Øving 2: DNS, Epost og Web

# Del 1: DNS

Det er flere siktemål med denne DNS-øvingen.

* Det første målet er å bli kjent med konfigurering på din egen PC. Du skal finne adressen til din lokale navnetjener og hva den heter. Her brukes kommandovindu på PC (ledetekst eller CMD på Windows).
* Videre skal du gjøre spesifikke DNS-oppslag hvor ett domenenavn brukes for ulike tjenester, web og epost, dette for å undersøke typefeltet i Ressursrecord.
* Til slutt skal Wireshark brukes til å finne levetiden for svar på DNS-oppslag, dvs hvor lenge de kan mellomlagres i cache.

1. **Hva heter din lokale navnetjener? Vi søker etter domenenavnet.**

Steg1: Finn hvilken/hvilke DNS-tjenere din PC er konfigurert med når du er *koplet på campusnettet.*

Start Ledetekst/CMD og gi kommando: *CMD>ipconfig /all.* Merk: Uten tilføyelsen */all* vil bare de mest sentrale parameterne vises, og ikke DNS-adresser.

**SVAR:** (Bruk gjerne skjermklipp)

Steg 2: Når dette DNS-oppslaget gjøres med tilkopling til NTNU-nettet vil det vises at jeg kan velge blant *fire* ulike IP-adresser til lokale navnetjenere, to IPv4 og to IPv6. Bruk *CMD>nslookup <IP>* til å finne navnet til den første IPv4-adressen.

SVAR:

Merk: Et tjenernavn kan være registrert både med IPv4- og IPv6 adresse men allikevel være på samme maskin.

1. **Finn IP-adressen til eposttjener for NTNU**

Domenenavnet til NTNU er ntnu.no. Dette navnet brukes jo blant annet for adressering i epost og web, og disse tjenestene er plassert på ulike maskiner av kapasitetsmessige årsaker. Dermed må et domenenavn kunne ha flere IP-adresser, avhengig av hvilke tjenester man bruker. Hvert ressursrecord i DNS har derfor angitt hvilken tjenestetype dette gjelder for.

Et forbehold: NTNU kjører store tjenester fordelt på flere serverer for redundans og lastbalansering. Når vi i oppgaven her søker på adresser og navn på tjenere så viser ikke dette hele bildet av NTNUs serverpark og hvordan de er knyttet sammen. Svarene kan derfor variere fra en PC til en annen.

**Steg 1**: Søke med nslookup på bestemt type ressursrecord (RR)

Det er 15-20 talls tjenestetyper av RR, de vanligste er

* A (IPv4)
* AAAA (IPv6)
* MX (Mail Exchange)
* NS (Name Server)
* CNAME (Alias, et navn som peker til et annet navn)
* PTR (Pointer)

Sett MX (for «Mail eXchange») og søk på ntnu.no i ledeteksten:

Et bilde som inneholder tekst, Font, skjermbilde

Automatisk generert beskrivelse

Når man søker på ntnu.no med type MX vil man få *navnet* på en ny tjener til svar. Hva er navnet til NTNUs eposttjener?

SVAR:

Nå må man gjøre et nytt oppslag for å spørre etter IP-adresse for denne navngitte tjeneren. Endre til «set type= AAAA» for å finne IPv6 til eposttjener.

Hva IPv6 til NTNUs eposttjener?

SVAR:

1. **Finn IP-adressen og det «egentlige» navnet til tjeneren for «ntnu.no»**

Gjør et nslookup-søk med type AAAA på «ntnu.no». Kjør så et nytt søk på denne IPv6-adressen, da får man navnet på tjeneren som er knyttet til denne adressen. Merk: Her vil man finne at tjeneren har et annet navn enn «ntnu.no». Det betyr at *ntnu.no* er «merkenavnet» på Internett, men at de ulike tjenerne bak dette merkenavnet har sine egne navn.

Hva er IPv6 til ntnu.no, og hva er navnet til denne tjeneren?

SVAR:

# Del 2 - EPOST

Hensikten med denne øvingen er å vise hvor enkelt det er å kople seg på en epost-tjener for å sende en epost, og med det vise hvordan SMTP-dialogen mellom klient og tjener fungerer.

