DevOps – Essentials

Git/GitHub

**Branches**

Branches er kopier av den originale koden. Den originale koden ligger i branchen kalt «main» eller «master». Man kan ha egne brancher hvor man tester ut eller utvikler ny funksjonalitet til programvaren, uten at det påvirker main-koden. Når man har utviklet noe på en branch og man mener det er på tide å implementere dette i den originale koden, kan man sammenflette branchen man har jobbet på med main-branch. Dette kalles merging. Det finnes forskjellige brancher som er vanlige å komme over:

* **Main/Master**: Grenen som inneholder koden som blir sluppet til kunden. Ved oppdatering av denne betyr som regel at man går fra f.eks 3.0.0 til 4.0.0 hvis det er snakk om store endringer, 3.0.0 til 3.1.0 hvis det er mindre endringer som er gjort, eller 3.0.0 til 3.0.1 hvis det er snakk om bugfix og patches,
* **Hotfix**: Brukes til å patche bugs som er enkle å gjøre noe med. Dette er da altså små feil i koden som enkelt kan endres uten særlig mye arbeid,
* **Develop**: Brukes for å utvide ny funksjonalitet. Dette er kode som når den flettes med main ofte gjør at man får en ny versjon av programvaren, framfor en bugfix.
* **Feature**: Brukes for å utvikle en spesifikk ny funksjon. Slike branches merges som regel med develop-branchen framfor direkte inn i main.

Hvordan skifte branch:

*git checkout [branchname]* (Lager en ny branch med det navnet dersom den ikke finnes)

Hvordan lage en feature-branch til en spesifikk feature:

*git checkout feature*

*git checkout -b feature/[branchname]*

*git checkout [branchname]*

**Merging**

Å merge brancher betyr å sammenflette dem. Har du utviklet på en develop-branch of ønsker implementere det du har skrevet inn i original-koden i main-branchen. Så kan du merge dem.

Hvordan merge branch:

*git checkout [branch du ønsker å få ny kode til]*

*git merge [branch med den nye koden]*

Hvordan sjekke merge-aktivitet

*git log –graph*

**Stashing**

En stashing er å lagre ikke-comitted endringer uten at det forsvinner hvis du vil jobbe med en annen gren. Dersom man har skrevet på en branch, men ikke ønsker å comitte den nye koden helt ennå, så kan man bruke git stash for å lagre denne. Det brukes som midlertidig lagring av kode.

Hvordan stashe

*git stash*

Hvordan flytte stash

*git checkout[target branch]*

*git pop [stashname]*

Hvordan implementere stashen

*git apply [stashname]*

Hvordan fjerne stashen

*git remove [stashname]*

Porter

Porter er addressepunkter i en logisk forbindelse mellom to programmer som kommuniserer. Datakommunikasjonen foregår ved hjelp av pakker. En pakke inneholder data, samt metainformasjon om hvor og til hvem pakken skal. En port angis av et tall mellom 0 og 65525 i metainformasjonen, som forteller transportlaget hvilken prosess pakken er ment for.

På en tjenermaskin er det typisk flere prosesser som til enhver tid står klar til å ta imot forbindelser. Prosessene skaper da en socket, som identifiseres ved hjelp av kombinasjonen av en IP-adresse og et portnummer. En socket «lytter» så etter forbindelser. Transportlaget gir pakker videre til den riktige prosessen på applikasjonslaget basert på portnummeret i den innkommende forespørselen. Portnummer lavere enn 1024 følger standariserte regler for hvilke typer tjenester som bruker dem. Dette tilordnes av Internet Assigned Numbers Authority.

Noen port-standarder:

* 23 – telnet
* 25 – SMTP (Simple Mail Transfer Protocol)
* 80 – http (Hypertext Transfer Protocol)
* 20 – FTP Datatrafikk
* 21 – FTP Kontrollmekanismer
* 22 – SSH (Secure Shell)
* 38 – Route Access Protocol
* 42 – Host Name Server Protocol
* 43 – WHOIS Protocol
* 66 – Oracle SQL \*NET
* 443 – https (Hypertext Transfer Protocol Secure)

Protokoller

En protokoll er et set med regler som bestemmer hvordan data blir overført mellom forskjellige enheter på samme nettverk. De lar tilkoblede enheter å kommunisere med hverandre selv om de skulle ha forskjeller i interne prosesser, struktur eller design. Nettverkprotokoller er grunnen til at man enkelt kan kommunisere med andre rundt om på jorden, og spiller derfor en viktig rolle i moderne digital kommunikasjon.

