JENSEN yrkeshögskola

TRÄNING FÖR VERKLIGHETEN

Examensarbete

Linux vs Windows inom DevOps

En teknisk jämförelse av operativsystemens roll i automatiserad mjukvaruutveckling och driftsättning

Klassens namn: DEV23M

Termin och år: vårterminen 2025

Student: Salah Abukar

Handledare: Ulf Eriksson

Sammanfattning

Valet av operativsystem är en kritisk faktor inom DevOps, där verktyg för kontinuerlig integration och distribution (CI/CD), containerhantering och säkerhet är centrala komponenter. Denna studie undersöker de tekniska skillnaderna mellan Linux och Windows inom DevOps, med fokus på prestanda, kompatibilitet och säkerhetsmodeller.

Genom en kombination av litteraturstudier och praktiska observationer från en LIA-period analyseras hur dessa operativsystem påverkar DevOps-arbetsflöden. Studien visar att Linux erbjuder bättre prestanda, större kompatibilitet med containerbaserade teknologier såsom Docker och Kubernetes samt en mer flexibel säkerhetsmodell. Windows har dock fortsatt en stark position i företag som är beroende av Microsofts ekosystem, inklusive .NET-plattformen och Active Directory.

Resultaten pekar på att hybridlösningar, där både Linux och Windows används beroende på behov, kan vara den mest optimala strategin för många organisationer. Studien bidrar med en vägledning för DevOps-team och beslutsfattare vid val av operativsystem, där både tekniska och affärsmässiga faktorer vägs in.

Innehåll

Inledning

I takt med att DevOps-metodiken blir alltmer central inom mjukvaruutveckling och IT-drift, spelar valet av operativsystem en avgörande roll för organisationers arbetsflöden och tekniska infrastruktur. DevOps innebär en sammansmältning av utveckling och IT-drift med målet att förbättra mjukvaruleveranser genom automatisering, kontinuerlig integration (CI) och kontinuerlig distribution (CD). För att möjliggöra detta används verktyg som Docker, Kubernetes, Jenkins och Ansible, där operativsystemets kompatibilitet och prestanda kan påverka effektiviteten avsevärt.

Två av de mest använda operativsystemen inom DevOps är Linux och Windows, vilka har fundamentala arkitektoniska skillnader som påverkar deras funktion i DevOps-miljöer. Linux har länge varit förstahandsvalet för containerbaserade arbetsflöden, CI/CD-pipelines och molnbaserade applikationer, medan Windows förblir populärt i traditionella företagsmiljöer med starkt beroende av Microsofts ekosystem.

Denna studie undersöker hur valet av operativsystem påverkar DevOps-arbetsflöden genom att analysera prestanda, kompatibilitet och säkerhetsmodeller i Linux- och Windows-miljöer. Vidare undersöks vilka utmaningar och begränsningar som finns vid implementering av DevOps-verktyg på respektive plattform. Studien syftar till att ge en tydligare förståelse av vilket operativsystem som är mest lämpligt för olika typer av DevOps-implementationer och om hybridlösningar kan vara en realistisk strategi för många organisationer.

Problemformulering

Valet av operativsystem påverkar prestanda, säkerhet, kompatibilitet och skalbarhet i DevOps-miljöer. Linux är ofta förstahandsvalet för containerbaserade arbetsflöden, men hur skiljer sig CI/CD-processer och containerteknologi mellan Linux och Windows?

Säkerhetsmodellerna skiljer sig åt, där Linux har en modulär struktur medan Windows erbjuder inbyggda säkerhetsverktyg. Hur påverkar dessa skillnader DevOps-säkerhet och DevOps-modeller?

Kompatibilitet är också en avgörande faktor. Jenkins, Kubernetes, Docker och Ansible är i grunden utvecklade för Linux, men fungerar även på Windows. Vilket operativsystem erbjuder bäst stöd och minst begränsningar för dessa verktyg?

Slutligen spelar stabilitet och skalbarhet en stor roll i hanteringen av DevOps-pipelines och containerorkestrering. Vilket system erbjuder bäst stabilitet och prestanda för komplexa arbetsflöden?

Syfte

Syftet med detta arbete är att analysera hur Linux och Windows hanterar DevOps-verktyg och arbetsflöden. Studien fokuserar på prestanda, kompatibilitet och säkerhet, med målet att identifiera vilket operativsystem som är mest lämpligt för DevOps-miljöer.

Genom att jämföra operativsystemens styrkor och begränsningar syftar studien till att ge praktisk vägledning för utvecklare och företag i valet av OS för DevOps-arbetsflöden.

Frågeställningar

För att uppnå syftet med studien besvaras följande frågor:

1. Hur påverkar operativsystem valet av CI/CD-verktyg och containerhantering?
2. Vilka prestandaskillnader finns mellan Linux och Windows inom DevOps?
3. Hur skiljer sig säkerhetsmodeller och sårbarheter mellan operativsystemen?
4. Vilka utmaningar och begränsningar finns vid användning av DevOps-verktyg på Linux respektive Windows?

Metod

För att besvara frågeställningarna i detta arbete används en kombination av litteraturstudie och observationer från tidigare erfarenhet och LIA. Syftet är att få både en teoretisk och praktisk förståelse av hur Linux och Windows påverkar DevOps-arbetsflöden.

Litteraturstudie

Litteraturstudien omfattar forskning och teknisk dokumentation om CI/CD, containerhantering, prestanda och säkerhet i olika operativsystem. Information samlas in från vetenskapliga artiklar, tekniska rapporter och dokumentation från relevanta plattformar. Källorna väljs utifrån deras aktualitet och relevans för ämnet.

Observationer från tidigare erfarenhet och LIA

För att komplettera litteraturstudien används observationer från arbete med DevOps-verktyg och operativsystem i praktiska sammanhang. Erfarenheter från arbete med CI/CD-pipelines, containerhantering och operativsystemens säkerhetsstrategier dokumenteras och analyseras.

Genom att kombinera teoretiska insikter från litteraturen med praktiska observationer ger detta arbete en bred förståelse av operativsystemens roll i DevOps-miljöer.

Resultat

I detta avsnitt presenteras de huvudsakliga resultaten från litteraturstudien samt observationer från praktisk erfarenhet och LIA. Syftet är att sammanfatta och analysera hur Linux och Windows påverkar DevOps-arbetsflöden utifrån de faktorer som identifierats i problemformuleringen.

Litteraturstudie

I denna litteraturstudie analyseras flera relevanta källor för att besvara studiens frågeställningar om operativsystemens roll i DevOps-miljöer. Genom en systematisk genomgång av akademiska artiklar, tekniska rapporter och dokumentation från branschledande aktörer kartläggs de viktigaste aspekterna av Linux och Windows inom DevOps. Fokus ligger på att identifiera operativsystemens påverkan på CI/CD-verktyg, containerhantering, prestanda och säkerhet.

För att säkerställa en tydlig struktur och en grundlig analys har svaren på frågeställningarna delats upp i separata sektioner. Varje avsnitt presenterar relevanta resultat från litteraturen och belyser specifika för- och nackdelar med Linux och Windows inom DevOps-miljöer.

Frågeställning 1: Operativsystemets påverkan på CI/CD-verktyg och containerhantering

