Plattformsteknik

- Abstraktionslager
- Databuttar (S3-lagring)
- Introduktion till Linux
- Lösenordsguiden
- Telemetri
- Öppen källkod

Abstraktionslager

1. Vad är ett abstraktionslager?
2. Hur har det påverkat tillgängligheten för öppen källkod?
3. Problemet med att motarbeta utvecklingen

1. Vad är ett abstraktionslager?

Ett abstraktionslager är ett tekniskt skikt som döljer underliggande komplexitet och detaljer för användaren eller administratören. Istället för att tvingas hantera rå kod, manuell serverkonfiguration eller invecklade terminalkommandon, interagerar användaren med ett förenklat och standardiserat gränssnitt.

I praktiken innebär detta att ett lager (till exempel en plattform som Pikapods eller ett verktyg som Docker) hanterar den bakomliggande tekniska strukturen, medan slutanvändaren bara ser kontrollpanelen och applikationen.

2. Hur har det påverkat tillgängligheten för öppen källkod?

Abstraktionslager har varit helt avgörande för att demokratisera och sprida öppen källkod (open source) utanför en snäv krets av datatekniker. Effekterna är framför allt:

- **Sänkta trösklar:** Tidigare krävdes djup systemförståelse i Linux för att ens installera och drifta en enkel tjänst. Med moderna abstraktionslager kan vem som helst starta en säker applikation med några få klick.
- **Fokus på slutanvändarnytta:** När teknisk administration automatiseras flyttas fokus till vad applikationen faktiskt gör för användaren. Det gör open source till ett reellt och användarvänligt alternativ till kommersiella stängda plattformar.
- **Ökad portabilitet och säkerhet:** Genom att paketera tjänster i standardiserade behållare (containrar) blir driften isolerad, stabil och enkel att flytta mellan olika miljöer utan risk för systemkonflikter.

3. Problemet med att motarbeta utvecklingen

Inom vissa delar av Linux-gemenskapen finns en tendens att motarbeta denna typ av förenklingar, ofta med argumentet att administratörer "bör" förstå och konfigurera allt manuellt från grunden. Detta motstånd skapar flera påtagliga problem:

- **Exkludering och elitism:** Att kräva manuell terminalhantering stänger ute vanliga användare och mindre organisationer från att använda fri programvara. Det motverkar rörelsens grundidé om digital frihet för alla.
- **Hämmad tillväxt för öppen källkod:** Om tröskeln för att använda öppna alternativ förblir för hög, tvingas slutanvändare istället in i armarna på proprietära techjättar som erbjuder färdiga, proprietära lösningar.
- **Ineffektiv tidsanvändning:** Att spendera timmar på att manuellt konfigurera grundläggande infrastruktur som kan automatiseras tar resurser från att faktiskt utveckla, förbättra och använda tjänsterna. Abstraktion är inte lathet – det är effektivisering.

> “ Abstraktion är inte att göra saker luddiga, utan att skapa en ny exakt nivå där man kan vara absolut effektiv.
>
> *Fritt efter* Edsger W. Dijkstra (pionjär inom datavetenskap)

Databuttar (S3-lagring)

S3-lagring (Simple Storage Service) är en objektbaserad molnlagringstjänst som lagrar data som objekt (filer + metadata) i "buckets" (mappar). Den är skalbar, billig och tillgänglig via API:er, och används för att lagra allt från backups till mediafiler och dokument.

Fördelar med S3-lagring

 Skalbarhet: Lagra petabyte-data utan problem.
 Tillgänglighet: 99,99% upptid (data är alltid tillgängligt).
 Billigt: Låg kostnad per GB. Betalning sker för den använda mängden data, inte fast datamängden.
 Flexibelt: Är enkelt att skapa och använda flera olika för olika ändamål..
 Hållbart: Versionshantering och kryptering ingår (kan vara tillval).
 Molnbaserat: Åtkomst från var som helst (via API eller klienter som `rclone`).

Viktiga funktioner för att använda databuttar rätt

1. Kryptering

- **I vila**: **AES-256** (alla filer krypteras innan de lagras).

- **Under överföring**: **TLS** (säker transport).
- **Nyckelhantering**: Använd **egna nycklar** (t.ex. via `rclone crypt`) för extra säkerhet.

2. Versionshantering

- **Automatiska versioner**: Spara **gamla versioner** av filer (återställ om något går fel).

3. Datalås (Object Lock)

- **Oföränderlig lagring**: Lås filer så de **inte kan raderas eller ändras** under en tid (t.ex. för compliance).
- **Används för**: **Känsliga dokument**, backups, eller lagring som måste bevaras.

4. Åtkomstkontroll (IAM/ACL)

- **Behörigheter**: Ställ in **vem som får läsa/skriva** (t.ex. via `rclone` eller S3-policyer).
- **Principen om minsta rättigheter**: Ge **endast den åtkomst som behövs**.

5. Montering och synkronisering

- **Monteras till servertjänster:** Monteras enkelt till tjänster som södjer S3-lagring.
  Montera som en mapp: Använd `rclone mount` eller `s3fs` för att komma åt filer som en lokal mapp.

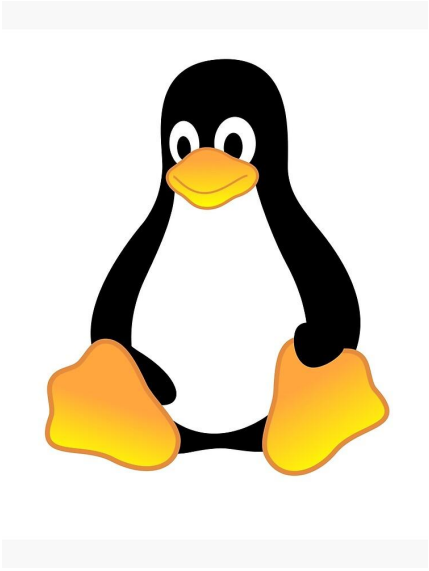
Kontakta Björknets support för mer information.

> Data är den nya oljan. Det är värdefullt, men om det inte är raffinerat och säkert lagrat kan det inte användas. Det måste omvandlas till något strukturerat för att ha ett verkligt värde.
>
> Clive Humby

Björknet - Infrastruktur på mänskliga villkor

"*Humanism i digital gestaltning.*"

Introduktion till Linux



1. Vad är Linux?
2. Linux vs. Windows vs. macOS: Varför specialisering vinner
3. Integritet och spårning: Du äger din data
4. Nyckelfaktorer vid val av Linux-distribution
5. Användargränssnitt: Skrivbordsmiljön bestämmer utseendet
6. Pakethanterare: Hur program installeras och körs
7. Uppdateringsmodell: Stabil, rullande eller oföränderlig
8. Skrivbordsmiljö (Desktop Environment)
9. Community och dokumentation
10. Företag i ryggen vs. Community-drivet
11. Val av distribution
12. Lycka till med Linux!

