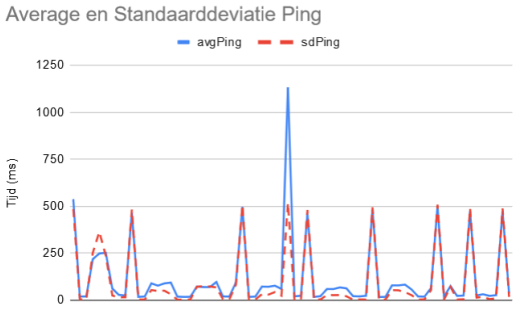
Klok skew kan problemen geven voor correct timing, en zoals eerder vermeld zal OSX dit tegen gaan via software. We zullen niet dieper gaan over de problemen van skew en OSX.

2.4 Analyse van de data

Als premisse is er gesteld dat de aangesloten clients gesynchroniseerd zouden zijn met de wereldklok, gebruikmakend van het interne synchronisatiesysteem op elk toestel. Het is duidelijk in figuur 4a dat er op elk soort systeem een afwijking gemeten wordt tegenover de referentieklok. Bij sommige operating systems al een groter verschil dan de andere, maar de reden waarom is niet altijd te verklaren of vrijgegeven en zullen hier dus niet verder op in gaan. Om dit te testen is er een poging gedaan om het verschil tussen device-klok en server-klok te vergelijken door de ping mee in rekening te brengen. De drift van beide klokken gaan over deze tijdsspan minimaal zijn en gaan geen invloed hebben op deze verschillen. Bij de testen werd ook de gemiddelde ping en standaardafwijking van de ping per test opgeslagen. Uit deze waarden valt af te leiden dat er bij grote ping fluctuaties ook een grote standaardafwijking mee gepaard ging en wijst naar inconsistenties binnen het netwerk van ofwel de client ofwel de belasting op de server en is er bijgevolg meer kans op een grotere fout binnen de vergelijking van de klokken.



(a) Mediaan van tijdsverschil van elk device.



(b) Gemiddelde en standaardafwijking van de ping met de server.

Figuur 4: Resultaten van de testen

In figuur 4b is het duidelijk zichtbaar dat er een pieken zijn met hoge standaardafwijkingen. Dit wilt zeggen dat er heel grote fluctuaties zijn in de ping tijdens het synchronisatieproces met 10 metingen. Hiermee zal de gesynchroniseerde tijd een afwijking hebben met de werkelijke tijd tot +-$(maxPing - minPing)$ milliseconden.

3 Besluit

Uit de testen is bewezen dat er een groot verschil kan zijn tussen de klok van de clients en de wereldtijd. Bijgevolg moet er dus rekening mee gehouden worden bij de synchronisatie van apparaten. In het project van pno is dit probleem opgelost door het NTP-protocol in te voeren waarmee deze verschillen gecorrigeerd worden.

Vervolgens zijn er ook fluctuaties in het netwerk dat er voor kunnen zorgen dat de synchronisatie een slechte benadering heeft dat zelfs tot een halve seconde kan oplopen. Dit kan dan opgelost worden door 10 keer de ping te berekenen, vervolgens de standaarddeviatie af te leiden en dan zal de client zich opnieuw moeten synchroniseren wanneer de berekende standaarddeviatie zich boven een vooraf gekozen waarde bevindt.

Uit de tests is het dan ook duidelijk dat de klokdrift weinig impact heeft over een korte tijdsspanne en is daardoor verwaarloosbaar.

Referenties

[1] *NTP protocol Wikiperdia.*