Kunnskap for en bedre verden

Virtuelle tjenere - introduksjon

Tor Ivar Melling, Institutt for datateknologi og informatikk (IDI), NTNU

Lærestoffet er utviklet for faget IFUD 1034 Virtuelle tjenere (VMware vSphere)

Resymé: *Denne leksjonen inneholder en introduksjon til virtualisering og hypervisoren*

1. Innhold

2. Innledning

Virtualisering innebærer at en maskin kan utføre jobben til flere maskiner ved å dele ressursene til denne ene maskinen. Det vil si en abstraksjon av de fysiske komponentene til et logisk objekt, en teknikk for å skjule de fysiske komponentene i en datamaskin. Ved å virtualisere et objekt kan en dra mer nytte ut av objektet enn objektet alene. Ved å tilby flere virtuelle maskiner (VM) samme CPU vil en kunne utnytte denne CPUen bedre enn om kun en maskin skulle benytte seg av den. Denne teknikken gir deg muligheten til å kjøre virtuelle servere og klientmaskiner med ulike operativsystem og forskjellige applikasjoner på en og samme maskin.

Vi finner også flere eksempler fra andre områder, eksempelvis nettverk med VLAN og lagring med SAN og LUN. Et virtuelt LUN kan opprettes på tvers av flere fysiske LUN og overskride den totale kapasitet til faktiske fysiske enheten. Virtuelt LUN eller flere som blir opprettet med større total kapasitet enn den fysiske kapasiteten hjelper til med å optimalisere lagringen siden fysisk lagring ikke er allokert før faktisk dataene er skrevet. En slik virtuell LUN er noen ganger referert til som et *thin* (tynn) LUN (Rouse, 2015).

I dette faget se på siste versjon av vSphere ESXi og dets funksjonaliteter ved bruk at vCenter. ESXi er hypervisoren som står for virtualiseringen av hardware som gjør det mulig å installere flere operativsystem på toppen av VMene.

3. Virtualisering sin historie

Historien bak virtualisering går mye lengre tilbake enn man skulle tro. Faktum er at virtualisering ble diskutert allerede på slutten av 1950-tallet. IBM lanserte i 1961 Time Sharing som var den originale drivkraften bak virtualisering. I 11.9.64 lanserte IBM IBM System/360 som hadde få og enkle virtualiseringsmuligheter og senere det samme året kom CP-40. CP-40 var maskinen som brakte på banen betegnelser som virtuelle maskiner og virtuell ram. I 1967 kom en ny lansering av CP-40 sammen med CMS og disse hadde et system som støttet opptil 14 virtuelle maskiner hvor hver hadde 256K med virtuelt minne. Disse maskinen var første generasjon av virtuelle maskiner, og er i dag best kjent som *mainframe* maskiner. Maskinene var svært store og tok gjerne opp et helt rom. For å benytte maskinene brukte man hullkort (Portnoy, 2016).

Fra 1967 og frem til 1971 kom det hele tre nye versjoner av CR40, kalt for CP-67 versjon 1, 2, 3 og 3.1. Den siste versjonen CP-67 3.1 kom med høyhastighets I/O forbedringer som økte både ytelse og stabilitet. I 1972 kom 370 Advanced Function, som blant annet kom med støtte for fire nye operativsystem. IBM-miljøet utpekte seg som spesielt interesserte i virtualisering og på bakgrunn av dette ble MVMUA (Metropolitan VM User Association) etablert i New York i 1973. Fra 1974 til 1987 dabbet interessen og utviklingen innen virtualisering. Fra 1987 kom en gradvis utvikling og bruk av *Personal Computer (PC)* og internett. Denne utviklingen bøy på nye muligheter og behov for TCP/IP støtte. Det samme året ble det utviklet en VM TCP/IP (FAL) som gjorde det mulig å benytte internett på de virtuelle maskinene.

I 1988 kom et lite firma ved navn Connectix Corporation på banen innen virtualisering. Connectix Corporation hjalp Apple med å finne nye innovative løsninger og introduserte virtual memory lenge før det ble implementert i System 7. Blant annet så utviklet de noen programmer som løste et problem

med 24-bits minne adresser, SpeedDoubler som emulerte mellom Motorola prosessorer og Power PC baserte prosessorer. Som et resultat av erfaringen og suksessen fra SpeedDouble, laget de et nytt produkt som ble kalt for Connectix Virtual PC 1.0 for MAC. Programmet kunne oversette instruksjoner fra en virtuell Intel x86 prosessor til en fysisk Power PC prosessor brukt av en MAC. Det var denne emuleringen mellom prosessorene som førte Connectix på kartet innen virtualiseringsteknologi

11998 ble VMware startet av et ektepar og en kollega fra Berkley, pluss to studenter fra Stanford University. De lanserte i 1999 VMware Virtual Platform. Dette regnes av mange som den første kommersielle plattformen for virtualisering og produktet anses som forgjengeren til VMware Workstation. I slutten av 2000 lanserte VMware sin første plattform beregnet mot server virtualisering ved navn VMware GSX Server 1.0. Programvaren ble installert som et programtillegg på toppen av et installert operativsystem av typen Linux eller Windows.

Ved lanseringen av VMware ESX Server 1.0 tok VMware virtualisering til et helt nytt nivå. ESX Server var en server som ble installert direkte på maskinvaren og fungerte som et selvstendig operativsystem. Fordelene med å installere hypervisoren direkte på jernet var blant annet bedre ytelse, stabilitet og mindre overhead. Fra 2002 og utover så har VMware fortsatt med å levere oppdaterte versjoner av både GSX og ESX plattformene – etter hvert med bedre ytelse og flere muligheter.

Connectix gikk senere inn i et samarbeid med Microsoft og laget blant annet PocketPC emulering som er innebygd i Microsofts Visual Studio.NET. De gikk inn i x86 server virtualisering i 2003, hvor de kom med et produkt kalt Connectix Virtual Server. Produkt nådde aldri markedet som et Connectix produkt, fordi Microsoft fikk eiendomsretten til både Virtual PC for MAC, Windows og Connectix Virtual Server. Virtual PC 2004 fra Microsofts ble planlagt lansert i 2004, men på grunn av sikkerhetsproblematikk, måtte programmet oppdateres og ble ikke lansert før slutten av 2004 ved navnet Virtual PC 2005.

I dag har både Intel og AMD tatt i bruk nye teknologier for å bedre støtte virtualisering. Denne teknologien inkluderer prosessorer med flere kjerner hvor med Intel‟s Virtualization Technology og AMD‟s Pacifica prosjekt.

Kronologisk oppsett av utviklingen

1961 Time Sharing fra 1DM
1964 1DM SystemI36O
1964 CP-40
1965 1DM Systeinf3ôO Modell 67 og TSS
1967 CP-40 og CMS
1968 CP-40 version 1
1969 CP-40 versjon 2
1970 CP-40 versjon 3
1971 CP-40 versjon 3.1
1972 1DM SystenV36O Advanced Function
1973 1VIVMUA blir etablert
1974 VM/370 versjon 2
1988 Connecfix etableres
1991 CMS Multitasking
1991 P1370
1996 Connectix VPC 1.0 til Iv1AC
2004 EMC tildeles VTvlware
2004 VMware GSX Server3.0, 3.1
2004 Microsoft Virtual Server 2005
2004 VMware ESX Server 2.5
2005 VMware ESX Server 3.2, Dual-Core support
2005 Microsoft Virtual Server 2005 P2
2005 Sun Solaris 10 x86/x64
2006 VMware ESX 3.0 Beta
2006 VMware Server
2006 Microsoft Virtual PC
2006 HP Integrity Virtual Machine
2007 VirtualBox Open Source Edition
2007 VMware ESX 3.5
2008 VMware Thinstall
2008 VMware Workstation 6.5 with DirectX 9

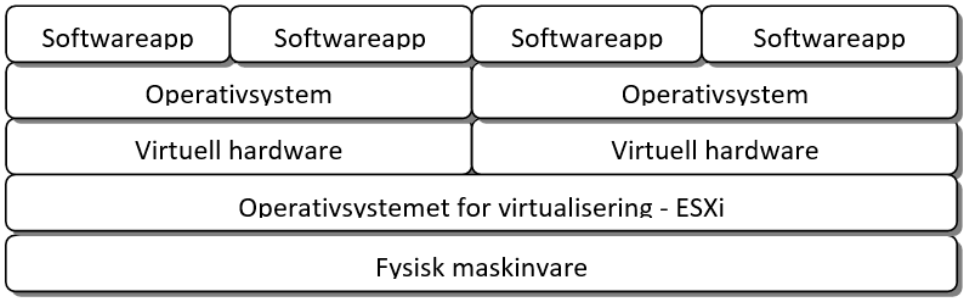
1998 VMware etableres
1999 VMware introduserer VMware Virtual Platfonn
2000 VMware GSX Server 1.0 for Linux og Windows
2001 VMware ESX Server 1.0
2002 VMware ESX Server 1.5, OSX Seiver 2.0
2003 VMware ESX Server 2.0, GSX Server 2.5
2003 VMware Virtual Center
2003 Connectix Virtual Server 1.0 RC
2003 Microsoft får Connectix VPC og Virtual Server
2009 VMware vSphere 4
2011 VMware vSphere 5
2013 Microsoft Hyper-V R2
2013 VMware vSphere 5.5
2015 Citrix XenServer 6.5
2015 VMware vSphere 6.0
2016 Citrix XenServer 7.0
2016 Hyper-V 2016
2016 VMware vSphere 6.5

4. Hypervisor

I dette kapittelet skal vi se nærmere hva en hypervisor er og hvordan den fungerer. Vi vil også se på flere forskjellige hypervisorer, selv om vi i den praktiske gjennomføringen forholder oss til VMware og vSphere ESXi.

En hypervisor er programvaren som jobber mellom den fysiske maskinvaren og en VM som kjører på serveren. Normalt sett er det operativsystemet som kommuniserer direkte med maskinvaren, diskoperasjoner går direkte kontrolleren og minne skrives / leses direkte på det fysiske minnet. For å unngå at flere VMer ønsker å samtidig ta kontroll over en ressurs, eksempelvis disk, trengs det en hypervisor som administrerer tilgangen til ressursen som alle VMene deler på.

Det finnes to typer hypervisorer, type 1 og type 2. Type 1 kjører direkte på maskinvaren uten noe mellomliggende operativsystem, som også kjent som en bare-metal hypervisor, gir en direkte kommunikasjon med maskinvaren under. Sammenlignet med type 2 hypervisor, som er en programvare installert på et underliggende operativsystem, er type 1 en kjent for å være mer effektiv hypervisor uten det ekstra laget ned mot maskinvaren samt bedre fra et sikkerhetsperspektiv. En VM kan kun ødelegge for seg selv ved at denne maskinen går i stå eller feiler. Andre VMer og den fysiske maskinen vil fortsette å kjøre som normalt uten noen påvirkning fra denne ene VMen som har feilet. I tillegg til eventuelle spart lisenskostnader ved bruk av type 2 hypervisor, har type 1 mindre overhead enn et tradisjonelt operativsystem og kan derfor kjøre flere VMer på hver fysisk maskin.

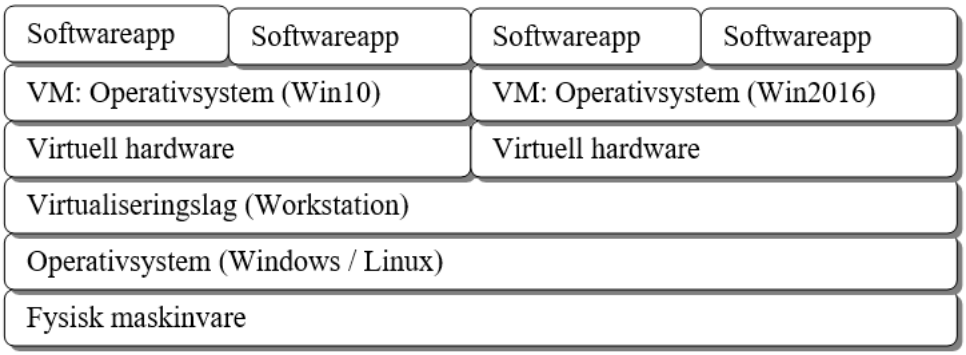


Figur 1 - Type 1 Hypervisor

I det nederste laget på figuren illustreres det fysiske maskinvarelaget. Lag nummer 2 nedenifra er hypervisoren, eller det dedikerte operativsystemet for virtualisering. Dette laget inkluderer en kjerne som behandler dataflyten mellom fysisk maskinvare, hypervisoren, og det påfølgende virtualiseringslaget (lag 3 nedenifra) som vises som virtuelle komponenter. Lag nummer 4 nedenifra er hvor operativsystemet installeres på VMene.

Type 2 hypervisor er en programvare som en installerer på et alt installert operativsystem. Siden vi allerede hadde et operativsystem som håndterte kommunikasjonen med den fysiske maskinvaren var det lettere å utvikle en hypervisor som kommuniserte med operativsystemet heller enn maskinvaren direkte. Dette er hovedårsaken til at vi først så type 2 x86 virtualisering. X86 virtualisering vil si når det ble mulig å virtualisere x86 arkitektur (“x86 virtualization,” 2017). Fordelene med type 2 hypervisor var dets maskinvarestøtte som ble arvet fra det underliggende operativsystemet, samt at installasjonen var kjent fra annen programvareinstallasjon. Ulempen er som vi nevnte ovenfor at for hver gang en VM skal kommunisere med den fysiske maskinvaren, eksempelvis disk eller nettverk, må type 2 hypervisoren videresende instruksjonen til operativsystemet som håndterer I/O forespørsler

i stedet for å direkte kommunisere med maskinvaren. Operativsystemet må deretter sende informasjonen tilbake til hypervisoren som gir beskjed til VMen. I tillegg vil alt som påvirker operativsystemet kunne påvirke hypervisoren samt VMene som kjører på hypervisoren. Eksempelvis sikkerhetsoppdateringer som krever omstart av operativsystemet vil resultere i at en må slå av VMene. Type 2 hypervisor typisk bruk som arbeidsstasjonvirtualisering av utviklere for testing og utprøving med en eller flere VMer, i motsetning til type 1 som er en dedikert hypervisor kun for å kjøre VMer.



Figur 2 - Type 2 Hypervisor

Det nederste laget på figur 2 representerer den fysiske maskinen. Påfølgende lag representerer operativsystemet som er installert på servermaskinen. Dette er et operativsystem som ikke er dedikert som hypervisor, men funksjonaliteten installeres som et tillegg på operativsystemet. Lag tre nedenfra representerer programvaren som står for selve virtualiseringen, og det er her vi oppretter og behandler VMer og andre virtuelle komponenter som programmet tilbyr, eksempelvis virtuelle svitsjer. Videre ser vi to forskjellige virtuelle maskiner, som kjører på den virtuelle maskinvaren, som er tilgjengelig ved hjelp av virtualiseringsprogrammet som ble beskrevet ovenfor. Disse maskinene er helt uavhengige av hverandre og opptrer som helt vanlige fysiske maskiner og må kommunisere med hverandre via nettverket eller et virtuelt nettverk. I eksemplet over kunne det totale antallet VMer vært flere, avhengig av maskinvaren, som avgjør hvor mange VMer den takler å kjøre samtidig (Golden, 2008; Portnoy, 2016).

Portnoy (2016) tar utgangspunkt i Popek and Goldberg (1974) når han beskriver hypervisorens primæroppgaver:

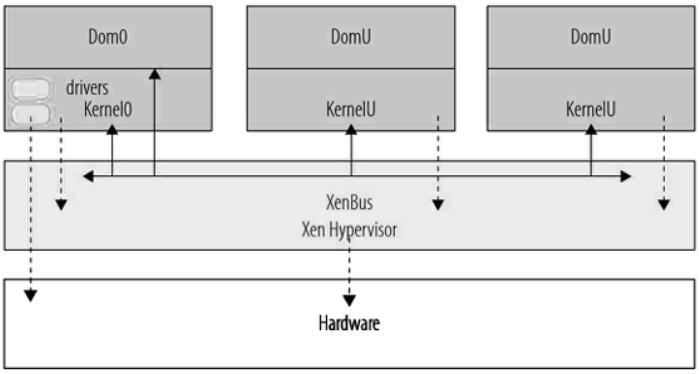
- Tilby et virtuelt område eller miljø tilnærmet likt det fysiske maskinvaremiljøet
- Tilby miljøet med et minimum av ytelseskostnader.
- Beholde fullstendig kontroll over systemets ressurser.

Med andre ord må hypervisoren sørge for at VMer som skal dele samme fysiske maskinvare få tilgang på den ressursen den trenger for å fungere optimalt. Eksempelvis skrive og lese til disk, aksessere minne samt kommunisere på nettverket. Fra et gjesteoperativ sitt perspektiv tror den i hvert fall at den gjør det. Det er her hypervisoren viser seg fra sin beste side, teknologisk sett. Ved å presentere et miljø så tilnærmet lik virkeligheten at den tror kommuniserer direkte med maskinvaren. Hypervisoren får en enkelt fysisk ressurs, eksempelvis en harddisk, til å fremstå som flere logiske ressurser. I tillegg må hypervisoren håndtere forespørslene om tilgang på fysiske ressurser fra alle kjørende VMer og balansere arbeidsbelastningen mellom dem så ingen må vente for lenge. Hvis en VM må vente lenge før hypervisoren tildeler den ønsket ressurs vil sluttbrukeren av operativsystemet eller applikasjonen få en følelse av treghet. I noen tilfeller må en sette prioritering på enkelte I/O

intensive VMer for å sikre at alle forespørslene blir tatt hånd om på en fornuftig måte. Dette gjelder for både prosessor, minne, disk og nettverk.

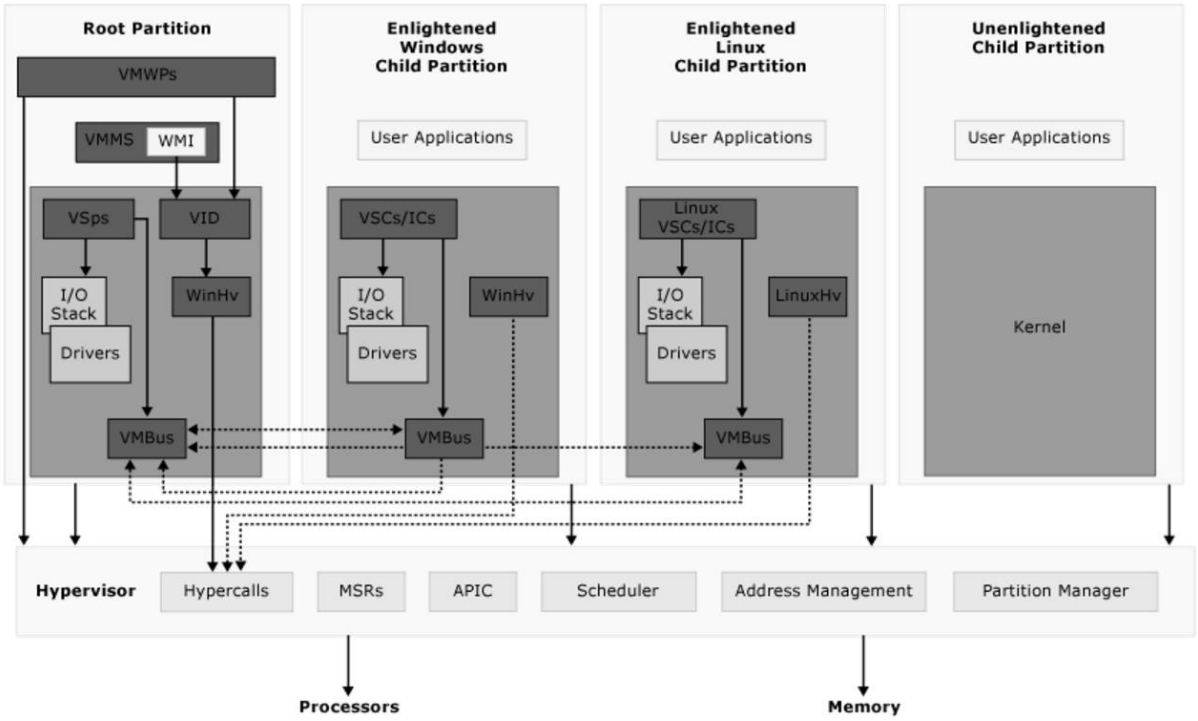
Av store hypervisorer finner vi VMware sin ESX og ESXi, Citrix Xen og Microsoft Hyper-V. VMware sin type 1 hypervisor ble etter versjon 5 (2011) kun lansert i form av ESXi. VMware valgte å gå bort fra ESX sin oppbygning med et veldig stort Service Console (900mb) som også kunne være en sikkerhetsrisiko for hypervisoren (32mb). VMware sin ESX var den første rene type 1 hypervisoren på markedet og har lenge vært og fortsatt er den mest utbredte hypervisoren.

Citrix Xen sin hypervisor er også en bare-metal løsning som jobber direkte mot den fysiske maskinvaren. I motsetning til VMware sin ESXi har Xen en forhåndsinstallert gjest, en VM som startes opp sammen med hypervisoren. Denne omtales som Dom0 (Domain 0) og er maskinen som håndterer ressursstyringen fra andre VMer. Dom0 benytter seg av ressurser som alle andre kjørende VMer på hypervisoren.



Figur 3 - Citrix Xen (Portnoy, 2016)

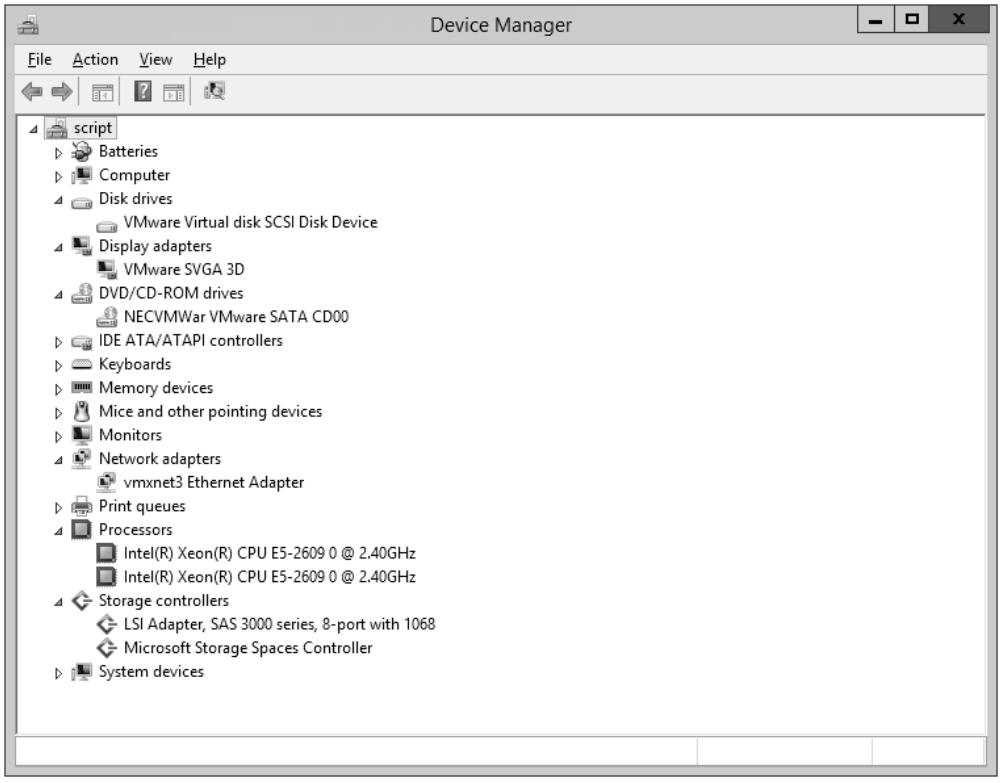
Før Hyper-V ble lansert hadde Microsoft, som de fleste andre leverandørene, en type 2 hypervisor ved navn Virtual Server. Denne ble tatt ut av produksjon og erstattet av Hyper-V i 2008 som en rolle en kunne installere på Windows 2008 Server operativsystemet. Hyper-V er på mange måter lik XenServer hypervisoren med en gjeste, kalt *Parent Partition*[i], som kjører Windows Server og står for kommunikasjonen med maskinvaren og administrasjonen av *Child Partition*[ii] som inneholder gjesteoperativsystemene. *Child Partition* har ikke direkte kontakt med maskinvaren og alle forespørsler om tilgang på ressursene går via *VMBus*[iii], som er en intern kommunikasjonskanal, og ned til hypervisoren ("Hyper-V Architecture," n.d.) (Portnoy, 2016).



Figur 4 – Hyper-V arkitektur ("Hyper-V Architecture," n.d.)