1. Vad är Linux?

Linux skapades ursprungligen av Linus Torvalds år 1991. En vanlig missuppfattning är att Linux är ett komplett operativsystem, men i själva verket är det bara en **operativsystemskärna** (kernel).

Kärnan är den innersta delen av mjukvaran som gör att datorns hårdvara kan prata med dina program. Eftersom Linux är fritt och har öppen källkod kan vem som helst ta denna kärna och bygga ett eget operativsystem runt den. Ett sådant komplett operativsystem kallas för en **distribution** (eller "distro"). Linux är uppbyggt på en logisk och modulär princip: **allt är separata filer**. Det betyder att systemet inte är en enda stor, låst klump, utan består av utbytbara byggstenar som samarbetar. Vilken är just det som är så bra att den kan användas efter behov.

Ett Linux-system består av tre huvuddelar

För att kärnan ska bli ett fungerande operativsystem som du kan använda på en vanlig dator krävs tre komponenter:

1. Kärnan (Kernel)

Själva Linux-kärnan. Det är motorn under huven som hanterar processorn, minnet och hårddiskarna.

2. Pakethanteraren

Det system som installerar, uppdaterar och tar bort programvara. Istället för att ladda ner lösa filer från webbsidor sköter pakethanteraren allt centralt och säkert.
Traditionella format: apt (använder .deb-filer, vanliga i Debian och Mint) eller rpm (i Fedora och openSUSE).
Moderna/Fristående format: Flatpak, Snap eller Nix. Dessa innehåller allt programmet behöver i ett och samma paket, vilket minskar risken för systemkonflikter.

3. Användargränssnitt & Fönsterhanterare

Det grafiska lagret som gör att du kan se och interagera med systemet.
Fönsterhanteraren (Window Manager): Ansvarar specifikt för att rita upp, flytta och ändra storlek på programmets fönster.
Skrivbordsmiljön (Desktop Environment): Det kompletta gränssnittet som inkluderar fönsterhanteraren samt paneler, menyer, ikoner och grundläggande verktyg (till exempel GNOME, KDE Plasma eller Xfce).

2. Linux vs. Windows vs. macOS: Varför specialisering vinner

Windows och macOS är byggda som generella operativsystem som ska försöka passa alla samtidigt. Det gör dem tunga och oflexibla. Linux är istället helt modulariserat. Genom att kombinera rätt kärna, rätt gränssnitt och rätt verktyg skapar man ett system som är extremt specialiserat för sin uppgift. De låser in dig vid deras egna pakethanterare.

Systemiska brister i Windows

Där Linux bygger på en ren, modulär arkitektur dras Windows med omfattande arkitektoniska problem:

- **Inkonsekvent kodbas och teknisk skuld:** Windows är i grunden ett lapptäcke av gammal och ny kod för att bibehålla bakåtkompatibilitet. Detta leder till en fragmenterad struktur där inställningar är utspridda mellan moderna gränssnitt och kvarlevor från äldre versioner. Denna komplexitet gör systemet svåröverskådligt och skapar svårlösta buggar.
- **Resurshantering och nätverksinstabilitet:** Systemets nätverksstack och drivrutinshantering lider ofta av ineffektivitet, vilket kan leda till att nätverkskort överbelastas eller slutar svara under hög belastning.
- **Integritetsaspekter och telemetri:** Moderna versioner av Windows integrerar omfattande telemetri och datainsamling i kärnsystemet. Funktioner som marknadsförs som användarstöd fungerar i praktiken som övervakning (spyware), vilket kompromet terar användarens digitala integritet.
- **Monetarisering och inlåsning:** Microsoft flyttar allt fler basfunktioner mot prenumerationsmodeller och molnberoende tilläggstjänster, vilket gör operativsystemet dyrt och beroende av en konstant internetuppkoppling mot deras servrar.

Begränsningar i macOS

macOS erbjuder en högre grad av stabilitet än Windows, men dras med andra fundamentala begränsningar:

- **Ekonomisk och hårdvarumässig inlåsning:** Systemet är strikt bundet till Apples egen, högt prissatta hårdvara. Priserna för både produkter, uppgraderingar och tillhörande molntjänster är abnormt höga i förhållande till prestandan.
- **Snävt och kontrollerat ekosystem:** Användaren tvingas anpassa sig till Apples strikta design- och funktionsval. Det finns mycket litet utrymme för egen konfiguration, och utbudet av programvara utanför Apples officiella kanaler är begränsat på grund av systemets slutna natur.

Hur Linux eliminerar de stängda systemens problem

Linux löser dessa grundläggande arkitekturbrister genom sin unika konstruktion:

- **Ingen teknisk skuld – Sanering av kodbasen:** Till skillnad från Windows lapptäcke av kod, rensas Linux-kärnan kontinuerligt. Gammal, oanvänd kod och föråldrade drivrutiner fasas ut i stället för lagras på hög. Eftersom källkoden är öppen kan tusentals utvecklare världen över identifiera, städa och optimera strukturen, vilket ger ett strömlinjeformat system utan dolda fel.
- **Effektiv resurshantering och nätverksstabilitet:** Linux har en extremt robust nätverksstack utvecklad för världens mest krävande servermiljöer. Systemet hanterar hög nätverksbelastning, minnesallokering och processorkraft med minimal overhead, vilket eliminerar risken för att hårdvara som nätverkskort överbelastas under tung drift.
- **Total integritet utan dold telemetri:** I ett Linux-system existerar ingen påtvingad datainsamling eller inbyggda spionprogram under täckmantel av "funktioner". Användaren har fullständig kontroll över varje process som körs, och ingen data skickas till externa aktörer utan explicit konfiguration.
- **Frihet från inlåsning och licenskostnader:** Linux är fritt från kommersiella prenumerationsmodeller och dyra tilläggstjänster. Distributioner och tillhörande mjukvara är tillgängliga utan kostnad och utan dolda avgifter. Dessutom är systemet helt hårdvaruoberoende – det kan köras på allt från billiga enkortsdatorer till kraftfulla servrar, vilket eliminerar den typ av ekonomiska inlåsningseffekter som präglar macOS.

