Definisjonen ...

A cluster is a group of servers and other resources that act like a single system and enable high availability and, in some cases, load balancing and parallel processing

Definisjon 2

In computers, clustering is the use of multiple computers, typically PCs or UNIX workstations, multiple storage devices, and redundant interconnections, to form what appears to users as a single highly available system

Kort om bruksområder

- Load balancing
  - Deler arbeidsoppgaver mellom flere maskiner, kan implementeres både gjennom HW og SW.
  - Vanligvis med en masterserver
  - Typisk på webfarms
  - Round robin DNS
- High availability
- Parallel prosessering
  - Beowulf ..

Cluster computing can be used for load balancing as well as for high availability. Advocates of clustering suggest that the approach can help an enterprise achieve 99.999 availability in some cases. One of the main ideas of cluster computing is that, to the outside world, the cluster appears to be a single system.
A common use of cluster computing is to load balance traffic on high-traffic Web sites. A Web page request is sent to a "manager" server, which then determines which of several identical or very similar Web servers to forward the request to for handling. Having a Web farm (as such a configuration is sometimes called) allows traffic to be handled more quickly.
Load balancing is dividing the amount of work that a computer has to do between two or more computers so that more work gets done in the same amount of time and, in general, all users get served faster. Load balancing can be implemented with hardware, software, or a combination of both. Typically, load balancing is the main reason for computer server clustering.
On the Internet, companies whose Web sites get a great deal of traffic usually use load balancing. For load balancing Web traffic, there are several approaches. For Web serving, one approach is to route each request in turn to a different server host address in a domain name system (DNS) table, round-robin fashion. Usually, if two servers are used to balance a work load, a third server is needed to determine which server to assign the work to. Since load balancing requires multiple servers, it is usually combined with failover and backup services. In some approaches, the servers are distributed over different geographic locations.

High Availability (HA) clusters are highly fault tolerant server systems where 100% up-times are required. In the event of failure of a node, the other nodes which form the cluster will take over the functionality of the failed node transparently. HA clusters are typically used for DNS server, proxy and web servers.

Cluster computing can also be used as a relatively low-cost form of parallel processing for scientific and other applications that lend themselves to parallel operations. An early and well-known example was the **Beowulf** project in which a number of off-the-shelf PCs were used to form a cluster for scientific applications.

Litt historie...

- Tilgjengelig siden 1980`s – brukt i DEC`s (Digital Equipment Corporation) VMS system
- IBM`s sysplex er en cluster for mainframe
- Microsoft, og Sun har og hadde HW og SW løsninger for clustering, skalering og økt tilgjengelighet.

Clustering has been available since the 1980s when it was used in DEC's VMS systems. IBM's sysplex is a cluster approach for a mainframe system. Microsoft, Sun Microsystems, and other leading hardware and software companies offer clustering packages that are said to offer scalability as well as availability. As traffic or availability assurance increases, all or some parts of the cluster can be increased in size or number

Mer historie (VMS)

- Står for Virtual Memory System
- Ble utviklet i 1979, for VAX maskiner – 32 bits maskiner som tok i bruk virtuell minne (swapping)
- Ble døpt om til OpenVMS når den ble videreutviklet for Alpa cpu`er.
- Open kommer fra støtten for POSIX standard
- Eksisterer fortsatt

VMS (Virtual Memory System) is an operating system from the Digital Equipment Corporation (DEC) that runs in its older mid-range computers. VMS originated in 1979 as a new operating system for DEC's new VAX computer, the successor to DEC's PDP-11. VMS is a 32-bit system that exploits the concept of virtual memory. VMS was renamed OpenVMS when it was redeveloped for the Alpha processor. (OpenVMS is also the name now used on the VAX computer.) The "Open" suggests the added support for the UNIX-like interfaces of the Portable Operating System Interface (POSIX) standard. Programs written to the POSIX standard, which includes a set of standard C language programming functions, can be ported to any POSIX-supporting computer platform.
Among other features, OpenVMS can be used with special software that facilitates its use with Windows NT servers

Mer historie (sysplex)

- **Sys**tem Com**plex**
- Kom sammen med OS/390, kjørt på mainframes
- Konseptet går ut på å koble sammen flere maskiner i et system
- Bygd for parallel prosessering, samt store OLTP jobber.

A sysplex is IBM's systems complex (the word sysplex comes from the first part of the word system and the last part of the word complex), introduced in 1990 as a platform for the MVS/ESA operating system for IBM mainframe servers. An enhanced version, Parallel Sysplex, was subsequently introduced for the newer operating system, OS/390. The sysplex consists of the multiple computers (the systems) that make up the complex. A sysplex is designed to be a solution for business needs involving any or all of the following: parallel processing; online transaction processing (OLTP); very high transaction volumes; very numerous small work units - online transactions, for example (or large work units that can be broken up into multiple small work units); or applications running simultaneously on separate systems that must be able to update to a single database without compromising data integrity.