**HTTP**

En nettverksprotokoll som blir brukt for å overføre det meste av informasjon på internett. HTTP styrer kommunikasjonen mellom en http-klient, typisk en nettleser og en http-tjener. Dette blir gjort ved at klienten sender en forespørsel til tjeneren, og denne vil da gi en statuslinje som «HTTP/1.1 200 OK» og sin egen melding. Meldingen inneholder som regel det som blir spurt om, for eksempel en bestemt datafil, men det kan også være en feilmelding eller annen informasjon. Protokollen har standardport 80.

**HTTPS**

En sikrere utgave av HTTP, som er kommunikasjonsprotokollen til World Wide Web. Den tilbyt autentisering og kryptering av kommunikasjon i forbindelse med elektronisk handel ved hjelp av digitale sertifikater. En HTTPS-sesjon blir krypter enten ved bruk av SSL-protokollen (Secure Socket Layer) eller TLS-protokollen (Transport Layer Security), og tilbyr beskyttelse mot tyvlytting og at noen endrer på de sendte dataene.

Graden av beskyttelse avhenger av hvor korrekt implementasjonen av HTTPS er i nettleseren, tjeneren og hvilken krypteringsalgoritme som er brukt.

HTTPS er ingen egen protokoll, men heller en kombinasjon av vanlig HTTP over en SSL- eller TLS-konling. Dette skaper beskyttelse mot avlytting og mellomleddsangrep.

Brannmurer

En brannmur er en nettverkssikkerhets enhet som overvåker og filtrer inkommende og utgående nettverkstrafikk basert på en organisasjons tidligere fastsatte retningslinjer. I grunnleggende trekk så er en brannmur den barrieren som sitter mellom et privat internt nettverk og det offentlige internettet. Brannmurens hovedoppgave er å slippe forbi ikke-truende trafikk, og stoppe farlig trafikk.

Det finnes 5 forskjellige typer brannmurer.

* **Packet filtering firewall** – Opererer på møtepunkter hvor enheter som rutere og brytere gjør deres arbeid. De sammenligner inn og utgående pakker mot et sett med kriterier, som de tillatte IP-adressene, pakketyper, portnumer og andre aspekter av pakke-protokoller.
* **Circuit-level gateway** – En brannmur som i steden for å inspisere selve datapakkene, så ser den på nettverksprotokoll sesjons initierings-meldinger på tvers av nettverket slik de er etablert mellom de lokale og de eksterne vertene for å avgjøre hvorvidt en sesjon som blir initiert er gyldig, og hvorvidt det eksterne systemet kan bli stolt på.
* **Application-level gateway** – Dette er teknisk sett en proxy of blir av og til kalt proxy-brannmur. Det fungerer som det eneste inngangs- og utganspunktet fra et nettverk. Slike brannmurer filtrerer datapakker ikke bare etter hvilke funksjon de har, som er definert av destinasjonsporten, men også etter karakteristikken deres, som HTTP forespørselsmeldingen.
* **Statefull inspection firewall** – Slike enheter inspiserer ikke bare hver datapakke, men den holder også telling med hvorvidt pakken er en del av en etablert TCP eller andre nettverkssesjoner. Dette tilbyt mere sikkerthet enn å kun filtrere bakker eller med circuit monitoring alene, men det påvirker nettverkets ytelse i større grad.
* **Next-generation firewall** – De kombinerer ofte pakkeinspeksjon med tilstands-inspeksjon samt at de også tar med diverse former for dyp pakkeinspeksjon (DPI), i tillegg til andre nettverkssikkerhets systemer som IDS/IPS, malware filtrering og antivirus.

Proxy-servere

En proxy-server er et datasystem eller en ruter som fungerer som en videresendings-enhet mellom en klient og en server. Den hjelper med å forhindre en angriper fra å invadere et privat nettverk og er et av flere verktøy som brukes til å bygge en brannmur. Ordet proxy betyr «å handle på vegne av en annen», og en proxy-server handler på vegne av brukeren. Alle forespørseler til internettet går gjennom proxy-serveren først, som evaluerer forespørselen og videresender den til internettet. På samme måte går responser gjennom proxy-serveren og deretter til brukeren.

Proxy-serveren er en to-hjemmet vert med to nettverks IP-addresser. IP-addressen på den utgående siden er den internettet kan se. Proxyer er ofte brukt i kombinasjon med nettverks addresse oversettelse (NAT), som gjemmer brukerens IP-addresse på det interne nettverket også. Proxy-servere kan også cache nettsider, slik at neste forespørsel for den nettsiden kan bli hentet opp igjen mye raskere.

* **Anonyme proxyer** – Servere som lar brukere surfe på nettet og holde IP-addressen deres private. Det trengs ikke nødvendigvis være en proxy, men det deler likheter med internet email (SMTP) og Usenet new system (NNTP), siden meldinger blir videresendt fra senderen til mottakeren.
* **Applikasjonsnivå og Circuitnivå proxy** – Proxyer som er dedikert til spesifikt innhold som HTTP og FTP. Circuitnivå proxyer vil i motsetning gi støtte til alle applikasjoner.
* **Framover og reversproxyer** – Dette er servere som skjuler detaljene om klienten for serveren. Den kan også bli implementert for nettsiden slik at den skjuler dets detaljer for klientene.