3. Integritet och spårning: Du äger din data

Moderna versioner av Windows och macOS innehåller omfattande inbyggd telemetri (telemetri är automatisk datainsamling som skickas tillbaka till tillverkaren). Systemen spårar användarmönster, sökningar och hårdvarudata.

Eftersom merparten av Linux-distributionerna drivs som open source-projekt utan vinstintresse saknas denna form av spårning helt. Ingen data samlas in i dolt syfte, inga riktade annonser trycks in i menyerna, och du har fullständig kontroll över vad din dator kommunicerar mot internet.

4. Nyckelfaktorer vid val av Linux-distribution

Att välja en distribution handlar inte om att hitta det "bästa" systemet, utan om att matcha systemets arkitektur med dina specifika behov. Valet kokas ner till fem kritiska faktorer:

1. Användningsområde och hårdvara

Operativsystemet måste matcha datorns primära syfte och de fysiska komponenterna.

Kontor och vardag: Kräver hög tillförlitlighet och minimal konfiguration. Fokus ligger på webbläsare, kontorsprogram och mediakonsumtion.

Spel: Kräver distributioner med färska grafikdrivrutiner (särskilt för Nvidia) och optimerade kärnor (t.ex. Pop!_OS eller Bazzite).

Äldre hårdvara: Kräver resurssnåla distributioner med lätta skrivbordsmiljöer (som Xfce eller MATE) som kan ge nytt liv åt datorer med begränsat arbetsminne och svaga processorer.

Det finns en uppsjö av specialiserade linuxdistrubtioner att utforska.

5. Användargränssnitt: Skrivbordsmiljön bestämmer utseendet

I Windows och macOS är det grafiska gränssnittet fastlåst. I Linux är gränssnittet (skrivbordsmiljön) bara ett utbytbart programlager ovanpå kärnan.

- **Traditionella gränssnitt (t.ex. Cinnamon, Xfce):** Påminner om Windows med en startmeny och aktivitetsfält. Mycket resurssnåla och logiska.
- **Moderna gränssnitt (t.ex. GNOME, KDE Plasma):** Erbjuder avancerade animationer, flexibel fönsterhantering eller extrem anpassningsbarhet för den som vill skräddarsy allt.

6. Pakethanterare: Hur program installeras och körs

Sättet mjukvara installeras på skiljer sig fundamentalt i Linux jämfört med andra system. Det finns fyra huvudsakliga spår:

- **Traditionella pakethanterare (t.ex. APT, RPM):** Installerar program som systempaket. De delar resurser och bibliotek med själva operativsystemet. Det är effektivt men kan ibland skapa konflikter vid uppdateringar.
- **Flatpak (Flathub):** Det viktigaste formatet för vanliga skrivbordsprogram idag. Varje program körs isolerat (sandlåda) med sina egna bibliotek. Det gör systemet stabilt och säkert eftersom applikationer inte kan krascha operativsystemet.

- **Snap:** Canonicals eget format. Bör undvikas på vanliga skrivbordsdatorer på grund av långsamma starttider och stängd backend-infrastruktur. Kan ha ett syfte i temporära miljöer eller specifika serverapplikationer, men är sällan optimalt för hemanvändaren.
- **Nix-pakethanteraren:** Representerar framtiden för terminalprogram och utvecklingsmiljöer. Den tillåter isolerade, reproducerbara installationer av kommandoradsverktyg som inte påverkar resten av systemet.

7. Uppdateringsmodell: Stabil, rullande eller oföränderlig

Hur och när systemets komponenter uppdateras avgör både stabilitet och tillgång till ny mjukvara.

- **Punktutgåvor (Standard/Stable)**: Systemet uppdateras i fasta cykler (t.ex. var sjätte månad eller vartannat år). Programvaran fryses i stabila versioner, och endast säkerhetsuppdateringar skjuts ut. Detta ger maximal förutsägbarhet (t.ex. Linux Mint).
- **Rullande utgåvor (Rolling Release)**: Inga fasta versionsnummer existerar. Systemet uppdateras kontinuerligt så fort utvecklarna släpper ny kod. Du har alltid de senaste funktionerna och drivrutinerna, men risken för tillfälliga buggar ökar (t.ex. Arch Linux eller openSUSE Tumbleweed).
- **Oföränderliga system (Immutable)**: Operativsystemets kärna och systemfiler är helt skrivskyddade under drift. Uppdateringar sker atomiskt (allt eller inget) i bakgrunden. Om en uppdatering mot förmodan orsakar fel, startar du bara om datorn och backar till föregående fungerande tillstånd. Applikationer körs helt isolerade via Flatpak (t.ex. Fedora Silverblue).

8. Skrivbordsmiljö (Desktop Environment)

Eftersom gränssnittet är modulärt i Linux måste du välja hur du vill interagera med datorn. Detta påverkar både arbetsflöde och systemresurser.

- **Traditionellt (t.ex. Cinnamon)**: Efterliknar det klassiska Windows-upplägget. Logiskt, funktionellt och kräver ingen inlärningströskel.
- **Modernt/Innovativt (t.ex. GNOME)**: Ett mer tangentbordsfokuserat och minimalistiskt gränssnitt som påminner om mobila operativsystem eller macOS, med fokus på virtuella skrivbord och rena ytor.
- **Höggradigt anpassningsbart (t.ex. KDE Plasma)**: Tillåter användaren att förändra precis allt i det visuella gränssnittet utan att behöva programmera.

- **Lättviktiga gränssnitt (t.ex. Xfce, MATE):** Avskalade gränssnitt som har valts bort tunga visuella effekter och animationer för att spara resurser. De drar minimalt med ström och arbetsminne, vilket gör dem perfekta för att ge nytt liv åt äldre datorer eller maximera batteritiden på bärbara datorer.

9. Community och dokumentation

När problem uppstår är distributionens användarbas din primära support.

- Stora, etablerade distributioner har omfattande wikis, aktiva forum och miljontals användare. Om du stöter på ett fel är sannolikheten närmast 100 % att någon annan redan har löst det och skrivit om det på internet.
- Nischade eller mindre distributioner kan erbjuda intressanta funktioner, men lämnar dig ofta ensam med terminalen om något slutar fungera.

10. Företag i ryggen vs. Community-drivet

Vem som styr distributionen påverkar dess framtid, ideologi och hur systemet utvecklas.

- Företagsägda (t.ex. Ubuntu/Canonical, Fedora/Red Hat): Har stora finansiella resurser, vilket ofta innebär mycket stabil hårdvarukompatibilitet och snabb utveckling. Nackdelen är att kommersiella intressen kan gå före användarens (som Ubuntus push för Snap-paket eller inslag av reklam för tilläggstjänster).
- Community-drivna (t.ex. Linux Mint, Debian): Styrs av volontärer och donationer. De fattar beslut helt baserat på vad som är bäst för användarna (som när Mint valde bort Snap). Risken är istället att utvecklingen kan gå långsammare om resurserna tryter.