5. Virtuelle maskiner

Virtuelle maskiner (VM) er det fundamentale konseptet av virtualisering, og selv om innsiden av en VM ser temmelig lik ut som en fysisk maskin, er utsiden betydelig forskjellig. En VM er et sett med filer hvor konfigurasjonsfilene beskriver den virtuelle maskinvaren og diskfil(ene) inneholder det installerte operativsystemet, programvare samt annen data som lagres på VMen. Operativsystemet har tilgang til et sett med virtuell maskinvare definert i konfigurasjonsfilene, men fra et operativsystemets ståsted, vet den ikke om denne maskinvaren er ekte eller ikke. Virtuelle maskinvare er et sett med standardenheter for alle maskiner. Denne standardiseringen gjør VMene fleksibel på kryss av den underliggende fysiske maskinvaren og en trenger ikke forholde seg til nye drivere om en skulle kopiere eller flytte en VM sine konfigurasjonsfiler og diskfiler til en annen fysisk maskin med samme hypervisor. Ved å åpne *Device Manager* på en VM ser vi de standardiserte enheten som operativsystemet alt har drivere til, eller får installert gjennom VMware Tools for VMware sine hypervisorer. Denne er identisk



Figur 5 – Device Manager på en virtuell maskin

Vi nevnte ovenfor at det er en stor fordeler med VMer og standardisert virtuell maskinvare. Denne fysiske maskinvareuavhengigheten gjør maskinene fleksible og lett å håndtere. En Virtuell maskin er:

- Enkelt å flytte / kopiere. VMen består bare av filer.
- Er ikke bundet til en bestemt fysisk maskinvare
- Enkel tilgang på konsoll for administrasjon
- Enkel å migrere ved behov for å lastfordeling.
- Enkel å oppgradere virtuell maskinvare, også mens VMen kjører.

6. Fordeler ved å ta i bruk virtualisering

Noen av fordelene ved å ta i bruk virtualisering å bedre utnyttelsen av tilgjengelige ressurser. Maskiner med kraftige prosessorer og mye minne som står uten arbeidsoppgaver er dårlig utnyttelse av ressurser, og en kan for eksemeple se på sin egen arbeidsstasjon at belastningen på CPU er under 10 %, som er svært vanlig, over lengre perioder. Ved å virtualisere den fysiske maskinvaren og kjøre flere servere, og eller arbeidsstasjoner, på en fysisk server, kan ressursene utnyttes mye bedre og en får mer igjen for maskinvareinvesteringen. I og med at enkelte virtuell maskin kan ha intervaller med belastning av CPU og står den resterende tiden med 10% belastning, vil det være fornuftig å kjøre flere, eksempelvis 5-10, maskiner samtidig på en og samme fysiske maskin. Konsolideringen av maskiner, vil i tillegg til økt ressursutnyttelse, kunne redusere kostnader i forbindelse med den fysiske infrastrukturen. En reduksjon i fysisk maskinvare vil påvirke behovet for strøm og kjøling, som igjen gir en reduksjon i behov for kjølesystemer og strømkostnader.

Tilgjengelighet er kritisk for tilfredsheten blant sluttbrukere. Ved å ta i bruk virtualisering har man enkelt muligheten til å sette opp gode sikkerhetsrutiner som gjør det kjapt og enkelt å få opp systemet etter uforutsette hendelser og datakrasj. Man kan også eliminere planlagt nedetid ved at man kan migrere komplette virtuelle miljøer mellom fysisk maskinvare og geografisk lokasjon uten stans i driften.

Virtualisering tilbyr en ny metode å administrere enkeltmaskiner og infrastrukturen. Ved å virtualisere både nettverk, lagring og maskiner kan administratorene forholde seg til ett og samme brukergrensesnitt. Reduksjon i repeterende oppgaver innen konfigurasjon, vedlikehold og overvåkning er en stor kostnadsbesparelse. I tillegg kan en sentralisere administrasjon og overvåking ansatte sine klientmaskiner, både på og utenfor bedriftens fire vegger.

Rask utrulling av nye servere og klienter er en enorm besparelse i både tid og arbeidsmengde. Ved å konfigurere maler av servere og klient, tar ikke mer enn 5-10 minutter å rulle ut flere nye servere og klienter. Utrullingen kan også automatiseres som gjør det mulig å tilby nye servere *on demand*[iv]. *On Demand Computing* (ODC), sammen med *Software-as-a-Service*[v] (SaaS) og *Cloud Computing*[vi] er modeller som kommer av de teknologiske mulighetene for å tilby ressurser via programvare.

Virtuelle labber har gjort det mulig å lage små miniatyrutgaver av store komplekse systemer for testing og opplæring. Man kan gjøre de forandringene og utprøvingene man ønsker, uten å være redd for at noe skal påvirke andre eller at en har ødelagt for seg selv. En funksjonalitet som snapshot, gjør det mulig å ta et tilstandsbilde av maskiner og konfigurasjon hvor man enkelt kan gå tilbake til dette stedet ved behvo. En slipper å bruke flere timer på reinstallasjon av maskinen og konfigurasjon. Det er heler ingen fare å la helt nye og uerfarne brukere bruke de virtuelle maskinene for læring.

7. Maskinvare

Vi må ha alle de fysiske komponentene som en hvilken som helst sever må ha. Forskjellen blir i all hovedsak ytelsen og størrelsen på de enkelte enhetene. Maskinvare tiltenkt virtualisering har ofte flere fysiske CPUer med mange kjerner som igjen tilbyr *hyperthreading*[vii]. Av minne er det ikke uvanlig at vi ser tall som 256 og 512GB med minne pr server.

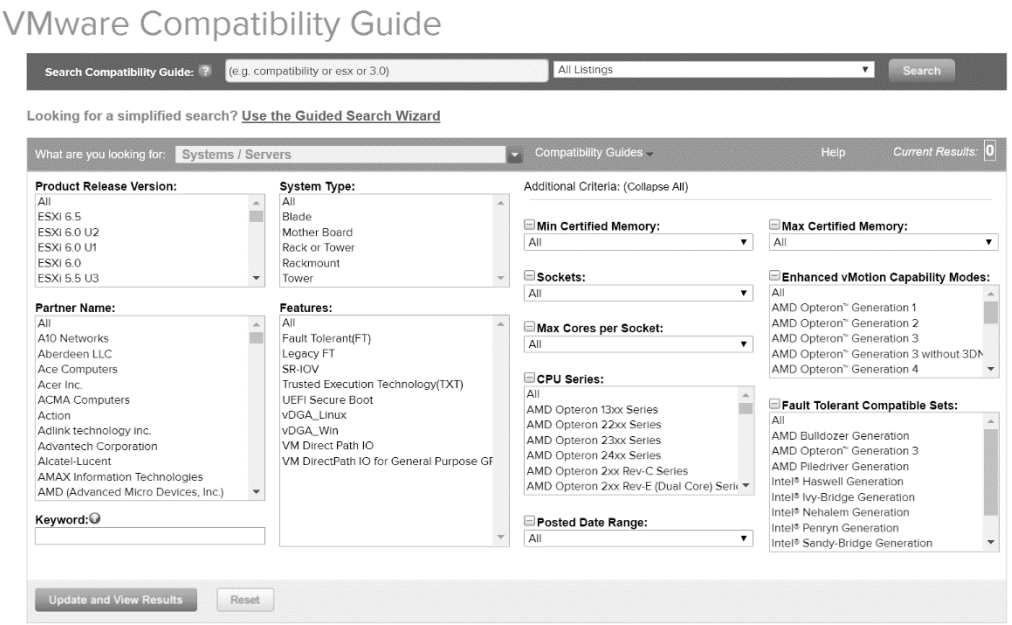
Den største svakheten til ESXi Server er at de i mindre grad gir støtte til alle typer maskinvarekomponentene. VMware praktiserer en streng driver modell for å kunne tilby en best mulig driftssikkerhet. Dette gjør at ESXi er mere bundet til en bestemt type maskinvare enn Microsoft Hyper-V og Citrix XenServer. Operativsystemene som Microsoft og Linux har en bredere driverstøtte i bunnen. Den positive siden med denne minimalistiske støtten, er at VMware har valgt å bruke maskinvare som gir best mulig ytelse og muligheter for kontroll av ressurser.

For å finne hva som støttes av hardware har VMware laget sine egne lister over systemkrav og hvilken maskinvare som støttes. Her kan du velge versjon og finne HCL – Hardware Compatibility List. http://www.vmware.com/resources/compatibility/search.php

Her kan du velge hva for versjon av ESX / ESXi du ønsker å finne kompatibel hardware.

Eksempel:

- Versjon: ESXi
- System type: Blade
- Sockets (antall plasser for prosessorer)
- Max cores pr socket (antall kjerner pr prosessor)
- Partner Name: DELL



Figur 6 – Vmware HCL

Minimumskravet for å kjøre ESXi, siste versjon:

- ESXi 6.5 requires a host machine with at least two CPU cores.
- ESXi 6.5 supports 64-bit x86 processors released after September 2006. This includes a broad range of multi-core processors.
- ESXi 6.5 requires the NX/XD bit to be enabled for the CPU in the BIOS.
- ESXi 6.5 requires a minimum of 4GB of physical RAM.
- To support 64-bit virtual machines, support for hardware virtualization (Intel VT-x or AMD RVI) must be enabled on x64 CPUs.
- One or more Gigabit or faster Ethernet controllers.
- SCSI disk or a local, non-network, RAID LUN with unpartitioned space for the virtual machines.
- For Serial ATA (SATA), a disk connected through supported SAS controllers or supported on-board SATA controllers. SATA disks will be considered remote, not local. These disks will not be used as a scratch partition by default because they are seen as remote.

ESXi har veldig godt på minnehåndtering for virtuelle maskiner. Vi kan tildele minne til virtuelle maskiner som til sammen blir høyere enn det totale minnet ESXi hosten har fysisk tilgjengelig. Dette er god praksis ettersom flere av kjørende VMene ikke vil ta i bruk alt av tildelt minnet og det ledige minne kan tildeles en VM som trenger det mer der og da. Faren med denne praksisen er hvis VMene opptar hele det fysiske minnet til ESXi servere, og sidevekslingsfilen(e) blir tatt i bruk. Dette medfører at data skrives og leses til diskløsninger i stedet for minne. En kan øke ytelsen ved tilfeller av sidevekslingsfiler med SSD-lagringsløsninger for ESXi serveren. Det er ikke en ønskesituasjon ettersom VMer som tar i bruk sidevekslingsfiler kan få dårligere ytelse.