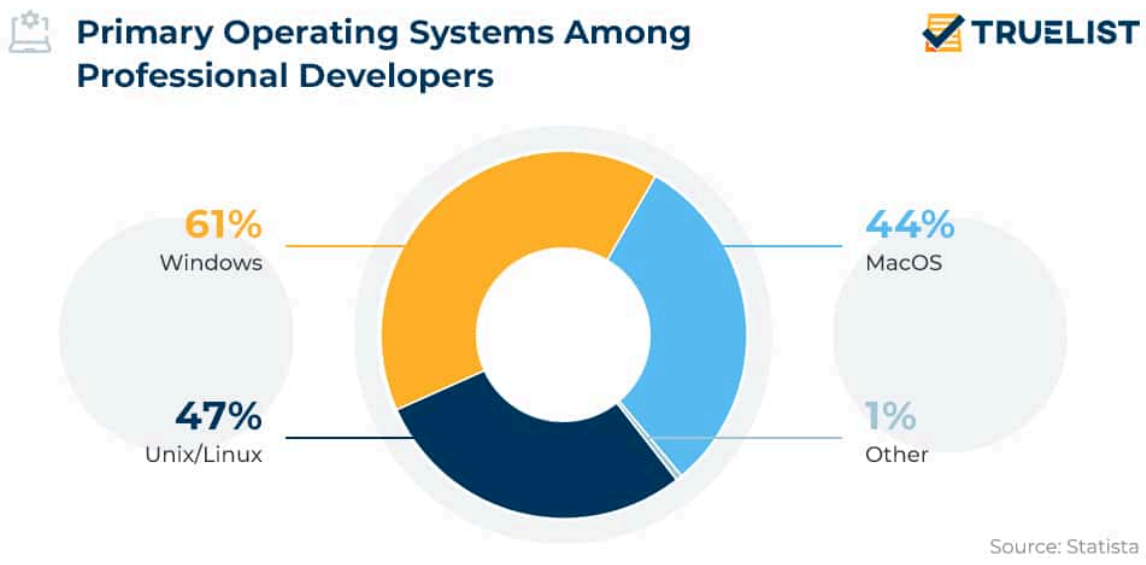
Valet av operativsystem har en direkt inverkan på implementeringen av CI/CD-verktyg och containerhantering i DevOps-miljöer. De tekniska skillnaderna mellan Linux och Windows påverkar arbetsflöden, prestanda och kompatibilitet med etablerade DevOps-verktyg. Flera forskare och experter inom DevOps har analyserat hur dessa skillnader påverkar effektiviteten vid automatiserad mjukvaruutveckling och driftsättning *(Krief, 2022; de Kort, 2016; Lakhera, 2024)*.

CI/CD-verktyg: En teknisk jämförelse mellan Linux och Windows

CI/CD (Continuous Integration/Continuous Deployment) utgör en kärnkomponent inom DevOps, och valet av operativsystem påverkar hur väl dessa processer kan implementeras. Forskning av *Krief (2022)* visar att Linux är det mest kompatibla operativsystemet för DevOps, eftersom verktyg såsom Jenkins, GitLab CI/CD, GitHub Actions och Ansible ursprungligen utvecklades för Linux-miljöer. *De Kort (2016)* påpekar att detta leder till en mer sömlös integration av CI/CD-verktyg, utan behov av omfattande manuella anpassningar.

För att förstå vilket operativsystem som är mest använt inom utvecklingsvärlden kan vi analysera aktuell statistik över primära operativsystem bland professionella utvecklare. Statista (2024) och TrueList (2024) har sammanställt data över användningsfördelningen mellan Windows, Linux och macOS.

Primära operativsystem bland professionella utvecklare (Statista, 2024; TrueList, 2024).

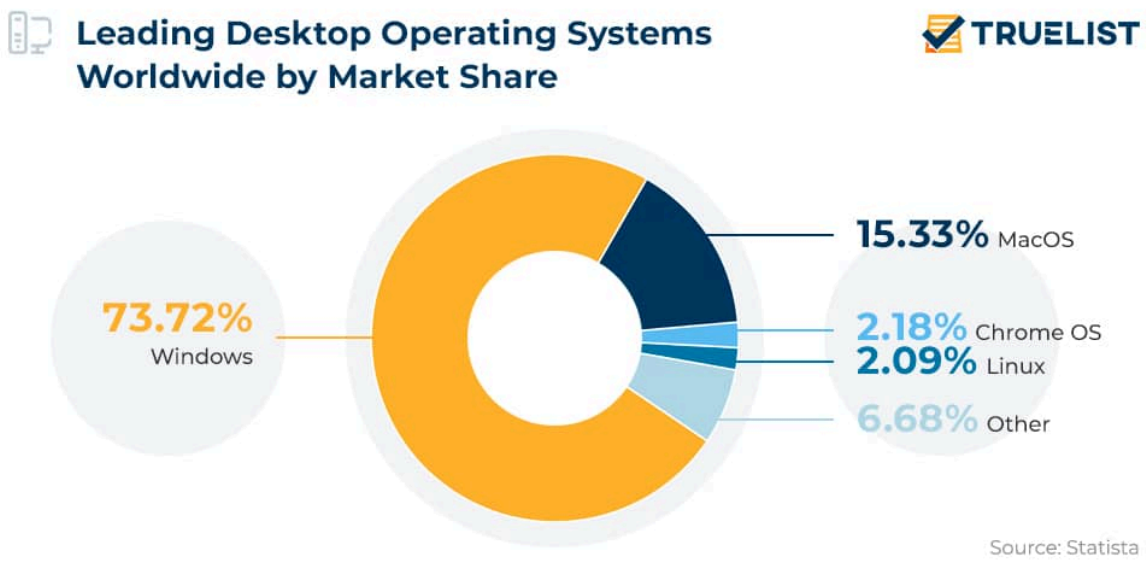


Bilden ovan visar att Windows fortfarande är det mest använda operativsystemet bland professionella utvecklare, med 61 % marknadsandel, jämfört med 47 % för Unix/Linux och 44 % för macOS (Statista, 2024; TrueList, 2024). Detta är dock en generell statistik och inkluderar även utvecklare utanför DevOps-fältet. Enligt forskning av Krief (2022) är Linux fortfarande det dominerande valet inom DevOps och molninfrastruktur, särskilt för CI/CD-pipelines och containerhantering.

Windows är däremot populärare inom företagsmiljöer där Microsoft-teknologier som .NET och Azure DevOps används. Enligt Awan & Khan (2022) föredrar många organisationer Windows på grund av dess användarvänlighet, breda tillgång till kunnig personal och bättre kompatibilitet med populära företagsapplikationer. Det grafiska gränssnittet (GUI) och det intuitiva "peka-och-klicka"-systemet gör det dessutom enklare att installera och konfigurera, vilket många företag ser som en fördel jämfört med Linux, som ofta kräver mer teknisk expertis.

Trots att Linux är dominerande inom DevOps och molnbaserade arbetsflöden, är Windows fortfarande det mest använda operativsystemet globalt. Statistiken nedan visar den totala marknadsandelen för olika operativsystem.

Global marknadsandel för desktop-operativsystem (Statista, 2024; TrueList, 2024).



Enligt Statista (*2024*) och TrueList (*2024*) har Windows 73.72 % av desktop-marknaden, vilket speglar dess dominans i företagsmiljöer. Däremot visar flera studier *(Krief, 2022; Gonzalez, 2017)* att Linux dominerar inom DevOps och molnbaserade arbetsflöden tack vare dess effektivitet och bredare stöd för CI/CD-verktyg och containrar..

Linux erbjuder också en mer robust hantering av beroenden genom pakethanteringssystem som APT (Advanced Package Tool) för Debian-baserade system och YUM/DNF för Red Hat-baserade distributioner. Dessa system möjliggör en effektiv och stabil installation av programvarupaket och bibliotek, vilket är avgörande för att upprätthålla en pålitlig och effektiv CI/CD-pipeline *(Krief, 2022)*. Dessutom har Linux inbyggt stöd för Bash-skript, vilket förenklar automatisering av CI/CD-processer och möjliggör mer effektiva arbetsflöden jämfört med Windows-miljöer *(Lakhera, 2024)*.

Å andra sidan har Microsoft gjort betydande framsteg för att förbättra Windows kompatibilitet med CI/CD-processer genom introduktionen av Windows Subsystem for Linux (WSL), vilket gör det möjligt att köra Linux-kommandon direkt i Windows-miljö (de Kort, 2016). Trots detta kvarstår flera utmaningar. Enligt *Lakhera (2024)* är hanteringen av beroenden mer komplex i Windows jämfört med Linux, där paketinstallationsverktyg som Chocolatey och Windows Package Manager (winget) saknar samma robusthet och effektivitet som de Linux-baserade alternativen. Dessutom kräver Windows ofta anpassade PowerShell-skript för att interagera med CI/CD-verktyg, vilket kan orsaka kompatibilitetsproblem i blandade miljöer där Bash och Linux-baserade verktyg används *(Krief, 2022)*.

Vid en jämförelse av pipeline-exekvering visar forskning av *Gonzalez (2017)* att CI/CD-processer i Windows ofta har högre resursförbrukning och längre exekveringstider än motsvarande processer i Linux-miljöer. Detta beror delvis på att Windows har en tyngre operativsystemarkitektur, vilket leder till ökad belastning på CPU och RAM vid körning av pipelines.

Containerhantering: Linux vs Windows

Containerteknologi är en central del av moderna DevOps-miljöer, och valet av operativsystem spelar en avgörande roll i hur effektivt dessa teknologier kan implementeras. Linux har en tydlig fördel inom detta område, då Docker och Kubernetes utvecklades med Linux som primär plattform *(Kim et al., 2016)*. *Enligt Krief (2022)* erbjuder Linux inbyggt stöd för namespaces och cgroups, vilket möjliggör effektiv resursisolering och hantering av containers. Tack vare Linux kärnstruktur och resurshantering kan fler containrar köras parallellt med minimal overhead jämfört med Windows-containrar.

