

1. Iteration

Client og Server

I vores projekt, bliver der på nuværende tidspunkt brugt nogle generelle netværksbegreber. Vores webserver har fx. en **IPv4 adresse** samt en **MAC adresse** på vores enhed, ligeledes har brugerne også en **IPv4 adresse** og **MAC adresse** på den enhed de benytter. Gennem disse to informationer, kan der kommunikeres mellem vores webserver og bruger ved hjælp af **TCP/IP (OSI-modellen)**, hvor data bliver genereret når det går gennem **OSI-modellens 7 lag (Encapsulation & De-encapsulation)** i takt med at det bliver leveret rundt ved hjælp af **Layer 2 (MAC-addressing scheme)** og **Layer 3 (IP-addressing scheme)** til forskellige enheder, og til sidst til enten vores webserver eller vores bruger.

2. Iteration

Direkte og indirekte protokoller der bliver brugt i vores projekt

Internet Protocol (IP)

er en logiske adresse for en bestemt enhed på et netværk, adresserne bliver uddelt af protokollen DHCP.

Dynamic Host Configuration Protocol (DHCP)

En DHCP server, står for at uddelegere IP-adresser til enheder på et netværk. Typisk så er disse IP-adresser kun til lån, dvs. at en enhed har kun en IP-adresse i en given længde, hvor den så skal anmode om at genlåne IP-adressen hvis den stadig er aktiv. IP-adresse kan dog også blive reserveret, dvs at DHCP sætter IP-adressen op mod enhedens MAC-adresse, så hver gang at enheden anmoder om at få en IP-adresse, så prøver den at give den samme IP-adresse igen. Dette er især vigtigt for fx en Web server som Kestrel og IIS, da IP adressen, specificeret på DNS serveren skal stemme overens, for at klienter kan sende requests til vores webserver.

Transmission Control Protocol (TCP)

er en protokol der skaber en sikker måde at transporterer data over et netværk, den sikrer at alt data bliver modtaget af den modtagende enhed, det gøres ved at den initiere et "Three Way Handshake" hvor der forhandles betingelser mellem de to enheder hvor der aftales f.eks størrelse på hver forsendelse af data i byte. håndtrykket bliver lavet ved at "host A" sender et SYN (synkroniserings sekvens nummer). til "Host B" som besvarer Den med et et SYN. samt et ACK (andkendese). for at angive hvilket SYN den forventer næste gang. til sidst sender Host A et ACK til Host B, derefter kan dataen blive sendt mellem de to hosts.

TCP er mest brugt til overførelse af data hvor det er vigtigt at alt data bliver overført uden tab f.eks Mail, Downloading/Overførsel af filer, og ved HTTP protokollen for at sikre at man får hele webstedets ressourcer.

men i tilfælde hvor det ikke er vigtigt at at dataen bliver overført men hastigheden har større betydningen, f.eks ved videosamtaler og streaming services vil det være bedre at bruge UDP (User Datagram Protocol) som ikke laver et "Three ways Handshake".

File Transfer Protocol (FTP) (måske er ikke sikker på om azura server bruger den eller om de bruger en anden protokol til overførsel af filer. Det er jeg 99% sikker på at de gør)

bruges til at dele overføre filer fra en host til en anden oftest i form af at overføre fra en computer til web server.

Hypertext Transfer Protocol (Secure) (HTTP(s))

er en protokol der kan hente ressourcer fra en kilde, den er hovedsageligt brugt på internettet til at kunne hente html dokumenter og css dokumenter og andre ressourcer fra et websted

fungerer også som mellemløbet mellem .netværket og den server som webstedet er lagret på.

Fordelen for http er at den er extensible, hvilket betyder at man kan nemt udvide på protokollen. Yderligere så bliver alt data transfer også krypteret, hvilket har gjort den til standarden for især forretnings hjemmesider.

3. Iteration

Overvejelser omkring implementering på flere servers

Man kan splitte sin hjemmeside op i dele, hvor hver del kører på sin egen server. Når en bruger så requester en specifik del fra index siden, så bliver requesten redirected (rerouted) til den server der håndterer den del. Man kan vælge en netværkstopologi som grundlag for den måde man sætter sin hjemmeside / servere op med. Her kunne man f.eks kigge på Mesh og Star pattern, til at sende requests og responses rundt.

Fordelene er at hvis én server går ned, så er de andre stadig oppe og køre.

Hvis en server går ned, så er den isoleret. Man kan derfor gå i gang med at fejlsøge. Alt efter størrelsen, kan det derfor være mere overskueligt at fejlsøge i sådanne tilfælde.

Ulemperne vil være at en request og response nok kan tage lidt mere tid for at komme frem og tilbage, hvis man har en "server-hub" der tager imod alle requests og redirecter til en anden server, for så at modtage og sende et response tilbage igen.

Der vil også være noget ekstra server tilsyn der skal foregå, da man nok kunne ende med at have mange servere kørende, alt efter hvor stor hjemmesiden er.