IDATT2104 – DKO 1 – m/ revidert TLS

Utførelse: Arbeidet i denne oppgaven kan utføres alene eller i frivillig sammensatte grupper på maksimalt 3 deltakere, det samme som for programmeringsøvinger. Oppgaven er tellende på sluttkarakter. Alle innleveringer skal derfor være unike for gruppene. Det betyr at tekstlige forklaringer, pakkefangst og skjermklipp skal utføres selvstendig i gruppen og ikke deles med andre.

Innlevering: Oppgaven leveres i Blackboard innen oppgitt frist. Hver enkelt laster opp sin besvarelse, også de som har arbeidet i grupper. Derfor må det skrives inn hvem som har deltatt på arbeidet, både i selve oppgaven og med merknad i BB.

Oppgave 1: Wireshark og DNS (10%)

Hensikt: Forstå virkemåten til lokal navnetjener

Gjør et navneoppslag på *datakom.no* med kommando *nslookup* og fang pakker med Wireshark.

a) Vis med skjermklipp fra kommandovindu navnet på din lokale navnetjener og dens IP-adresse, samt svar på navneoppslaget. Marker på figuren hva som er hva.

b) Vis deretter konfigurering av egen PC med **ipconfig /all** og ta et skjermklipp som viser full beskrivelse av nettverkskortet som er aktivt i kommunikasjon. Hvordan passer IP-adressen til DNS-tjener med svaret i oppgave a)?

c) Ta skjermklipp fra Wireshark av Answers-seksjonen i DNS-svaret. Hva er verdiene på Type og Levetid, og hva brukes denne informasjonen til?

d) PC-en utfører flere liknende navneoppslag som følge av oppslag på datakom.no. Hva er det disse andre oppslagene gjelder?

Oppgave 2: Wireshark og HTTP (15%)

Hensikt: Forstå hensikten og virkningen av to utvalgte headerlinjer i HTTP

Gjør et oppslag på datakom.no med nettleser. Start Wireshark og gjøre et nytt oppslag på datakom.no. Finn HTTP-headerlinjer som brukes for følgende funksjoner og svar på spørsmålene:

a) Vedvarende forbindelser: Hvilken headerlinje sender klient, og hva er svaret fra tjener? Vis med skjermklipp fra Wireshark hvor lang tid det går mellom siste svar fra webtjener og til det kommer kommer en TCP-pakke med FIN-flagget satt.

b) Bruk av lokalt mellomlager på PC: Hvilken headerlinje sender klient for denne funksjonen, og hva er svaret fra tjener?

Oppgave 3: Wireshark og TCP (20%)

Hensikt: Forstå hvordan TCP kan tilby en pålitelig overføringstjeneste

Gjør et nytt oppslag på Datakom.no slik at *alle* filer for HTML-siden lastes ned i sin helhet.
Hint 1: Siden du allerede har gjort oppslag tidligere må du slette mellomlagrede Internett-filer på PC. CTRL+SHIFT+DEL fungerer bra for nettlesere utenom Safari for macOS.
Hint 2: Nettlesere oppretter som regel parallelle tilkoplinger til webtjener for raskere nedlasting av websider. Det er tilstrekkelig å følge «hovedstrømmen» for å kunne svare på spørsmålene under. Velg Wireshark «Follow TCP-stream» på den første GET requesten som er for indeks-fila.

3A. Oppkopling 3-way Handshake

Vis skjermklipp av pakkefangst for de tre pakkene og fyll ut tabellen for feltene:
- Source port (SRC) og Destination port (DST) og - Flaggene SYN og/eller ACK
- Relativt sequence number (SEQNR) og acknowledgement number (ACKNR)

SRC	→	DST	FLAGG	SEQNR	ACKNR
	→	80	SYN		
	→				
	→				

Hvilket sekvensnummer vil alltid første byte med nyttelast ha, og hvilket kvitteringsnummer vil vi få tilbake dersom nyttelasten bare er 1 byte?

3B. Dataoverføring

Vis et skjermklipp som viser aktuelle pakker for overføring av indeks-fila og fyll inn tabellen under med noe av innholdet i pakkene som sendes frem og tilbake mellom nettleserklient og webtjener. Avslutt med et kort sammendrag av hva som er utført i weboppslaget.

