Linux-applikasjoner

Noen nyttige Linux-applikasjoner *

- Teksteditorer:
  - vi, emacs
- GUI “Office-pakker”:
  - LibreOffice, OpenOffice
- Tekstformatering:
  - LaTeX
- Kryptering:
  - openssl
- E-post:
  - mailx, mail, Mail, pine, emacs(!), GUI-klienter
- Nettverksprogramvare:
  - Remote access
  - Filoverføring
  - Testing og søk på nettverket

*: Dette er de samme applikasjonene som er beskrevet i kapittel 5 i læreboken. Temaer markert med grå skrift ovenfor er ikke å regne som del av pensum i kurset.

Redigering av tekst: Editorer

- Brukere *må* lære redigering – “alt” er tekst i Linux
- Redigering av tekst gjøres med en *teksteditor*:
  - For “ren” tekst *
  - Ikke for tekst*formatering* / “word processing”
- Editorer brukes bl.a. til å redigere:
  - Tekstlige data
  - Programkode skrevet i f.eks Bash, Java eller C
  - Dokumenter formatert med markup, f.eks. HTML eller LaTex
  - E-post
  - Systemfiler for administrasjon og konfigurasjon

*: Tekstfiler der innholdet typisk er tegn fra et Unicode-tegnsett

GUI- vs. terminalvindu-editorer

- Finnes enkle teksteditorer i Linux GUI:
  - Tilbyr pek-og-klikk, cut-and-paste etc.
  - Fordel: Superenkelt å lære
  - Ulemper: Tungvint, lite produktivt, musesyke
  - Eksempel: *gedit* under Gnome desktop
- Ikke-GUI editorer:
  - Kjører i terminalvinduet
  - Kommandoer fra tastaturet, (nesten) ikke bruk av mus
  - Fordeler: Effektivt og raskt
  - Ulempe: Vanskeligere å lære, ikke “vanlig” grensesnitt

Vanlige teksteditorer i Linux

- Enkle verktøy for “småjobber”:
  - pico, nano, joe m.fl.
  - Selvdokumenterende, “tar 5 minutter å lære”
- “Fullblods” editorer:
  - vi / vim og emacs
  - Kraftigere og mer fleksible
  - Enkel, grunnleggende bruk er raskt å lære
  - Tar lenger tid å lære seg å *mestre* verktøyene

vi

- vi (“visual”) – første(?) skjermorienterte editor, 1976
- vim (“vi improved”) brukes i dag, overbygg på vi
- Svært populær blant Linux-brukere*
- Skal alltid finnes som del av et Unix/Linux-system
- vi kan være en forvirrende opplevelse for nybegynnere pga. bruken av ulike *modus* (kommando- og input-modus) og den stadige switchingen mellom disse
- Det er *ikke* del av pensum i dette kurset å lære seg å bruke vi (men dere anbefales å teste det ut)

* vi-brukere liker oftest ikke emacs, og de fleste emacs-brukere holder seg langt unna vi...

emacs

- Skjermorientert editor fra 70-tallet*
- Videreutvikles fortsatt som open-source:
  - Svært mange bidragsytere
  - Et av de mest vellykkede open-source prosjektene
  - “Verdens beste program”(?) – og det er *gratis*!
- emacs er egentlig en hel *familie* med editorer:
  - Finnes i mange ulike versjoner, også med GUI
  - Er “alltid” tilgjengelig på Linux-system
  - Vanligste versjon er GNU Emacs

*: Opprinnelig skrevet av (selveste) Richard Stallman

emacs “kan alt”*

- Inneholder over 2000 innebygde kommandoer
- Brukere kan i tillegg lage egne makroer for enkelt å automatisere arbeidet med emacs
- Har et eget programmeringsspråk, Emacs Lisp, som kan brukes til å skrive nye emacs-kommandoer
- Tilbyr svært gode omgivelser for programutvikling
- Kan også brukes til f.eks. e-post, web og twitter