Det skal kjøres en manuell Telnet-sesjon mot eposttjeneren på NTNU som om man selv var en ordinær epostklient. Denne sesjonen er ikke ment å fullføres, altså sende en epost, men hvis man allikevel forsøker på det så skal det uansett IKKE benyttes annen avsenderadresse enn sin egen legitime epostadresse. Det vil i så fall kunne gi meget uønskede konsekvenser for avsender.

1. Åpne en telnet-sesjon: CMD>*telnet*
2. Opprett forbindelse mot NTNUs eposttjener: *o mx.ntnu.no 25*(Betyr open TCP mot port 25 som SMTP lytter på)
3. Det opprettes en TCP-sesjon og Tjeneren presenterer seg med kode 220, tjenesten er klar
4. Neste steg i dialogen er at klienten skal presentere seg, men for denne testen hopper man over det og skriver bare <*mail from:xx*>
5. Tjeneren gir da en feilmelding og forteller deg hva du egentlig skulle ha sendt.

Oppgave: Hva er det SMTP svarer i linje 6?

**SVAR:**

Dette er en utmerket demonstrasjon av dialogen mellom klient og tjener, og hvor enkelt det er å kople seg på en tjener. Det kreves ingen pålogging. Tilkoplingen kan riktignok krypteres med TLS ved å kople seg på port 587, men det er bare for å ha konfidensiell forbindelse mellom klient og tjener.

I teknisk forstand er det strengt tatt ikke noe som heter «falske eposter», men vi har epost med uønsket innhold, falsk avsender osv. Derfor gjør epost-tjenere en stor jobb med å filtrere innholdet på svartelistet IP-adresser og nøkkeltekst i innholdet. For å gi en indikasjon på omfanget så får NTNU omtrent 1 million eposter daglig, og litt forsiktig anslått er ca 80% av dette uønsket epost.

# Del 3 - Webtjenesten og HTTP

Hensikten med denne øvingen å forstå hvordan HTTP fungerer, nærmere bestemt hvordan headerlinjer styrer dialogen mellom klient og tjener. Dette skal illustreres med Wireshark pakkefangst av HTTP/ 1.1. Vi kan ikke gjøre en tilsvarende undersøkelse av **HTTPS** fordi den har innhold kryptert med TLS. TLS vil bli undersøkt nærmere i neste øving.

1. **Innholdet av HTTP-meldinger**

I første del av øvingen benyttes det en ferdig Wireshark-pakkefangst fra et oppslag på *datakom.no* (hentes på Blackboard). Studer innholdet i pakke nr 13, respons fra HTTP-tjener. Denne pakken er siste del i overføringen av hele index.html fila

Når ble denne pakkefangsten tatt?

* **Svar**:

Når var denne index-fila sist endret?

* Svar:

Hvor mange TCP-segmenter ble benyttet for å laste ned index-fila, og hvor store var de?

* **Svar**:

I pakke nr 13 viser Wireshark svaret fra tjener på klientens tidligerer request i linje 5 om å bruke *vedvarende forbindelse*. Formatet kalles «kolonseparert navn/verdi-par». Hvordan er denne utvekslingen?

* **Svar:**   
  Klient spurte (i linje 5): Connection: \_\_\_\_

Tjener svarer: Keep-Alive: \_\_\_

Og dette svaret betyr: \_\_\_\_

HTTP gjøres tilstandsfull ved bruk av cookies. Hva betyr det at en forbindelse er tilstandsfull? Hvordan brukes cookies for å oppnå dette i dialogen mellom klient og tjener?

* **Svar:**

Det er unødvendig å laste ned uendrede filer som allerede finnes i mellomlager. Hvordan oppnår HTTP dette i dialog mellom klient og tjener?

Merk1: Dersom man sletter mellomlagrede filer så kan man kunne bruke Wireshark for å studere TCP-segmenter for nedlasting av store filer.

Merk 2: PHP-genererte html-filer vil alltid lastes ned fordi de er dynamisk genererte, alltid nye versjoner.

* **Svar:**

Finn en pakke med HTTP-GET. Høyreklikk på denne pakken og velg Follow/HTTP-stream. Hva skjer?

**Svar**:

Pakke 1 og 2 inneholder begge et TCP-SYN-flagg. Observer at ulike TCP-portnummer knyttes til disse. Hvilken effekt har dette for webtjenesten?

**Svar**: