IN1020: Ukeoppgaver Operativsystemer (Uke 38)

Oppgave 1:

a) Hvorfor har vi operativsystemer?

- *Portable programmer, som fungerer på datamaskiner med forskjellig maskinvare. Det gjør at det blir enklere å forholde seg til.*
- *Abstrahere bort maskinvaredetaljer. Detaljer om hvordan lagre filer? Hvordan sende data til nettverket? Hvordan bruke grafikkort.*
- *Deling av ressurser mellom tjenester og programmer som kjører (scheduling, allokering). Operativsystemet sørger for rettferdig fordeling av tid mellom prosesser, tjenester og programmer.*
- *Sikkerhet. Programmer skal ikke kunne se og forandre hverandres minneområder.*

b) Hvor finner vi operativsystemer?

Dette er noen av stedene hvor det finnes: datamaskiner, biler, smarte enheter (IoT), konsoller, mobiler, Nesten alle digitale enheter har et operativsystem, selv om de i mange tilfeller kan være veldig enkle.

c) Hvilke to perspektiver bruker vi for hva et operativsystem er?

- *Extended machine (utvidet maskin): Gjemmer unna detaljene om maskinvare slik at den blir enklere å bruke å bruke datamaskinen*
- *Resource manager (Ressurshåndterer): Deler ressursene mellom programmene som kjører på datamaskinen.*

Oppgave 2:

Hvilke hovedkomponenter finner man i et operativsystem?

- *Filhåndtering*
- *Brukergrensesnitt (grafisk og tekst)*
- *Prosesshåndtering*
- *Enhetshåndtering (Device management)*
- *Kommunikasjon (Internt mellom programmer og over datanettverk)*
- *Minnehåndtering (allokering, aksess)*

Oppgave 3:

Hvilke mekanismer bruker et operativsystem for å ivareta sikkerhet, og sørge for at et program ikke får tilgang til data fra andre programmer?

Operativsystemet kar et klart skille mellom «user mode / brukermodus» og «kernel mode / kjernemodus».

Vanlige programmer kjører i brukermodus, og har da begrenset tilgang til prosessoren, samt kan kun se sitt eget minne. Interne komponenter i operativsystemet kjører i kjernemodus har vil da kunne kjøre alle typer instruksjoner på prosessoren og se hele minne.

Det er innebygd maskinvare i alle prosessorer som sørge for at disse modellene håndheves. Man bør derfor være veldig forsiktig med å gi programmer man ikke kjenner til tilgang til kjernemodus. (sudo på Linux eller Administrator på Windows)

Oppgave 4:

Hva skjer når vi kaller en systemkall (f.eks. read) fra et program?

1. *Argumentene (hva som skal leses, hvor mye, og hvor det skal lagres) sendes til minne*
2. *Hvilket systemkall som skal gjennomføres legges i et register.*
3. *API kaller kjernen på operativsystemet og ber om at kallet gjennomføres.*
4. *Kjernen håndterer systemkallet*
5. *Kjernen returnerer, og programmet kan kjøre videre.*

Oppgave 5:

Hvilke to typiske kjerne-arkitekturer har vi? Hva er forskjellen?

- *Monolitiske kjerner og mikrokjerner.*

- *Monotolitiske kjerner har mye funksjonalitet inkludert i kjernen. Dette gjør denne arkitekturen effektiv, på bekostning av størrelse og kompleksitet.*
- *Mikrokjerner har minimal funksjonalitet i selve kjernen. Funksjonalitet implementeres i prosesser som kjører i userspace. Bruk av funksjonalitet krever en del sending av meldinger, som gjør denne arkitekturen ineffektive. Fordeler med mikrokjerner er at de er små, modulere, utvidbare og portable.*