BACHELOROPPGAVE:

Digital overføring Vestoppland Politidistrikt

FORFATTER(E):

- DANIEL KRISTOFFERSEN
- EIRIK LINTHO BUE
- EIRIK ELLEGÅRD

Dato: 15.05.2015

SAMMENDRAG

Tittel:	Digital overføring Vestoppland Politidistrikt	Dato : 15.05.2015
Deltaker(e)/	Daniel Kristoffersen Eirik Lintho Bue Eirik Ellegård	
Veileder(e):	Harald Liodden	
Evt. oppdragsgiver:	Vestoppland Politidistrikt / HiG IT-tjenesten v/ Stian Husemoen	
Stikkord/nøkkel ord	Kommunikasjon, sikkerhet, Java, Android, TLS	

Antall sider/ord: 188 / 32 500	Antall vedlegg: 20	Tilgjengelighet (åpen/konfidensiell): Åpen

Kort beskrivelse av master/bacheloroppgaven:

Vestoppland Politidistrikt har utfordringer vedrørende overføring av digital informasjon direkte på åsted og har derfor et behov for et system som kan bistå de med å løse denne utfordringen. Særlig å se på muligheten for å kunne utveksle bilder, film og lydopptak på en sikker og verifiserbar måte mellom enheter og operasjonssentralen. Aspekter som data integritet, reliabilitet, validitet og autentisering er sentrale begreper som må inngå i utvikling av et slikt system.

Formålet med oppgaven er å se på mulige løsninger og utvikle et system ved å gjøre et analysearbeid som legger grunnlaget for dette. Systemet vil bestå av en Android applikasjon og en serverapplikasjon som vil vise den praktiske implementasjonen.

Rapporten vil bestå av en omfattende sikkerhetsvurdering som vil omhandle ulike protokoller, kjente angrep/svakheter og forbrukerprodukter. Den vil også inneholde drøftinger og avgjørelser om design, utvikling og testing av dette systemet.

Abstract

Title:	Digital overføring Vestoppland Politidistrikt	Date : 15.05.2015
Participants/	Daniel Kristoffersen	
	Eirik Lintho Bue	
	Eirik Ellegård	
Supervisor(s)	Harald Liodden	
Employer:	Vestoppland Politidistrikt / HiG IT-tjenesten v/ Stian Husemoen	
Keywords	Communication, security, Java, Android, TLS	
Number of pages/words:188 / 32 500	Number of appendix: 20	Availability (open/confidential): Open

Short description of the bachelor thesis:

Vestoppland Politidistrikt experience challenges transfering digital information directly from the crime scene and they are in need of a system that could provide solutions for these challenges. Especially the possibility to exchange images, videos and audio recordings in a secure and verifiable manner between units and the operations center. Aspects like data integrity, reliability, validity and authentication are key concepts that need to be considered during the development of such a system.

The purpose of this thesis is to consider possible solutions and develop a system by doing an initial analysis. The system will be composed of an Android application and a server application that will show the practical implementantion.

The report consists of an extensive security assessment that will consider different protocols, known attacks/weaknesses and consumer products. Discussions and decisions about design, development and testing of the system will also be a part of this report.


HEED
POLITI
JD 66833

Forord

Vi vil innlede hele rapporten med å takke Vestoppland Politidistrikt for all deres tid og imøtekommelse vedrørende dette prosjektet. Selvom de ikke har vært direkte tilknyttet denne bacheloroppgaven har de vært til stor hjelp både med å avsette tid og ressurser for oss slik at vi fikk introdusert systemet og holdt brukertesten hos dem. De har også, spesielt ved seksjonsleder Pål Erik Teigen, kommet med god hjelp for å fremme videre arbeid ved å gi oss kontaktinformasjon direkte inn til Politidirektoratet slik at det er muligheter for at prosjektet kan realiseres i stor skala.

Det er på sin plass å sende takk til sikkerhetskonsulent Christoffer Vargtass Hallestensen som har vært en nyttig ressursperson for dette prosjektet ved å være tilgjengelig, behjelpelig og ha gode råd i vanskelige situasjoner. Dette i tillegg til å avsette tid i sin allerede hektiske hverdag til å verifisere sikkerheten i systemet vårt på et svært teknisk og dypt nivå.

Vår veileder Harald Liodden har vært til god hjelp med det administrative under prosjektperioden og hans kunnskaper har vært til stor hjelp for å drøfte avgjørelser og fremgangsmåter.

Takk også til ressurspersonene Rune Hjelsvold, Tom Røise og ikke minst våre korrekturlesere for deres tid og samarbeidsvilje.

Innhold

Innholdsfortegnelse Figur og tabeller:

1 – Innledning

1.1 – Prosjekt beskrivelse

1.1.1 - Problemområde

Vestoppland Politidistrikt har en rekke varierte utfordringer bestående i alt i fra «vanlige» oppgaver, til mer moderne utfordringer som har dukket opp i takt med teknologiens utvikling. Det ligger utfordringer knyttet til muligheten i å overføre digital informasjon fra enheter ute på åsted til operasjonssentralen (fra nå: OPS) og motsatt. Utfordringer i denne omgang er mangel på et slikt system som kan hjelpe de med disse utfordringene, men vi må også se på aspekter som sikkerhet, data integritet, relabilitet og validitet i forbindelse med implementasjon av et slikt system.

Dette vil i hovedsak gjelde internt for Vestoppland Politidistrikt, men oppgaven definerer også problematikk rundt manglende løsning for publikum som kan brukes som en tipskanal mot Politiet.

1.1.2 – Oppgave definisjon

Helt konkret vil det utvikles et system bestående av en app for mobile enheter og ett serverprogram som vil kjøre på en vanlig desktop PC eller laptop. Hvor appen er tiltenkt å bli brukt av politibetjentene ute på åsted og samsnakke med serverapplikasjonen benyttet på OPS. Det vektlegges kjernefunksjonalitet og praktisk implementasjon av dette som tilbyr deling av media mellom enheter og OPS på en sikker og verifiserbar måte. Verifiserbar på den måten at man skal kunne garantere identiteten til alle parter i informasjonsutvekslingsprosessen og sikker på den måten at uvedkommende ikke skal kunne alterere utvekslingen som foregår. Hashing av alt innhold er også meget relevant for å vite om media ikke har blitt alterert slik at det kan brukes som bevismateriale.

Serverløsningen vil bestå av en applikasjon som hovedsakelig består av ett enkelt GUI og en database med nødvendig funksjonalitet for å kunne svare opp i mot appens kjernefunksjonalitet.

Hovedvekten i oppgaven omhandler å finne mulige løsninger, utover selve utviklingen av et slikt system vil det inngå ett omfattende analysearbeid for å avgjøre hvilke sikkerhetsteknologier som er gode nok til bruk i et slikt system. Det vil bli foretatt en analyse av dagens løsning og av mulige eksisterende løsninger. Som en del av analysen vil det også foreligge en sikkerhetsvurdering av plattformer som omhandler om det er trygt å benytte forbrukerprodukter fra Apple og Google.

Avslutningsvis leveres det et proof-of-concept system som vil vise praktisk implementasjon av ønsket kjernefunksjonalitet basert på analysearbeidet gjort i forkant og underveis i prosjektet. Kjernefunksjonaliteten vil omhandle muligheten for å sende digitale media, herunder bilder, video og lyd mellom enheter og OPS. De andre deloppgavene vil ikke være en del av proof-of-concept systemet, men vil bli drøftet for å finne mulige teoretiske løsningsmetoder.

1.1.3 - Avgrensning

I dette prosjektet avgrenser vi til å se på utfordringer knyttet til digital informasjonsutveksling internt i Vestoppland Politidistrikt. Dette defineres i oppgavebeskrivelsen som «Politiet på stedet → OPS». Vi skal se på hvilke mulige tredjepartsløsninger som finnes for å evaluere om noen av disse kan brukes og analysere løsningsmetodikk og fremgangsmåter for hvordan gruppen best mulig kan utvikle et eget system for dette formålet.

På grunnlag av kunnskaper innad i gruppen, samt tiden tatt i betraktning, utvikles appen kun for Android, dette til tross for at Vestoppland Politidistrikt hovedsakelig bruker iOS enheter. Det blir sett bort ifra all økonomi i forbindelse med en eventuell utskiftning av enheter eller hvilke ressurser som måtte trenges for å videreutvikle til iOS. Det vil gjenbrukes systemfunksjoner så langt det lar seg gjøre for å skape en så brukervennlig og gjenkjennelig app som mulig for brukerne. Mest hensiktsmessig er det å benytte de knappe tidsressursene på sikkerhet og funksjonalitet kontra bruke ressurser på å utvikle «native» funksjoner som allerede eksisterer. Vi vil også avgrense til å kreve at brukeren har uavbrutt edgde dekning for kommunikasjonen mellom app og server. Android appen vil benytte seg av Android biblioteket «SSLEngine», og på grunnlag av kravet fra oppdragsgiver om bruk av TLSv1.2 og minst 128-bits krypteringsnøkler er man dermed nødt til å forholde seg til API nivå 20+. Dette

skyldes at ingen lavere API nivåer støtter den sterke krypteringen som kreves og resulterer i at systemet må utvikles for Android 5.0 (Lollipop).

Serverapplikasjonen vil kjøres på kun én OPS PC, noe som vil begrense bruken noe, men fordi det er et proof-of-concept system anses dette som en fornutig avgrensning.
Vi avgrenser også til å se bort ifra sikkerheten på innsiden av Vestoppland Politidistrikts brannmur, som i praksis betyr at det ikke implementeres noen sikkerhetsteknologier for serverprogrammet. Derfor fokuserer vi kun på sikkerheten rundt kommunikasjonskanalen og appen.

Deloppgave 2 «Publikum på stedet → OPS» tas ikke med i utviklingsfasen og vil derfor heller ikke inngå som en del av proof-of-concept systemet som leveres grunnet vurdering av viktigheten og etter diskusjon med oppdragsgiver.

Deloppgave 3 «OPS→Politi/personer på stedet» vil hovedsakelig bli sett bort ifra, men muligheten for OPS å kunne videresende mottatt media til andre patruljer vil bli implementert siden dette ble ansett som en viktig del og med god tilhørighet til hovedoppgaven.

1.2 – Målgruppe

Systemet som leveres vil rettes mot det varierende teknologiske nivået blant betjentene, og er ment å kunne brukes av hvem som helst uavhengig av alder, bakgrunn og vil ikke kreve noen teknisk forståelse. Sikkerhet og brukervennlighet er to av de viktigste målene, hvorav sistnevnte forsterker hensikten med at systemet er ment å kunne brukes av hvem som helst til tross for høy sikkerhet. Sikkerheten er underliggende og berører derfor ikke brukeren på noen som helst måte, men forklares og dokumenteres i rapporten.

1.3 – Formålet

Formålet vil være å bistå Vestoppland Politidistrikt med å vise hvordan deres problematikk rundt sending av digitalt media kan løses uten å lage komplekse systemer eller forstyrre deres arbeidsrutiner, men heller smelte sammen med eksisterende arbeidsmetodikk. Ved å lage et enkelt system som vil kunne brukes uten noen grad eller i svært liten grad av opplæring så vil det hjelpe de med å løse hvertfall ett av deres teknologiske utfordringer i hverdagen. I tillegg til det leverte proof-of-concept systemet vil det leveres en prosjektrapport som har som formål

å lyssette utfordringer møtt underveis i perioden. Prosjektets progresjon, videre arbeid for hvordan å iverksette dette prosjektet i full skala samt spesifiseringer av sikkerhetsstandarder og annen nødvendig informasjon vil også inngå i denne rapporten.
Kort oppsummert skal systemet være brukervennlig på et slikt nivå at hvem som helst kan bruke det uten at det går på bekostning av nødvendig sikkerhet som man forventer av et system brukt av Vestoppland Politidistrikt.

1.4 – Bakgrunn og kompetanse

Alle tre gruppemedlemmene kommer fra samme årskull på faglinjen Dataingeniør og har nesten lik bakgrunn som man oppnår i løpet av studiet. Utover dette har gruppemedlemmene litt variert bakgrunn tatt i betrakning valgte valgfag femte semester hvor Eirik Lintho Bue tilegnet seg kunnskaper innen Java fra faget «Programvareutvikling» mens de to øvrige gruppemedlemmene har erfaring fra «Systemadministrasjon». Utover dette skiller to av gruppemedlemmene seg ut ved at de har fagbrev, og sammen har gruppen god og noe variert kompetanse for å møte oppgavens utfordringer.

Kompetanseområdene vi må tilegne oss nye kunnskaper innenfor er informasjonssikkerhet, kryptografi og mobil applikasjonsutvikling. Derfor startet prosjektperioden med å innhente nødvendig kunnskap innenfor nevnte fagområder for å møte denne oppgaven på en forsvarlig måte.

1.5 – Rammeverk

1.5.1 – Systemutviklingsmodell

Vi har valgt å benytte en kombinasjon av modellene SCRUM og inkrementell i løpet av dette prosjektet, til tross for at disse er noe like, ble det ansett som veldig gunstig for prosjektet. Det benyttes SCRUM som verktøy til konfigurasjonsstyring og vi benytter oss også av modellens prosessflyt og møteanordninger. Møtene vil bestå av daglige SCRUM møter, «Sprint Planning Meeting» og «Sprint Review Meeting», hvorav de to sistnevnte møtene slåes sammen til ett. Den inkrementelle modellen er blitt fulgt mindre enn planlagt, men har fortsatt til en viss -eller modifisert grad fulgt modellens karakteristika.

SCRUM passer veldig godt opp i mot oppgaven ved at de forskjellige oppgavene er definert på en slik måte at de enkelt kan overføres til «user stories» uten at man står i fare for at kravene endres i denne overføringen. Dette bidro til å skape en god og fyldig kravspesifikasjon, kalt backlog, som er en arbeidsliste med ønsker til sluttproduktet. Det er også enkelt å kunne bryte ned de forskjellige kravene til mindre deler om det oppdages at en «user story» er for tidkrevende for den pågående sprinten. Daglige SCRUM fungerer som en god anledning for gruppemedlemmene for å få en oversikt over hverandres arbeidsoppgaver og eventuelle problemer som har oppstått fordi det arbeides med forskjellige «user stories», gjerne fra forskjellige lokasjoner. Hvis gruppen ikke arbeider sammen denne dagen blir møtet holdt over «Skype».