Mer om paralell prosessering

- Def: prosessering av programinstruksjoner ved å dele dem opp mellom flere cpu`er for å bruke mindre tid
- Før i tiden – vi hadde mulighet til å kjøre kun 1 program om gangen – 1t + 1t = 2t
- Første forsøk : 1t + 1t = 1,25t
- Neste steget – multiprogrammering – flere programmer blir kjørt samtidig (i alle fall nesten)... Da kom også ressursproblemer...

In computers, parallel processing is the processing of program instructions by dividing them among multiple processors with the objective of running a program in less time.

In the earliest computers, only one program ran at a time. A computation-intensive program that took one hour to run and a tape copying program that took one hour to run would take a total of two hours to run.

An early form of parallel processing allowed the interleaved execution of both programs together. The computer would start an I/O operation, and while it was waiting for the operation to complete, it would execute the processor-intensive program. The total execution time for the two jobs would be a little over one hour.

The next improvement was multiprogramming. In a multiprogramming system, multiple programs submitted by users were each allowed to use the processor for a short time. To users it appeared that all of the programs were executing at the same time. Problems of resource contention first arose in these systems. Explicit requests for resources led to the problem of the deadlock. Competition for resources on machines with no tie-breaking instructions lead to the critical section routine.

Mer ...

- Vektorprosessering – sender inn 2 arrays av tall. Egner seg kun der hvor data er naturlig struktrurert i vektorer.
- Multiprosessering – flere CPU`er (maskiner) samarbeider for å løse en eller flere oppgaver
  - Fungerer godt nok, så lenge det er få enheter.
  - Avhengig av masternoden, som forteller alle hva de skal gjøre
- Symmetric multiprosessing system (SMP)
- Message passing systems. Programmer snakker med hverandre og forteller hva som ble endret.

Vector processing was another attempt to increase performance by doing more than one thing at a time. In this case, capabilities were added to machines to allow a single instruction to add (or subtract, or multiply, or otherwise manipulate) two arrays of numbers. This was valuable in certain engineering applications where data naturally occurred in the form of vectors or matrices. In applications with less well-formed data, vector processing was not so valuable.

The next step in parallel processing was the introduction of **multiprocessing** . In these systems, two or more processors shared the work to be done. The earliest versions had a master/slave configuration. One processor (the master) was programmed to be responsible for all of the work in the system; the other (the slave) performed only those tasks it was assigned by the master. This arrangement was necessary because it was not then understood how to program the machines so they could cooperate in managing the resources of the system

Solving these problems led to the **symmetric multiprocessing system**. In an SMP system, each processor is equally capable and responsible for managing the flow of work through the system. Initially, the goal was to make SMP systems appear to programmers to be exactly the same as single processor, multiprogramming systems. However, engineers found that system performance could be increased by someplace in the range of 10-20% by executing some instructions out of order and requiring programmers to deal with the increased complexity. (The problem can become visible only when two or more programs simultaneously read and write the same operands; thus the burden of dealing with the increased complexity falls on only a very few programmers and then only in very specialized circumstances.) The question of how SMP machines should behave on shared data is not yet resolved. .

To get around the problem of long propagation times, **message passing systems** were created. In these systems, programs that share data send messages to each other to announce that particular operands have been assigned a new value. Instead of a broadcast of an operand's new value to all parts of a system, the new value is communicated only to those programs that need to know the new value. Instead of a shared memory, there is a network to support the transfer of messages between programs. This simplification allows hundreds, even thousands, of processors to work together efficiently in one system. Hence such systems have been given the name of massively parallel processing systems

Forskjellige typer clusters..

- Computational intensive
  - Beowulf
  - Mosix
- High Availability
  - HA – Linux High Availability Project
- I/O intensive
  - Supercomputers
  - Mosix + mfs
- Distributed Systemes
  - Seti@home
  - Distributed.org
  - GRID
  - UD
- Multiprosessor
  - SGI`s

Linux based clusters are pretty much in fashion these days. The ability to use cheap off-the-shelf hardware together with open source software makes Linux an ideal platform for supercomputing. Linux clusters are either: "Beowulf" clusters, "MOSIX " clusters or "High-Availability" clusters. Cluster types are chosen depending on the application requirements which usually fall in one of the following categories: computational intensive (Beowulf, Mosix), I/O intensive (supercomputers) or high availability (failover, servers...). In addition to clusters, there are two related architectures: **distributed systems** (seti@home, folding@home, Condor...) that can run on computers with totally different architectures spread out over the Internet; and **multiprocessor machines** (those, like the SGIs are much superior to clusters when memory needs to be shared fast between processes).

Beowulf – Definisjon

It's a kind of high-performance massively parallel computer built primarily out of commodity hardware components, running a free-software operating system like Linux or FreeBSD, interconnected by a private high-speed network.

Clusters are used where tremendous raw processing power is required like in simulations and rendering. Clusters use parallel computing to reduce the time required for a CPU intensive job. The workload is distributed among the nodes that make up the clusters and the instructions are executed in parallel. More nodes means faster execution and less time taken.