Docker

En docker brukes for å kunne teste en nettside lokalt framfor å implementere den på en ekstern server. Dockere er programmer som laster opp programmkoden pakker det til en såkalt container, som deretter blir gjort tilgjengelig på localhost. Standard porten for localhost er 8000, altså finner man nettsiden på <http://localhost:8000> eller <https://localhost:8000>. Dette er en løsning for et testmiljø som ikke er på det offisielle nettet, men som heller blir kjørt på den lokale maskinen. For å dra nytte av docker trenger man applikasjonen docker, samt docker compose, og en mappe med docker-configurering for nettsiden man skal kjøre. Hele simuleringen kan settes i gang med bash-kommandoer i terminalen.

Docker tillater at man kan kjøre forskjellige versjoner av en nettside samtidig på forskjellige, egendefinerte porter. Det har også den praktiske fordelen at nettsiden oppdaterer seg i sann tid dersom man endrer på koden til nettsiden.

En ting som er viktig å vite er om nettsiden kjører med http eller https som protokoll, ettersom man kun kan nå nettsiden gjennom localhost med den tilsvarende protokollen.

SSH

SSH, eller Secure Shell, er en protokoll som bruker kryptering for å sikre tilkoblingen mellom en klient of en server. Den tillater at person til å få tilgang til en annen datamaskin, som oftest en server, og utføre handlinger med den maskinen slik som man ville gjort med maskinen dersom man var fysisk til stede. Man kommuniserer med den eksterne maskinen gjennom terminalen, og kan bruke alle kommandoer man ellers ville ha brukt derfra.

Prosessen bak tilkoblingen med SSH mellom klient og server kan deles inn i 4 punkter:

1. Klienten initierer tilkoblingen ved å kontakte serveren
2. Serveren responderer med å sende serverens offentlige nøkkel
3. Serveren behandler parametere gitt fra klienten og åpner en sikker kanal
4. Klienten kan logge inn på serveren

For å koble seg på en server med SSH må man åpne terminalen og kjøre følgende kommando:

*ssh [brukernavn]@[serveraddresse]*

*[passord]*

Eks:

*ssh* [*jorgfi@iirmoodle.it.ntnu.no*](mailto:jorgfi@iirmoodle.it.ntnu.no)

*[passord]*

Linux filsystem

Alle filer og mapper i et Linux-system er strukturert på en spesiell måte slik at det skal være oversiktlig. Dette trengs slik at en bruker kan ha en viss kontroll over hvor executables havner når man installerer noe, for å få bedre dokumentasjon og mye mer. Filsystemet har en bestemt mappestruktur, hvor alle mappene har et navn som beskriver innholdet i dem. Her er en liste over mye brukte mappenavn i linux sitt filsystem:

* /**root**: Dette er hoved-mappen til administratoren av systemet. Denne er separert fra resten av brukerens mapper ettersom at endringer på ting i denne mappen kan være fatalt for systemet.
* /**bin**: Inneholder binaries, altså kompilert kode som tillater et program å bli installert uten å måtte kompilere kildekoden. Dette er ofte applikasjoner og programmer man kan kjøre, som for eksempel ls programmet for å liste opp filer, og programmer for å opprette og fjerne filer og mapper. Man finner flere bin-mapper i andre deler av filsystem-treet også-
* /**boot**: Inneholder filer som er nødvendig for å starte opp systemet. Disse blir altså kjørt i BIOS og er helt nødvendige for at systemet kan starte riktig. Denne mappen bør helst ikke røres, ettersom at feil i denne kan gjøre at hele operativsystemet maskinen kjører ikke kan startes.
* /**dev**: Inneholder device filer. Mange av disse er generert i boot-fasen eller etterhvert som hendelser oppstår. For eksempel, hvis du plugger inn en USB minnepinne eller kobler til et nytt webkamera, så vil en ny devuce oppføring dukke opp i dev-mappa.
* /home: Dette er mappen hvor man finner brukernes personlige mapper. Man kan f.eks. ha stien /home/jorgen/bilder.
* /**lib**: Dette er hvor biblioteker ligger. Biblioteker er filer som inneholder kode som applikasjonene dine kan bruke. De inneholder utdrag av kode som applikasjoner bruker til å tegne faner og vinduer på skjermen, sende filer til harddisken og masse mer.
* /**media:** Dette er hvor ekstern lagring automatisk vil bli plassert når du konler det til og prøer å få tilgang til det. Dette kan f.eks. være SD-kort, minnepinner eller harddisker.
* /mnt: Ikke ofte brukt lenger, men tidligere var det hvor man manuellt kunne konle lagringsenheter.
* /**opt**: Dette er hvor kompilert kode (kode som du selv skriver) havner. Applikasjoner ender opp i /opt/bin mens biblioteker havner i /opt/lib.
* /**proc**: En virtuell mappe slik som dev. Den inneholder informasjon om maskinen, slik som informasjon om CPU og kernelen som Linux systemet kjører på.
* /**run**: Systemprosesser bruker denne mappen til å lagre midlertidig data for deres egne diverse nødvendigheter. Dette er en mappe man helst ikke burde røre.
* /**usr**: Inneholder ofte en blanding av mapper som igjen inneholder applikasjoner, biblioteker, dokumentasjon, bakgrunnsbulder, ikoner og masse annet som trengs å bli delt mellom applikasjoner og tjenester.
* /**srv**: Inneholder data for servere. Dersom du kjører en web-server fra Linux, så vil HTML-filer f.eks. gå i /srv/http eller /srv/www.
* /**sys**: Inneholder inromasjon om enheter tilkoblet til maskinen din. I noen tilfeller kan man manipulere disse enhetene, det går an f.eks. å endre lysstyrken på skjermen gjennom å finne den riktige filen i sys-mappen, og endre verdien i den. Det lureste er nok fremdeles å holde seg unna denne mappen uansett.
* /**tmp**: Inneholder midlertidige filer, som regel plassert der av applikasjoner som kjører på det tidspunktet. Filene og mappene der inne kan ofte inneholde data som en applikasjon ikke nødvendigvis trenger her og nå, men som kanskje kan være nødvendig senere. Man kan bruke denne mappen for å lagre sine egne midlertidige filer.
* /**var**: Dette er en mappe hvor innholdet som oftest er variabelt, altså det skiftes ofte. Det kan være logger for forskjellige prosesser og hendelser i datamaskinen.

Linux terminalkommandoer

* **pwd**: Path of working directory. Viser hvor i datamaskinen man er,
* **cd**: Change directory. Skifter mellom mapper.
* **ls**: List. Lister opp filer og mapper i den plasseringen man er i.
* **cat**: Concatenate: Brukes for å lage en fil, slå sammen filer, eller endre filnavn.
* **cp**: Copy. Brukes for å kopiere filer fra en mappe til en annen.
* **mv**: Move. Brukes til å flytte filer, men kan også brukes til å gi dem nye navn.
* **mkdir**: Make new directory. Lager en ny mappe.
* **rmdir**: Remove directory. Sletter en mappe så lenge den er tom.
* **rm**: Remove. Sletter en mappe og alt den inneholder.
* **touch**: Lager en ny fil som mofidiseres og lagres gjennom terminalens kommandolinje.
* **locate**: Brukes for å lokalisere en fil. Akkurat som å søke på noe.
* **find**: Brukes på samme måte som locate, men den brukes inni en bestemt søkeplass.
* **grep**: Brukes til å søke gjennom all teksten i en gitt fil.
* **sudo**: Gir deg autoriseringen til å kjøre kommandoer som administrator.
* **df**: Skriver en rapport på systemets lagringsplass.
* **du**: Disk usage. Viser hvor mye lagringsplass en fil eller mappe bruker.
* **head**: Brukes for å kunne se de første linjene i en tekstfil.
* **tail**: Samme som head, bare for å kunne se de siste linjene isteden.
* **diff**: Sammenligner innholdet mellom to filer, linje for linje.
* **tar**: For å komprimere en mappe til f.eks. en .zip fil.
* **chmod**: Brukes for å endre read, write og execute tillatelser til filer og mapper.
* **chown**: Brukes til å skifte eller overføre eierskapet til en spesiell fil til en annen bruker.
* **jobs**: Viser alle prossesser som foregår i shellet på det tidspunktet.
* **kill**: Brukes fir å terminere et program dersom det ikke responderer.
* **ping**: Brukes for å sjekke tilkoblingsstatus til en server.
* **wget**: Brukes for å laste ned filer fra internett.
* **uname**: Skriver ut detaljert informasjon om systemet som OS, maskinnavn og kernel.
* **top**: Tilsvarer task manager. Viser pågående prosesser og deres bruk av CPU.
* **history**: Skriver ut loggen av kommandoer som du har kjørt tidligere.
* **man**: Viser manualen til de forskjellige kommandoene.
* **echo**: Brukes for å flytte data inn til en fil.
* **zip**: Pakker sammen en mappe til en .zip-fil.
* **unzip**: Pakker opp en .zip-fil.
* **hostname**: Viser tilkoblet nettverks sitt vertsnavn og ip-addresse.
* **useradd**: Brukes til å lage en ny bruker på systemet.
* **userdel**: Brukes til å fjerne en bruker på systemet.