11. Val av distribution

Det finns hundratals Linux-distributioner, men för en nybörjare kokas de bästa valen ner till dessa fem. De är stabila, har stora användarbaser och löser specifika uppgifter extremt väl.

Det bör noteras att de flesta distributioner som är baserade på linux är ofta små och är instabila hobbyprojekt. Men väljer du någon av dessa så har du inte de problemen.

Edubuntu

- **Profil:** För yngre barn och utbildning.
- **Format:** Traditionella systempaket och Snap. Flathub/flatpak kan installeras manuellt.
- **Varför välja den?** Detta är en specialanpassad version fylld med pedagogiska program och spel för barn i alla åldrar. Även om systemet använder Snap-paket (vilket ofta kritiseras på vanliga skrivbordsdatorer), har det en klar fördel här: det gör installationen av tunga, utbildningsrelaterade programvaror helt problemfri för föräldrar och lärare. Det bara fungerar, vilket är precis vad som krävs i en barnmiljö.

Fedora Workstation/Atomic

- **Profil:** För dig som vill ha det senaste och mest moderna.
- **Format:** RPM-paket och Flatpak.
- **Varför välja den?** Fedora ligger i teknikens framkant och använder de senaste versionerna av mjukvara och gränssnitt (GNOME). Det är ett rent, snyggt och innovativt system som är mycket populärt bland utvecklare. Det kräver lite mer modern hårdvara men erbjuder en av de bästa och mest opåverkade Linux-upplevelserna som finns. Finns som traditionellt system och oföränderligt.

Linux Mint

- **Profil:** Det säkra kortet för nybörjaren.
- **Format:** Traditionella systempaket (.deb) och Flatpak (standard).
- **Varför välja den?** Om du vill ha ett system som känns bekant om du kommer från Windows. Det har ett klassiskt utseende med startmeny, är extremt stabilt och fungerar direkt efter installationen utan att du behöver konfigurera något. Det har medvetet rensat bort allt som rör Snap-paket till förmån för Flatpak.

openSUSE Leap/Tumbleweed/MicroOS

- **Profil:** För dig som vill ha full grafisk kontroll över systemet.
- **Format:** RPM-paket och Flatpak.
- **Varför välja den?** openSUSE är en professionell och stabil distribution som har en unik fördel: kontrollpanelen YaST. Via YaST kan du hantera avancerade systeminställningar, brandväggar, användare och hårdvara helt grafiskt. Det är det perfekta valet om du vill ha djup kontroll över ditt system utan att behöva skriva kommandon i terminalen. Finns tre utgåvor stabill traditionell, rullande traditionell, rullande oföränderligt. Det är marknadens mest stabila rullande linuxdistribution.

Pop!_OS

- **Profil:** För spel, skapande och produktivitet.
- **Format:** Traditionella systempaket (.deb) och Flatpak.
- **Varför välja den?** Om du har ett Nvidia-grafikkort eller vill spela spel på datorn. Pop!_OS erbjuder en separat installationsfil där de stängda drivrutinerna för Nvidia är inbakade

från start. Det har också ett modernt gränssnitt med smart, automatisk fönsterhantering (tiling) som gör det mycket effektivt att arbeta i.

Distributioner som är stabila – men som inte rekommenderas för de flesta

Det finns flera andra mycket stora, stabila och funktionella distributioner som ofta diskuteras i Linux-sammanhang. Många erfarna användare älskar dem, men av olika anledningar är de fel val för de flesta:

- **Debian:** Modersystemet som både Ubuntu och Linux Mint bygger på. Det är extremt stabilt och pålitligt, men det fokuserar på äldre, djupt testad mjukvara. Det kräver också en hel del manuell konfiguration och teknisk kunskap för att få igång drivrutiner och vardagliga program.
- **NixOS:** Ett unikt och extremt kraftfullt system där hela operativsystemet konfigureras via en enda textfil (deklarativ konfiguration). Det är fantastiskt för utvecklare och systemadministratörer, men inlärningströskeln är monumental för en vanlig användare.
- **RHEL (Red Hat Enterprise Linux)**: Ledande av företagssystem. Det är maximalt stabilt och driver en enorm del av världens servrar, men det är helt låst till företagets ekosystem och licensmodeller, vilket gör det helt ointressant för en privatperson.
- **Ubuntu:** Som tidigare nämnts är det i grunden ett stabilt och funktionellt system med enormt mjukvarustöd. Problemet är att det numera är optimerat för storföretag och servrar, vilket gör och hemanvändare tvingas dras med Canonicals egna kommersiella lösningar som Snap-paket och systemreklam.

Specialiserade skrivbordsdistributioner för specifika behov

Dessa system är förkonfigurerade för vanliga datorer med skärm och tangentbord, men de är skräddarsydda för att lösa en specifik uppgift på skrivbordet maximalt effektivt.

antiX

- **Syfte:** Att köra ett fullt fungerande skrivbordssystem på extremt gammal eller svag hårdvara.
- **Hur det fungerar:** antiX är helt fritt från systeminit (systemd), vilket gör det otroligt resurssnålt. Det använder inga tunga skrivbordsmiljöer utan förlitar sig på mycket lätta fönsterhanterare. Det kan starta och köras smidigt på datorer med så lite som 256 MB arbetsminne och gamla 32-bitars processorer. Perfekt för att förvandla en 15–20 år gammal dator, som annars hade kastats, till en fullt fungerande skrivmaskin eller enklare

surfstation.

Bazzite

- **Syfte:** Renodlad speldator.
- **Hur det fungerar:** Detta är ett modernt, oföränderligt (immutable) skrivbordssystem helt optimerat för datorspel. Det är byggt för att efterlikna operativsystemet på Steam Deck (SteamOS), har färdiga grafikdrivrutiner för Nvidia och AMD, och startar direkt in i ett strömlinjeformat spelgränssnitt som gör att du slipper konfigurera något i terminalen för att spela.

LibreELEC

- **Syfte:** Att förvandla en dator eller en Raspberry Pi till ett renodlat mediacenter för TV:n.
- **Hur det fungerar:** LibreELEC är ett extremt minimalistiskt operativsystem som bara har en enda uppgift: att starta och driva mediaprogrammet Kodi. Systemet saknar ett vanligt skrivbord med ikoner och webbläsare. Istället startar datorn direkt in i ett TV-anpassat gränssnitt som styrs helt med en fjärrkontroll eller handkontroll. Det drar minimalt med systemresurser, vilket gör att det kan spela upp högupplöst film och musik helt utan lagg, även på mycket strömsnål hårdvara.