Mer om Beowulf

- beowulf.tar.gz ???? (MPI, LAM, PVM)
- Programmeringsmodell for parallel prosessering
- Billig og tilgjengelig for alle
- Krever distribuert programmeringsmiljø – PVM (Parallel Virtual Machine) eller MPI (Message Passing Interface)
- Består helst av dedikerte maskiner som jobber i flere måneder på maks load
- Bruker en eller flere masternoder

Beowulf is a *programming model for parallel computation*. Beowulf clusters need distributed application programming environments such as PVM (Parallel Virtual Machine) or MPI (Message Passing Interface). LAM (Local area multicomputer). PVM is the standard interface for parallel computing. But MPI is becoming the industry standard. PVM still has an upper edge over MPI as there are more PVM aware applications when compared to MPI based ones. But this could soon change as MPI becomes popular.

Enda mer om Beowulf

- Utviklet primært for hastighet, ikke for stabilitet
- Første beowulf ble utviklet i 1994 (16 CPU`er).
- Masse beowulfs overalt nå.
- Flere universiteter og høgskoler slår seg sammen i GRID`s nå.

Mosix

- Prosessmigrering implementert i kernel
- Konkret applikasjonspakke (rpm eller src)
- Migrerer prosesser for å jevne ut minne/cpu/io forbruk på forskjellige maskiner
- Fungerer best når flere CPU-tunge prosesser går samtidig. Trenger ikke dedikerte maskiner.
- Fungerer dårlig på delt minne, men har MFS
- Kan godt fungere sammen med MPI
- Kan ha en masternode. Hidden for brukere.

OpenMosix is a kernel implementation of process migration

MOSIX is a software package that was specifically designed to enhance the Linux kernel with cluster computing capabilities. The core of MOSIX are **adaptive (on-line) load-balancing**, memory ushering and file I/O optimization algorithms that respond to variations in the use of the cluster resources, e.g., uneven load distribution or excessive disk swapping due to lack of free memory in one of the nodes. In such cases, MOSIX initiates process migration from one node to another, to balance the load, or to move a process to a node that has sufficient free memory or to reduce the number of remote file I/O operations. MOSIX clusters are typically used in data centers and data warehouses.
Mosix works best when running plenty of separate CPU intensive tasks. Shared memory is its big drawback, like Beowulf: for applications using shared memory, such as Web servers or database servers, there will not be any benefit from [Open]Mosix because all processes accessing said shared memory must resided on the same node.
MPI and openmosix play very nicely together, the processes that MPI spawns on remote nodes are migrated by the OpenMosix load balancing algorithm like any other process. So in a sense its better in a dynamic enviroment where differnt MPI programs will be running that are unaware of each other and they could both try to overload any individual node.

MFS stands for Mosix File System and allows all nodes access to all node filesystems

Andre cluster pakker der ute ..

- OSCAR - http://oscar.sourceforge.net
- FAI - http://www.informatik.uni-koeln.de/fai
- Rocks - http://www.rocksclusters.org
- Click - http://clic.mandrakesoft.com/

OSCAR Components

- LAM/MPI - ··http://www.lam-mpi.org/
- Maui PBS Scheduler - ··http://supercluster.org/maui/
- MPICH - ··http://www-unix.mcs.anl.gov/mpi/mpich/
- PBS - ··http://www.openpbs.org/
- PVM - ··http://www.csm.ornl.gov/pvm/
- System Installation Suite - ··http://www.sisuite.org/

FAI - Fully Automatic Installation for Debian

With ROCKS, you get:

- SCSI Support
- Easily reinstalled nodes
- Pre-installed queue software
- brain dead admin tools

The CLIC Phase 1 (9.0) includes :

MPICH 1.2.4
LAM 6.5.6
PVM 3.4.4

Hva som er viktig – cluster @ hiof

- Forskning
  - Paralell prosessering – Beowulf (MPICH)
  - Mer kraft – GRID
- Daglig drift - (studenter)
  - Load distribution – mosix med mfs
- Daglig drift – (IA-Drift)
  - Enklet å vedlikeholde
  - Enkelt å ta maskiner inn og ut

Beowulf @ hiof

- Sleiper2
  - God gammel sak – fases ut
- Sleipner
  - Oscarbasert cluster (+ enkelte modifikasjoner)
  - GRID
- Ny cluster til høsten

Mosix @ hiof

- Studentserver
  - Flere CPU-tunge oppgaver samtidig
  - Mye delt minne og treads – migrerer andre prosesser
  - Se på mosix eksempel (mtop, mosmon, hastighet)
- Labben
  - Mange maskiner som har lite å gjøre...
  - Prosjekter
  - distributed.net / United Devices

1. Se på mosix conf (mosix.map), rpm pakker, kernel, mosix status
2. mosmon
3. kjøre testen med mange prosesser
4. endre hastighet på noden, kjøre opp 1 prosess

mosctl setspeed 4000
mosctl getspeed
7475 – riktig på sd1

mosix og I/O oppgaver – ca 30% forbedring

Grid @ hiof

- Vi trenger mer CPU kraft
- Vi har ikke penger
- Andre har penger og masse CPU
- Kobler sammen flere beowulfs...
- Vi kan bidra med ledig CPU – det er mangel på CPU der ute ..
- Oppkobling til NorduGrid