Vidare påpekar de Kort *(2016)* att Linux-distributioner såsom Ubuntu, CentOS och Alpine har optimerats för containerbaserade arbetsflöden. Dessa operativsystem har lättviktskärnor och minimala systemkrav, vilket möjliggör snabbare uppstartstider och bättre skalbarhet. Detta gör Linux till det föredragna valet för organisationer som arbetar med containerorkestrering och mikrotjänstarkitekturer. Windows har dock gjort framsteg inom containerteknologi genom utvecklingen av Windows Containers och Hyper-V-containrar *(de Kort, 2016)*.

Enligt Lakhera *(2024)* erbjuder dessa lösningar god säkerhet genom Hyper-V-baserad isolering, men de lider fortfarande av högre resurskrav jämfört med Linux-containrar. En av

de största skillnaderna är att Windows Containers ofta kräver en Hyper-V-instans för att säkerställa bättre isolering, vilket resulterar i längre uppstartstider och högre CPU-/minnesförbrukning jämfört med Linux-containrar *(Krief, 2022)*.

En annan begränsning är att Windows Containers har lägre kompatibilitet med Kubernetes, vilket ofta kräver att master-noder körs på Linux medan Windows används enbart för specifika arbetarnoder. Detta ökar komplexiteten i hanteringen av Kubernetes-kluster och skapar en högre administrativ börda *(Krief, 2022)*. Ett potentiellt sätt att hantera dessa begränsningar är genom användning av Windows Subsystem for Linux 2 (WSL2), vilket möjliggör körning av Linux-kärnan inom Windows *(Gonzalez, 2017)*.

Detta har förbättrat Windows kompatibilitet med Linux-containrar, men introducerar samtidigt en extra virtualiseringsnivå, vilket kan leda till prestandaförluster *(Kim et al., 2016)*.

Påverkan på DevOps-arbetsflöden

De tekniska skillnaderna mellan Linux och Windows påverkar DevOps-arbetsflöden på flera sätt. Enligt Davis & Daniels (*2016*) möjliggör Linux en mer agil utvecklingsprocess genom effektivare resurshantering, kortare byggtider och mer robust containerstöd. Windows är däremot bättre lämpat för organisationer som är starkt beroende av Microsofts ekosystem, inklusive .NET-plattformen och Azure DevOps *(de Kort, 2016)*.

Hybridlösningar, där både Windows och Linux används parallellt, har visat sig vara en effektiv strategi för många organisationer. Kim et al. (*2016*) argumenterar för att företag kan dra nytta av Linux för containerhantering och CI/CD, samtidigt som Windows används för specifika applikationer och arbetsflöden där Microsoft-teknologi är dominerande. Denna strategi kan minska begränsningar i Windows-baserade DevOps-miljöer samtidigt som den möjliggör integration med befintliga Windows-baserade system.

Slutsats: Operativsystemens påverkan på DevOps

Sammanfattningsvis visar både teoretisk forskning och praktiska observationer att Linux är det föredragna operativsystemet för CI/CD och containerhantering. Dess starka integration med DevOps-verktyg, effektivare resurshantering och bättre prestanda gör det till ett optimalt val för organisationer som vill optimera sina DevOps-processer. Windows har gjort framsteg genom WSL och bättre integration med DevOps-verktyg, men har fortfarande begränsningar gällande prestanda, kompatibilitet och resurshantering.

För organisationer som primärt arbetar med open-source DevOps-verktyg och containerbaserade arbetsflöden är Linux det naturliga valet. Däremot kan företag som är beroende av Microsofts teknologi dra nytta av hybridlösningar, där Windows används för specifika applikationer medan Linux hanterar CI/CD-pipelines och containermiljöer. Genom att förstå de tekniska och praktiska skillnaderna mellan operativsystemen kan organisationer

fatta mer informerade beslut om hur de bäst strukturerar sina DevOps-miljöer för att uppnå högre effektivitet och stabilitet.

Frågeställning 2: Prestandaskillnader mellan Linux och Windows i DevOps

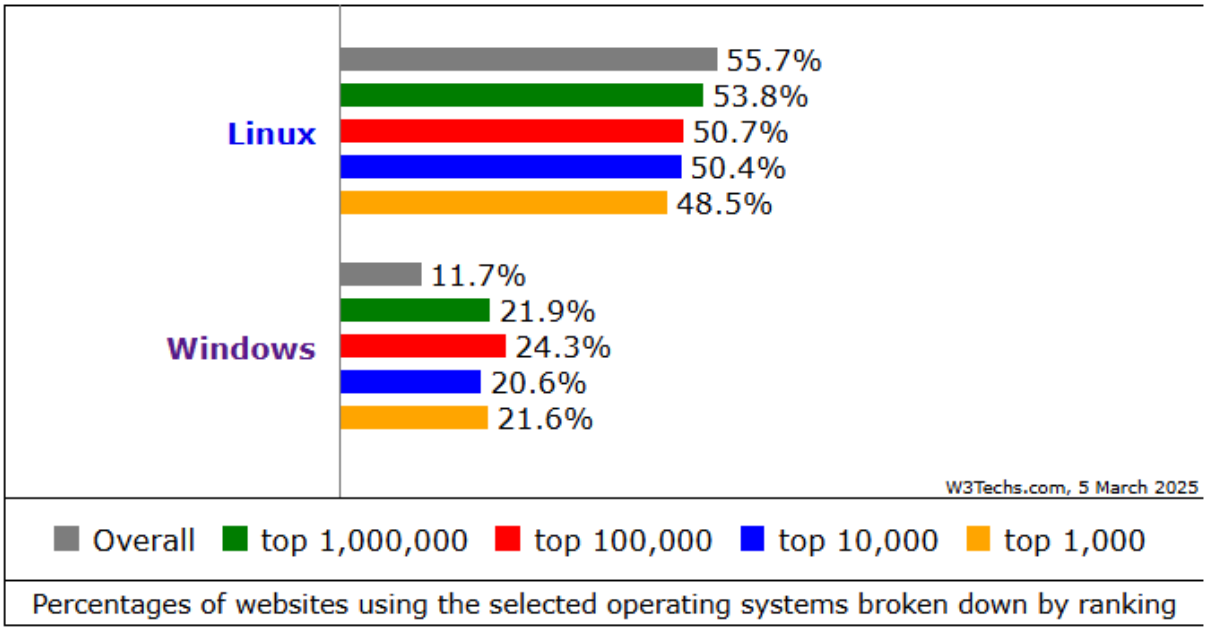
Prestanda är en avgörande faktor vid val av operativsystem i DevOps-miljöer, eftersom det direkt påverkar utvecklingshastighet, skalbarhet och kostnadseffektivitet. Flera studier har jämfört Linux och Windows ur ett DevOps-perspektiv och identifierat betydande skillnader i byggtider, resurshantering och containerprestanda *(Krief, 2022; de Kort, 2016; Kim et al., 2016)*.

Byggtider och effektivitet

Flera forskare, däribland Krief (*2022*), har konstaterat att Linux generellt erbjuder kortare byggtider och snabbare testcykler tack vare dess minimalistiska arkitektur och effektiva resurshantering. Operativsystemets låga overhead innebär att processer som kompilering och testning kan exekveras snabbare jämfört med Windows. Linux hanterar också filsystem och processer mer effektivt, vilket möjliggör smidigare och mer resurseffektiva CI/CD-pipelines *(Gonzalez, 2017)*.

Enligt de Kort (2016) kan skillnaderna i byggtider mellan operativsystemen särskilt observeras i containerbaserade arbetsflöden. Linux-baserade system uppvisar snabbare byggtider tack vare optimerade kärnprocesser och mer effektiva hanteringsmekanismer för systemresurser. Windows, å andra sidan, har vanligtvis längre byggtider och högre resurskrav, vilket kan förlänga utvecklingscykler och minska produktiviteten i DevOps-miljöer *(Kim et al., 2016)*. En avgörande faktor vid val av operativsystem för DevOps är dess prestanda i servermiljöer. Statistik från W3Techs (2025) visar att Linux är det dominerande operativsystemet för webbservrar, medan Windows har en betydligt lägre andel.

Operativsystemsmarknadsandel inom serverdrift enligt W3Techs (2025).