Qubes OS

- **Syfte:** Extrem säkerhet genom total isolering.
- **Hur det fungerar**: Detta är skrivbordssystemet för den som är paranoid gällande säkerhet. Istället för att köra alla program i samma system, delar Qubes upp ditt skrivbord i isolerade, virtuella "kuber" (t.ex. en för bankärenden, en för arbete och en för osäkra webbsidor). Om du råkar ladda ner ett virus i din "surfkub" är resten av datorn och dina filer fortfarande helt opåverkade.

Raspberry Pi OS

- **Syfte:** För barn och unga som vill lära sig programmera och bygga elektronikprojekt.
- **Hur det fungerar:** Detta är det officiella operativsystemet för enkla enkortsdatorer (Raspberry Pi). Det är extremt lättviktigt och stabilt eftersom det i grunden bygger på Debian. Systemet levereras helt färdigkonfigurerat med allt en ung kodare behöver för att komma igång: visuella programmeringsverktyg som Scratch, utvecklingsmiljöer för Python (Thonny), samt speciella bibliotek för att kunna styra fysiska komponenter som lampor och sensorer som kopplas till datorn. Det är standarden i skolsalar över hela världen för att lära ut datorteknik på ett praktiskt sätt.

Tails

- **Syfte:** Extrem integritet och anonymitet i vardagen.

- **Hur det fungerar:** Tails körs direkt från ett USB-minne på din vanliga dator och lämnar inga spår efter sig på hårddisken. Systemet skickar all internettrafik krypterat genom Tor-nätverket och raderar allt du gjort ur arbetsminnet så fort du stänger av datorn. Perfekt för journalister eller när du tillfälligt behöver surfa helt spårlöst på en lånad dator.

Ubuntu Studio

- **Syfte:** Produktion av ljud, video och grafik.
- **Hur det fungerar:** Att konfigurera Linux för ljud- och videoproduktion kräver ofta speciella drivrutiner och en modifierad systemkärna för att undvika fördröjningar (latens). Ubuntu Studio har en så kallad low-latency-kärna inbyggd från start och levereras färdig med professionella open source-verktyg för ljudmixning, videoredigering och 3D-animering.

Distributioner som rekommenderas ibland – men som du bör undvika

Det man bör förstå när man börjar titta på Linux är att vem som helst som kan skriva lite kod kan göra en distribution. Detta har gett upphov till en mängd instabila och dysfunktionella hobbysystem. Tyvärr rekommenderas dessa ibland i forum och guider, vilket kan leda till en frustrerande upplevelse för en nybörjare.
Följande system dyker ofta upp i diskussioner, men bör undvikas av alla:

- **Arch Linux:** Saknar grafiskt installationsprogram som standard och kräver att du bygger upp operativsystemet textbaserat via terminalen. Minsta felsteg vid uppdateringar kan göra att systemet inte startar. Detta system fungerar mest som en lekstuga för vuxna och är för instabila eller nischade för att tas på allvar i yrkeslivet.
- **Deepin:** Visuellt tilltalande men tungt, dåligt optimerat för servrar utanför Kina och innehåller inbyggd telemetri.
- **Elementary OS:** Skrivbordsmiljön Pantheon och den tillhörande mjukvarubutiken är buggiga och har ett mycket begränsat programutbud**.**
- **Manjaro:** En instabil kompromiss som ofta går sönder när dess egna uppdateringar krockar med externa Linux-program. Byggt på Arch.
- **Zorin OS:** Modifierar gränssnittet hårt för att efterlikna Windows, vilket leder till buggar och småstrul vid systemuppdateringar.

12. Lycka till med Linux!

Att ta steget till Linux handlar inte bara om att byta operativsystem – det handlar om att ta tillbaka kontrollen över din egen hårdvara och din digitala integritet. Genom att välja en stabil distribution slipper du påtvingade uppdateringar, dold telemetri och licenskostnader.

Det bästa sättet att komma igång är att testa systemet i lugn och ro. Kom ihåg att du kan köra de flesta av de rekommenderade distributionerna direkt från ett USB-minne (så kallad "Live-USB") utan att ändra någonting på din befintliga hårddisk. Det gör att du kan prova gränssnittet, kontrollera att ditt Wi-Fi fungerar och bekanta dig med miljön innan du fattar beslutet att installera.

Om du stöter på problem under resans gång finns det ett enormt globalt community. Eftersom du har valt en av de stora, etablerade distributionerna finns i stort sett alla tänkbara frågor och lösningar redan dokumenterade i forum och wikis på nätet.
Välj den distribution som passar dina behov bäst, ladda ner ISO-filen och påbörja installationen.

Lycka till!

> “ Det bästa med Linux är att ingen äger det, men alla kan bidra till att göra det bättre. När koden är öppen tillhör framstegen oss alla.
>
> Linus Torvald (grundare av Linux)

Björknet - Infrastruktur på mänskliga villkor.
"*Humanism i digital gestaltning.*"

Lösenordsguiden

1. Varför lösenordshanterare behövs
2. Tre typer av lösenord – Välj rätt efter syfte
3. Säker hantering och delning
4. Tvåfaktorsautentisering (2FA) – Det extra skyddet
5. Hantering vid misstänkt läcka

1. Varför lösenordshanterare behövs

Varje konto och tjänst kräver ett unikt lösenord. Om samma lösenord används på flera ställen innebär en dataläcka hos en enskild tjänst att alla dina konton blir sårbara.

Eftersom det är mänskligt omöjligt att memorera dussintals unika och komplexa lösenord, är den generella rekommendationen att använda en lösenordshanterare. Verktyget sparar och krypterar dina uppgifter så att du endast behöver komma ihåg ett enda huvudlösenord.

Webbläsarens inbyggda hanterar

Det är vanligt att webbläsare (som Chrome, Edge eller Firefox) erbjuder sig att spara lösenord automatiskt. Dessa är ofta sämre skyddade. Om någon får tillgång till din dator eller ditt synkroniserade Google/Microsoft-konto, får de ofta direkt tillgång till alla sparade lösenord i klartext. Stäng därför av funktionen "Erbjud att spara lösenord" i webbläsaren och låt en dedikerad lösenordshanterare sköta all lagring.