Hint: Noen segmenter vil inneholde HTTP nyttelast, andre vil være bare TCP-kvitteringer. Tabellen skal vise følgende innhold for hver pakkene:

- Nummer, Avsender (Klient eller tjener)
- DST-port (mottaker portnummer)
- SEQNR (Sekvensnummer, alltid i senderetningen)
- ACKNR (Kvitteringsnummer tilbake for mottatte data)
- TCP PAYLOAD (Størrelse på nyttelasten fra applikasjonslaget)
- Merknad (Si noe om hva pakken inneholder)

NR	Avsender	DST-port	SEQNR	ACKNR	TCP PAYLOAD	Merknad
1	Klient	80	1	1		GET-request til webtjener
2	Webtjener					

Oppgave 4: Nettverk subnetting (20%)

Hensikt med oppgaven: Forstå prinsippet med subnetting.

Du har gitt et IPv4-nettverk a.b.c.0/24. Dette skal deles i subnett med antatt følgende behov:

A. 80 ansatte
B. 10 tjenermaskiner
C. 20 gjestebrukere

Subnettene skal disponeres slik at de får færrest mulig ledige adresser ut over antatt behov, og lavest mulige adresser skal benyttes innenfor a.b.c.0/24-nettet. Sett opp en nettverkplan som viser tildelte og ledige subnett. Illustrer gjerne med figur. Tabelln inneholder:

- CIDR: Antall bit i nettmasken
- Min Host og Max host: laveste og høyeste tilgjengelige IP-adresse for noder (hosts) tilkoplet subnettet
- Broadcast: IP-broadcast for subnettet
- Tilgjengelige: Alle tilgjengelige IP-adresse for noder/hostadresser på subnettet
- Alle subnett må også ha egen IP-adresse for default gateway/ruter

Nett	Nett-adresse	CIDR	Min host	Max host	Broadcast	Tilgjen gelige	Behov	Ledig

De to neste oppgavene består i å dokumentere de to første programmeringsøvingene. Man skal dokumentere hvordan forbindelsen mellom klient og tjener etableres og hvilke data som utveksles mellom partene.

Oppgave 5: Dokumentasjon av Programmeringsøving 1 – Tråder og socketprogrammering (15%)

Øvingens Del 1: Enkel tjener/klient (kalkulator)

a) Vis aktuelle programlinjer for utveksling av data mellom klient og tjener og beskriv hvordan de fungerer. Ta et skjermklipp av pakkefangsten for pakker som sendes mellom applikasjonene. Vis også meldingsinnholdet i klartekst med «Follow TCP-Stream».

Øvingens Del 2: Enkel webtjener (returnere HTTP-header)

a) Vis med skjermklipp pakkeutveksling mellom klient og tjener, og vis i tillegg meldingsinnholdet i klartekst med «Follow TCP-stream» i Wireshark. Angi spesielt hvilke kvitteringsmeldinger som sendes mellom klient og tjener.

b) Web-tjener skal avslutte forbindelsen etter at HTTP-svar er sendt til nettleseren. Dokumenter tilhørende programkode for dette og vis aktuelle pakker med skjermklipp. Beskriv prosessen for nedkopling.

Oppgave 6: Dokumentasjon av programmeringsøving 2 – UDP, TLS og ASIO (20%)

Øvingens Del 1: Kalkulator med UDP
Hensikten med denne oppgaven: Forstå faktiske forskjeller mellom TCP og UDP

a) Vis aktuelle programlinjer for kommunikasjon og tilhørende skjermklipp av pakkefangst som viser *alle* pakker som utveksles mellom klient og tjener for en enkelt kalkulasjon. Sammenlikne denne kommunikasjonen med kalkulatoroppgaven i øving 1 og diskuter fordeler og ulemper ved å velge henholdsvis TCP eller UDP

Øvingens Del 2: Etablere sikker kommunikasjon med TLS

Hensikten med denne oppgaven: Kunne identifisere stegene i oppkopling mellom klient og tjener på en TLS-forbindelse.

a) Beskriv hvordan klient og tjener settes opp i programmeringsøvingen for å støtte TLS-kryptering, inkludert bruk av sertifikat. Hvor lagres privat nøkkel?

b) Steg 1 oppkopling: Client Hello
Det første som skjer er at partene må enes om hvilke algoritmer de skal bruke for kryptering. Og det er ikke bare en enkelt algoritme som trengs, men en hel suite. Vis et skjermklipp av de siffersuitene (Ciphersuits) som klienten tilbyr tjener å velge blant.

c) Steg 2A: Server Hello, siffersuite
Vis skjermklipp av den bestemte siffersuiten som tjener velger. Denne må nødvendigvis være med på listen fra klienten. *Til informasjon: Hva ligger i siffersuiten? Her er en forklaring på hvordan man kan tolke den «kryptiske» strengen: https://scotthelme.co.uk/https-cheat-sheet/*

d) Steg 2B: Server Hello, sertifikat
Tjener må sende sitt sertifikat for å autentiserer seg når nye sesjoner settes opp. Vis hva innholdet i serversertifikatet er for følgende to elementer:

- «Issuer/common name», altså utsteder
- «subject/common name», altså innehaver

e) Steg 3: Nøkkelutveksling

Partene genererer tall som brukes som nøkler for videre symmetrisk kryptert kommunikasjon. For dette brukes Diffie-Hellman algoritmen. DH etablerer en delt hemmelighet over en usikker kanal. Denne delte hemmeligheten kan brukes direkte som en (symmetrisk) nøkkel, eller til å avlede nye symmetriske nøkler (for eksempel et dobbelt sett av nøkler, en for hver senderetning).

DH har også flere settinger. Disse parameterne angis i seksjonen «Handshake Protocol/Server Key Exchange». Vis med skjermklipp hva som er valgte server- og klient-parametere for DH-algoritmen.

Oppgave 6 - Revisjon TLS

Det er to versjoner av TLS som brukes om hverandre i dag. Vi har den eldre TLS 1.2 som fortsatt har stor utbredelse og den nyere TLS 1.3 som ble definert i RFC 8446 i 2018.

Oppgaveteksten om TLS-programmering var basert på TLS 1.2. Denne versjonen av TLS blir benyttet dersom man kompilerer prog-øvingen med Java 8, har jeg fått bekreftet. Her skjer TLS-handshake over flere meldinger, og hvor den etterspurte informasjonen i Oppgave 6 kan finnes med Wireshark.

Dersom man benytter Java 15.0.2 vil prog-øvingen kjøre på TLS 1.3. Oppgave 6 får derfor en justering for de som bruker TLS 1.3:

Revidert oppgave 6 for TLS 1.3

Fra programmeringsøvingen

a) Beskriv hvordan klient og tjener settes opp i programmeringsøvingen for å ta i bruk TLS 1.3. Hvor lagres privat nøkkel?

b) Vis med skjermklipp hvordan TLS 1.3 etableres og deretter fortsetter med kryptert pakker. Undersøk hvilke siffersuiter som tilbys i Klient-Hello og hvilken av disse som Tjener svarer med i Tjener-Hello.

Fra Blackboard kommunikasjon

I den videre delen av oppgaven gjøres det pakkefangst av oppslag på Blackboard. Blackboard HTTPS bruker TLS 1.2. Spørsmålene er de samme som før revidering, og er kopiert inn her for å få en komplett oppgavetekst. Det er altså bare datagrunnlaget som er endret.

c) Steg 1 oppkopling: Client Hello
Det første som skjer er at partene må enes om hvilke algoritmer de skal bruke for kryptering. Og det er ikke bare en enkelt algoritme som trengs, men en hel suite. Vis et skjermklipp av de siffersuitene (Ciphersuits) som klienten tilbyr tjener å velge blant.

d) Steg 2A: Server Hello, siffersuite
Vis skjermklipp av den bestemte siffersuiten som tjener velger. Denne må nødvendigvis være med på listen fra klienten. *Til informasjon: Hva ligger i siffersuiten? Her er en forklaring på hvordan man kan tolke den «kryptiske» strengen: https://scotthelme.co.uk/https-cheat-sheet/*

e) Steg 2B: Server Hello, sertifikat
Tjener må sende sitt sertifikat for å autentiserer seg når nye sesjoner settes opp. Vis hva innholdet i serversertifikatet er for følgende to elementer:

- «Issuer/common name», altså utsteder
- «subject/common name», altså innehaver

f) Steg 3: Nøkkelutveksling

Partene genererer tall som brukes som nøkler for videre symmetrisk kryptert kommunikasjon. For dette brukes Diffie-Hellman algoritmen. DH etablerer en delt hemmelighet over en usikker kanal. Denne delte hemmeligheten kan brukes direkte som en (symmetrisk) nøkkel, eller til å avlede nye symmetriske nøkler (for eksempel et dobbelt sett av nøkler, en for hver senderetning).

DH har også flere settinger. Disse parameterne angis i seksjonen «Handshake Protocol/Server Key Exchange». Vis med skjermklipp hva som er valgte server- og klient-parametere for DH-algoritmen.