Fra og med ESX 2.5 hadde en mulighet til å boote ESX installasjon fra et SAN. Dette kan by på store fordeler siden det er blitt mer vanlig å bruke bladservere - noen av disse kan være uten en lokal disk. Det er en god skikk å bruke all form av lagring på et SAN siden dette gir oss god feiltoleranse, ekspanderingsmuligheter og mulighet for replikasjon av bootdisk. Dette kan selvsagt også være en mulighet selv om vi har disker på en bladserver. Men hvis en feil skulle oppstå med systemet vil det bli lettere å feilsøke på en server der installasjonen er på den fysiske disken på en server. Det kan også være greit å utnytte den lagringsplassen man har og ikke bruke SAN-diskplass til ESXi-installasjon.

VMer bør alltid lagres på SAN, i testformål kan en så klart velge hva en ønsker selv, men hvis man har sentralisert lagring, i form av SAN / NAS, er det absolutt å anbefale. Hvis en ESXi server skulle krasje, vil det være mye lettere å hente inn VMer fra sentralisert lagring enn fra en fysisk disk som er på en server som det kanskje ikke vil starte. En viktig ting med tanke på lagring er disk I/O. For å kunne ha høy ytelse på I/O-operasjoner bør vi ha et SCSI-RAID system med høy ytelse eller en Fibre Channel (HBA) som er tilkoblet et SAN. Jo større cache og antall disker vi har, jo høyere ytelse får vi. Aksesstiden, båndbredden og forsinkelsen til disken bør nøye kontrolleres for å oppnå den ytelsen vi ønsker i det virtuelle miljøet. For å forbedre ytelsen på systemet bør OS, VMFS partisjoner og sidevekslingsfiler være på hver sin kontroller og disk.

På en server med installert ESXi er det absolutt anbefalt med flere enn ett nettverkskort. Dette for å kunne ha redundante løsninger og separere administrativ trafikk fra annen nett trafikk. For en test server er det ikke nødvendig med mer enn ett nettkort. Mens maskiner i produksjon trenger løsninger med høy ytelse og god sikkerhet.

8. Oppsummering

Vi har nå vært igjennom teori om hvorfor og hvordan vi benytter oss av virtualisering. Sett på fordeler ved bruk av virtuelle maskiner kontra fysiske. Sett på virtualiseringens historie og litt på forskjellene mellom noen eksempelvis VMware Workstation og vSphere ESXi . Videre i neste leksjon skal vi se på installasjon av ESXi og vCenter.

For de som ikke har tilgang på eget ustyr er det mulig å sette opp en testlab her på skolen som dere kan styre hjemmefra via vSphere Client (klienten som benyttes for å administrere vSphere ESXi og vCenter). Hva vCenter er kommer vi mer inn på i neste leksjon.

VMware har laget en fin presentasjonsvideo med veldig forklarende. Noen av temaene tatt opp i videoen kommer vi til i senere leksjoner. Anbefaler å se denne etter gjennomgått leksjon som et supplement for forståelsen av virtualisering.

9. Referanser

Golden, B. (2008). *Virtualization for dummies*. Hoboken, N.J: Wiley.

Hyper-V Architecture. (n.d.). Retrieved February 1, 2017, from https://msdn.microsoft.com/en-us/library/cc768520(v=bts.10).aspx

Mell, P., Grance, T., & others. (2011). The NIST definition of cloud computing. Retrieved from http://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf

Popek, G. J., & Goldberg, R. P. (1974). Formal requirements for virtualizable third generation architectures. *Communications of the ACM*, *17*(7), 412–421. https://doi.org/10.1145/361011.361073

Portnoy, M. (2016). *Virtualization essentials* (2nd edition). Indianapolis, IN: John Wiley and Sons.

Rouse, M. (2015). Working with a SAN. Retrieved January 27, 2017, from http://searchstorage.techtarget.com/essentialguide/LUN-storage-Working-with-a-SANs-logical-unit-numbers

x86 virtualization. (2017, January 25). In *Wikipedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=X86_virtualization&oldid=761906905

[i] Parent (Root) Partition – Manages machine-level functions such as device drivers, power management, and device hot addition/removal. The parent (or root) partition is the only partition that has direct access to physical memory and devices.

[ii] Child Partition – Partition that hosts a guest operating system - All access to physical memory and devices by a child partition is provided via the Virtual Machine Bus (VMBus) or the hypervisor.

[iii] VMBus – Channel-based communication mechanism used for inter-partition communication and device enumeration on systems with multiple active virtualized partitions. The VMBus is installed with Hyper-V Integration Services.

[iv] On-demand self-service. A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider (Mell, Grance, & others, 2011).

[v] Software as a Service (SaaS). The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user specific application configuration settings (Mell et al., 2011).

[vi] Private cloud. The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.

Community cloud. The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.

Public cloud. The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider(Mell et al., 2011).

Hybrid cloud. The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

[vii] Hyperthreading uses processor resources more efficiently, enabling multiple threads to run on each core. As a performance feature, it also increases processor throughput, improving overall performance on threaded software.