2. Tre typer av lösenord – Välj rätt efter syfte

Vilken typ av lösenord du ska välja beror helt på hur det ska användas. Lösenord kan delas in i tre kategorier:

Kategori A: Genererade lösenord (För webbtjänster och appar)

Används för alla konton på internet och lokala tjänster (t.ex. Nextcloud, Baserow, e-post).

- **Björknets rekommenderade standard:** 20 tecken. Ska bestå av en helt slumpmässig blandning av stora och små bokstäver, siffror och specialtecken.
- **Undvik:** Tvetydiga tecken som kan förväxlas (exempelvis stort I, litet l och siffran 1).
- **Hantering**: Låt lösenordshanteraren generera och fylla i dessa automatiskt. Du ska inte memorera dem.

Kategori B: Huvudlösenord (För memorering)

Används till själva lösenordshanteraren (t.ex. Vaultwarden) eller för att låsa upp krypterade diskar.

- **Björknets rekommenderade standard:** Minst 12 tecken. Ska bestå av ord som uppfattas som slumpmässiga för utomstående, men som är möjliga för dig att memorera. Kombineras med siffror och specialtecken.
  Exempel på struktur: "f*ED|k.m@5rId¤gU&k@-135" (Fredrik Madrid Gurka)
- **Hantering:** Måste kommas ihåg utantill. Skriv aldrig ner detta digitalt.

Kategori C: Lokala lösenord (För snabb enhetsåtkomst)

Används för fysiska enheter som du har direkt tillgång till, exempelvis för att logga in på din personliga dator eller mobil.

- **Björknets rekommenderade standard:** 8–12 tecken med en blandning av stora och små bokstäver, siffror och specialtecken.

- **Hantering:** Kan hållas något kortare för att tillåta snabb inmatning, eftersom säkerheten här primärt skyddas av att enheten blockerar inloggningsförsök efter ett visst antal felaktiga inmatningar.

3. Säker hantering och delning

Ett starkt lösenord förlorar sitt värde om det hanteras oaktsamt. Följ dessa fasta regler för hantering:

Skicka aldrig lösenord i klartext: Lösenord ska aldrig skrivas direkt i chattar (såsom Nextcloud-chatt) eller i vanliga e-postmeddelanden.

Använd säkra delningsverktyg: Björknet rekommenderar och tillhandahåller verktyg för säker delning. Använd Password Pusher eller de inbyggda delningsfunktionerna i Vaultwarden när du behöver skicka ett lösenord till någon annan.

Begränsa livslängden: När du genererar en delningslänk för ett lösenord, ställ alltid in en kort giltighetstid (t.ex. några timmar) samt ett lågt maximalt antal öppningar (förslagsvis att länken raderas efter att den har öppnats en gång).

Fysisk omgivning: Var uppmärksam på din omgivning så att ingen ser när du skriver in känsliga lösenord.

Lånade enheter: Logga aldrig in på känsliga konton från publika eller lånade datorer, då du inte kan kontrollera om enheten är övervakad eller infekterad med skadlig programvara.

4. Tvåfaktorsautentisering (2FA) – Det extra skyddet

Ett lösenord – oavsett hur starkt det är – utgör bara en enda försvarslinje. Björknet rekommenderar att du alltid aktiverar tvåfaktorsautentisering (2FA) på de tjänster där det är möjligt, särskilt för kritiska konton som Nextcloud och din lösenordshanterare.

- **Så fungerar det:** Vid inloggning krävs både ditt lösenord och en tidsbegränsad engångskod (ofta 6 siffror) som genereras i en app på din telefon.
- **Varför det behövs:** Om ditt lösenord skulle läcka kan en obehörig person ändå inte logga in, eftersom de saknar den fysiska enhet som genererar koderna.

- **Rekommenderade verktyg:** Använd en dedikerad autentiseringsapp för att hantera dina koder, till exempel den inbyggda funktionen i Vaultwarden, Aegis eller Proton Authenitcatior.

5. Hantering vid misstänkt läcka

Om du misstänker att ett lösenord har kommit på avvägar måste du agera omedelbart:

- **Byt lösenord direkt:** Om ett lösenord har läckt, eller om du av misstag råkat skicka det i en öppen chatt, ska det bytas ut direkt i den berörda tjänsten.
- **Kontrollera läckor:** Du kan använda tjänster som Have I Been Pwned (vilket finns integrerat direkt i Vaultwarden) för att kontrollera om dina e-postadresser eller lösenord har förekommit i kända publika dataläckor på internet.

> “ Komplexitet är säkerhetens fiende.
>
> Gary McGraw

Björknet - Infrastruktur på mänskliga villkor.
"*Humanism i digital gestaltning.*"

Telemetri

1. Definition av telemetri
2. Kategorisering: Konstruktiv kontra problematisk telemetri
3. Legitimering enbart genom öppen källkod och samtycke
4. Standardisering: OpenTelemetry

1. Definition av telemetri

Telemetri härleds från grekiskans tele (fjärr) och *metron* (mått) och avser fjärrmätning samt automatisk överföring av data. Inom programvaruarkitektur innebär telemetri att en applikation samlar in och exporterar drifts- och användningsdata till utvecklare eller systemadministratörer.

Processen omfattar primärt tre datakategorier:

- **Prestandadata:** Metrik gällande resursallokering, såsom processor- (CPU) och minnesutnyttjande (RAM).
- **Användningsdata:** Aggregerad information om funktionsnyttjande och navigationsmönster i gränssnittet.
- **Felrapportering:** Diagnostisk information och stackspårningar som genereras vid applikationsfel eller systemkrascher.

2. Kategorisering: Konstruktiv kontra problematisk telemetri

Telemetrins funktion och legitimitet bestäms av dess transparens, användarkontroll och bakomliggande syfte.

Kraschrapportering och felanalys

Teknisk felrapportering och kraschrapporter är nödvändiga verktyg för underhåll och stabilitetsförbättringar, då de isolerar systemdata vid applikationskrascher. Men inte heller denna form av telemetri är universellt acceptabel. Om programvaruarkitekturen är proprietär kan innehållet i rapporterna inte granskas. Det innebär att unika identifierare, minnesdumpar eller dolda personuppgifter (PII) kan överföras utan administratörens vetskap eller kontroll under förevändning av felhantering.

Proprietär kod och kommersiella intressen

Inbäddning av telemetri i proprietär (stängd) programvara introducerar betydande risker. Eftersom källkoden inte är tillgänglig för extern granskning, är det omöjligt att verifiera exakt vilken information som samlas in. I kommersiella ekosystem uppstår en inneboende intressekonflikt där gränsen mellan produktoptimering och monetärisering av användardata ofta suddas ut.