*: emacs har også et eget kirkesamfunn og en innebygget psykiater

Kommandoer og opplæring i emacs

- emacs opererer *ikke* i ulike input-modus, slik som vi
- Kommandoer startes med `Ctrl-` eller `Esc-`
- Grunnleggende kommandoer i emacs er de samme som for redigering av kommandolinjen i bash
- For å bli en oppegående emacs-bruker:
  - Gå gjennom emacs sin egen tutorial: `Ctrl-h t`
  - Test eksemplene i avsnitt 5.2.2 i læreboken
- Å lære seg å bruke emacs *er* del av pensum i kurset

“Office-pakker” i Linux *

- To vanlige programvarepakker:
  - LibreOffice
  - OpenOffice
- Begge er Linux-alternativer til Microsoft Office
- Grensesnitt og utseende ligner MS Office
- Gratis programvare
- Open-source, open-format (og minst like bra!)
- M$-kompatibelt, med tidsforsinkelse

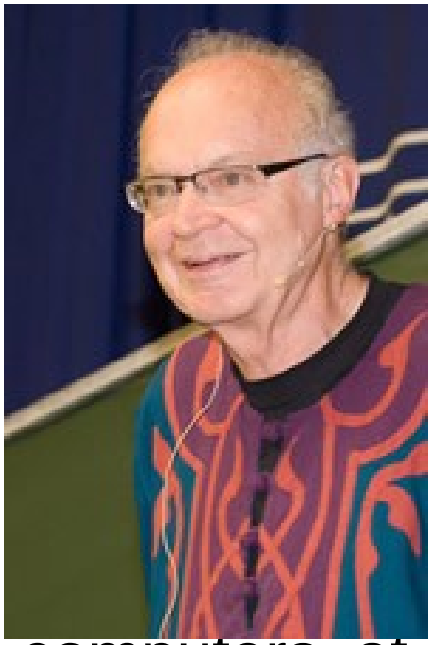
*: “Productivity software” i læreboken

Programvare i LibreOffice og OpenOffice

Begge office-pakkene i Linux innholder mye av den samme programvaren:

- *Writer* Word processor (“MS Word”)
- *Calc* Spreadsheet (“MS Excel”)
- *Impress* Presentation graphics (“MS PowerPoint”)
- *Base* Database management system (“MS Access”)
- *Math* Mathematical package program
- *Draw* Drawing software

Litt IT-historie: Donald Knuth og TeX



- Donald Knuth (1938 - , Stanford):
  - Informatikk-pionér: "Father of the analysis of algorithms”
  - Ville fått Nobel-prisen i informatikk...
  - Lagde TeX-systemet på 70-tallet, for å produsere/sette høykvalitets trykkeklare dokumenter med matematisk innhold
- “TeX should give exactly the same results on all computers, at any point in time”
- “The most sophisticated digital typographical system in the world”
- TeX (og Metafont) alltid tilgjengelig på Linux-systemer

TeX-kode

- Dokumentene skrives som ren tekst, med “markup”/ formateringskoder inne i teksten
- Ligner på HTML, men er fokusert mer på typografi og utseende av dokumentet, mindre på struktur og metadata
- Eksempel:

Markup	Renders as
`The quadratic formula is $$-b \pm \sqrt{b^2 - 4ac} \over 2a$$ \bye`	The quadratic formula is $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

TeX og WYSIWYG

- WYSIWYG : What You See Is What You Get
  - Tekstbehandling der det du ser på skjermen ser likt ut som det som skrives ut på printer
  - F.eks. MS-Word er WYSIWYG
- TeX er ikke WYSIWYG:
  - Fil med tekst og TeX-koder redigeres i en editor
  - Filens leses av TeX-programmet, som produserer en DVI-fil (device independent) med “trykksats”
  - DVI-filen kan konverteres til PDF, Word, HTML etc., for utskrift eller publisering på nett

LaTex: Et overbygg på TeX

- TeX er relativt komplisert og svært detaljert
- **LaTex** (Leslie Lamport, 1985) er en forenkling av TeX som fokuserer mindre på typografiske spissfindigheter
- LaTex har blitt en internasjonal standard for skriving av vitenskapelige og tekniske artikler og bøker

Hvorfor lære seg LaTex?