Inkrementell modellen brukes for å backtracke hvis noe går feil mens det arbeides mot en endelig løsning. Prosjektet er basert på å levere ett første utkast til Vestoppland Politidistrikt som inneholder de mest kritiske funksjonene, kalt kjernefunksjonalitet, som kunden deretter gir tilbakemelding på som former utviklingen videre. Dette gjøres i en noe modifisert form ved at det ble holdt en brukertest hos Vestoppland Politidistrikt, og feedbacken fra denne brukertesten la fundamentet for den videre utviklingen. Dette møtet ble holdt litt senere i prosessen enn først planlagt, men ga likevel et godt utbytte og viktige påpekninger for sluttspurten av prosjektet.

«SCRUM board» bidrar med å opprettholde struktur i prosjektet og dokumentasjon blir produsert i slutten av hver sprint under «Sprint Review Meeting». Dette møtet indikerer avslutningen på sprinten og hvor det blir foretatt en eventuell videreføring av «user stories» som ble for tidkrevende. Samme dag som dette møtet ble holdt foretok vi også «Sprint Planning Meeting» som indikerer starten på neste sprint og inneholder en kort planlegging av hvilke «user stories» som skal bli utviklet under denne sprinten. Vi så det som mest effektivt og tidsnyttig å benytte samme dag for disse to møtene istedet for to separerte dager og det har vist seg å fungere svært godt.

Karakteristika ved selve oppgaven som lå til grunn for valg av den systemutviklingsmodell hybriden som ble valgt følger:

- Produktet skal utvikles i inkrementer.
- Levere ett proof-of-concept som viser kjernefunksjonalitet og hvordan et slikt system kan implementeres i praksis.

Det ble også vurdert andre systemutviklingsmodeller på det samme grunnlaget som nevnt ovenfor og hver modell med eventuell kombinasjoner ble nøye veid opp i mot hverandre for å se hvilke som passet best. De andre modellene som ble vurdert var:

- Xtreme Programming (XP)
- Rational Unified Process (RUP)

XPs store fokus på testing anså vi som feil bruk av tidsressursene fordi det kun skulle leveres et proof-of-concept system. Vi mente også at mange bugs -og kodefeil kunne lukes ut ved testing av utviklede funksjoner som en helhet i systemet istedet for forhåndsdefinerte tester.

RUP anså vi som litt for komplisert for dette prosjektet med dets forskjellige perspektiver og faser, tiltross for at vi til en viss grad kunne implementert de ulike perspektivene. Dette på bakgrunn av at det er gjort mye forarbeid i forbindelse med en bacheloroppgave og derfor vil mye av fokus punktene til RUP bli unødige. Derfor gjenstod ikke disse to modellene med gode nok fordeler til at vi valgte en av disse.

1.5.2 – Plan for gjennomføring

Det er utarbeidet ett GANTT-skjema for å visuelt illustrerer hvordan prosjektperioden er planlagt med sprinter og andre viktige aktiviteter oppført. Viser til *vedlegg M* for oversikt over selve GANTT-skjemaet, men noen viktige punkter vil nevnes her.

Alle sprinter vil være utvikling på proof-of-concept systemet, hvor hver sprint inneholder disse stegene:

- **System og programvare design:** Skaper en overliggende arkitektur og designe løsningene man har kommet frem til. Involverer å identifisere og beskrive de fundamentale kravene for den kommende sprinten samt hvordan å utvikle disse.
- **Implementasjon og testing:** De utvalgte «user storiesene» for den gjeldende sprinten vil bli utviklet til en fungerende utvidelse av programmet og testet for å verifisere at implementasjonen er vellykket.
- **Integrasjon og system testing:** Utvidelsene vil bli integrert som en helhet i systemet og testet som en del av det. Ved fullførelse av dette steget vil man ha en ferdig inkrement og indikrere avslutningen på sprinten.

SCRUM modellen poengterer at det er vanlig å ha to konkrete møter, et for planlegging av oppkommende sprint, et annet for avslutning av daværende sprint, som nevnt under punkt 1.5.1- Systemutviklingsmodell. Disse møtene er som nevnt tidligere slått sammen og holdes henholdsvis annenhver torsdag, i skifte mellom to sprinter.

Sprint én skulle utifra planen inneholde en brukertest hos Vestoppland Politidistrikt, men denne ble flyttet til 27. mars grunnet manglende tidsressurser hos dem og det oppstod ingen konkrete utfordringer for utviklingen ved flytting av denne testen.

De tre statusrapportene som er oppført i Gantt-skjemaet er plassert i henhold til viktige eller spesielle aktiviteter i tilfelle det skulle ha oppstått viktige hendelser verdt å rapportere. Resultatet av brukertesten som i følge planen skulle være en del av statusrapport én, bestod istedet som en del av statusrapport tre.

Den siste perioden «Testing og feilretting», vil være relativt lik en sprint, men med mindre eller ingen fokus på utvikling. Denne perioden inneholder situasjonstesting av systemet som er ment for å avdekke eventuelle feil/mangler og tid til å utbedre disse.

1.5.3 – Konfigurasjonsstyring

Som nevnt i punkt 1.5.1 – Systemutviklingsmodell ble det benyttet «SCRUM board» under prosjektet og ble løst ved bruk av tjenesten «Trello». Dette er en nettbasert løsning hvor alle gruppemedlemmene fikk en oversikt over prosjektet. «user stories», backlog, release backlog, pågående og sluttført arbeid ble oppført her i samme struktur som ved bruk av et tradisjonelt «SCRUM board».

«Bitbucket» ble benyttet som «repository», med andre ord et digitalt tilgjengelig oppebevaringssted for koden og ble benyttet sammen med «Source Tree» som er et versjonshåndteringsprogram som ble knyttet opp mot «Bitbucket».

For organisering av dokumenter, illustrasjoner og andre viktige dokumenter eller filer i forbindelse med prosjektet ble det benyttet «Dropbox». Slik hadde alle gruppemedlemmene kontinuerlig tilgang til disse filene.

1.6 – Prosjekt organisering

1.6.1 – Ansvarsforhold og roller

Daniel Kristoffersen har fungert som SCRUM Master noe vi kom frem til under et fellesmøte holdt første uka i prosjektet. Det vil også inngå som en del av SCRUM Masterens ansvar å sørge for å utarbeide statusrapportene og sørge for en oppdatert nettside for bacheloroppgave slik at utviklerne ikke trenger å tenke på administrative detaljer.

1.6.2 – Øvrige roller

Oppgaven er definert for Vestoppland Politidistrikt, men det er Stian Husemoen fra IT-tjenesten ved Høgskolen i Gjøvik som fungerer som oppdragsgiver under denne bacheloroppgaven. Han er den som vil fungere som vår kontaktperson samt ta beslutninger som er avgjørende for oppgaven i fellesskap med oss. Andre personer fra IT-tjenesten som bistår som ressursperson for dette prosjektet er sikkerhetskonsulent Christoffer Vargtass Hallestensen som vil bidra med sikkerhetsvurdering av systemet og generell rådgivning. Vestoppland Politidistrikt har liten delaktighet iløpet av prosjektetsgang med unntak av et møte vi arrangerer hos dem hvor vi skal gjennomføre en brukertest og muligens noe delaktighet fra deres side iløpet av perioden vi har satt av til testing.

Veileder for vår gruppe er Harald Liodden som er en nyttig person med relevant og god bakgrunn som passer godt for vår oppgave.

1.7 – Selve rapporten

Selve rapporten vil bli skrevet på en slik måte at den vil være forståelig for den gjennomsnittlige IT-student og begreper vidt kjent for denne gruppen vil ikke få utbredt definering. Rapporten vil inneholde en definisjonliste, se *vedlegg N*, som presiserer begreper og forkortelser vi mener ikke er kjent kunnskap for målgruppen. Det vil innledningsvis være to innholdsfortegnelser, hvor den ene representerer innholdet og den andre vil være en sammenslåing av innholdsfortegnelse for både figurer og tabeller. Bildet som brukes til den personlige forsiden ble tatt av oss da vi var hos Vestoppland Politidistrikt.

Når det gjelder terminologi vil vi fornorske ord og uttryk det er naturlig for og vil være konsekvent på dette:

- «Brukeren» vil omhandle brukeren av systemet enten det er appen eller serverapplikasjonen, og vil bli brukt isteden for henholdsvis han/hun.
- «Oppdragsgiver» vil alltid være Stian Husemoen, og det vil bli tydelig presisert om det er snakk om Vestoppland Politidistrikt som rolle.
- OPS er en fortkortelse for operasjonssentralen. OPS er de som benytter serverapplikasjonen, ikke appen.
- «App» eller «appen» skrives i henhold til norsk ordbok.
- Server er alltid ment som applikasjonen som kjører på en vanlig laptop eller desktop maskin benyttet av OPS.
- «Database» vil vi utype i teksten om gjelder for appen eller for serveren.
- «POD» vil bli brukt som forkortelse for Politidirektoratet.
- «Metode» og «funksjon» vil bli brukt om hverandre, men vil bety det samme uavhengig av bruk.
- En «aktivitet» er en av de mulige skjermbildene i appen, hvor hvert skjermbilde er ny aktivitet. Eksempelvis er velkomstbildet aktivitet én og hovedmeny aktivitet to.
- System som utvikles vil bli kalt VP Secure.

Strukturen i rapporten vil bestå av totalt 8 kapittler og en kort beskrivelse av hvert kapittel følger:

1 – Innledning: Her vil bakgrunnen og formålet med oppgaven bli tydeliggjort og mye av innledningen baserer seg på prosjektplanen. Prosjektorganisering og rammeverk under prosjektperioden lyssettes også.

2 – Forarbeid: Dette kapittelet tar for seg forarbeidet som måtte gjøres for å kartlegge fremgangsmåten og det praktiske behovet utover hva oppgaven definerer.

3 – Kravspesifikasjon: Her vil oppgaven brytes opp i «user stories» som er med på å definere en backlog med funksjonalitet som skal være en del av sluttproduktet. Vil også inneholde krav for både analysearbeid og utviklingsarbeid.

4 – Analyse: Alt analysearbeid i forbindelse med prosjektet vil bli tatt opp i dette kapittelet.

5 – Design: Det planlagte designet av app og serverapplikasjon vil utgreies i dette kapittelet og vil fortelle hvordan den overliggende arkitekturen ble planlagt og gjennomført.

6 – Implementering, koding og produksjon: Vil fortelle om detaljer rundt utviklingen og hvordan forskjellig funksjonalitet ble implementert.

7 – Testing og kvalitetssikring: Omhandler all testing av systemet i form av testing av funksjonalitet og brukertest av grensesnitt. Samt kvalitetssikring som forteller hvordan å opprettholde kvaliteten og en gjennomgang av sikkerheten i systemet.

8 – Avslutning: Dette kapittelet vil avslutte rapporten og konkluderer hele prosjektet.

2 - Forarbeid

2.1 – Kartlegging av behov og fremgangsmåte

Oppgaveteksten definerer de faktiske behovene systemet må fylle, sett opp i mot de teknologiske utfordringene som Vestoppland Politidistrikt har. En kartlegging av deres faktiske behov og hvordan en kan nå frem til disse behovene er like viktig som å tilfredsstille de definerte behovene fra oppgaveteksten.

Deres hverdag er preget av mange forskjellige og tungvinte løsninger med alle de elektroniske hjelpemidlene de bruker i dag. Dette er roten til mange irritasjonsmomenter og de trenger et system som kan erstatte alle disse hjelpemidlene på en sømløs måte. Dette ville bidratt til å øke effektiviteten og lette arbeidet deres ute på åsted. I dag bruker de flere SLR-kameraer for å ta bilder, egne lydopptakere til lydopptak, nettbrett i bilene og alle disse apparetene kan bli en unødig stressfaktor i deres hverdag. Under besøk hos Vestoppland Politidistrikt ga de uttrykk for at det er et irritasjonsmoment å måtte passe på at det hele tiden er plass på minnekort og oppladede batterier i tillegg til alt styret med etterarbeidet.

Det viste seg at behovet er større enn det oppgaven definerer, og det er derfor svært viktig å se på disse faktorene for å kunne skape et system som på best mulig måte dekker disse elektroniske hjelpemidlenes funksjon. Systemet som leveres tiltross for å kun være ett proof-of-concept vil vise at det lar seg gjøre å enkelt samle disse teknologiene inn i ett og samme system.

Fremgangsmåten for å svare på dette ble gjort ved at det ble skissert en løsning for hvordan det teknologiske måtte fungere for å skape et sømløst system. Brukertesten bidro med å skape en dialog for å få innblikk i deres hverdag og illustrere det praktiske behovet for et slikt sytem. Denne dialogen bidro i noen grad til å styre hvordan utviklingen av systemet skulle gjøres, samt påpekning av hva som er det mest nødvendige og ikke minst at et slikt system er høyt ønsket.

2.2 – Risikoanalyse

2.2.1 - Teknologi risikoer:

- Endring i API-nivå fra Android som kan føre til at funksjoner blir utdatert eller ikke støttes.

- Bugs i ny versjon av OS (Android 5.0 nettopp lansert vår 2015)

2.2.2 - Forretningsmessige risikoer:

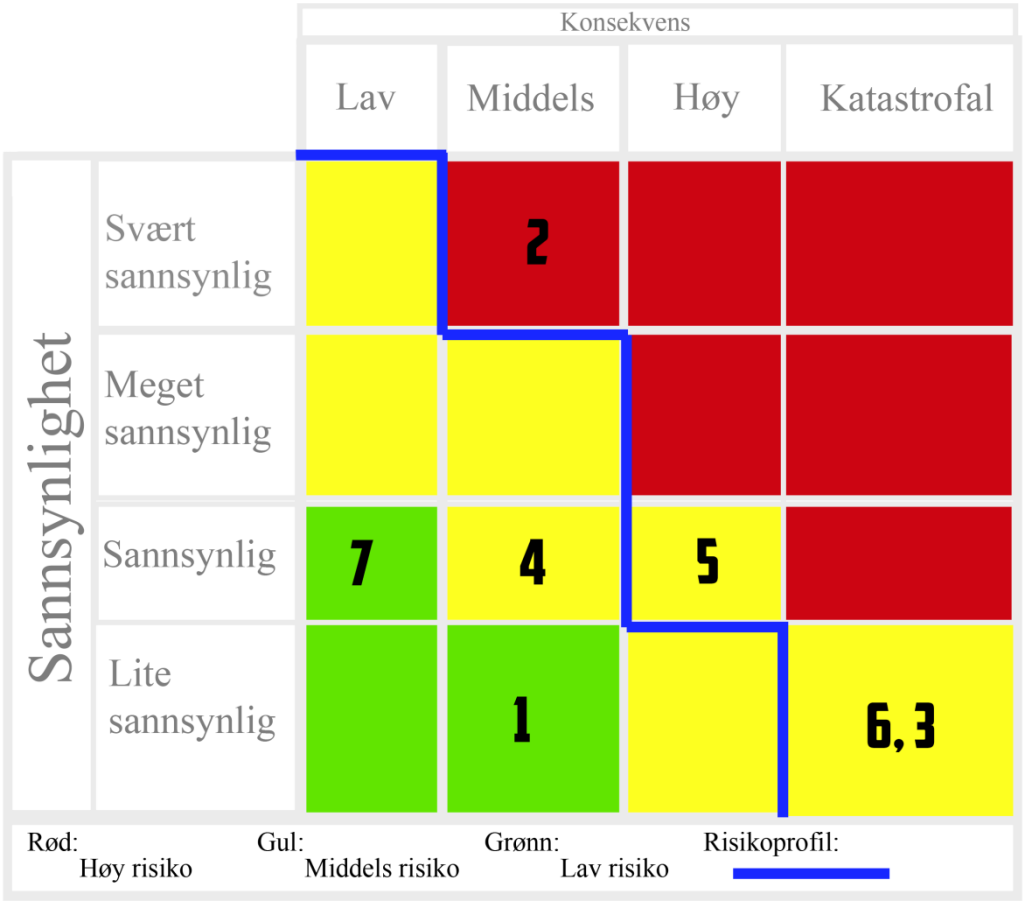
- Systemet tilfredsstiller ikke kravene fra oppdragsgiver og Vestoppland Politidistrikt
- Endringer i kravspesifikasjonen

2.2.3 - Prosjekt risikoer:

- Sykdom blant en eller flere av gruppemedlemmene
- Avskjedigelse av en eller flere av gruppemedllemene
- Uforutsette problemer som resulterer i at tidsestimeringen slår feil

Risikoanalyse				
Nr.	Beskrivelse	Sannsynlighet	Konsekvens	Tiltak
1	Endring i API-nivå i forbindelse med ny Android versjon.	Lite sannsynlig	Middels	Tiltakene varierer ut ifra endringstype, men genereltsett vil det omhandle å omskrive kode.
2	Bugs i ny OS versjon.	Svært sannsynlig	Middels	Avhengig av hva slags type bug, vil løsninger være å vente på utbedring fra Google eller gjøre endringer i implementasjon for å arbeide oss rundt feilen.
3	Produktet tilfredsstiller ikke kravene.	Lite sannsynlig	Katastrofalt	Gjøre endringer som tilfredsstiller kravene. Forflytte arbeidsressursene fra utvikling og forstørrelse av systemet til å håndtere endringene som må gjøres.
4	Endringer i kravspesifikasjonen enten fra brukertesten eller oppdragsgiver	Sannsynlig	Middels	Legges inn for å inngå i neste inkrement av produktet. Flytte fokus fra opprinnelig plan, til nye ønsker.
5	Sykdom hos en eller flere av gruppemedlemmene over en sammenhengende periode som overskrider én uke.	Sannsynlig	Høy	Omdelegering av arbeid, arbeide hjemmefra er en løsning ved bruk av Skype for møter og annen interaktivitet Hvis det overstående ikke kan gjøres må backlog-en justeres og resterende gruppemedlemmer må ta det mest kritiske arbeidet.
6	Avskjedighet av et gruppemedlem.	Liten sannsynlighet	Katastrofalt	Noe av det samme som ovenfor, gjøre sterkere avgrensninger.
7	Uforutsette problemer som forårsaker feil tidsestimering.	Sannsynlig	Lav	Basert på release backlog-en og justere backlog-en for å delegere arbeidet til det mest kritiske funksjonen. Følge med på progresjonen.

Tabell 1: Viser de ulike risikoene nummerert for bruk i matrisen nedenfor. Sannsynlighet, konsekvens og tiltak er presentert for å danne risikonivået og løsningene om noen av de skulle forekomme.



Figur 1: Illustrerer risikoene grafisk satt inn i en matrise representert med farger for å illustrere graden. Alt under risikoprofilen markerer hva som aksepteres, alt over starter tilhørende risikohåndteringsplan. Utarbeidet i Adobe Photoshop.

3 – Kravspesifikasjon

3.1 – Analysearbeid

Innledningsvis skal det ses kort på dagens løsning hos Vestoppland Politidistrikt og hvordan de løser problematikken per i dag uten et slikt system og tilhørende rutiner. Som en etterfølger av dette vil det undersøkes om det finnes mulige løsninger der ute som enten er utviklet med tilsvarende funksjonalitet, eller annen programvare som kan benyttes for å løse nevnt problematikk. Siden oppgaven vektlegger at det skal utvikles et proof-of-concept vil det kun ses på noen mulige løsninger, og ikke vektlegges i like stor grad som utvikling.

Hovedaspektet med analysearbeidet vil være å innhente informasjon om sikkerhetsaspektene som må foreligge for å utvikle et system som tilfreddstiller kravene. Oppgaven definerer tydelig ønskene til hva et sluttprodukt bør inneholde, men definerer ikke noe om underliggende teknologi så det må derfor gjøres analyse i forkant. Oppdragsgiver har definert noe ved at det må benyttes TLSv1.2 og minimum 128-bits krypteringsnøkler. Analysen vil derfor finne ut av hvilke protokoller, cipherpakker, sertifikater og algoritmer som er sikre nok til bruk i et så viktig system. Det vil også foreligge et analysearbeid for å se på kjente svakheter og angrep på TLS og cipherpakker.

Sikkerhetsvurdering av plattformer vil også inngå som en del av analysen, hvor det vil ses på i hvilken grad det kan benyttes forbrukerprodukter fra Apple og Google. Siden vi har avgrenset til å kun utvikle for Android, vil den avgrensningen også gjelde her.

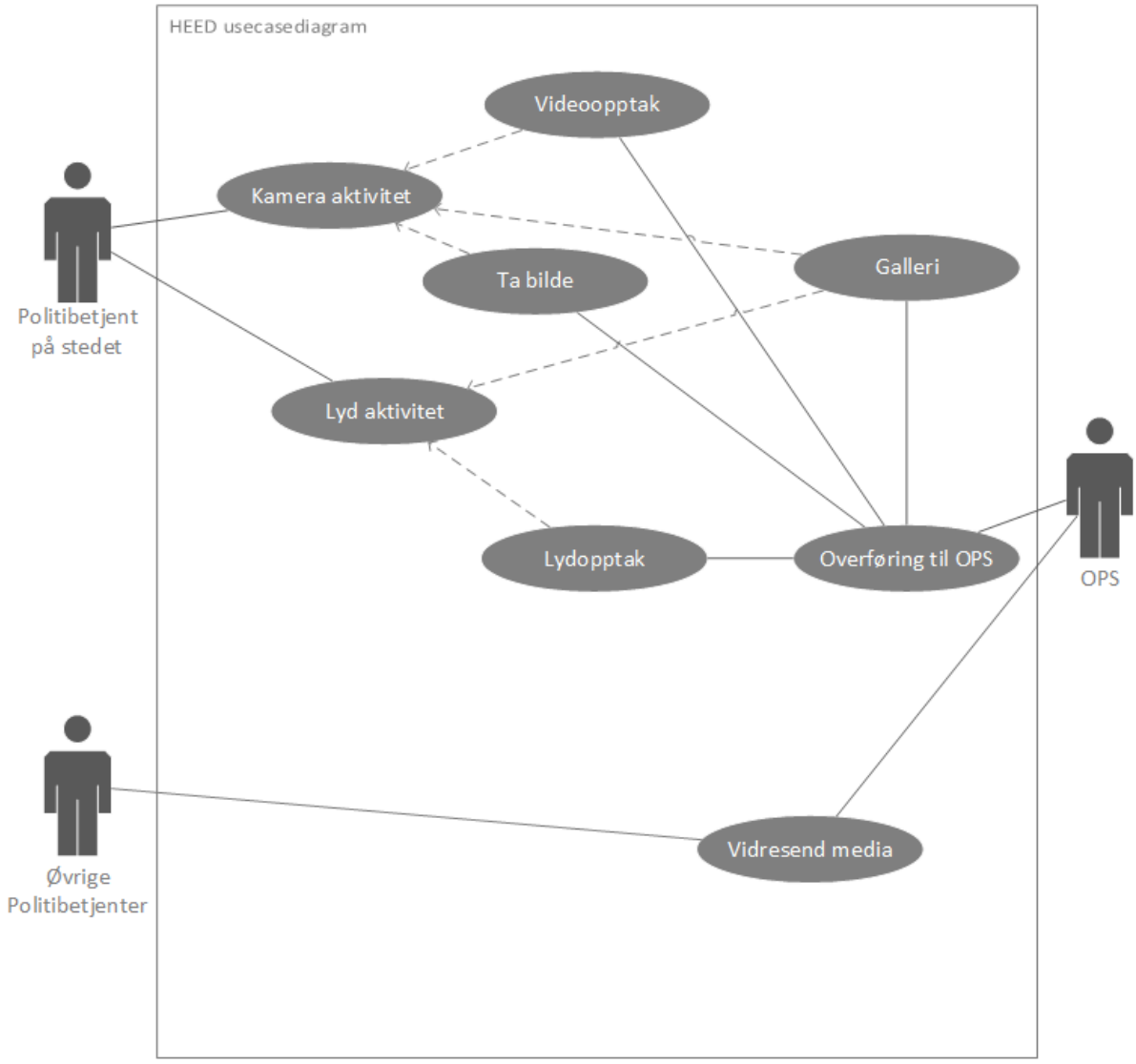
Avslutningsvis vil det vedligge teoretiske forslag på hvordan man kan løse deloppgave 2 «Personer på åsted → OPS», siden dette ikke vil inngå i utviklingsfasen som nevnt i 1.1.3 – Avgrensning.

3.2 - Utviklingsarbeid

Analysearbeidet som blir gjort både i forkant og underveis i prosjektet vil være styrende for utviklingen i den grad det er nødvendig, avhengig av hva analysen vil belyse av kjent problematikk, gode praktiske løsninger og eventuelle svakheter. Dette vil da i hovedsakelig omhandle valg av cipherpakke siden dette er essensielt for hele nøkkelutveksling, og derfor den totale sikkerheten.

Det vektlegges å utvikle kjernefunksjonalitet som gir muligheten for politibetjenter på åsted å kunne sende digitalt media fra sin mobile enheter til OPS. Dette gjøres på en sikker og verifiserbar måte slik at man skal kunne verifisere identiteten til de involverte partene og være garantert at informasjon ikke er endret i transaksjonen, eksempelvis MITM angrep. Det skal også implementeres muligheten for OPS å videresende media til enhetene. Det vil tas hensyn til dårlig dekning og lave overføringshastigheter i henhold til avgrensningen og hashing av alt innhold bidrar til at man kan sjekke om media har blitt alterert.

3.3 – Use case



Figur 2: Viser de forskjellige use casene og deres tilhørlighet. Utarbeidet i Microsoft Visio 2013

3.3.1 – Høy nivå use case

Use case:	Sende bilde lagret på enhet
Aktør:	Politi på stedet → OPS
Mål:	Enkelt, raskt og sikkert kunne sende ett/flere bilde(r) fra enheten til OPS.
Beskrivelse:	Politi på stedet skal dele et forhåndslagret bilde(r) fra enheten til OPS på en sikker og verifiserbar måte.
Pre. betingelser:	Politi på stedet må minst ha edge-dekning og forutsetter at man er koblet til serveren.
Detaljert hendelsesforløp («Main success scenario»):	1.1 - Politiet på stedet ønsker å sende ett eller flere forhåndslagrede bilder fra enheten til OPS. 1.2 - Politiet på stedet åpner appen og trykker på kamera ikonet. 1.3 - Trykker på firkant ikonet som fører til galleriet. 1.4 -Velger det eller de bildene som skal sendes. Disse blir da markert. Trykker fullfør. 1.5 – Trykker på send knapp i høyre hjørne. 1.6 - Bildet/bildene blir nå sendt over en sikker TLS linje til OPS. 1.7 - OPS mottar bildet/bildene.
Alternative scenarer:	1.1 – Politiet på stedet ønsker å sende ett eller flere forhåndslagrede bilder fra enheten til OPS. 1.2 – Har ikke dekning og får ikke koblet til serveren/mister kontakten med serveren. 1.3 - Meldingen «Not connected to server» dukker opp og man får ikke sendt noe. 1.4 - Avventer til man er i et område med dekning og fortsetter fra punkt 1.2 i main scenario. 2.1 – Politiet på stedet ønsker å sende et eller flere forhåndslagrede bilder fra enheten til OPS. 2.2 – Har dekning, men er ikke koblet til server. 2.3 – I bakgrunnen startes en «reconnect handler» som etter forhåndsbestemte intervaller gjenopptar kontakten med serveren og brukeren får meldingen «Not connected to server». 2.4 – Avventer til kommunikasjonen med serveren er på plass og forsøker igjen fra punkt 1.2 i main scenario.

Tabell 2: Høy nivå use case diagram for sending av forhåndslagret bilde, men prosessen er nesten identisk for sending av andre forhåndslagrede media. Viser til brukermanualen under *vedlegg F* for ytterliggere forklaring av ikoner eller andre oppgaver.

Use case:	Sending av nytt lydopptak
Aktør:	Politi på stedet → OPS
Mål:	Enkelt, raskt og sikkert kunne sende et/flere lydopptak fra enheten til OPS.
Beskrivelse:	Politi på stedet skal ta et lydopptak med enheten for deretter å sende det til OPS.
Pre. betingelser:	Politi på stedet må ha minst edge-dekning og forutsetter at man er koblet til serveren.
Detaljert hendelsesforløp («Main success scenario»):	1.1 – Politiet på stedet ønsker å ta opp et lydopptak for deretter å sende det til OPS. 1.2 - Åpner appen og trykker på ikonet av en mikrofon. 1.3 – Knappen med en rød sirkel indikerer opptak. Trykker på denne. 1.4 – Trykker stopp når opptaket er fullført. 1.5 – Lokaliserer send knappen nederst i høyre hjørnet, illustrert med hjelp av en pil, trykker på den for å sende. 1.6 - OPS mottar opptaket.
Alternative scenarer:	1.1 – Politiet på stedet ønsker å høre gjennom klippet for å verifisere at de fikk med alt. 1.2 – Benytter da enten knappene som indikerer spoling eller bruker fingeren på glidebryteren for å spole til ønsket sted i opptaket. 1.3 – Trykker play for å spille av. 2.1 – Politiet på stedet ønsker å ta opp et lydopptak for deretter å sende det til OPS. 2.2 - Har ikke dekning og får ikke koblet til serveren/mister kontakten med serveren. 2.3 - Meldingen «Not connected to server» dukker opp og man får ikke sendt noe. 2.4 - Avventer til man er i et område med dekning og fortsetter fra punkt 1.2 i main scenario. 3.1 - Politiet på stedet ønsker å ta opp et lydopptak for deretter å sende det til OPS. 3.2 - Har dekning, men er ikke koblet til server. 3.3 - I bakgrunnen startes en «reconnect handler» som etter forhåndsbestemte intervaller sørger for at kontakten med serveren gjenopptas og brukeren får meldingen «Not connected to server». 3.4 - Avventer til kommunikasjonen med serveren er på plass og forsøker igjen fra punkt 1.2 i main scenario.

Tabell 3: Høy nivå use case diagram sending av nytt lydopptak. Viser til brukermanualen under *vedlegg F* for ytterliggere forklaring av ikoner eller andre oppgaver.

Use case:	Sjekke dekningsstatus og tilkoblinger med server og type nettverk.
Aktør:	Politi på stedet
Mål:	Finne ut om man har dekning og at man er tilkoblet.
Beskrivelse:	Sjekke om man har dekning og i hvilke grad utifra grafen, samt sjekke om man er koblet til server og hvilke nettverkstype.
Pre. betingelser:	Ingen
Detaljert hendelsesforløp («Main success scenario»):	1.1 – Politiet på stedet ønsker å sjekke om de har kontakt med serveren og internett og graden av dekning. 1.2 – Åpner appen og trykker på knappen som illustrerer dekning. 1.3 – Internettilgang og serverkontakt vises enten ved hake eller kryss. Viser også nettverkstypen og hvor god dekningen er.
Alternative scenarer:	1.1 – Tilkoblingen til serveren er krysset ut, betyr at det ikke er kobling opp mot serveren. 1.2 – Restarter appen ved å lukke og åpne den igjen. 2.1 – Internettilgang er krysset ut, betyr at man ikke har tilgang til internett som mest sannsynlig skyldes mangel på dekning. 2.2 – Se på «Network type», om beskjeden «UNKNOWN» er fremme så indikerer det null dekning og må man re-lokalisere til nytt sted.

Tabell 4: Høy nivå use case diagram sending av nytt lydopptak. Viser til brukermanualen under *vedlegg F* for ytterliggere forklaring av ikoner eller andre oppgaver.

4 – Analyse

Det ble foretatt en analyse av omstendighetene rundt oppgaven og dagens løsning hos Vestoppland Politidistrikt. Vi fikk en innføring i deres arbeidsrutiner for ha best mulig grunnlag for å finne en løsning og det ble gitt et innblikk i hvordan Vestoppland Politidistrikt kan ta i bruk dette. Et grundig analysearbeid vil gjøres av TLS protokollen og tilhørende detaljer for å avgjøre implementasjonsdetaljer rundt sikkerheten slik at dette er på et tilfredsstillende nivå.

4.1 – Dagens løsning

Per i dag har Vestoppland Politidistrikt ingen bestemte rutiner for hvordan de skal behandle bilder, video og lydopptak fra åsted og dette skyldes mangelen på ett slikt system. Dette ble belyst på møtet sammen med Seksjonsleder Pål Erik Teigen og IKT-konsulent Jan Erik Ødegård og ved en eventuell anvendelse av et slikt system vil det bli opprettet rutiner for hvordan det skal bruke i hverdagen. Det foreligger noen vage rutiner per i dag for hvert enkelt område, på den måten at det hovedsakelig er kun en av betjentene som benytter nettbrettet i tjenestebilen for å ta bilder som deretter sendes til ett nettbrett på OPS via e-post.

4.2 – Mulig eksisterende løsninger

En eksisterende løsning som kunne ha blitt tatt i bruk er muligheten for å ha mobile enheter knyttet opp mot en skylagringstjeneste og laste opp media til denne. Denne skylagringstjenesten kunne eksempelvis ha blitt levert av en tredjepart, men det er også mulighet for å sette opp en egen ved hjelp av open source programvare. Det er mange skylagringstjenester på markedet, og de forskjellige leverandørene leverer produktet sitt med forskjellige egenskaper, som type sikkerhet, bruksmuligheter, lagringsstørrelser og prisalternativer. Som et eksempel har vi valgt å se nærmere på tredjepartstjenestene «Dropbox» og «OneDrive», to av de største skylagringstjenestene, hvor først nevnte er selvstendig og leverer kun denne tjenesten, mens siste nevnte er Microsofts egne løsning. Det vil ses på deres sikkerhetsrutiner til lagring og overføring av data samt hvor brukervennlig det ville vært med bakgrunn i oppgaven og ønskene til Vestoppland Politidistrikt.

Ved å ta i bruk «Dropbox» har man muligheten for automatisk opplastning av bilder og video gjennom mobilapplikasjonen, noe som vil dekke noe av problemområde rundt oppgaven.

Under opplastning av data er tilkoblingen beskyttet med SSL/TLS med minimum 128 bits AES kryptering, grundigere detaljer er ikke spesifisert. Når det gjelder lagring av data blir det benyttet AES256 kryptering [5]. Ett av kravene fra oppdragsgiver var en kryptering med bruk av minst 128-bits nøkler, så denne løsningen vil tilfredsstille dette kravet, og da kun basert på sikkerheten være en aktuell kandidat. Om man skal se på muligheten for hashing og at det vil være en kompleks implementering vil dette fort vise seg å ikke være en god kandidat.

«OneDrive» har også muligheten for automatisk opplastning fra mobilapplikasjonen og bruker SSL/TLS på lik linje med «Dropbox». Ved lagring av data benyttes det to typer krypteringer, hvor den ene gjelder datahallen deres og den andre for hver enkelt fil. Diskene i datahallen som dataene lagres på er kryptert med «BitLocker», som er en funksjonalitet utviklet av Microsoft som benyttes for å kryptere hele harddisker med AES128/256 kryptering i CBC modus [6]. Hver enkelt fil og hver enkelt oppdateringer er kryptert enda en gang med tilsvarende kryptering.

Om krypteringen av kommunikasjonskanalen ville vært tilfredsstillende, enten bruk av AES256 eller en annen tjeneste som ikke nevnes her med sterkere kryptering ville det også være naturlig å se på lønnsomheten, forholdet mellom lagringsplass og pris og sett på andre sikkerhetsrisikoer. «Dropbox» reklamerer med et bedriftsabonnement med en pris på 750 EURO (tilsvarende ca. 6300 kr) for fem brukere med en lagringsplass på 1 TB hver [7] mens «OneDrive» reklamerer med en pris på 300 EURO (tilsvarende ca. 2500 kr) for tilsvarende løsning [8]. Ut ifra prisene disse selskapene tilbyr ser man at kostnadene ville blitt meget høye ettersom det er behov for mange lisenser for å dekke behovet hos Vestoppland Politidistrikt. Noen av de andre sikkerhetsrisikoene som også må vurderes kan eksempelvis være å kjenne til leverandørens protokoller for gjenoppretting etter katastrofe, hvor sikker de fysiske datalagringssentrene deres er, og hvilke rutiner de har for å sikre seg mot hacking.

Det er også mulig å sette opp egen skylagring, ved å gjøre dette eliminerer man flere av problemene ved bruk av tredjeparts skylagring ved at man har full kontroll over sikkerheten og dataene som blir lagret. I tillegg til at man selv kan ha full kontroll over de fysiske serverne, og kan gjøre de grepene man selv føler er nødvendige for å sikre seg mot at uvedkommende får tilgang. Det er flere aktører som leverer open source skylagringstjenester, og ved å ta i bruk et slikt system kan man selv velge hvilken kryptering som brukes, hvor mange brukere systemet skal ha, hvor mye plass hver enkelt bruker skal ha og åpner opp

muligheten for å utvikle egne utvidelser som eksempelvis kan være imeplementering av hashing [9][10].

En annen negativ effekt ved å bruke skylagringstjeneste ville være at det hadde gått mye arbeid og tid til å sortere og håndtere all innkommende data. Disse måtte organiseres i mappestrukturer og nøye navngis for å kunne lokalisere dataene igjen senere. Uten noen form for utvikling av egne utvidelser ville ikke denne løsningen tilfredsstilt kravet om hashing av alle filer. Alle aspektene tatt i betraktning viser at en tredjeparts skylagringstjeneste ikke vil være en ideell løsning for vår oppdragsgiver, primært på grunnlag av sikkerhetsrisiko, men også på grunn av ubeleiligheten til en slik løsning kontra ett skreddersydd system.

VPN –Virtual Private Network er en teknikk som skaper ende-til-ende kobling ved bruk av tuneller gjennom usikre nettverk og kan fungere som en mulig løsning for oppgavens krypteringskrav. Det vil gi inntrykk av at brukeren sitter i samme bygg som man er koblet til, noe som også forsterkes ved at man vil få utdelt IP adresse fra det lokale nettverket tunellen er koblet til [29]. Dette kunne fungert som en mulig løsning enten ved at det hadde blitt utviklet en app eller benyttet en tredjepartsapplikasjon fra Play store [61]. Ved å benytte denne fremgangsmåten som løsningen ville man fortsatt vært nødt til å bruke manuell mappestruktur og sortere mottatt media for å systematisere dette. I tillegg blir det ansett som et større arbeid med tanke på innføringer av rutiner og opplæring av personell enn ved å skreddersy et system for å dekke ønsket behov.

Når det gjelder vurdering av andre systemer som enten er utviklet for å fylle tilsvarende behov eller som med noe modifikasjon kunne blitt benyttet til å løse denne problematikken er det ikke tilsynelatende mange muligheter ute på markedet. Det var henholdsvis kun én rapport som skisserte en tilsvarende løsning for samme problematikk, men uten tilgang til hele rapporten vil det ses bort ifra den.

4.3 – Overblikk TLS

TLS –Transport Layer Security er en videretuvikling av SSL 3.0 (Secure Socket Layer) og kort fortalt er det en kryptografisk protokoll som brukes for å sikre kommunikasjonen over usikre linjer, eksempelvis internett. Dette gjøres ved bruk av asymmetriske nøkler for utveksling av symmetriske sesjonsnøkler som brukes videre i kommunikasjonen. Protokollen

er per vår 2015 oppe i versjon 1.2. Hovedpoenget med TLS sikkerhet er beskyttelse av kommunikasjonen mellom klient og tjener som skal sikre mot avlytting, alterering av oversendt data noe som ofte er relatert til MITM angrep og fabrikkering av ugyldige meldinger [14]. RFC 5246 definerer standarden til protokollen og har som hovedmål å tilby data integritet og sikkerhet mellom to kommuniserende parter eller applikasjoner [11]. Protokollen er bestående av to lag: «TLS Record Protocol» og «TLS Handshake Protocol» og blir mer forklart i de to følgende underpunktene.

4.3.1 - TLS Record Protocol/Layer:

Dette er laget som sørger for den faktiske sendingen og mottakelsen av den krypterte traffiken og sørger for at kommunikasjonen mellom partene foregår på en sikker og privat måte ved bruk av symmetrisk krypteringe. Krypteringen av data gjøres med funksjoner som AES eller RC4 hvor sistenevnte har betraktagelige kjente svakheter, utdypt under punktet 4.7 – Sikkerhet og levetid [41].

Å sikre pålitelig kommunikasjon mellom de involverte partene er også oppgaven til dette laget og gjøres ved bruk av en integritetssjekk ved bruk av en kryptert MAC (Message Authentication code) og generes ved hjelp av hashing funksjoner, eksempelvis SHA-1 [11] [41].

4.3.2 - TLS Handshake Protocol/Layer:

Her autentiseres klient og server og de utveksler valg av algoritmer og kryptografisk nøkkelutveksling før selve sendingen begynner. Det benyttes asymmetriske nøkler for å starte sesjonen og deretter symmetriske sesjonsnøkler videre i kommunikasjonen, disse ble generert som et resultat av den asymmetriske offentlige nøkkelen og sertifikatet fra serveren.
Utover dette foreligger det tre konkrete sikkerhetsprinsipper i bunn for dette laget:

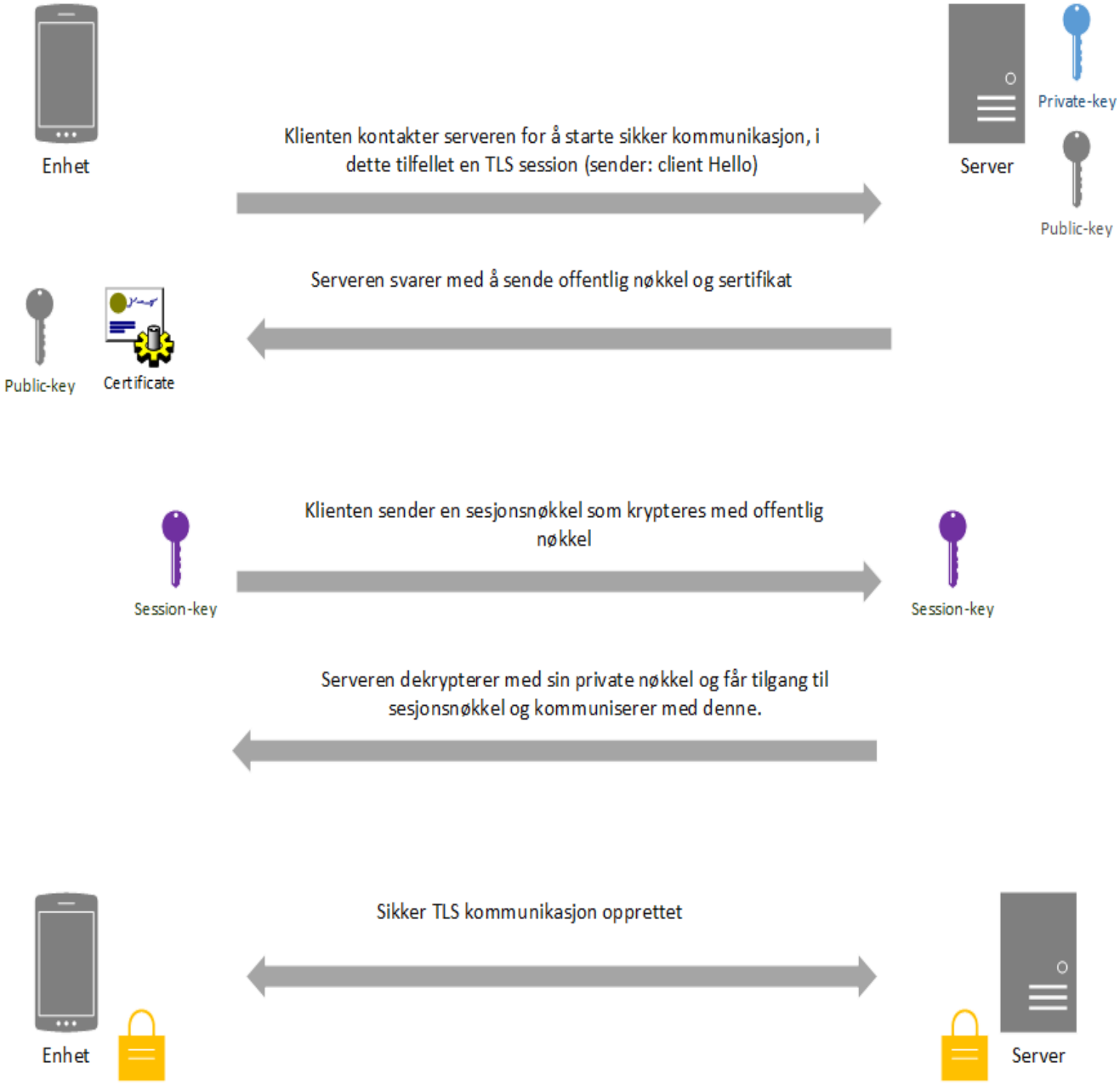
- Asymmetrisk –eller offentlig nøkkel kryptografi for autentisering av involverte parter som gjøres ved bruk av RSA, DSA eller «anon».
- Hindre at eventuelle lyttere på kommunikasjonen ikke kan fange opp utveksling av protokollhemmeligheter slik som valg av algoritmer, nøkkelutveksling, parametere og sertifikater.
- Pålitelighet sørges for i form av at ingen angripere, MITM, kan alterere forhandlingsdetaljene, dette ved at TLS detekterer «støy» eller

modifikasjonsforsøk om det oppstår og vil resultere i at forbindelsen termineres [11].

Grunnet TLS' mangel på implementert feilhåndteringsprotokoll avhenger den seg derfor av TCP protkollen i den grad at TCP må foreta håndteringen av typiske nettverksfeil og pakker som går tapt sendes på nytt [12] [13]. Utover dette vil feilhåndteringen foregå slik at om kommunikasjonen termineres vil alle inngående parter varsles og partene glemmer all informasjon forbundet med den terminerte sesjonen.
Det derfineres noen konkrete mål som oppnås ved bruk av TLS protokollen i følge RFC 5246 standarden og er som følgende:

- Det skal oppnåes kryptografisk sikkerhet som kan garantere kommunikasjonen mellom involverte parter.
- Rammeverk som tilbyr utvidelsesmuligheter for at nye offentlige nøkler (public keys) og krypteringsmuligheter kan implemeneres etter behov istedet for at det hver gang utvikles nye protokoller og utskiftning av sikkerhetsbibliotek.
- CPU effektivisering ved bruk av sesjonscaching for å lette CPU lasten nødvendig for å foreta tunge kryptografiske operasjoner noe som vil begrense nettverksaktivitetet og antallet oppkoblinger som må etableres fra bunn.

4.4 – TLS prosessen



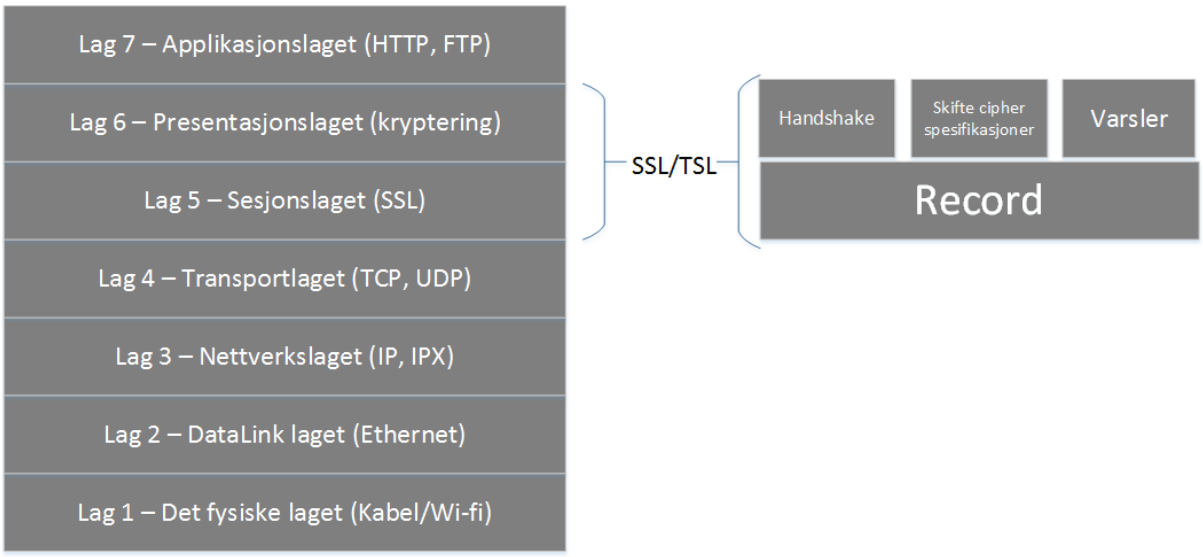
Figur 3: Illustrerer enkel virkemåte for etablering av TLS kommunikasjon mellom enhet og server. Utarbeidet i Microsoft Visio 2013.

1. «Handshake» prosedyren starter med at klienten («Enhet» i illustrasjon på forrige side, og omtales videre som dette) forespør serveren etter en sikker kobling. Dette gjøres ved at det sendes ut en «ClientHello» melding for å initialisere TLS kommunikasjonen, og dette gjøres for å bli enige om algoritmer, utveksle tilfeldige verdier (til bruk i algoritmene) og sjekke etter om det er tidligere sesjoner som kan gjenoptas. På dette tidspunktet har ikke enhet noen nøkler, men kan ha sitt eget sertifikat [11] [12] [25].
2. Sertifikatet fra serveren sendes som en såkalt «sertifikatsmelding» som benyttes for å verifisere identiteten ovenfor enheten og må sendes når det er blitt avtalt å benytte autentisering. Dette sertifikatet sendes rett etter «ServerHello» meldingen, med unntak av når «anon» benyttes. Avsnittet 4.6 – Sertifikater/Digitale signaturer går mer i detalj om bruk og betydningen av sertifikater. «ServerKeyExchange» melding kan bli sendt om det er påkrevd, eksempelvis om serveren ikke har noe sertifikat eller sertifikatet kun benytes til signering. Hvis serveren blir autentisert kan den be om sertifikat fra enheten for å kunne verifisere dens identitet, men dette vil avhenge av hvilke cipherpakke som er valgt og hvorvidt denne krever det. «ServerHelloDone» melding vil indikere avslutningen på dette steget [11] [12] [25].
3. Enheten vil etter å ha mottat serversertifikatet inspisere dette for å verifisere gyldigheten og at det hører til den forespurte tjenesten. Om det ble sendt et «CertificateRequestMessage» så er enheten tvunget til å sende sitt sertifikat, som nevnt i punktet ovenfor [11] [12] [25].
4. En «ClientKeyExchange» melding blir sendt og dens innhold avhenger av hvilke offentlig nøkkel algoritme som ble valgt i prosessen mellom «ClientHello» og «ServerHello». Om enheten sendte et sertifikat med mulighet for signering, vil det bli sendt en digital signerings melding for å verifisere besittelsen av den private nøkkelen i sertifikatet. På dette stadiet vil det sendes en «ChangeCipherSpec» melding av enheten, hvor enheten kopierer de avventede cipher detaljene til å være de gjeldende og avslutter med å sende «FINISH» melding med bruk av de nye algoritmene, nøklene og hemmelighetene. Dette er fra nå den gjeldende symmetriske sesjonsnøkkelen [11] [12] [25].
5. Serveren responderer med å sende sin egen «ChangeCipherSpec» melding, kopierer de avventede cipher detaljene over til å være de gjeldende, slik enheten gjorde i forestående punkt, og sender sin «FINISH» melding med de nye cipher detaljene.

Etter fullførelse av dette steget er «Handshake» prosedyren fullført og overføringen kan begynne [11] [12] [25].

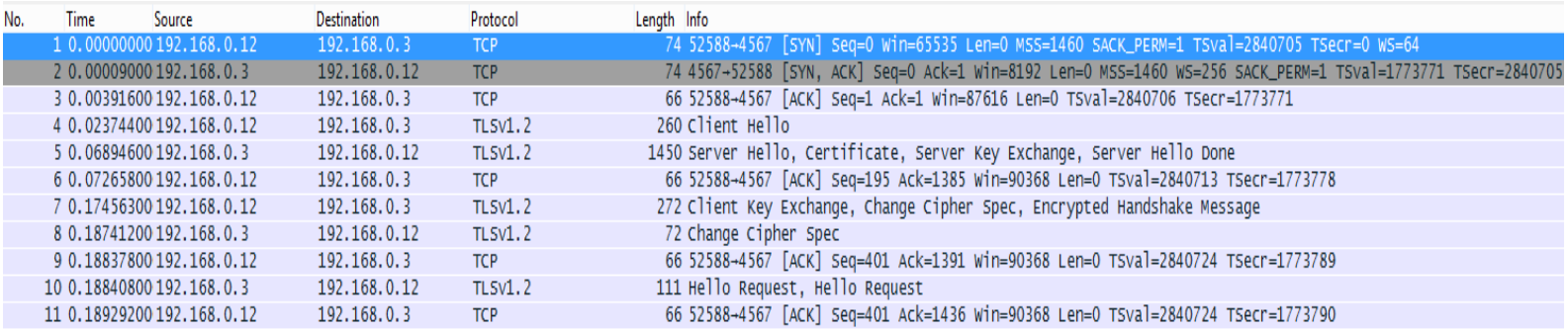
4.5 – TLS Dypdykk

OSI-modellen (Open Systems Interconnection) er en konseptuell modell designet for å standardisere og karakterisere de forskjellige funksjonalitetene innenfor kommunikasjonssytemer ved å dele disse opp i syv abstrakte lag. Den er med her for å illustrere hvor TLS inngår i denne abstrakte oppdelingen og samsvaret mellom disse som er nødvendig for at protokollen skal være operativ [27].



Figur 4: Illustrerer OSI-modellen til venstre og til høyre en oversikt over hvilke lag SSL/TLS benytter seg av og hvilke lag dette gjelder i OSI-modellen. Utarbeidet i Microsoft Visio 2013.

Illustrasjonen over viser hvor TLS befinner seg fordelt over de to lagene, sesjons –og presentasjonslaget, hvor det er innunder førstnevnte TLS Record Protocol/Layer opererer, utdypt under punktet 4.3.1 – TLS Record Protocol/Layer. I presentasjonslaget opererer både TLS sin «handshake» prosess sammen med varsler og utskiftning av cipher detaljer. Siden TLS opererer mellom applikasjonslaget (lag syv) og transportlaget (lag fire) åpner dette opp muligheten for at sikkerhetsprotokollen kan benytte seg av andre protokoller, deriblant TCP som nevnt i 4.3 – Overblikk TLS [26].



No.	Time	Source	Destination	Protocol	Length	Info
1	0.00000000	192.168.0.12	192.168.0.3	TCP	74	52588→4567 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 SACK_PERM=1 TSval=2840705 TSecr=0 WS=64
2	0.00009000	192.168.0.3	192.168.0.12	TCP	74	4567→52588 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1 TSval=1773771 TSecr=2840705
3	0.00391600	192.168.0.12	192.168.0.3	TCP	66	52588→4567 [ACK] Seq=1 Ack=1 Win=87616 Len=0 TSval=2840706 TSecr=1773771
4	0.02374400	192.168.0.12	192.168.0.3	TLSv1.2	260	Client Hello
5	0.06894600	192.168.0.3	192.168.0.12	TLSv1.2	1450	Server Hello, Certificate, Server Key Exchange, Server Hello Done
6	0.07265800	192.168.0.12	192.168.0.3	TCP	66	52588→4567 [ACK] Seq=195 Ack=1385 Win=90368 Len=0 TSval=2840713 TSecr=1773778
7	0.17456300	192.168.0.12	192.168.0.3	TLSv1.2	272	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
8	0.18741200	192.168.0.3	192.168.0.12	TLSv1.2	72	Change Cipher Spec
9	0.18837800	192.168.0.12	192.168.0.3	TCP	66	52588→4567 [ACK] Seq=401 Ack=1391 Win=90368 Len=0 TSval=2840724 TSecr=1773789
10	0.18840800	192.168.0.3	192.168.0.12	TLSv1.2	111	Hello Request, Hello Request
11	0.18929200	192.168.0.12	192.168.0.3	TCP	66	52588→4567 [ACK] Seq=401 Ack=1436 Win=90368 Len=0 TSval=2840724 TSecr=1773790

Figur 5: Viser skjermdump av avlytting på kommunikasjonen mellom enhet og server i «handshake» prosessen. Identiteter: 192.168.0.3 (server) og 192.168.0.12 (enhet). Utledet ved bruk av Wireshark.

Illustrasjonen ovenfor viser etablering av kommunikasjonen mellom enhet og server i systemet, fra start til fullført utveksling hvor sending kan begynne. Dette er det faktiske hendelsesforløpet slik det fungerer i systemet som er blitt utviklet og viser samarbeidet mellom de forskjellige protokollene i OSI-modellen. De viktigste linjene ovenfor er som følgende:

- Linje 4: «ClientHello» fra enhet til server som vil være forespørselen til serveren om å etablere TLS kommunikasjon.
- Linje 5: Serveren svarer med «ServerHello» og starter «handshake» prosessen. Dette etterfølges da med utveksling av sertifikater, valg av algoritmer, og hemmeligheter som utdypt i avsnittet 4.4 – TLS prosessen. Denne utvekslingen avsluttes ved «ServerHelloDone».
- Linje 6: Utveksling av flagg ved bruk av TCP protokollen, hvor den faktiske TLS prosessen fortsetter på neste linje.
- Linje 7: Utveksler sesjonsnøkkelen og sender meldingen som varsler om å endre cipher spesifikasjonene fra å være midlertidige til å være de gjeldende. Avsluttes med «FINISH» melding.
- Linje 8: Serveren foretar samme prosessen som enheten, nevnt i forrige linje. Avsluttes også med «FINISH» melding.
- Linje 10: På dette stadiet er TLS forbindelsen opprettet og all videre kommunikasjon vil nå foregå i cipher tekst som dekrypteres med den genererte sesjonsnøkkelen.

4.6 – Sertifikater / Digitale signaturer

Det benyttes digitale sertifikater for å autentisere eierskapet av en offentlig nøkkel, det vil si identiteten til den forespurte tjenesten og er en viktig del av tilkoblingsprosessen i TLS siden dette er med på å skape autentisitet. Sertifikater skal sørge for autentiseringen av server og klient og vil kun være gyldig i bestemte tidsperioder. Navnet på utgiver, gyldigheten, navnet

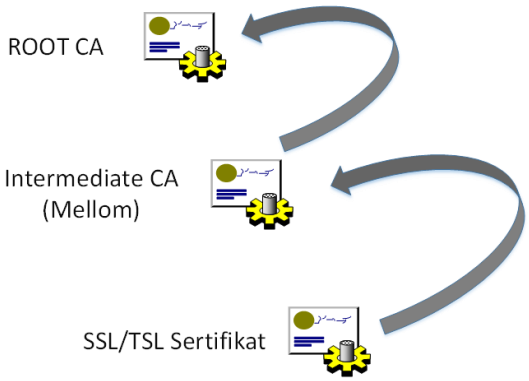
på innehaveren, offentlig nøkkel og innhold som identifiserer nivået er blant detaljene som inngår i et slikt sertifikat og vil kun være gyldig om det er signert av en CA (Certificate Authority) [24]. Slike CA-er tilbyr som regel to typer tjenester hvorav disse vil være å enten selge forhåndsgenerte sertifikater gjennom årlige abonnementer eller at man sender inn et eget produsert sertifikat som de verifiserer og signerer.

«Trusted certificate» er digitale sertifikater som er signert av en CA, blant mange eksisterende er dette noen eksempler [23]:

- Comodo
- Symantec Group
- Go Daddy Group
- Digicert
- Globalsign

I tillegg til dette har man muligheten for å benytte «self-signed certificate» som vil være, som navnet tilsier, et eget generert sertifikat som man selv signerer. Dette vil fungere utmerket i private sammenhenger siden man kan stole på sine egne sertifikater, men utenforstående forespørsler som gjøres på et slikt sertifikat vil gi varsel, tiltross for at det ikke nødvendigvis er noen grunn for uro. I løpet av bacheloroppgaven vil det bli benyttet et slikt «self-signed certificate» av praktiske årsaker for å slippe å kjøpe ett av en CA, samtidig som at det i praksis vil fungere uten merkbar forskjell.

«Chain of Trust» er ett begrep brukt i sertifikat sammenheng og omhandler tillitsforholdet mellom de involverte identitetene hvor «ROOT CA» vil være selve tillitsankeret i kjeden. Dette prinsippet baserer seg på at hvert ledd er avhengig av hverandre, ved at sikkerheten ligger i at de forskjellige nivåene validerer hverandre. En SSL/TLS forbindelse vil være kryptert og fungere uten en slik kjede tilstede og man vil heller ikke være helt sikret mot falske sertifikater tiltross for «chain of trust» [24].



Figur 6: Viser tillitsforholdene mellom partene i «Chain of trust» kjeden og viser hvordan partene avhenger av hverandre. Utarbeidet i Microsoft Visio 2013.

4.7 – Sikkerhet og levetid

Det er implementert mange sikkerhetstiltak som en del av TLS protokollen, hvorav disse er:

- Beskyttelse mot nedgradering til eldre protokollversjoner eller svakere cipherpakker, som oftest kalles et «version rollback attack». Dette er en av de største risikoene for webservere siden det i disse tilfellene vil være vektlagt kompatibilitet over sikkerhet, og vil ikke være tilstede i dette prosjektet siden det i utviklingen blir valgt konkret versjon og cipherpakke. Sett bort ifra implementasjonen i dette prosjektet, så vil man i andre tilfeller kunne risikere at en angriper påtvinger en protokollversjon eller cipherpakke med kjente svakheter [12] [34].
- «HMAC (Keyed-Hashing for Message Authentication)» brukes for autentisering av meldinger ved hjelp av hashing funksjoner som eksempelvis SHA og MD5 [12] [35]. Avsnittet 4.3.1- TLS Handshake Protocol/Layer nevner hva en slik melding vil bidra med.

Tiltross for mange tiltak som gjøres for å opprettholde høy sikkerhet vil man på lik linje med alle andre protokoller oppleve svakheter og angrepsforsøk. Dette kan være svake cipherpakker som inneholder for lav kryptering som allerede er knekt med dagens prosessorkraft eller det kan omhandle direkte angrepsmetoder.

4.7.1 - Heartbleed

«Heartbleed» er den mest nylige omtalte svakheten som ble avdekket for bibilioteket fra OpenSSL hvor det viste seg å være mulig å stjele informasjon som opprinnelig skulle være

beskyttet med SSL/TLS kryptering. Denne metoden gjorde det mulig å uthente informasjon om nøkler benyttet til x.509 sertifikater, brukernavn, passord, e-post adresser og annen informasjon som ligger på serveren. Svakheten berører versjonene 1.0.1 til 1.0.1f og den beste foreslåtte løsningen er å benytte en nyere versjon av protokollen [15].

4.7.2 – Renegotiation Attack

«Renegotiation attack» ble oppdaget i august 2009 og er nå allerede blitt en velkjent risiko, men i motsetning til «Heartbleed» som berørte veldig spesifikke versjoner gjelder denne for SSL 3.0 og alle versjoner av TLS. Det ble kjent at det var mulig å injeksere «plaintext» kommandoer som i praksis betyr at man kan sende forespørseler til serveren som ikke kom fra klienten, men med klientens legitimasjon. For serveren virker det derfor som om forespørselen kommer fra klienten, men i realiteten kommer fra en angriper og er muliggjort ved at svakheten utnytter reforhandlingsfunksjonen implementert som en del av protokollen. Denne vil, med gode intensjoner, bli benyttet for at partene kan reforhandle seg frem til nye parametere uten å måtte etablere en helt ny sesjon, og det er derfor mulig å sikre seg mot angrepet ved å deaktivere denne funksjonen [16].

4.7.3 – RC4 Attack

«RC4 attack» berører alle versjoner av TLS som benytter cipherpakker som inkluderer RC4 krypteringen og har vært kjent som en svak cipher i mange år. Det er derfor ikke anbefalt å benytte denne av Enisa, som er den Euorpeiske foreningen for nettverk –og informasjonssikkerhet. I grove trekk vil denne angrepsmetoden gjøre det mulig å uthente hemmeligheter ved hjelp av plaintekst injeksjoner, på samme måte som «Renegotiation attack». Dette kan forekomme eksempelvis ved at man påtvinger en terminering av sesjonen som foregår, noe som tvinger applikasjonen til å gjøre en gjenoppkobling og sender dermed sensitive informasjonskapsler og passordinformasjon som da kan plukkes opp av en angriper. Denne typen angrep kan man sikre seg mot ved å benytte enten AES-GCM eller CBC cipherpakke, men må da kontrollere at den CBC-modusen er beskyttet mot kjente «BEAST» og «Lucky 13» angrep [12] [17].

Levetiden og anbefalinger i forbindelse med valg av nøkkel –og algoritmetyper er også et viktig poeng når man undersøker TLS sikkerheten siden sikkerheten avhenger hovedsakelig av disse konseptene. Både amerikanske National Institute of Standards and Technology

(NIST) og European Union Agency for Network and Information Security (Enisa) har laget en oversikt over anbefalinger til bruk av algoritmer, nøkkelstørrelser og en ca. antatt levetid for disse. Den Europeiske standarden vektlegges over den amerikanske, men tok med noe data for å benytte som referansepunkter opp i mot den europeiske.

- **Block cipher:** Anbefalt å benytte AES-128, men for langsiktig bruk anbefaler Enisa AES-256 [41], hvorav NIST anslår en levetid til år 2030 med bruk av AES-128 [18].
- **Hash funksjoner:** SHA-256 for kortsiktig bruk, og for langsiktig bruk anbefaler Enisa SHA-512 [41].
- **Public key:** 256-bit «elliptic curves» for kortsiktig langtidsbruk, og 512-bit for lenger langtidsbruk [41].
- **RSA:** Per i dag sikkert med bruk av 1024-bit, men anbefalt så høyt som 3072-bit for kortsiktig langtidsbruk.

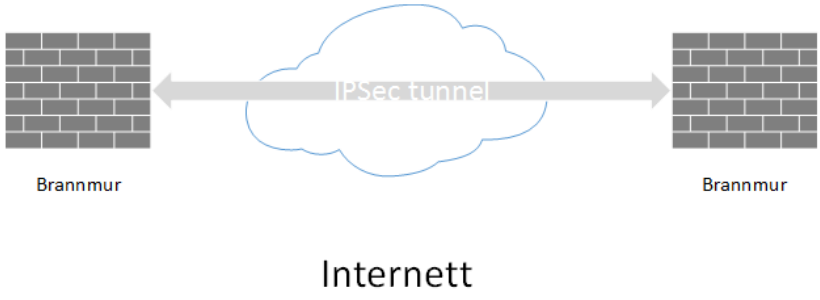
4.8 – Andre protokoller

Ipsec og SSH var to andre sikkerhetsprotokoller som ble undersøkt for å avgjøre hvorvidt disse var så gode alternativer til TLS at det heller burde benyttes en av disse fremfor TLS. Undersøkelsen av disse tre løsningene vil ikke omhandle hver eneste detalj, men gi et innblikk i virkemåte, forskjeller og bistå med å komme frem til en konklusjon om TLS er det beste valget for dette prosjektet eller ikke.

4.8.1 - IPsec

IPsec (Internet Protocol Suite) sørger for sikkerhet i nettverkslaget, lag tre i OSI-modellen (se fig. 4) og skiller seg fra blant annet TLS og SSH siden disse benytter applikasjonslaget for sikkerhet. For IPsec er det fordelsmessig at det benyttes lag tre siden det da ikke kreves noen omgjøring av applikasjonene for å dra nytte av IPsecs sikkerhetsfordeler. Bruken av IPsec baserer seg hovedsakelig for sikkerhet innenfor VPN, se 4.2 – Mulige eksisterende løsninger for utbroderende informasjon om VPN, men kan også brukes til generelle løsninger. Protokollen er blitt definert i flere RFC-er, mens denne undersøkelsen av protokollen forholder seg til standarden RFC 4301 [41].

Tunnelmodus og transportmodus er de to grunnleggende modusene som IPsec protokollen kan bli brukt til og fungerer litt forskjellig. Helt grunnleggende vil sikkerheten basere seg på å fungere som en tilleggsbehandling til vanlig IP pakker og foretar den nødvendige prosesseringen for å kryptere de før de sendes videre som vanlig. I tunnelmodus gjøres sikkerheten ved veiskiller som brannmurer eller rutere og gjøres på vegne av endepunkt tjenere. Dette gjøres ved at hver enkelt IP pakke behandles som en ny last der hele pakken behandles som en ny last utenpå den eksisterende IP pakken med sin egen header, kalt «outer header». Den originale pakken, som da befinner seg «innerst» vil da være innkapslet inni den ytre IP pakken og er sikret på denne måten [41] [30].



Figur 7: Illustrerer tunnelmodus ved at det lages en «tunnel» gjennom et usikkert nettverk. Utarbeidet med Microsoft Visio 2013

Figur 8: Illustrerer tunnelmodus, hvordan IP pakken blir innkapsulert og at det legges til en ny header utenpå den originale. Utarbeidet i Microsoft Visio 2013.

Transportmodus brukes som regel når det ønskes å oppnå full ende-til-ende kryptering og tilbyr sikkerhet ved å kryptere hver last på en slik måte at det ikke vil berøere headeren. Se figur 9.

Figur 9: Illustrerer hvordan en IP pakke vil se ut med bruk av transportmodus. IPsec headeren legges inn som et tillegg i den orignale pakken uten noen større grad for innblanding. Utarbeidet i Microsoft Visio 2013.

IPsec belager seg på en kombinasjon av mange underliggende protokoller for å fungere og tilbyr egenskaper som autentisering, integritet, dataopphavs autentisering, konfidensialitet og

nøkkelutveksling [41] [19]. «Security Policy Database (SPD)» er en del av hver IPsec implementasjon hvor hver oppføring definerer regler for spesifikke typer trafikk og hver oppføring i denne SPD-en peker på enten en eller et flertfall «Security Associations (SA)». Det er disse SA-ene som inneholder informasjon om blant annet kryptografiske nøkler og IV-er som enten kan bli gjort manuelt eller ved bruk av automatiserte metoder som «Internet Key Exchange (IKE)» [41].

Anbefalinger fra Enisa når det kommer til hvilke algoritmer som bør benyttes avhenger litt av hva man ønsker å oppnå, siden de forskjellige algoritmene tilbyr forskjellige egenskaper. Om det eneste som ønskes er autentisering kan man benytte IPsec implementert med «Authentication Header (AH)» eller «Encapsulation Security Payload (ESP)» og har følgende valg av MAC algoritmer[43]:

- HMAC-SHA2-256
- HMAC-SHA-384
- HMAC-SHA2-512

Om det ønskes konfidensialitet så bør IPsec implementeres ved bruk av ESP i en kombinasjon med en av de overnevnte algoritmene og av følgende [41]:

- AES-CTR
- CAMELLIA-CTR

4.8.2 – SSH

SSH –Secure Shell er en transportprotokoll utviklet for å sikre datakommunikasjon og var tiltenkt å ersatte usikre fjentilkoblingsprotokoller som telnet og FTP, men har med tiden blitt et mer generelt verktøy for å ha sikre koblinger mellom to tjenere. SSH benytter mange av de samme sikkerhetsprinsippene som TLS og tilbyr sterk kryptering, kryptografisk autentisering og integritetsbeskyttelse. Hovedprinsippet til SSH sin virkemåte er at det opprettes en sikker kanal i en klient/tjener arkitektur over en usikret linje og som da kan benyttes for fjerntilgang for kommandolinje eller filoverføring. SSH finnes i to forskjellige utgaver der SSHv1 med tiden har blitt kompromittert og inneholder en del designfeil og anbefales derfor ikke å brukes. Derimot er SSHv2 ansett som trygg å benytte og defineres i standarden RFC 4253 [33] [41] [32] hvor OpenSSH er den mest benyttede implementasjon av SSH protokollen [44].

Nøkkelutveksling innen bruk av SSH er basert på Diffie-Hellman algoritmen hvor autentisering av server gjøres ved en kombinasjon av algoritmen sammen med en signatur. Det er mulig med autentisering av tjenere, men da må dette gjøres i en selvstendig implementasjon fra en annen standard hvor denne metoden da benytter seg av enten passord, offentlig nøkkel kryptografi (DSA, RSA, X.509), «interaktivt-tastatur» utfordring eller en «GSSAPI», hvorav sistenevnte åpner muligheten for bruk av eksterne mekanismer, eksempelvis [41].

«Fingerprints» er noe som brukes innenfor SSH og er et øyeblikksbilde av klientens offentlige nøkkel, som vanligvis er 128-bit, og benyttes for å autentisere klienten. Kommunikasjonskryptering gjøres med bruk av 3DES, AES, IDEA eller det kan implementeres en annen algoritme [25] [33].

Listen med cipherpakker som tilbys til SSH er omfattende, men av de mest sikre og anbefalte av Enisa er følgende:
“

- AES128-ctr with HMAC-SHA2-256 eller HMAC-SHA2-512
- AES192-ctr with HMAC-SHA2-256 eller HMAC-SHA2-512
- AES256-ctr with HMAC-SHA2-256 eller HMAC-SHA-512
- AEAD_AES_128_GCM
- AEAS_AES_256_GCM “[41].

4.9 – TLS versjoner og Cipher suites

Nedenfor følger alle protokoll versjoner av TLS og tilhørende cipherpakker som tilfredsstiller oppgavens og oppdragsgiver krav.

4.9.1 - Protokoller [20]:

- TLSv1
- TLS1.1
- TLSv1.2

4.9.2 - Cipherpakker [20]:

CBC modus:	GCM modus:
TLS_DHE_DSS_WITH_AES_128_CBC_SHA256	TLS_DHE_DSS_WITH_AES_128_GCM_SHA256
TLS_DHE_DSS_WITH_AES_256_CBC_SHA256	TLS_DHE_DSS_WITH_AES_256_GCM_SHA384

TLS_DHE_RSA_WITH_AES_128_CBC_SHA256	TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256	TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
TLS_DH_anon_WITH_AES_128_CBC_SHA256	TLS_DH_anon_WITH_AES_128_GCM_SHA256
TLS_DH_anon_WITH_AES_256_CBC_SHA256	TLS_DH_anon_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256	TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384	TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384
TLS_RSA_WITH_AES_128_CBC_SHA256	TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256	TLS_RSA_WITH_AES_256_GCM_SHA384

Tabell 5: Viser en oversikt oversikt over alle cipherpakkene tilgjengelige som oppfyller kravene fra oppdragsgiver, med det ment at det er minimum AES-128.

4.9.3 – Forklaring versjoner

TLSv1.0: Denne versjonen av TLS er basert på SSL 3.0 og har tilsvarende funksjoner og sikkerhetsegenskaper som inngår innenfor denne protokollen. Forskjellene er ikke veldig store, men relevante nok til at det ble gitt en egen versjon, deriblant en mekanisme som gjør det mulig å kjøre kommunikasjonen ned til SSL 3.0 [36]. Noen svakheter med denne versjonen kort oppsummert: «BEAST attack», «FREAK attack» (gjelder om OpenSSL bibilioteket benyttes), «Version Rollback attack», «POODLE attack» (om kommunikasjonen nedgraderes til SSL 3.0) og svakhet i protokollens CBC modus [37]. Denne versjonen vil også være utsatt for RC4 svakheten om denne flytchifferen benyttes.

TLSv1.1: Dette er en oppdatert versjon av den foregående versjon 1.0 med små sikkerhetsmessige forbedringer, hvorav de største forskjellene er som følgende:

- Beskyttelse mot CBC angrep er blitt implementert ved å foreta utskiftninger med «Initialization vector (IV)» [38].
- Foretatt endringer i forbindelse med hvilke varselmeldinger som brukes, dette for å øke sikkerheten og redusere sjansen for CBC angrep [38].
- «IANA (Internet Assigned Numbers Authority)» registre er definert for protokoll parametere [38].
- Termineringer som oppstår på et for tidlig stadie vil ikke kunne restartes igjen, og vil redusere angrepsmulighetene [38].
- Noen flere endringer av mindre viktighet.

Denne versjonen kan også selvfølgelig berøres av RC4 svakheten på samme måte som versjonen nevnt ovenfor, men utover dette ingen «store» kjente sikkerhetsproblemer.

TLSv1.2: Er hittil den nyeste versjonen av TLS som er blitt lansert og inneholder blant annet forbedringer i forbindelse med implementasjon fleksibilitet og noe endringer i hvordan forhandlingsprosessen av kryptografiske algoritmer forekommer. Større endringer av viktighet:

- MD5/SHA-1 hashing funksjoner er blitt erstattet med cipherpakker som benyttes SHA-256 [11].
- Signerte elementer inneholder nå et felt som spesifiserer hashing algoritmen som benyttes [11].
- Økt støtte for autentisert kryptering [11].
- Gjort opprydding i hvilke muligheter klient/tjener har for å spesifisere hvilke hash –og signaturalgoritmer som aksepteres [11].
- Andre vesentlige, men mindre viktige sikkerhetsforbedringer.

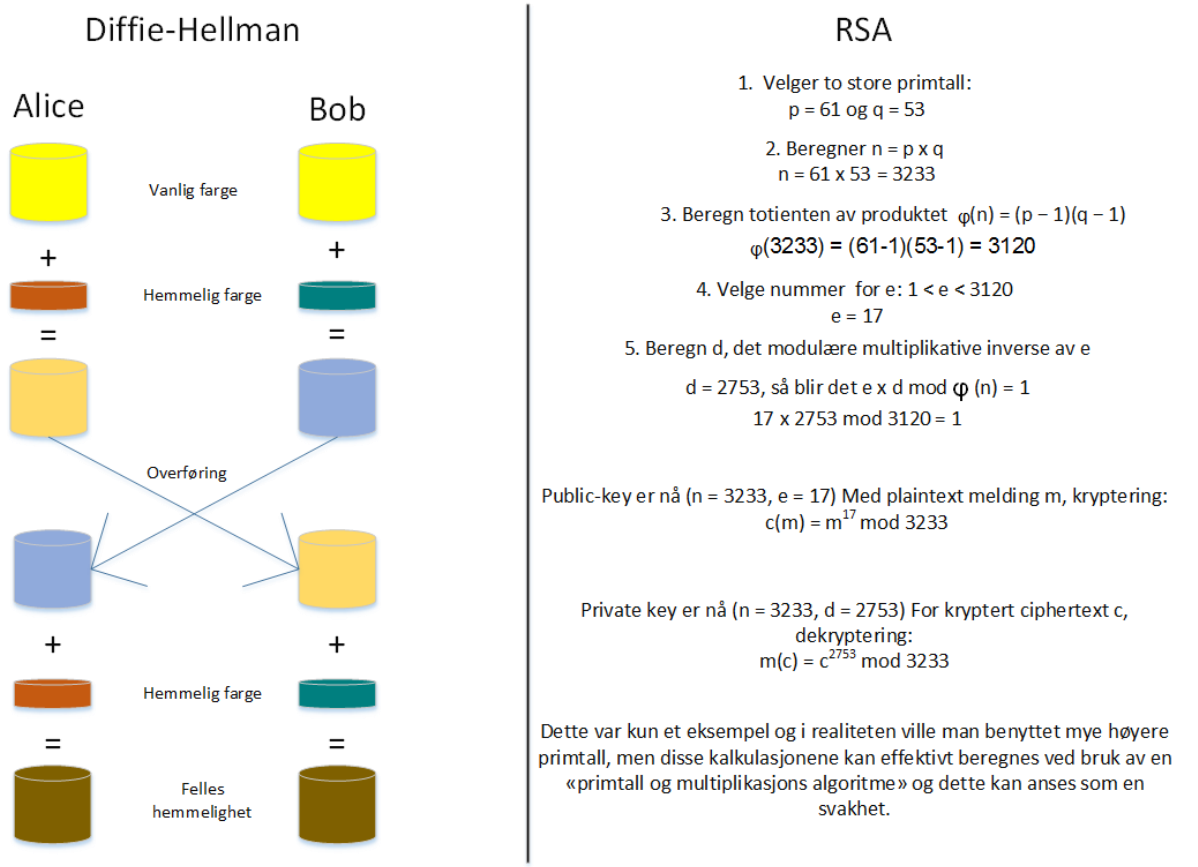
4.9.4 – Forklaring cipherpakker

Diffie-Hellman er en mye brukt nøkkelutvekslingsalgoritme og tillater at to tjenere som ikke vet om hverandre kan utveksle en hemmelighet. Dette gjøres ved en kombinasjon av en privat nøkkel og den ene partens offentlige nøkkel som både sender og mottaker må inneha for at de skal kunne bruke disse til å produsere den delte hemmeligheten [42]. Kommer i flere varianter hvorav disse er:

- **DH:** Den originale Diffie-Hellman modusen.
- **DHE:** Mulighet for å kjøre «Ephemeral» modus av Diffie-Hellman som kort fortalt betyr at det vil bli generert et nytt nøkkelpar for hver nye melding og sendes ved bruk av den originale nøkkelen [42].
- **ECDHE:** «Elliptic curve» versjon av Diffie-Hellman , hvor det brukes algebraiske kurver for å produsere nøklene.

Hovedforskjellen mellom de forskjellige versjonene ligger i at ordinær DH eller DHE benytter modulær aritmetikk mens, som nevnt, benytter ECDHE algebraiske kurver i tillegg, hvorav sistenevnte er anbefalt av blant annet Enisa [41]. Det som er unikt med DHE og ECDHE er at disse tilbyr «forward secrecy» som vil bety at om en sesjonsnøkkel blir kompromittert vil ikke den tilhørende private nøkkelen i det settet (privat og offentlig) bli kompromittert i fremtiden [22] [23].

RSA (Rivest-Shamir-Adleman) er en av de første praktiske offentlig nøkkel kryptosystemene og er vidt brukt for å sikre datakommunikasjon.



Figur 10: Illustrerer virkemåte for de forskjellige nøkkelutvekslingsalgoritmene. Utarebeidet i Microsoft Visio 2013.

Digitale signaturer benyttes for å kunne tilby autentisering og integritet. Blant de overnevnte cipherpakkene er det hovedsakelig tre alternativer: DSA, ECDSA og RSA. «Anon» er også et alternativ på noen av cipherpakkene, men vil se bort ifra disse siden dette betyr at det brukes anonym Diffie-Hellman uten autentisering. DSA inngår innunder standarden «Digital Signature Standard (DSS)» definert av NIST, mens ECDSA er en variant av DSA som benytter elliptiske kurver for signering [13].

CBC: «Cipher block chaining» er den mest brukte modusen for block cipher og består hovedsakelig av en sekvens med bits, som blir kryptert som en hel blokk [41][21].

GCM: «Galois/Counter Mode» er en forbedring for CWC og bruker en kombinasjon av «GMAC» hvorav de underliggende hashing funksjonene er basert på polynome felter. Denne modusen er mye brukt og er et anbefalt valg for blant annet IPsec, SSH og TLS [41] [21].

Forskjellen mellom disse er hovedsakelig hva som blir sendt gjennom block cipher og hva som behandles med XOR [21].

4.10 – Implementasjon

Basert på analysearbeidet som ble foretatt viser det seg at det bør benyttes TLS som protokoll sammenlignet med de andre protokollene grunnet sikkerhetsdetaljer, implementasjonsdetaljer og vanlig sedvane. Tekniske grunner for valg av TLS er dets egenskaper og at det er enkelt å implementere denne uavhengig av type system.

SSH ble valgt bort, hovedsakelig siden denne protokollen er utviklet for å tilby kommandolinjetilgang for fjerntilkoblede enheter, men også fordi man må implementere flere standarder for å oppnå ønsket funksjonalitett, eksempelvis autentisering av klienter. Sikkerhetsmessig er SSH et godt alternativ med sine implementerte sikkerhetsaspekter og ved bruk av GCM er man også sikret mot RC4 angrep. De anbefalte algoritmene fra Enisa ville tilfredsstilt kravene fra oppdragsgiver med hensyn på oppgaven, men grunnet større kompleksitet rundt implementering og bruk ble ikke SSH et aktuelt alternativ.

IPsec som hovedsakelig benyttes for sikkerhet innenfor VPN ble analysert som et mulig alternativ, men ble tidlig valgt bort hovedsakelig på grunnlag av høy kompleksitet vedrørende implementering og bruk i et slikt system som oppgaven etterspør. Sekundært ville ikke IPsec fungert som et godt alternativ tatt i betrakning all ekstra overhead som produseres, spesielt i tunnelmodus. Noe som ville hatt en negativ effekt med tanke på at systemet skal fungere under dekningsforhold som er dårlige og/eller berørt av lave overføringshastigheter hvor man bør ha minst mulig overhead for å øke muligheten for suksessfull overføring.

Det naturlige valget for TLS versjon falt på 1.2, tiltross for små forskjeller fra TLSv1.1 så inneholder TLSv1.2 noen forbedringer og vil være det beste valget med tanke på at systemet skal være sikkert i lang tid fremover. Når det kommer til valg av cipherpakket ble det avgjort å benytte «TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA256» på grunnlag av hva analysearbeid har belyst, etter drøftinger internt i gruppen med oppdragsgiver og sikkerhetskonsulent. «Elliptic curve Diffie-Hellman» anses som den mest sikre nøkkelutvekslingsalgoritmen og ved hjelp av tester utført tidlig i utviklingen viste det seg å fungere svært effektivt. RSA benyttes for signering og autentisering av identiter, og per i dag er 768-bit den høyeste RSA verdien som har blitt knekt, noe som krevde 1500 CPU år, tilsvarer flere hundre maskiner over en tidsperiode på to år [39]. På dette grunnlaget ble det avgjort å benytte RSA verdi på 2048 noe som anses som svært sikkert og «uknekkelig» i nærmeste fremtid. GCM ble valgt til fordel for CBC primært på grunnlag av at den skal fungere svært effektivt og sikkert, men også fordi ved bruk av AES-GCM sikrer man seg mot RC4 angrep. Vedrørende implementasjonsdetaljer for hashing algoritmer har det blitt valgt å benytte to: SHA256 og MD5. Tiltross for at MD5 anses å være kompromittert benyttes denne for å kunne generere hashing raskt, mens SHA256 vil stå for den garanterte sikkerheten samt at viktigheten av hashing er så høy at systemet måtte inneholde redundans på det området. Se *vedlegg S* for praktisk implementasjon av hashing funksjonalitet.