Exploatering av användardata

När telemetri frikopplas från funktionella syften kan den transformeras till ett verktyg för beteendeanalys och övervakning. Detta sker huvudsakligen genom tre mekanismer:

- **Beteendemanipulation:** Genom att mäta skärmtid och interaktionsfrekvens kan gränssnitt optimeras för att maximera användarens digitala kvarhållning (retention) och skapa digitala beroenden.
- **Kommersiell profilering:** Användningsmönster och metadata aggregeras till digitala profiler som säljs vidare till datamäklare för riktad annonsering eller beteendebaserad påverkan.
- **Prisdiskriminering:** Analys av hårdvarudata, geografisk position och konsumtionsbeteende kan ligga till grund för dynamisk prissättning av tjänster baserat på den unika användarprofilen.

3. Legitimering enbart genom öppen källkod och samtycke

Kontentan är att telemetri – oavsett om det rör sig om användarstatistik eller kraschrapporter – endast är rättfärdigad när den implementeras inom ramen för öppen källkod och är konfigurerad som Opt-in (avstängd som standard).
När källkoden är öppen och transparent kan hela dataströmmen granskas av oberoende parter för att säkerställa att inga dolda eller kommersiella intressen existerar, samt att ingen känslig data

döljs under pseudonymiserade id-nummer. Inom öppen källkods-gemenskapen sker utvecklingen på användarnas villkor och med explicit samtycke. Syftet är alltid att förbättra verktyget för kollektivet, snarare än att generera aktieägarvärde genom datainsamling.

4. Standardisering: OpenTelemetry

Branschen rör sig mot en ökad standardisering genom initiativet OpenTelemetry (OTel).

OpenTelemetry är ett leverantörsneutralt ramverk under öppen källkod som standardiserar insamling av loggar, spårning (traces) och mätvärden (metrics). Genom att ersätta proprietära, slutna lösningar med en enhetlig och transparent infrastruktur återförs kontrollen till systemadministratören. Detta möjliggör full insyn i dataströmmarna och eliminerar risken för leverantörsinlåsning i kommersiella analysplattformar, då administratören själv definierar exakt var den insamlade datan ska deponeras.

> “ Övervakningskapitalism är en ny ekonomisk ordning som gör anspråk på den privata mänskliga erfarenheten som gratis råvara för dolda kommersiella metoder för utvinning, förutsägelse och försäljning.
>
> Clive Humby

Björknet - Infrastruktur på mänskliga villkor.
"*Humanism i digital gestaltning.*"

Öppen källkod

1. Introduktion
2. Vad är öppen källkod?
3. Varför öppen källkod?
4. Olika typer av licenser
5. Hur kan du ta makten över din data?
6. Rekommenderade öppen källkodsverktyg
7. Tillsammans för samarbeten, öppenhet och frihet

1. Introduktion

Denna sida fungerar som en strategisk och praktisk vägledning för hur vi förhållningssätt till digital infrastruktur, datalagring och mjukvaruval. I en digital miljö som alltmer domineras av stängda ekosystem och centraliserade molntjänster är valet av teknisk plattform avgörande för vår operativa självständighet.

Syftet med detta dokument är att definiera principerna bakom öppen källkod och tydliggöra hur dessa principer tillämpas i vår dagliga drift. Genom att basera vår infrastruktur på öppna system säkerställer vi full insyn, långsiktig kontroll över vår egen data samt minimerat beroende av enskilda externa programvaruleverantörer (så kallad digital suveränitet).

2. Vad är öppen källkod?

Öppen källkod (open source) innebär att den underliggande programkoden i en mjukvara är publikt tillgänglig för vem som helst att läsa, granska, modifiera och distribuera vidare. Istället för att koden hålls hemlig av ett enskilt företag, utvecklas programvaran ofta öppet och transparent, vilket gör det möjligt för oberoende utvecklare och organisationer över hela världen att samarbeta kring förbättringar och säkerhetsuppdateringar.

Motsatsen till öppen källkod är *proprietär* mjukvara (stängd källkod). I proprietära system är programkoden en affärshemlighet som ägs och kontrolleras strikt av en specifik leverantör. Användaren köper eller hyr endast rätten att använda den färdiga produkten, utan möjlighet att kontrollera hur programmet faktiskt fungerar under ytan eller vart informationen skickas.

3. Varför öppen källkod?

Om tjänsten är gratis så är du produkten!

Att basera en verksamhet på *proprietär (stängd)* mjukvara innebär att man överlåter den totala kontrollen till en extern part, vilket skapar fundamentala risker i form av leverantörsinlåsning, dolda säkerhetshål och dolda kostnader. Inom den proprietära mjukvaruvärlden finansieras gratis digitala tjänster nästan uteslutande genom kommersialisering av användarnas privatliv, där betalningen består av beteendedata och personlig information som samlas in i bakgrunden för att säljas vidare till annonsmarknaden. Om en sådan leverantör dessutom drabbas av driftstörningar, ändrar sina licensvillkor eller går i konkurs riskerar användaren att förlora tillgången till sina egna arbetsverktyg och sin historiska data över en natt.

Genom att istället välja mjukvara med *öppen källkod* elimineras dessa risker till förmån för en teknisk miljö som bygger på transparens, oberoende och långsiktig stabilitet. Eftersom koden är publik kan den kontinuerligt granskas av oberoende experter, vilket garanterar högre säkerhet och förhindrar dold spårning, profilering eller otillåten datainsamling. Öppen källkod bryter logiken kring övervakningsekonomin genom att finansieras av utvecklingsbidrag, donationer eller supportavtal, samtidigt som användningen av öppna dataformat säkerställer att informationen förblir tillgänglig, flyttbar och kontrollerad av verksamheten själv under överskådlig tid.

4. Olika typer av licenser

Mjukvara med öppen källkod styrs av licenser som juridisk garant för att koden ska förbli tillgänglig och att upphovsmännen skyddas. Licenserna delas generellt in i två huvudkategorier: tillåtande licenser (permissive) och copyleft-licenser.

Här är de vanligaste licensformerna och deras huvudsakliga innebörd:
MIT-licensen: En av de mest tillåtande licenserna som existerar. Den tillåter vem som helst att göra vad som helst med koden – inklusive att använda den i kommersiella syften, ändra den och stänga den i en proprietär produkt – så länge ursprungliga upphovsrättsmeddelanden och ansvarsfriskrivningar inkluderas.

Apache 2.0
Liknar MIT-licensen i att den är mycket tillåtande, men innehåller ytterligare explicita skydd och regler gällande patenträttigheter. Användare får modifiera och distribuera koden fritt, men ändringar i filerna måste dokumenteras.