Statistik från W3Techs (*2025*) visar att Linux är det dominerande operativsystemet för webbservrar, medan Windows har en betydligt lägre andel.
Bilden visar att Linux dominerar samtliga kategorier av webbplatser baserat på ranking:

- 55.7 % av alla kända webbplatser kör Linux, medan endast 11.7 % kör Windows.
- Linux används på 53.8 % av de 1 000 000 mest besökta webbplatserna, medan Windows endast används på 21.9 %.
- Bland de topp 100 000 webbplatserna har Linux en andel på 50.7 %, medan Windows endast når 24.3 %.
- För de topp 1 000 mest besökta webbplatserna är 48.5 % Linux-baserade, jämfört med endast 21.6 % som kör Windows.

Denna statistik förstärker slutsatsen att Linux är det föredragna operativsystemet för högpresterande och skalbara servermiljöer. Detta beror på dess effektiva resursanvändning, bättre prestanda i storskaliga DevOps-miljöer och stabilare containerhantering. I kontrast är Windows mer vanligt förekommande i mindre företagsmiljöer där Microsofts ekosystem används, men det har svårt att konkurrera med Linux inom server- och molnbaserade arbetsflöden.

Denna dominans är en direkt följd av Linux optimerade resurshantering, effektivare minnesanvändning och förmåga att hantera storskaliga DevOps-miljöer. Detta bekräftar att Linux är det mest optimala valet för högpresterande miljöer.

Microsoft har dock försökt förbättra Windows prestanda genom att introducera Windows Subsystem for Linux (WSL), vilket gör det möjligt att köra Linux-kommandon och utvecklingsverktyg direkt i Windows-miljö *(de Kort, 2016)*. Även om detta har minskat prestandagapet mellan operativsystemen, kvarstår vissa begränsningar. Enligt Gonzalez (*2017*) är WSL fortfarande beroende av Windows-kärnans processhantering, vilket innebär att vissa CI/CD-processer körs långsammare än i en nativ Linux-miljö.

Vidare visar tester av Lakhera (*2024*) att Windows-operativsystem ofta har mer omfattande overhead, vilket resulterar i längre byggtider och högre resursförbrukning jämfört med Linux. Även med förbättringar såsom Windows Containers och Azure DevOps, kräver Windows fler systemresurser för att uppnå samma prestanda som Linux i DevOps-arbetsflöden.

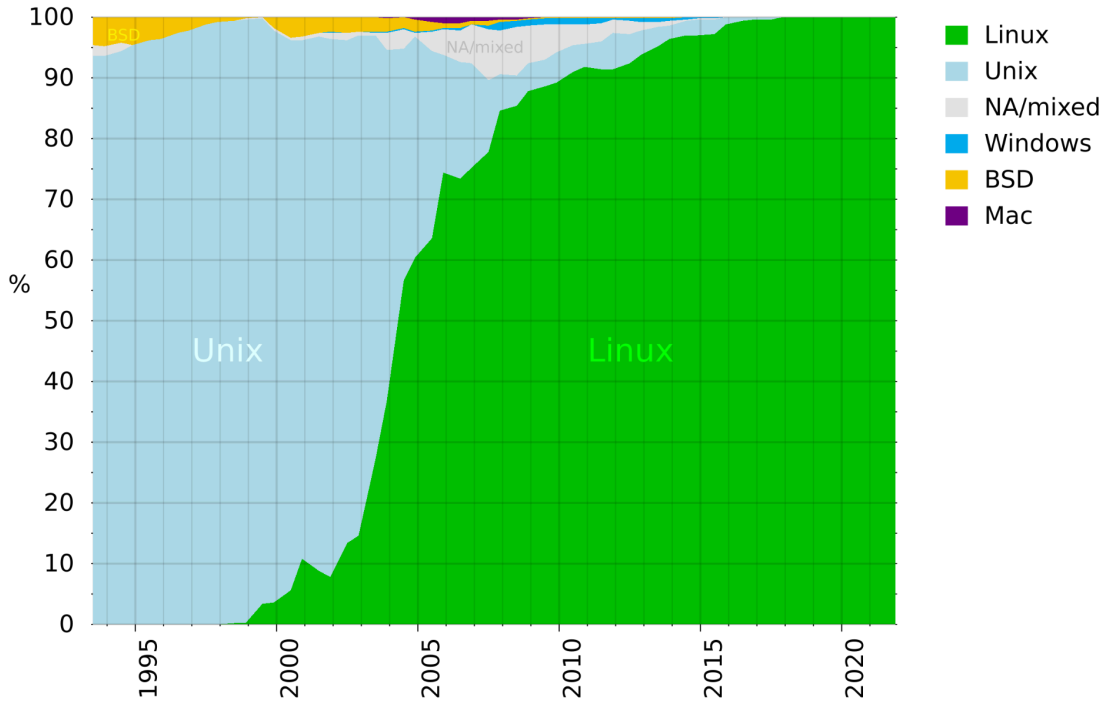
Resurshantering och skalbarhet

En central faktor vid utvärdering av DevOps-prestanda är hur effektivt ett operativsystem hanterar resurser såsom CPU, RAM och diskåtkomst. Forskning av Krief (*2022*) visar att Linux har en mer modulär och optimerad arkitektur, vilket gör det mer effektivt vid hantering av parallella processer. Detta gör Linux särskilt fördelaktigt för storskaliga CI/CD-pipelines och containerbaserade miljöer, där flera samtidiga jobb exekveras.

Windows, å andra sidan, kräver generellt sett mer systemresurser för motsvarande arbetsbelastning. Enligt Davis & Daniels (*2016*) leder detta till ökade driftkostnader, särskilt i molnbaserade DevOps-miljöer där företag betalar för CPU och minnesanvändning. Studier av Gonzalez (*2017*) har också visat att Windows har lägre skalbarhet vid storskaliga DevOps-implementationer, eftersom operativsystemets resurshantering inte är lika optimerad för samtidiga exekveringar som Linux.

Linux och superdatorer – bevis på överlägsen skalbarhet
Enligt den senaste TOP500-listan *(Wikipedia, 2025)* kör 100 % av världens snabbaste superdatorer Linux. Denna absoluta dominans i högpresterande databehandling är ett direkt resultat av Linux effektiva resurshantering, parallella beräkningskapacitet och stabilitet. Windows har ingen representation bland världens superdatorer, vilket bekräftar dess begränsningar inom skalbarhet och optimering för högpresterande miljöer.

Linux dominans i världens snabbaste superdatorer enligt TOP500-listan (Wikipedia, 2025).



Denna graf visar övergången från Unix till Linux i superdatorer och illustrerar Linux växande dominans.

Ett exempel på denna skillnad är virtualisering och containerhantering. Linux kan effektivt hantera tusentals samtidiga processer genom sin kärnfunktionalitet för resurshantering, medan Windows kräver ytterligare konfigurationsarbete och mer resurser för att uppnå motsvarande prestanda *(Kim et al., 2016)*. *De Kort (2016)* påpekar att även vid användning av Windows Server Core, som är en mer lättviktig version av Windows Server, är resurskraven fortfarande högre än för motsvarande Linux-distributioner.

Containerprestanda och orkestrering

Containerteknik utgör en av de mest kritiska komponenterna inom DevOps, och valet av operativsystem påverkar direkt hur effektivt containrar kan hanteras. Enligt Kim et al. (2016) har Linux en naturlig fördel inom containerteknologi, eftersom Docker och Kubernetes ursprungligen utvecklades för Linux. Linux erbjuder också inbyggt stöd för namespaces och cgroups, vilket möjliggör effektiv isolering av processer och resursallokering, något som resulterar i snabbare uppstartstider och lägre CPU-belastning *(Krief, 2022)*.

Windows har däremot en mer komplex arkitektur för containrar. De Kort (*2016*) beskriver hur Windows Containers och Hyper-V-containrar erbjuder säkerhet och isolering, men lider av högre overhead och längre starttider jämfört med Linux-containrar. Enligt Lakhera (*2024*) är detta särskilt påtagligt i Kubernetes-miljöer, där Windows ofta kräver hybridlösningar, där master-noder körs på Linux och Windows används för specifika arbetsnoder.

Enligt Gonzalez (*2017*) har Microsoft försökt förbättra kompatibiliteten genom att introducera WSL2, vilket gör det möjligt att köra Linux-containrar på Windows. Trots dessa förbättringar kräver Windows-containrar fortfarande mer resurser och längre uppstartstider, vilket påverkar den totala prestandan i containerbaserade DevOps-miljöer.

