DIS - Skalerbarhed 2 - Clusters of workstations & Database og servere

Clusters of workstations. Cluster-Based Scalable Network Services

Mainframes og de der "typiske" servere er ikke cluster of workstations

Mainframes skalerer dårligt

Derfor fandt vi ud af at sætte computere sammen som et cluster. Flere forskellige computere der er sat sammen på en eller anden måde. Derfor nemmere at putte flere computere på

Clusters er smarte fordi de skalerer, tilsammen er pålidelige og er relativt billige at bygge.

Hvorfor har man dem? Fordi det er smart at skalere. Man har ikke nogen øvrer grænse for hvor mange computere man kan sætte sammen. Tilsammen er et cluster pålideligt - én computer kan gå ned, mens de andre stadig virker. Det er billigt at bygge. Man skal man have mulighed for at bygge.

Husk at det ikke betyder at alle computere i et cluster er pålidelige.
Det betyder det netop ikke.

CouchDB
Couch? Cluster of unreliable commodity hardware.
En database man kan bruge på maskiner der ikke er så dyre. Tilsammen gør det dem reliable

To summarize, clusters have significant advantages in scalability, growth, availability, and cost. Although fundamental, **these advantages are not easy to realize.**

Det har noget at gøre med, hvorfor skat skalerer så dårligt fx. Det er svært for dem at lave det. Det hænger i øvrigt sammen med at dem der er ansatte, er fra en ældre generation, så de er vant til at bygge på en anden ældre måde.

Folding at home
Et eksempel på et af verdens største clusters. Et projektproteinholdning. Finde frem til mutationer af proteiner til forebyggelse af kræft. De skal lave nogle udregninger på de forskellige universiteter, som er så store, at de ikke kan gøre det på én computer. De er nødt til at dele det op – derfor har de lavet det her projekt.

Udfordringer

Det kan være besværligt at vedligeholde et cluster, fordi man skal holde styr på mange servere.

- Organisatorisk/strukturel problem – man er ikke vant til dem.
- Der er mange der ikke vedligeholder teknologi.
- Det kan være besværligt at vedligeholde et cluster, fordi man skal have styr på rigtig mange clustere – når der er en server der går ned i et cluster, så skal man helst skifte det ud.
- 2/3 af et distribueret system skal helst fungere, før hele systemet fungere – Det hedder **Quorum**

#

Dermed kan man tvinges til at opdele sit cluster, så forskellige servere har forskellige opgaver. Man skal aktivt tage stilling til hvor hvilken data skal lægge – altså dele det aktivt op.

Det er ikke altid muligt at have en hel service (system) spejlet på hver enkelt server i et cluster.

Noter

Scalability, a computer's or network's ability to function as the number of users increases. Et netværks evne til at virke i takt med at brugerbasen stiger.

Such systems must run continuously and reliably 7 days a week, 24 hours a day and must be capable of meeting wide service demands.

Because the system must ultimately be comprehensive and able to adapt to unknown future requirements, its framework must be general, and capable of evolving over time.

Når noget skalerer, så fungerer den samme service på den samme måde når antallet af brugere og udbuddet af hardware stiger. Det foregår som minimum lineært

Når en service er tilgængelig, så skal den være det hele tiden på trods af fejl i hardware eller software. Hvis en service går ned, så er den jo ikke tilgængelig - **available** - mere

Når en service skal kunne betale sig, så skal det ikke koste en milliard at udvide med flere servere.

Database og servere

Quorom

Flertalsbestemmelse, quorom.

Det er ofte en udfordring for alle servere i et cluster at blive enige om en værdi, en fælles sandhed.

Det er en udfordring at skabe ***shared state***, som der står i artiklen.

Soft state

Soft state, which can be regenerated at the expense of additional computation or file I/O, is exploited to improve performance; data is not durable.

Soft state kunne fx være logging af hændelser med henblik på genskabelse af selvsamme ved fejl.

Soft state – det kan hedde alt muligt andet – De nævner – det er værdier på bankkontoen fx, som kan regenerres – Data der lægger mellem servere er ikke holdbart, men vi skal kunne skabe det igen, skulle det gå ned.
Filer på computeren bruger journalizering. Quoromperer den hele filen. journalizeret filsystemer som gemmer marginaler på en fil hele tiden. Det er sådan noget her

Fault-tolerance

Det er evnen til at håndtere fejl, når tingene går ned. For det gør de - ofte! det faktum at der sker fejl og man på en eller anden måde skal kunne håndtere fejl

At være fejltolerant, så at sige, handler om at spejle data flere forskellige steder og dermed gøre data tilgængeligt fra forskellige kilder.

Sharding vs. Replication

Hvordan gør man data tilgængeligt fra forskellige kilder?

Sharding
SQL databaser som MySQL, ORACLE, PostgreSQL.
Sharding bygger typisk på nøglebaseret ting

Replication
NoSQL databaser som riak, cassandra, **amazon** DynamoDB
Replication replikerer blot

Amazon case

De havde shardet dem så meget, at de ikke kunne udvide deres SQL database mere. På et tidspunkt når man skalerer en SQL database, at det kan begynde at have den modsatte effekt, fordi der er for mange servere der skal begynde at kommunikere med hinanden.

De opfandt derfor en NoSQL database som hed Dynamo, som byggede på replication – hvor man lagde data forskellige steder.

Amazon er en af de største kilder for hvorfor vi har nosql i dag – ikke fordi de hader sql

Bursts

Hvordan reagerer man på dem? Der er nogen der vælger ikke at gøre det.
Amazon reagerede på det – uanset hvor mange kommer ind, så VIL man servicerer dem.

Billetnet kunne ikke håndtere de bursts de oplevede og gjorde ikke noget ved det, så de valgte bare fejle.

SKAT

Load Balancing

det betyder at man har en belastning som man forsøger at balancere. SKAT's er et forsøg på at skabe load balancing

Man går ind på skat.dk så rammer man en load balancer – en mekanisme, ofte en server eller kan også være en router. Den undersøger hvilken server af den underliggende cluster, kan man sende denne person hen til, som ikke har så meget at lave.

Ofte er det bare en webserver som man går ind på som tager imod ens forespørgsel

Hvordan sikrer man at load balanceren ikke er det svage led – at den ikke går ned?

- Det forholdler sig sådan at den er rimelig dum, så den laver ikke noget af det praktiske arbejde.
- Loadbalanceren skal bare tage imod én og så sige at man skal hen til én server. Den laver ikke noget arbejde

Caching

Caching er kunsten at gemme noget, som man sender til andre uden at opdatere det hver gang.

En anden metode til at skabe **load balancing** – som de kan spørge til eksamen

#

Caching er når man gemmer noget til senere brug. Når man gemmer noget som man sender til andre uden at opdatere det hver gang. Man har altså nogle forespørgsler på et eller andet – der er mange der spørger om det samme – kunne det være at jeg bare skal regne det ud én gang og så sende det samme svar ud til andre

Det er normal praksis ikke at lave SQL-kald i en database ved hvert page load.