- Svært utbredt i utdannings- og forskningsmiljø, spesielt innen tekniske og matematiske fag
- Brukes også mye ved IT-studiene på HiØ, både av ansatte og studenter, f.eks. til å skrive bachelor- og masteroppgaver
- Forenkler og effektiviserer skriveprosessen *mye*
- Ultraportabelt, konverteres enkelt til “hva som helst”
- Har en svært god bibliografi- og referansemekanisme
- Se f.eks.: “LaTeX for nybegynnere” (Dag Langmyhr, UiO)

Xelcgrevat

- Xelcgrevat:
  - Genafsbezrer qngn fyvx ng qr vxxr yratre re yrfoner bt xna sbefgåfle
- Qrxelcgrevat:
  - Genafsbezrer xelcgregr qngn gvyonxr gvy bevtvany, yrfone sbez
- Uibesbe xelcgrer qngn?
  - Fvxxreurg sbe frafvgvir qngn cå arggrg (xerqvggxbeg, uryfrvasb.)
  - Føetr sbe ng ceving bt crefbayvt vasbeznfwba bt xbzzhavxnfwba vxxr xna yrfrf ni bireiåxrer bt xevzvaryyr

Kryptering *

- Kryptering:
  - Transformere data slik at de ikke lenger er lesbare og kan forstås
- Dekryptering:
  - Transformere krypterte data tilbake til original, lesbar form
- Hvorfor kryptere data?
  - Sikkerhet for sensitive data på nettet (kredittkort, helseinfo.)
  - Sørge for at privat og personlig informasjon og kommunikasjon ikke kan leses av overvåkere og kriminelle

*: Teksten på forrige lysark er kryptert med en svært enkel algoritme, rot-13 aka Cæsar-koding

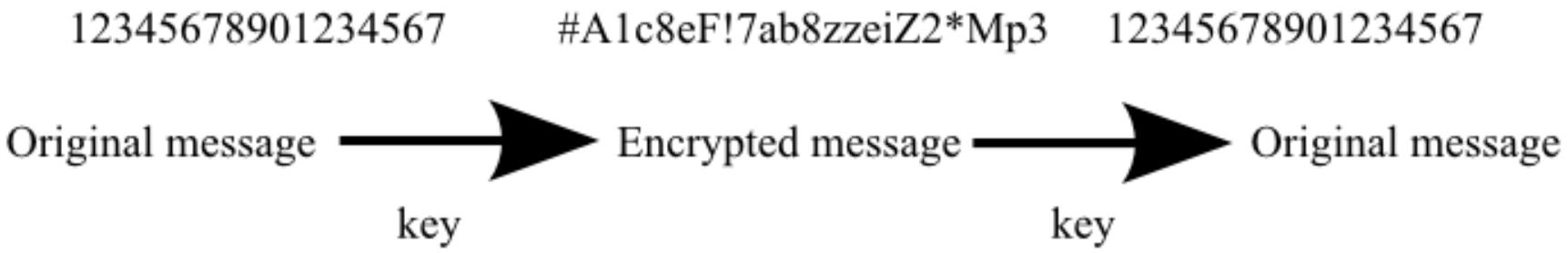
Sikker kryptering

- Kryptering er verdiløs hvis koden enkelt kan knekkes:
  - Effektive, gratis kodeknekkere lett tilgjengelig på nettet
- Standard krypteringsalgoritmer* er regnet som sikre:
  - Basert på permutering (omstokking) av mange elementer i en eller to *kodenøkler* (encryption / decryption key)
  - Sikre kodenøkler har typisk lengde på 1024 bits – antallet mulige permutasjoner blir “ufattelig “ stort
  - Tar “uendelig” lang tid å skulle knekke koden
  - Men: Sikkerheten er *helt avhengig* av at kodenøkkelen *ikke* er tilgjengelig for uvedkommende

*: Utvikling av krypteringsalgoritmer er et eget fagområde innen matematikk/informatikk

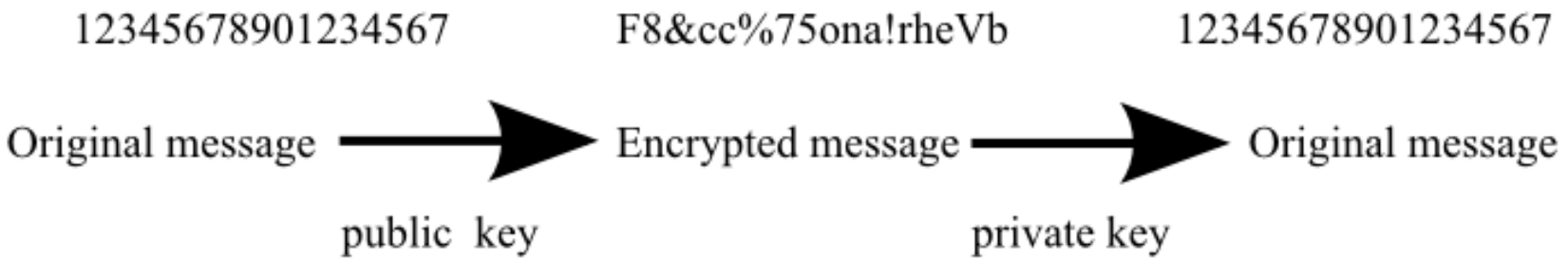
Symmetrisk kryptering

- Aka “private key encryption”
- Bruker en og samme kodenøkkel for kryptering og dekryptering
- Kodenøkkelen er en “private key” – skal *ikke* publiseres
- Brukes typisk til kryptering av egne data på disk, eller til å sende data til betrodde personer som kan få nøkkelen



Asymmetrisk kryptering

- Aka “public key encryption”
- To kodenøkler, en “private key” og en “public key”
  - Private key brukes til å generere public key
  - Public key brukes til å kode dataene
  - Private key brukes til å dekode dataene, må holdes skjult
  - Public key kan publiseres til alle uten sikkerhetsrisiko



Bruk av asymmetrisk kryptering

- Typisk for sikker betalingsformidling:
  - Mottaker av betaling sender public key til avsender
  - Betalingsdata sendes kryptert over nettet
  - Bare mottager har private key for dekryptering
- Er innebygd i sikre nettelesere:
  - Laster automatisk ned public keys ved behov
  - Lagres som “digitale sertifikater” hos bruker
- Merk:
  - Sikkerhet avhengig av at private key beskyttes

Kryptering i Linux: `openssl`

- Implementerer sikre protokoller for kommunikasjon
- Krypterer og dekrypterer data
- Lager public og private kodenøkler
- Lager digitale sertifikater og signaturer for sikre nettsteder
- Stort utvalg kodenøkkel- og krypteringsalgoritmer
- For hjelp og dokumentasjon:

```
openssl help
man openssl
man openssl-kommando
```

Symmetrisk kryptering av en fil med `openssl`

```
openssl enc -cipher –in file1 –out file2
```

cipher	Krypteringsalgoritmen som skal brukes
file1	Filen som skal krypteres
file2	Kryptert fil

- For å se alle tilgjengelige krypteringsalgoritmer:

  ```
  openssl enc -ciphers
  ```

- `openssl` vil be om et passord før kryptering
- Passordet og valgt krypteringsalgoritme bestemmer hvilken key som brukes til krypteringen

Symmetrisk dekryptering av en fil med `openssl`

`openssl enc -d -`*`cipher`* `–in` *`file1`* `–out` *`file2`*

`cipher`	Krypteringsalgoritmen som brukes
`file1`	Filen som skal dekrypteres
`file2`	Dekryptert fil

- `openssl` vil be om et passord før dekryptering, dette er det samme passordet som brukes for kryptering
- For mer om symmetrisk kryptering med `openssl`:

  man openssl-enc

Assymmetrisk kryptering med `openssl`: Generering av private key

- Eksempel som bruker kryptosystemet RSA:

```
openssl genrsa -out priv-key.pem 2048
```

- Lager en privat kodenøkkel med 2048 bits lengde
- Lagrer nøkkelen i filen `priv-key.pem`*
- Mange ulike måter å lage private keys på, se:

```
man openssl-genpkey
man openssl-genrsa
```

* PEM: Privacy-Enhanced Mail filformat

Assymmetrisk kryptering med `openssl`: Generering av public key

- Public key lages på grunnlag av private key
- Eksempel som bruker RSA:

```
openssl rsa -pubout -in priv-key.pem \
-out pub-key.pem
```

- Filen `priv-key.pem` inneholder privat kodenøkkel
- Den nye public-nøkkelen lagres i filen `pub-key.pem`
- Mer om håndtering og generering av public keys her:

```
man openssl-pkey
man openssl-rsa
```

Typisk bruk av assymetrisk krypto. med openssl

- Kari lager sin egen private key:

```
openssl genrsa -out priv-key.pem 2048
```

- Kari lager en public key og sender denne til Per:

```
openssl rsa -pubout -in priv-key.pem \
-out pub-key.pem
```

- Per bruker Karis public key til å kryptere en fil, og sender deretter den krypterte filen til Kari:

```
openssl rsautl -encrypt -in pers_fil.txt \
-out pers_fil.enc -inkey pub-key.pem
```

- Kari bruker sin egen private nøkkel til å dekryptere filen fra Per:

```
openssl rsautl -decrypt -in pers_fil.enc \
-inkey priv_key.pem
```

Mer om `openssl` og andre krypteringsverktøy

- Det er *mange* andre måter å bruke `openssl` på enn det som er vist i de enkle eksemplene her
- `openssl` kan bl.a. generere sertifikater for bruk i sikre nettsteder, se avsnitt 5.5.2 i læreboken
- `openssl` er egentlig bare et “overbygg”, som bruker innebygde Linux-programmer for kryptering
- Mange andre programmer i Linux for kryptering, f.eks. Gnu Privacy Guard (`gpg`)

Nettverk og hardware

- Alle datamaskiner på internett er koblet sammen i et fysisk nettverk
- Egen nettverks-hardware brukes for å håndtere kommunikasjon mellom maskinene:
  - Hubs og switches håndterer trafikken i lokale nett
  - Rutere kommuniserer mellom ulike nettverk
  - Gateways er spesial-routere som “oversetter” mellom nettverk med ulike protokoller for kommunikasjon

Nettverkskommandoer i Linux

- Linux har en rekke kommandoer og programmer for å sende og hente data i nettverk, som tilbyr abstraksjon *vekk* fra hardwaren som utgjør det fysiske nettet
- All kommunikasjon på internettet bruker protokollen TCP/IP * som “grunnleggende språk”
- Alle nettverkskommandoene er basert på at maskiner i nettet kan adresseres med *IP-adresser* og/eller *navn* på servere/maskiner.

*: Transmission Control Protocol/Internet Protocol

Internet Protocol (IP): Adresser

- Alle maskiner på nettet har en unik IP-adresse som identifiserer maskinen *
- Adresser i IPv4 (versjon 4):
  - 32 bits tall, 4 x 8-bits, skilt med punktum:

    00001010.00001011.11011001.001100111
  - Skrives oftest som fire desimale tall mellom 0 og 255:

    10.11.233.55
- Adresser i IPv6 består av 128 bits, skrives oftest som 32 heksadesimale siffer