Slutsats: Prestandaskillnader mellan Linux och Windows i DevOps

Analysen av prestandaskillnader mellan Linux och Windows i DevOps-miljöer visar att Linux konsekvent presterar bättre i avgörande områden som byggtider, resurshantering, skalbarhet och containerhantering.

Linux har kortare byggtider, en mer effektiv hantering av CI/CD-processer och en optimerad arkitektur som möjliggör parallell exekvering av fler processer med lägre systembelastning. Detta gör Linux till det mest kostnadseffektiva alternativet för storskaliga DevOps-miljöer där prestanda och stabilitet är avgörande.

En av de starkaste indikatorerna på Linux överlägsenhet är dess dominerande roll inom superdatorer och molninfrastruktur. 100 % av världens snabbaste superdatorer kör Linux, och majoriteten av världens största webbplatser samt molntjänster bygger på Linux-baserade system. Denna omfattande användning i prestandakritiska miljöer visar att Linux är det främsta valet för DevOps-team som arbetar med skalbara, resurskrävande arbetsflöden.

Windows har genom initiativ som WSL (Windows Subsystem for Linux) och Windows Containers försökt minska prestandagapet, men kvarstår som ett mindre effektivt alternativ för DevOps-arbetsflöden. Dess högre overhead, längre byggtider och högre resursförbrukning innebär att Windows kräver fler systemresurser för att uppnå samma prestanda som Linux. För företag som är beroende av Microsofts ekosystem kan hybridlösningar, där Linux hanterar CI/CD och containerbaserade arbetsflöden medan Windows används för specifika applikationer, vara en strategisk kompromiss.

Sammanfattningsvis är Linux det mest optimala operativsystemet för DevOps-miljöer som kräver hög prestanda, effektiv skalbarhet och kostnadseffektiv drift. Organisationer bör därför noggrant analysera sina arbetsflöden och prestandakrav för att säkerställa att deras val av operativsystem stödjer deras långsiktiga DevOps-strategi.

Frågeställning 3: Säkerhetsmodeller och sårbarheter i DevOps-miljöer

Säkerhet är en central aspekt inom DevOps, där operativsystemens säkerhetsmodeller och hantering av sårbarheter påverkar både effektivitet och riskhantering i automatiserade arbetsflöden. Linux och Windows har fundamentalt olika säkerhetsstrategier, vilket medför både styrkor och utmaningar beroende på användningsområde och krav på driftssäkerhet. Enligt forskning av Krief (*2022*) och Davis & Daniels (*2016*) spelar valet av operativsystem en avgörande roll för säkerhetsarkitekturen i DevOps-miljöer.

Säkerhetsmodeller: Linux vs Windows

Operativsystemens säkerhetsmodeller skiljer sig både i arkitektur och implementation, vilket har en direkt inverkan på hur säkerhetsprinciper såsom minsta möjliga åtkomst, processisolering och sårbarhetshantering tillämpas i DevOps-miljöer *(Kim et al., 2016)*.

Linux: En modulär och flexibel säkerhetsmodell

Linux erbjuder en modulär och flexibel säkerhetsmodell, vilket gör det till ett attraktivt operativsystem för DevOps-miljöer med höga krav på säkerhet och isolering. Forskning av Krief (*2022*) visar att Linux inkluderar avancerade säkerhetslösningar såsom:

- Security-Enhanced Linux (SELinux) – En obligatorisk åtkomstkontroll (MAC) som möjliggör finjusterad policyhantering och begränsar behörigheter för processer och användare.
- AppArmor – Ett alternativ till SELinux som erbjuder profiler för att begränsa applikationers åtkomst till systemresurser.
- Seccomp (Secure Computing Mode) – Ett system för att begränsa vilka systemanrop en process kan utföra, vilket minskar attackytan för skadlig kod.

Dessa säkerhetsmekanismer gör Linux särskilt lämpligt för containerbaserade DevOps-miljöer där processisolering och minimal åtkomst är avgörande *(Davis & Daniels, 2016)*. Enligt Gonzalez (*2017*) möjliggör Linux även granulär kontroll över användarrättigheter, vilket minskar risken för att enskilda processer eller användare får oönskade privilegier.

Windows: En centraliserad säkerhetsmodell med högre privilegier

Windows säkerhetsmodell har förbättrats avsevärt genom introduktionen av Windows Defender Application Control (WDAC) och Credential Guard, som skyddar mot obehörig åtkomst och förbättrar hanteringen av autentisering *(De Kort, 2016)*. Enligt Lakhera (*2024*) bygger Windows säkerhet på en centraliserad och integrerad modell, där säkerhetspolicys hanteras genom Active Directory (AD) och Group Policy Objects (GPO).

Trots dessa förbättringar har Windows fortfarande vissa säkerhetsutmaningar:

- Administrativa privilegier: Många säkerhetsinställningar i Windows kräver högre privilegier jämfört med Linux, vilket kan öka risken för att skadlig kod får utökade behörigheter (Kim et al., *2016*).
- Attackyta: Windows har historiskt sett varit en mer frekvent måltavla för cyberattacker, vilket ökar behovet av kontinuerliga säkerhetsuppdateringar och sårbarhetshantering *(Davis & Daniels, 2016)*.

I DevOps-miljöer där säkerhetsprinciper såsom minsta möjliga behörighet och snabb incidenthantering är centrala, kan Windows säkerhetsmodell innebära en högre administrativ börda jämfört med Linux *(Gonzalez, 2017)*.

Sårbarheter och hantering av säkerhetsrisker

Hur operativsystem hanterar säkerhetsuppdateringar och sårbarheter har en direkt inverkan på systemets driftssäkerhet och kontinuerliga tillgänglighet i en DevOps-miljö.

Linux: Snabb och effektiv hantering av sårbarheter

Linux är känt för sin snabba och transparenta hantering av säkerhetsproblem, vilket möjliggör effektiv respons vid upptäckta sårbarheter. Enligt Krief (*2022*) möjliggör Linux öppen källkod att säkerhetsproblem identifieras och åtgärdas snabbt av en global utvecklargemenskap. Säkerhetspatchar distribueras effektivt via pakethanteringssystem såsom:

- APT (Advanced Package Tool) – Används av Debian-baserade distributioner (t.ex. Ubuntu).
- YUM/DNF (Yellowdog Updater, Modified / Dandified YUM) – Används av RHEL och CentOS.
- Zypper – Används av openSUSE.

Denna snabba respons på säkerhetshot är en av de största fördelarna med Linux i DevOps-miljöer där hög tillgänglighet och minimala driftstörningar är avgörande *(Kim et al., 2016)*.

Windows: Centraliserad men långsammare uppdateringshantering

Windows har en mer centraliserad och kontrollerad process för säkerhetsuppdateringar, vilket kan leda till längre ledtider innan kritiska sårbarheter åtgärdas *(De Kort, 2016)*. Enligt Lakhera (2024) innebär detta att system kan vara exponerade för säkerhetsrisker under längre perioder än Linux-baserade system.

En ytterligare utmaning i Windows är att många säkerhetsuppdateringar kräver omstarter, vilket kan störa DevOps-arbetsflöden där kontinuerlig tillgänglighet och minimering av driftstopp är kritiska *(Gonzalez, 2017)*. Detta kan vara särskilt problematiskt i CI/CD-miljöer där system kontinuerligt distribuerar och exekverar kod.

Påverkan på DevOps-arbetsflöden

Skillnaderna i säkerhetsmodeller och hantering av sårbarheter påverkar hur väl ett operativsystem fungerar i DevOps-miljöer. Enligt Davis & Daniels (*2016*) gör Linux säkerhetsmodell det bättre anpassat för DevOps, då det möjliggör en högre grad av automatisering och isolering. Detta minskar risken för säkerhetsincidenter samtidigt som det ger utvecklare och driftsteam mer kontroll över systemets integritet.

Windows kan dock kräva mer omfattande säkerhetsplanering och resurser för att uppnå samma säkerhetsnivå som Linux. De Kort (*2016*) påpekar att DevOps-team som arbetar i Windows-miljöer ofta behöver hantera mer komplexa säkerhetskonfigurationer och en högre administrativ overhead. Trots förbättringar genom Azure Security Center och förbättrad integrering med Kubernetes, innebär Windows fortfarande en högre risk för säkerhetsrelaterade driftstörningar *(Kim et al., 2016)*.

Slutsats: Linux vs Windows i DevOps-säkerhet

Baserat på analysen av operativsystemens säkerhetsmodeller och hantering av sårbarheter kan det konstateras att Linux är det mest lämpliga valet för säkerhetskritiska DevOps-miljöer. Dess flexibla säkerhetsmekanismer, effektiva hantering av säkerhetsuppdateringar och modulära arkitektur gör det mer robust och anpassningsbart för moderna DevOps-arbetsflöden.

Windows erbjuder starka säkerhetslösningar, särskilt i Microsoft-ekosystemet, men har högre administrativ komplexitet och längre uppdateringscykler, vilket kan påverka systemets tillgänglighet och DevOps-effektivitet. Organisationer som prioriterar snabba säkerhetsuppdateringar och minimal driftstörning bör i första hand överväga Linux, medan hybridlösningar kan vara en kompromiss för företag som använder både Windows och Linux i sina DevOps-strategier.

Frågeställning 4: Utmaningar och begränsningar vid användning av DevOps-verktyg på Linux respektive Windows

DevOps-miljöer ställer höga krav på automatisering, kompatibilitet och säkerhet, vilket gör operativsystemets arkitektur och funktionalitet avgörande för hur effektivt DevOps-verktyg kan användas. Linux och Windows har båda styrkor inom olika områden, men också

begränsningar som kan påverka DevOps-arbetsflöden negativt. Enligt Krief (*2022*) och de Kort (*2016*) påverkas valet av operativsystem av faktorer såsom anpassningsmöjligheter, prestanda, säkerhet och kompatibilitet med befintlig infrastruktur.

Utmaningar och begränsningar i Linux-miljöer

Linux har blivit det dominerande operativsystemet inom DevOps tack vare sin flexibilitet, låga overhead och breda stöd för open-source-verktyg. Dock finns det vissa utmaningar och begränsningar som kan påverka företag vid implementering av DevOps-strategier.

Komplexitet och inlärningskurva

Enligt Krief (*2022*) är en av de största utmaningarna med Linux dess höga tekniska tröskel. Linux erbjuder omfattande anpassningsmöjligheter och kraftfulla verktyg såsom Bash, Ansible och Kubernetes, men kräver att användarna har djup teknisk kunskap. Administrering av Linux-system sker huvudsakligen via kommandoraden och konfigurationsfiler, vilket kan vara en utmaning för team som är vana vid GUI-baserade arbetsflöden.

Gonzalez (*2017*) betonar att även om Linux är optimalt för automatisering, kan verktyg som Ansible och Kubernetes kräva manuell konfiguration och tuning för att uppnå maximal prestanda. Detta kan vara en nackdel i organisationer där IT-personal har begränsad erfarenhet av Linux-administration.

Fragmentering och distributioner

Linux består av flera distributioner, såsom Ubuntu, CentOS, Red Hat Enterprise Linux (RHEL) och Debian, vilket kan skapa kompatibilitetsproblem mellan olika system och DevOps-verktyg *(de Kort, 2016).* Denna fragmentering innebär att företag kan behöva standardisera en specifik distribution för att säkerställa enhetliga arbetsflöden.

Enligt Davis & Daniels (*2016*) kan bristen på en enhetlig standard för pakethantering vara en utmaning i stora organisationer. Företag som hanterar storskaliga DevOps-miljöer kan därmed drabbas av ökade underhållskostnader och längre implementeringstider, särskilt om olika team använder olika Linux-distributioner.

Hårdvaru- och drivrutinskompatibilitet

En annan begränsning med Linux är dess begränsade stöd för proprietär hårdvara. Enligt Lakhera (*2024*) erbjuder vissa hårdvarutillverkare begränsat eller inget stöd för Linux, särskilt när det gäller specialiserad nätverksutrustning, lagringslösningar och GPU-acceleration. Detta kan skapa problem för företag som behöver använda Windows-baserade drivrutiner eller firmware, vilket kan påverka prestandan i CI/CD-pipelines och containerbaserade DevOps-miljöer.

Begränsat kommersiellt stöd

Trots att Linux har en stark open-source community för support, kan företag uppleva begränsningar i kommersiellt stöd, särskilt om de använder gratis distributioner såsom Ubuntu eller Debian *(Krief, 2022)*. För företag som kräver snabb support och hög tillgänglighet kan detta vara en nackdel jämfört med Windows, där Microsoft erbjuder omfattande Enterprise-support genom sitt kommersiella ekosystem.

Utmaningar och begränsningar i Windows-miljöer

Microsoft har investerat i att göra Windows mer DevOps-vänligt genom att förbättra PowerShell, Windows Subsystem for Linux (WSL) och Azure DevOps *(de Kort, 2016)*. Trots dessa förbättringar kvarstår vissa begränsningar som kan påverka organisationers förmåga att effektivt implementera DevOps-verktyg.

Licenskostnader och kommersiell inlåsning

En av de största begränsningarna med Windows i DevOps-sammanhang är licenskostnaderna. Enligt de Kort (*2016*) kan kostnaderna för Windows Server, Microsoft SQL Server och Azure-tjänster snabbt öka för företag som använder Windows-baserade DevOps-miljöer. Detta kan göra Windows till ett mindre kostnadseffektivt alternativ jämfört med Linux, där många verktyg och operativsystem är gratis att använda.

Dessutom kan Microsofts kommersiella ekosystem leda till vendor lock-in, där företag blir beroende av Microsoft-specifika lösningar såsom Azure DevOps och Windows Containers. Enligt Gonzalez (*2017*) kan detta minska organisationers flexibilitet att migrera till open-source-lösningar på Linux.

Begränsat stöd för containerteknologi

Windows har gjort framsteg inom containerteknologi genom utvecklingen av Windows Containers och Hyper-V-containrar, men dessa är fortfarande mindre optimerade än Linux-baserade lösningar *(Krief, 2022)*. Enligt Davis & Daniels (2016) har Windows Kubernetes-noder begränsad funktionalitet och kräver ofta hybridlösningar där Linux används som master-node. Detta ökar konfigurationskomplexiteten och underhållskostnaderna, särskilt i stora DevOps-miljöer.

Kompatibilitet med open-source verktyg

Många populära DevOps-verktyg såsom Kubernetes, Terraform och Ansible är primärt utvecklade för Linux, vilket innebär att stöd för Windows är begränsat *(Lakhera, 2024)*. Exempelvis använder Ansible SSH-protokollet, medan Windows istället kräver Windows Remote Management (WinRM), vilket kan skapa säkerhets- och kompatibilitetsproblem i blandade miljöer.

Begränsad flexibilitet och automationsmöjligheter

Windows är traditionellt sett mindre flexibelt än Linux eftersom det har färre möjligheter att anpassa kärnan och systemkonfigurationen *(de Kort, 2016)*. Enligt Kim et al. (2016) är Windows också mer beroende av GUI-baserade administrationsverktyg, vilket kan begränsa möjligheten att automatisera DevOps-processer i stor skala.

PowerShell har dock förbättrat Windows automatiseringsmöjligheter, men enligt Gonzalez (*2017*) är det fortfarande mindre effektivt än Bash och Linux-baserade skriptlösningar. Detta kan påverka hur snabbt DevOps-team kan implementera och optimera sina arbetsflöden.

Slutsats: Operativsystemens utmaningar och begränsningar i DevOps

Både Linux och Windows har specifika utmaningar och begränsningar vid användning av DevOps-verktyg. Linux erbjuder hög flexibilitet, bättre containerstöd och kostnadseffektivitet, men kan ha en brant inlärningskurva, fragmentering mellan distributioner och begränsat stöd för proprietär hårdvara. Windows, å andra sidan, är mer användarvänligt och fungerar väl i Microsofts ekosystem, men har högre licenskostnader, sämre kompatibilitet med open-source-verktyg och mer begränsad automationskapacitet.

Organisationer bör noggrant utvärdera sina behov och befintliga IT-miljöer innan de väljer operativsystem för sina DevOps-arbetsflöden. I många fall kan hybridlösningar där både Linux och Windows används parallellt vara den mest optimala strategin för att balansera kostnader, prestanda och kompatibilitet.

Observationer från tidigare erfarenhet och LIA

I detta avsnitt presenteras insikter och observationer från min LIA-period och tidigare praktiska erfarenheter inom DevOps. Dessa erfarenheter kompletterar den teoretiska analysen och belyser hur Linux och Windows skiljer sig åt i verkliga DevOps-miljöer. Genom att arbeta med olika DevOps-verktyg såsom Jenkins, GitHub Actions, Docker, Kubernetes och Ansible i både Linux- och Windows-miljöer har jag fått en praktisk förståelse av de tekniska och operativa skillnaderna mellan operativsystemen.

CI/CD-verktyg och DevOps-arbetsflöden

En av de mest framträdande skillnaderna mellan Linux och Windows i DevOps-miljöer är hur CI/CD-verktyg hanteras. Under LIA-perioden märkte jag att installationen och

konfigurationen av Jenkins var betydligt enklare i Linux-miljö tack vare dess inbyggda pakethanterare, såsom APT och YUM. Dessa system möjliggör smidig installation av Jenkins och dess beroenden med enstaka kommandon, vilket minimerar konfigurationsarbetet. I Windows-miljö var processen mer komplex och krävde manuella anpassningar med PowerShell-skript. Detta överensstämmer med Krief (*2022*), som påpekar att Linux-baserade CI/CD-pipelines generellt har lägre overhead och färre kompatibilitetsproblem jämfört med Windows-motsvarigheter.

Vid användning av GitHub Actions observerade jag dock att verktyget fungerade likvärdigt på båda plattformarna, eftersom det körs direkt på GitHubs infrastruktur snarare än lokalt. Därmed undveks många av de konfigurationsutmaningar och plattformsberoenden som annars kan uppstå vid lokal körning av CI/CD-verktyg.

När det gäller Ansible fanns det tydliga skillnader i hur verktyget fungerade i Linux- och Windows-miljöer. På Linux kunde Ansible installeras direkt och användas för att hantera fjärrservrar via SSH, vilket är den standardiserade metoden för fjärradministration i Linux-baserade system. I Windows däremot krävdes Windows Remote Management (WinRM) för att möjliggöra samma funktionalitet. Lakhera (*2024*) beskriver hur Windows-miljöer ofta kräver extra säkerhetskonfigurationer och kompatibilitetsinställningar för att fungera optimalt med open-source-verktyg som Ansible, vilket var tydligt under praktisk testning.

Containerhantering och Kubernetes

En annan viktig observation under min utbildning var hur Linux och Windows hanterar containerteknologi och Kubernetes-kluster. Linux visade sig vara det mest effektiva operativsystemet för containerbaserade arbetsflöden, vilket är i linje med forskning av Krief (*2022*) och Gonzalez (*2017*). Installationen av Docker och Kubernetes i Linux var smidig och optimerad, medan samma process i Windows innebar fler konfigurationssteg och kompatibilitetsproblem.

Under arbete med Kubernetes-kluster noterade jag att master-noder kördes på Linux medan arbetarnoder kördes på Windows. Windows-noder hade ofta längre uppstartstider och högre resursförbrukning, vilket även dokumenteras av de Kort (*2016*). Windows Containers och Hyper-V-containrar, som används för att köra Windows-specifika applikationer, hade högre systemkrav än deras Linux-motsvarigheter, vilket påverkade den totala prestandan.

Vid testning av Minikube, en Kubernetes-lokal utvecklingsmiljö, var installationen betydligt mer komplicerad på Windows eftersom den krävde aktivering av Hyper-V och Windows Subsystem for Linux (WSL). Detta ökade konfigurationskomplexiteten och ledde till nätverksrelaterade kompatibilitetsproblem. På Linux fungerade installationen däremot utan problem, vilket underlättade testning och utveckling av containerbaserade applikationer.

Windows-specifika utmaningar och användning av WSL

Under LIA-perioden arbetade jag även med LabWare8, en applikation specifikt utvecklad för Windows. När jag försökte köra den via Wine i Linux uppstod flera problem, inklusive kraschande applikationer, instabilitet och komplexa beroenden. LabWare8 krävde specifika DLL-filer, ODBC-drivrutiner, .NET Framework och Visual C++ Redistributables, vilket gjorde det svårt att återskapa en fungerande miljö i Linux. Krief (*2022*) beskriver hur Windows-miljöer ofta är mer kompatibla med företagsapplikationer som är beroende av Microsoft-teknologier, vilket stämde överens med mina egna observationer.

För att hantera dessa utmaningar använde jag Windows Subsystem for Linux (WSL), vilket visade sig vara en effektiv lösning. Genom WSL kunde jag köra Linux-anpassade DevOps-verktyg samtidigt som jag behöll kompatibiliteten för Windows-specifika applikationer. Detta tillvägagångssätt gjorde det möjligt att hantera både Docker, Kubernetes och Ansible direkt i Windows-miljön, utan att behöva använda separata fysiska eller virtuella Linux-servrar. De Kort (*2016*) argumenterar för att hybridlösningar såsom WSL kan förbättra kompatibiliteten mellan operativsystemen och göra det enklare att hantera blandade miljöer, vilket bekräftades under mina praktiska tester.

Slutsats av observation

Mina erfarenheter från LIA-perioden visar att Linux generellt är det föredragna operativsystemet för DevOps-verktyg som Jenkins, Docker, Kubernetes och Ansible. Dessa verktyg är i grunden utvecklade för Linux och fungerar mer stabilt och effektivt där, utan behov av omfattande anpassningar. Enligt Krief (*2022*) är Linux det mest kostnadseffektiva och flexibla alternativet för DevOps-miljöer, särskilt när det gäller containerhantering och automatisering.

GitHub Actions fungerade dock lika bra på både Linux och Windows, eftersom det är en molnbaserad lösning som inte är direkt beroende av operativsystemets arkitektur.

Vid användning av Windows för DevOps-arbetsflöden uppstod däremot flera kompatibilitetsutmaningar och högre resurskrav, särskilt vid hantering av containrar och CI/CD-verktyg. Windows är dock ett bättre val för organisationer som är beroende av Microsofts teknologi, såsom .NET-applikationer och Active Directory-integration.

Att kombinera Windows med WSL visade sig vara en effektiv hybridlösning, eftersom den möjliggjorde användning av Linux-baserade DevOps-verktyg direkt i Windows, samtidigt som kompatibiliteten med Windows-specifika applikationer bibehölls. Denna strategi kan vara särskilt fördelaktig för organisationer som vill utnyttja styrkorna hos både Linux och Windows, utan att behöva underhålla separata operativsystemmiljöer.

Baserat på dessa observationer rekommenderas att organisationer som vill optimera sina DevOps-processer väljer Linux som huvudplattform, men överväger hybridlösningar som WSL om Windows är ett krav för vissa applikationer eller affärskritiska system.

Diskussion

Resultatet av detta examensarbete belyser hur operativsystemen Linux och Windows påverkar DevOps-arbetsflöden på flera plan. Från början var min utgångspunkt att valet mellan Linux och Windows i huvudsak handlade om att jämföra tekniska egenskaper, där jag förväntade mig att Linux skulle vara överlägset i nästan alla avseenden. Enligt min mening var detta en rimlig förförståelse, eftersom Linux ofta framhålls som förstahandsvalet inom DevOps (*Krief, 2022*). Men när jag nu ser tillbaka på både mina litteraturfynd och mina observationer under LIA-perioden står det klart att frågan är betydligt mer komplex och att teknisk prestanda ibland väger lättare än affärsmässiga och organisatoriska faktorer.

Även om resultaten tydligt visar att Linux erbjuder bättre prestanda, kortare byggtider och naturlig integration med DevOps-verktyg som Kubernetes och Docker, är det inte alltid självklart i praktiken. Jag anser att många organisationer prioriterar affärsmässiga hänsyn som befintlig infrastruktur, leverantörsavtal och personalens befintliga kompetens. Det kan göra en övergång till Linux både mer riskfylld och kostsam.

Enligt min tolkning av resultaten antog jag i början av arbetet att valet mellan Linux och Windows huvudsakligen var en teknisk fråga – vilket operativsystem som erbjöd bäst prestanda, säkerhet och kompatibilitet med DevOps-verktyg. Jag trodde att Linux skulle vara det överlägsna valet i alla kategorier, inte minst eftersom det används i majoriteten av molntjänster (*Krief, 2022*). Att 100 % av världens superdatorer kör Linux var en insikt som först förvånade mig, men som vid närmare eftertanke är logisk med tanke på Linux stabilitet, skalbarhet och effektivitet (*TOP500-listan, 2025*). Samtidigt visar *Statista (2024)* och *TrueList (2024)* att Windows fortfarande är det mest använda operativsystemet på desktops med 73.72 % av marknaden. Detta tyder på att även om Linux dominerar inom moln, DevOps och superdatorer, så är Windows fortfarande en viktig aktör inom företagsmiljöer och personliga datorer.

På samma sätt dominerar Linux också inom servermarknaden. Statistik från *W3Techs (2025)* visar att Linux används i majoriteten av webbservrar globalt. Medan Windows fortfarande är det mest använda operativsystemet på desktops, används Linux på över 55 % av alla kända webbplatser, och andelen ökar ju högre upp i webbplatsrankingen man går. Detta bekräftar att Linux är det föredragna operativsystemet för skalbara och högpresterande servermiljöer, där effektiv resursanvändning och stabil containerhantering är avgörande faktorer.

Teknisk överlägsenhet kontra affärsmässiga faktorer

Resultaten i detta examensarbete visar att Linux är överlägset inom containerhantering, CI/CD-pipelines och prestanda, vilket jag själv förväntade mig redan innan jag påbörjade

studien. Linux är det primära operativsystemet för Kubernetes och andra DevOps-verktyg, och dess modulära arkitektur gör det mer anpassningsbart (*Krief, 2022*). Min reflektion är att dessa egenskaper gör Linux särskilt lämpat för arbetsflöden där hög skalbarhet och resurseffektivitet är centrala.

Samtidigt visar resultatet att Windows inte nödvändigtvis är sämre, utan att det föredras av andra skäl än prestanda. Enligt *Awan & Khan (2022)* väljer många företag Windows på grund av dess användarvänlighet, breda tillgång till kunnig personal och bättre kompatibilitet med populära företagsapplikationer. Windows erbjuder en mer intuitiv användarupplevelse genom sitt GUI, vilket gör det enklare att installera och administrera. Detta är en viktig aspekt, särskilt för organisationer där IT-avdelningen har mindre erfarenhet av Linux.

Ett annat viktigt skäl till att organisationer väljer Windows är programkompatibilitet. Många verksamhetskritiska applikationer, särskilt inom finans, sjukvård och offentlig sektor, är utvecklade för Windows-miljöer och kan inte enkelt migreras till Linux. Detta förklarar i viss mån också att Windows fortfarande dominerar desktop-marknaden globalt med 73.72 % (*Statista, 2024; TrueList, 2024*). Detta innebär att valet av operativsystem inte alltid styrs av teknisk överlägsenhet, utan av befintlig infrastruktur och affärskritiska behov.

Windows har flera styrkor, inklusive enkel installation och konfiguration, ett grafiskt användargränssnitt (GUI) som gör systemet mer tillgängligt, samt bred programkompatibilitet med affärskritiska applikationer. Dessutom finns en större tillgång till kunnig personal som har erfarenhet av Windows-administration, vilket kan minska utbildningskostnader för företag. Å andra sidan har Linux fördelar som öppen källkod, högre säkerhet, bättre stabilitet och effektivare resurshantering, vilket gör det mer lämpligt för servrar och storskaliga DevOps-miljöer (*Lakhera, 2024*). Linux möjliggör också större anpassningsbarhet och är kostnadseffektivt, eftersom många distributioner är gratis.

Hybridlösningar: den mest realistiska vägen framåt

Baserat på studiens resultat och mina egna praktiska erfarenheter har jag insett att en ren Linux- eller Windows-miljö inte alltid är den bästa lösningen. Istället verkar hybridlösningar, där både Linux och Windows används parallellt, vara den mest realistiska strategin för många företag. Jag uppfattar att detta låter organisationer dra nytta av Linux för containerbaserade arbetsflöden och CI/CD-pipelines, samtidigt som befintliga Windows-applikationer kan fortsätta köras utan större risktagande eller merkostnader för licenser och utbildning (*Krief, 2022; Gonzalez, 2017*).

Microsofts förbättrade stöd för Linux genom WSL (Windows Subsystem for Linux) och dess ökade kompatibilitet med Docker och Kubernetes tyder på att gränsen mellan operativsystemen blir alltmer flytande (*de Kort, 2016*). Detta stärker idén om att DevOps-miljöer i framtiden sannolikt kommer att bygga på hybridlösningar där Linux används för prestandakritiska arbetsflöden, medan Windows bibehålls för företagsapplikationer och Active Directory-integration (*Lakhera, 2024*).

Slutsats

Baserat på analysen av Linux och Windows inom DevOps står det klart att operativsystemens roll i automatiserad mjukvaruutveckling och driftsättning inte enbart är en fråga om prestanda och kompatibilitet, utan även om affärsmässiga, organisatoriska och strategiska val. Linux framstår som det tekniskt överlägsna alternativet inom DevOps, med effektivare resurshantering, snabbare CI/CD-pipelines och djupare integration med containerteknologier såsom Docker och Kubernetes (*Krief, 2022; Gonzalez, 2017*). Samtidigt visar forskning att Windows fortfarande är det dominerande operativsystemet inom företagsmiljöer, mycket tack vare dess användarvänlighet, bredare stöd för affärsapplikationer och etablerade Microsoft-ekosystem (*Awan & Khan, 2022; Statista, 2024*).

Denna dynamik tyder på att framtiden för DevOps-miljöer kommer att präglas av hybridlösningar, där företag kombinerar Linux för prestandakritiska arbetsflöden och Windows för kompatibilitet med befintliga system (*de Kort, 2016*). Hybridstrategier gör det möjligt för organisationer att dra nytta av de styrkor som båda operativsystemen erbjuder, utan att behöva göra drastiska förändringar i sin infrastruktur. Microsofts ökade stöd för Linux genom WSL och Azure Kubernetes Service (AKS) är exempel på hur företaget anpassar sig till denna utveckling, vilket gör det enklare för DevOps-team att arbeta plattformoberoende och mer flexibelt (*Lakhera, 2024*).

Ett annat viktigt resultat från denna studie är att operativsystemsvalet även påverkar säkerhetsstrategier och driftkostnader. Linux erbjuder större kontroll över säkerhetspolicyer, snabbare hantering av säkerhetsuppdateringar och lägre licenskostnader, vilket gör det särskilt attraktivt för DevOps-team som arbetar med storskaliga och säkerhetskritiska applikationer (*Kim et al., 2016*). Windows å andra sidan erbjuder en mer centraliserad säkerhetsmodell, vilket kan vara en fördel för företag som är beroende av Active Directory och Microsofts säkerhetstjänster (*de Kort, 2016*).

Slutligen visar denna studie att valet av operativsystem inom DevOps inte är en binär fråga, utan snarare en balansgång mellan tekniska krav, affärsmässiga mål och operativa förutsättningar. Organisationer som vill ligga i framkant inom DevOps bör inte begränsa sig till ett enda operativsystem, utan istället adoptera en flexibel strategi där Linux används för prestandakritiska och containerbaserade arbetsflöden, medan Windows kan bibehållas för specifika affärs- och företagsapplikationer. Denna strategi säkerställer optimal prestanda, hög säkerhet och långsiktig hållbarhet i DevOps-miljöer.

Sammanfattningsvis visar denna studie att Linux är det tekniska förstahandsvalet för DevOps-miljöer, men att Windows förblir ett viktigt alternativ inom företagsmiljöer. Hybridlösningar blir en allt vanligare strategi för att balansera prestanda, kompatibilitet och affärsbehov, vilket gör att företag kan dra nytta av båda operativsystemens styrkor beroende på deras specifika krav.

Källförteckning

Davis, J. & Daniels, K. (2016). *Effective DevOps: Building a Culture of Collaboration, Affinity, and Tooling at Scale.* O'Reilly Media.

de Kort, J. (2016). *DevOps on the Microsoft Stack*. Apress.

Gonzalez, J. (2017). *Implementing Modern DevOps: Enabling IT organizations to deliver faster and smarter.* Packt Publishing.

Kim, G., Humble, J., Debois, P. & Willis, J. (2016). *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. IT Revolution Press.

Krief, M. (2022). *Learning DevOps: Continuously Deliver Better Software, Second Edition.* Packt Publishing.

Lakhera, P. (2024). *Cracking the DevOps Interview: Prepare for DevOps Roles with Real-world Scenarios and Solutions.* BPB Publications.

Awan, M.T. & Khan, K. (2022). "Linux vs. Windows: A Comparison of Two Widely Used Platforms." *Journal of Computer Science and Technology Studies.* DOI: 10.32996/jcsts.2022.4.1.4.

TrueList (2024). "Linux usage statistics 2024." Hämtad från https://truelist.co/blog/linux-statistics/

Statista (2024). "Global desktop operating system market share 2024." Hämtad från https://www.statista.com/statistics/869211/worldwide-software-development-operating-system/

W3Techs (2025). "Usage of operating systems for websites, 2025." Hämtad från https://w3techs.com/technologies/comparison/os-linux,os-windows

Wikipedia (2025). "TOP500 list." Hämtad från https://en.wikipedia.org/wiki/TOP500