GNU General Public License (GPL)
Den mest kända copyleft-licensen. Den tillåter fri användning och modifiering, men har ett strikt villkor: om mjukvaran modifieras och distribueras vidare måste även den nya, modifierade koden göras tillgänglig under samma GPL-licens. Detta förhindrar att koden stängs in i proprietära program.

GNU Affero General Public License (AGPL)
En vidareutveckling av GPL, specifikt anpassad för molntjänster och nätverksprogramvara. Om en modifierad version av ett AGPL-program körs på en server där användare interagerar med den över ett nätverk, måste hela källkoden till den modifierade versionen göras tillgänglig för dessa användare.

Mozilla Public License (MPL)
En hybridlicens som ligger mellan de tillåtande licenserna och GPL. Den kräver att modifieringar av befintliga filer i det specifika MPL-projektet måste förbli öppna under MPL, men tillåter samtidigt att dessa filer kombineras med proprietär kod i ett större programvarukomplex.

5. Hur kan du ta makten över din data?

Att ta makten över sin data handlar i praktiken om att flytta sin digitala infrastruktur från centraliserade molnjättar till system som man själv kontrollerar genom egen drift (så kallad self-hosting). Istället för att lita på att externa storföretag hanterar informationen säkert och integritetsvänligt, tar man fullt ägandeskap över de servrar och applikationer där data lagras och bearbetas. Det innebär att ingen utomstående part har teknisk möjlighet att läsa, blockera eller sälja informationen vidare. I praktiken genomförs detta genom att använda en egen server eller specialiserade drifttjänster där applikationer med öppen källkod installeras.

Det är dock viktigt att ha med sig att öppen källkodsutvecklingen är en pågående och dynamisk process. Många lösningar är under aktiv uppbyggnad och alla funktioner är inte alltid helt mogna eller färdigutvecklade jämfört med kommersiella jättars miljardprojekt. Men utvecklingstakten är hög; communityt växer snabbt och mer robusta, polerade alternativ tillkommer hela tiden. Att ta makten över sin data innebär därför också att acceptera en viss inlärningskurva i utbyte mot fullständig digital suveränitet och vetskapen om att systemen förbättras för varje dag som går.

EU:s satsningar på digital suveränitet

Frågan om att äga sin egen data har även blivit en central politisk prioritet inom Europa. Genom lagstiftningar som dataförordningen (Data Act) och strategiska initiativ för digital suveränitet satsar EU stora resurser på att minska beroendet av utomeuropeiska molnjättar. Det görs omfattande offentliga investeringar i projekt baserade på just öppen källkod för och att främja öppna standarder, datadelning under användarens kontroll och utveckling av säkra europeiska molnalternativ.

6. Rekommenderade öppen källkodsverktyg

Att gå över till öppen källkod behöver inte ske över en natt. Det mest effektiva tillvägagångssättet är att börja byta ut enskilda program och tjänster steg för steg. Genom att ersätta proprietära standardverktyg med öppna alternativ minskas datainsamlingen omedelbart utan att arbetsflödet störs.

Här är de rekommenderade stegen för att komma igång:

Byt webbläsare
Ersätt Google Chrome eller Microsoft Edge med Mozilla Firefox. Firefox är en oberoende webbläsare som inte kontrolleras av de stora annonsplattformarna, vilket ger bättre inbyggt skydd mot spårning och spara-funktioner som inte kartlägger ditt beteende.
Byt kontorsvit: Ersätt Microsoft Office med LibreOffice eller OnlyOffice. Dessa verktyg hanterar dokument, kalkylblad och presentationer lokalt på din dator utan krav på molnkonto eller löpande licenskostnader.

Byt e-postklient
Ersätt proprietära e-postprogram med Thunderbird. Det är en kraftfull och säker klient för hantering av e-post, kalendrar och kontakter direkt från skrivbordet, med full kontroll över din datalagring.

Ersätt molntjänster
Centraliserade tjänster som Google Drive, Dropbox och iCloud kan fasas ut till förmån för Nextcloud (självdriftad) eller Twake (SAAS-tjänst). Dessa plattformar samlar filsynkning, samarbetsverktyg,

chatt och kalendrar i ett gränssnitt som du kan drifta själv eller hyra via en integritetsfokuserad leverantör.

Decentraliserad kommunikation och sociala medier

För att helt bryta sig loss från de proprietära plattformarnas algoritmer och övervakning används federerade nätverk där ingen enskild aktör äger infrastrukturen:

Matrix
Ett öppet protokoll för säker, krypterad realtidskommunikation som ersätter Slack, WhatsApp och Microsoft Teams. Genom att bygga på ett decentraliserat nätverk kan användare på olika servrar kommunicera sömlöst med varandra, precis som med e-post.

Mastodon
Ett decentraliserat alternativ till traditionella sociala medier som Twitter/X. Istället för att styras av en central algoritm som maximerar skärmtid och samlar data för annonsering, består Mastodon av tusentals oberoende servrar (instanser) som samverkar i ett globalt, användarstyrt nätverk.

7. Tillsammans för samarbeten, öppenhet och frihet

Digital suveränitet är i grunden inte ett soloprojekt, utan resultatet av ett globalt samarbete. När en verksamhet väljer mjukvara med öppen källkod blir man en del av ett större ekosystem där utvecklare, organisationer och slutanvändare delar kunskap, kod och resurser. Varje enskilt val att migrera till en öppen tjänst gör stor skillnad, eftersom det flyttar resurser och legitimitet från centraliserade monopol till transparenta alternativ. Ditt val är viktigt för att skifta balansen på marknaden; ju fler som väljer öppna system, desto starkare, säkrare och mer ekonomiskt bärkraftiga blir de gemensamma verktygen för alla.

Friheten att granska, modifiera och dela vidare mjukvara innebär också det gemensamma byggandet av en mer demokratisk digital infrastruktur. Istället för att gömma tekniska lösningar bakom stängda licensavtal, bidrar öppna system till att sänka trösklarna för andra som vill ta kontrollen över sin egen data. Genom att aktivt välja öppenhet säkras inte bara den egna verksamhetens oberoende och stabilitet, utan valet fungerar som ett direkt stöd för en digital framtid som premierar transparens, gemensam utveckling och långsiktig frihet.

> ***”Teknik är aldrig neutral, eftersom den tar på sig egenskaperna hos dem som uttänker, finansierar, reglerar och använder den.”***
> *Magnifica Humanitas, kap. 2*

Björknet - Infrastruktur på mänskliga villkor.

"*Humanism i digital gestaltning.*"