*: For å se egen IP-adresse, bruk kommandoene `ip` eller ifconfig

The Domain Name System: DNS

- Numeriske IP-addresser er vanskelige å huske
- Bruker i stedet IP-*aliaser*
- Nettet “deles opp” i logiske områder – domener
- Et IP-alias identifiserer en maskin med hostnavn, subdomene(r) og topp-domene
- Eksempel:

  `theband.hiof.no` → 158.39.165.8

DNS-servere

- Håndtering av aliaser gjøres av egne DNS-servere
- Er ordnet i et hierarki, med lokale servere nederst
- En DNS server som får et ukjent alias (name request) sender dette videre “oppover”, rekursivt i hierarkiet av servere
- Hvis aliset går helt til høyeste nivå i hierarkiet uten å kunne gjenkjennes, genererer dette en feilmelding (ukjent host)

Nettverks-protokoller *

- En beskrivelse av hvorledes to maskiner og/eller programmer skal kommunisere
- TCP/IP inneholder underprotokoller for ulike formål, bl.a.:
  - TCP, UDP — Håndtering av “pakker” med data
  - HTTP, FTP — Overføring av filer
  - HTTPS, SFTP — Sikker overføring av filer
  - SSL — Sikker kommunikasjon mellom maskiner
  - TELNET — Usikker maskin-kommunikasjon
  - DNS — IP alias to address mapping

*: Mer stoff om nettverksprotokoller og DNS i Linux kommer senere i kurset

Noen nettverksprogrammer i Linux

- `telnet`, `ssh`, `rsh`

  Pålogging/styring av en remote host

- `ftp`, `sftp`, `wget`

  Programmer for filoverføring

- `ping`, `traceroute`

  Tester konnektivitet, ytelse og nåbarhet i nettet

- `nslookup`, `dig`, `host`

  Sender DNS requests for å knytte et alias til en IP-adresse

Pålogging/styring av remote host *

- `telnet` – for å logge på en annen maskin
  - Åpen, ukryptert kommunikasjon – også passord!
  - Ofte stengt av pga. sikkerhetsproblemer
- `ssh` – secure shell
  - “Sikker telnet”, public key kryptering av datatrafikken
- `rsh` – remote shell
  - For å utføre shell-kommandoer på annen maskin
  - Er usikker, krypterer ikke data, oftest erstattet av `ssh`

*: “remote host” – en maskin som befinner seg et annet sted på nettet

Overføring av filer mellom maskiner

- `ftp` – file transfer protocol
  - Klient kontakter ftp server for å laste opp og ned filer
  - Gammelt program, relativt tungvint og lite fleksibelt
  - Kommunikasjonen går åpent (også passord!)
  - Er ofte stengt av pga. sikkerhetsproblemer
- `sftp` – secure file transfer
  - Bruker `ssh` til å kryptere all datatrafikk
  - Fleksibelt og enkelt å overføre mange filer
  - Enkelt å styre OS'et på remote host fra `sftp`

`wget` – nedlasting fra webservere

- Kommandosyntaks:

```
wget URL
```

- Vanlige opsjoner:

  `-A` Tillater jokernotasjon for å hente flere filer

  `-r` Recursive get – laster ned filen og alle filer som denne filen lenker til (web crawler)

- Støtter både sikker (https) og usikker (http) kommunikasjon

Kommandoer for å teste nettet

- `ping`
  - Sender pakker med data til en remote host
  - Hvis host kan nås, vises overføringstider for data
  - Har mange opsjoner, kan også brukes til “angrep” på servere (“ping of death”, “ping flood”)
- `traceroute`
  - Virker som `ping`, men viser også *ruten* til en host
- Mange andre “inspeksjonskommandoer”:
  - `ip`, `ss`, `netstat`, `route`, `ifconfig` ...

Kommandoer som bruker DNS

- `host`
  - Viser DNS informasjonen som finnes for en IP-alias eller en IP-addresse
- `dig`
  - Gjør omtrent det samme som `host`
- `nslookup`
  - Utfører en IP-alias → IP-adresse DNS request