4.11 – Sikkerhetsvurdering av platformer

Oppgaven definerer også at det skal ses på sikkerhetsvurdering av platformer i den grad at det kan avgjøres om det kan brukes forbrukerprodukter fra Google eller om man må ha kontroll over all hardware. Det ble derfor foretatt en analyse av personvern og vilkår for de forskjellige enhetene gruppen hadde og Google sitt operativsystem for å avgjøre i hvor stor grad de påkrevever seg retten for å uthente informasjon om bruk, koordinater eller andre essentielle data som ikke bør deles med de eller deres tredjeparter. I denne analysen gjøres det vurdering kun for appen og mobile enheter siden serverapplikasjonen vil fungere i et Windows miljø som alle andre av deres systemer gjør.

4.11.1 – Operativsystem Android

Android er et Google utviklet operativsystemet og vil være miljøet som appen på de mobile enhetene vil kjøre i, og det er derfor et godt sted å starte med sikkerhetsvurderingen. Dette fordi Google kan ha andre personvernregler for sitt operativsystem enn produsentene har for

enhetene. Det kunne blitt brukt utallige timer på å gjøre analysearbeid for å virkelig studere personvernreglene, men siden dette ikke er hovedpoenget i oppgaven vil kun de viktigtste poengene trekkes frem.

Enhetsinformasjon som vil bli hentet ut er maskinvaremodell, versjon av operativsystemet, type mobilnettverk enheten er tilkoblet og mobilnummeret. Dette er informasjon som ikke anses som veldig sensitiv og det vil derfor ikke være noen konkrete sikkerhetsrisikoer knyttet til at Google kan innhente disse opplysningene. I tillegg vil det logges telefonlogginformasjon som omhandler både SMS og samtaleinformasjon, men siden disse enhetene ikke vil bli benyttes hovedsakelig for dette vil ikke dette være avgjørende heller. Google tilskriver seg også retten til å logge hvordan man bruker enheten og om spesielle enhetshendelser. Med enhetshendelser menes aktiviteter som programstopp, systemaktivitet, maskinvareinnstillinger og andre detaljer på systemnivå. Vedrørende loggingen av enhetsbruk så kan dette være en avgjørende faktor for eller i mot om det bør benyttes egenkontrollert hardware. Etter en vurdering internt i gruppen ble det avgjort at dette ikke ble ansett som noen stor sikkerhetsrisiko på grunlag av at dette ikke vil gi innblikk i selve bruken av enheten på en slik måte som kan eksponere Vestoppland Politidistrikts arbeidsrutiner og metoder. Om det blir foretatt søk på nettet med enheten vil Google samle inn denne informasjonen, noe som kan eksponere saksdetaljer eller gi en tilhørlighet til Vestoppland Politidistrikts saker. Dette vil være verdt å tenke på siden det kan være en alvorlig sikkerhetsrisiko, men anses som så lite sannsynlig at det vil bli et problem, og derfor ses bort ifra. Utover dette tillater Google seg å samle inn posisjonsinformasjon og dette vil også være gjeldende for enhetene som benytter appen siden posisjonering må være aktiv for at man skal få lagret koordinater med media som sendes. Det spesifiseres at Google kan innsamle og behandle posisjonsinformasjon, men presiserer ikke i hvilke tilfeller dette vil gjelde. Tatt i betrakning hvordan GPS lokasjon innhentes ved at man aldri får forespurt nåværende, men sist kjente lokasjon så antas ikke dette som en av de største sikkerhetsrisikoen, selvom det i værstefalls tilfelle kan bety at det kan være mulig å innhente stedsinformasjon om alle Vestoppland Politidistriksenheter og derfor også eksponere patruljenes lokasjoner.

All informasjon som samles inn av Google kan brukes for å vedlikeholde, beskytte og forbedre tjenestene deres samt utvikle nye tjenester basert på informasjonen. Personopplysningene blir ikke delt med andre parter eventuelt kun hvis man eksplisitt har gitt samtykke for dette. Domeneadministratorer har tilgang til å vise statistikk, endre

kontopassord, fjerne kontotilgang (Google konto), hente ut informasjonen som lagres, hente ut informasjon etter forespørsel fra myndighetene og begrense brukerens mulighet for å slette informasjon. Google kan også dele opplysninger om det er skjedd noe kriminelt som gjør at offentlige etater trenger informasjonen, men vil i dette tilfelle hvertfall ikke være tilfellet. Derimot har Google retten til å dele informasjon om bruk som ikke er identifiserbar med sine partnere [56].

4.11.2 – Motorola mobil

Mobilen er blitt levert av Motorola og er i tett samarbeid med Google siden dette er deres Nexus serie. Dette samarbeidet gjelder kun for denne modellen siden Google for hver nye Nexus modell innhenter nye samarbeidspartnere så derfor vil denne sikkerhetsvurderingen kun gjelde for Motorola Nexus 6 og ingen andre enheter fra hverken Motorola eller Nexus. Det lyktes ikke gruppen å finne personvernregler som gjelder for spesifikt denne enheten siden Motorola ikke har publisert noen dokumenter i forbindelse med dette på sine hjemmesider. Det ble derfor benyttet et dokument som er blitt publisert i forbindelse med Android 4.0 og 4.1, og det gjøres en antagelse om at denne vil være gjeldende i hel eller stor grad for versjon 5.0. Det vil uansett gi et overblikk over deres rettigheter og innhentings tilganger i forbindelse med enheten. Tenkes at det er tre mulige årsaker til at det ikke er publisert et eget oppdatert dokument for versjon 5.0, hvor første grunnen kan være at kilden som refereres til her anses som gyldig grunnet så nylig oppdatering. Andre årsaken kan være at de ikke har fått produsert og publisert et slikt dokument enda. Siste årsak kan være at siden det er en Nexus enhet vil retningslinjene til Google gjelde, men disse tre årsakene er kun antagelser og har ikke referanser i fakta.

Motorola påskriver seg retten for å uthente kjernedata som mobilnummer og serienummer. Litt mer detaljerte data som blir uthentet er ytelsesdata, analysedata, personlig informasjon, applikasjonshistorikk, mobil lokasjon, offentlig informasjon fra sosiale medier og besøkte nettsider. Det vil derimot aldri bli hentet ut spesifikk informasjon fra enheten om kommunikasjon og innhold av filer. Det kan benyttes tredjeparts markedsførere eller reklamenettverk for innhenting av informasjon og de vil derfor dele informasjon med de tredjepartene som yter tjenester for Motorola og deres brukere [57].

4.11.3 – HTC nettbrett

HTC er leverandøren for nettbrettet som gruppen har og er også en del av Nexus serien på lik linje med telefonen fra Motorola. Derfor vil det samme som nevnt innledningsvis i 4.11.2 – Motorola mobil gjelde. Utover dette har HTC sine egne personvern og retningslinjer i forbindelse med sine enheter og er derfor nødvendig å se på dette for å gjøre kunne gjøre en velinformert avgjørelse.

HTC logger en del informasjon om enheten vedrørende aktiveringsdato, GPS lokasjoner, telefonnummer, enhetstype, serienummer og enhetsindikatorer. Dette skiller seg ikke nevneverdig ut fra overnevnte Motorola eller Google sine retningslinjer. I følge deres opplysninger brukes denne informasjonen til å forbedre, videreutvikle og tilpasse deres produkter for å gi økt brukeropplevelse. Derimot deler de ikke informasjonen med noen parter uten brukerens eget samtykke eller juridiske forespørsler. Om HTC skulle være innblandet i en fusjon med et annet selskap vil de kunne måtte oppgi informasjonen de har innsamlet til denne parten, men ikke i andre tilfeller. Informasjon som oppgis å ikke være identifiserbar deler de med tredjeparter uten brukerens viten eller samtykke og informasjonen lagres enten på enheten eller på HTCs servere [58].

4.11.4 - Konklusjon

Basert på retningslinjene fra de forskjellige produsentene så viser det seg at de ikke skiller seg så mye fra hverandre, med unntak av hardware produsentene som samler mer enhetsspesifikk informasjon enn hva Google gjør. Det er ikke noe som påpeker seg som ekstreme sikkerhetsrisikoer og vil med størst sannsynlighet ikke gjøre noe skade, men det er viktig å være klar over de. Innsamling av data er milliongeskjeft for produsentene og er derfor noe de vil gjøre i så stor grad de kan og med et system som skal brukes av Vestoppland Politidistrikt så er det verdt å ha et innblikk i hva de faktisk innhenter. Informasjonen som innhentes av produsentene vil ikke ha så stor betydning siden disse ikke avslører så mye om selve bruken, med unntak av applikasjonshistorikken som Motorola innhenter. Det defineres ikke hva denne historikken vil inneholde, men det vil nok ikke avsløre noe om bruken av prosjektets app utover hva som ikke er sensitiv informasjon. HTC presiserer tydelig at de ikke vil ha innsyn i filene på enheten, og det vil antas at dette vil gjelde de andre også med tanke på hvor personlig disse filene vil være sammenlignet med bruksmønster og annen innhentet informasjon.

Siden Vestoppland Politidistrikt allerede kjører sine systemer på mobile enheter gjøres det en forutsetning om at Politidirektoratet har foretatt en vurdering av dette og anser bruken av forbrukerprodukter som trygt for Politiets arbeid. Til tross for dette så anses det som i grenseland av gruppen, og det ville ved en videreutvikling blitt gjort grep for å sikre seg i større grad mot dette. En løsning som har blitt vurdert er å benytte «sandboxing» som er en måte å isolere et eget miljø i operativsystemet på enheten. Ved benyttelse av «sandboxing» vil man få et separert miljø fra det vanlige operativsystemet hvor man har forhåndsgodkjente applikasjoner. Da vil ikke applikasjoner på enheten kunne nå de som er lagt i «sandboxing» miljøet og motsatt og vil bidra mot å sikre seg mot overnevnt problematikk. Appen som utvikles kan derfor kjøre i et slikt miljø og vil da være sikker til og med om enhetens operativsystem er infisert med virus. Dette systemet ble opprinnelig utviklet av Samsung under navnet «Knox», men er videreutviklet av Google for å fungere for alle Android enheter og ble døpt «Android for Work» [59].

«Android for Work» er implementert som en del av Android 5.0, og for eldre versjoner er det tilgjengelig som en app man kan laste ned. Fungerer på den måten at man starter det som en hvilken som helst app, logger inn med et passord og får da tilgang til det separerte miljøet. Man blir da samtidig blokkert fra enhetens faktiske miljø og vil kun få funksjonaliteten til dette «sandboxing» miljøet til man logger ut. Det kan selv defineres rettigheter og retningslinjer for hva som er tillatt og er en glimrende løsning for å dele opp i et privat –og et jobbmiljø [60].

4.12 – Teoretisk løsning Publikum på stedet → OPS

Oppgavebeskrivelsen forteller at det er ønskelig å tilby en løsning for publikum som kan gjøre det enklere å sende inn tips og bistå Vestoppland Politidistrikt i den grad det lar seg gjøre for publikum. Innledningsvis i denne rapporten poengteres det at denne deloppgaven vil bli sett bort ifra i utviklingsfasen, men at den vil vurderes for mulige teoretiske løsninger.

I grove trekk ønskes det et system tilsvarende det som Vestoppland Politidistrikt skal benytte, men med noen mindre endringer i form av at hashen som generes bør knyttes opp i mot noe fysisk på telefonen istedet for i hovedsystemet hvor det knyttes opp i mot filnavn. I tillegg ønskes det at telefonnummer sendes med så dette må utvikles. Det foreslås i oppgaven at det

utvikles en app ment for dette formålet, og det vil ikke være en så stor prosess siden det allerede kan utnyttes allerede utviklet funksjonalitet i systemet som er laget. Serverapplikasjonen kan være den samme, men må utvikles en tilleggsmodul som vil ta i mot tipsene som kommer fra publikum slik at dette ikke blandes med innkommende media fra betjentene. Dette kan gjøres ved å lage serverapplikasjonen fanebasert på samme måte som en nettleser ved at en fane er opp i mot betjentene og den andre fanen er opp i mot publikumkanalen. Dette vil da gi muligheten for å tilby nøyaktig det samme brukergrensesnitet, forenkler overvåkingsområde og gjør det lettere å videresende innkommet media fra publikumkanalen til betjenter siden det allerede er i samme systemet.

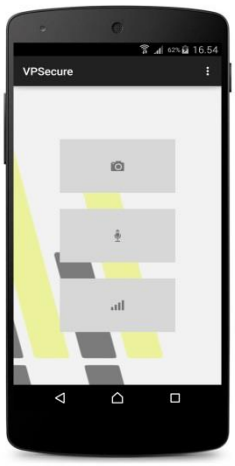
Appen kan være den samme som betjentene bruker, men hvor det er gjort noen endringer i funksjonalitet. Siden dette ville vært mer tipsrelatert så måtte det blitt implementert en større form for rapportering enn bare mulighet for å sende media for at det ikke skal bli misbrukt. Kunne eksempelvis vært at man måtte fylle ut noe grunnleggende informasjon om type hendelse, sted, kontaktinfo, nødvendig tiltak som tipseren følte at var nødvendig. Dette kunne vært alt i fra informering, eller om det var behov for én eller flere patruljer og innblanding av flere etater. Deretter kunne man kommet dit at media kunne legges ved rapporteringen. Selvom man ville skrevet inn sted -og kontaktinfo ville dette blitt forsøkt å hente inn og legges ved automatisk av appen, men foreslås som en ekstra sikkerhet og for å skape redundans. Det ville også vært mulig å implementert en akutt knapp, hvor man slapp å fylle ut et rapportskjema og kun sendt et media eller kort forhåndsdefinert beskjed for å indikere at man trenger hjelp. I tillegg ville man hatt sikkerhetsegenskapene fra betjentenes app ved at generert media hashes.

En annen løsning, og som anses som en mer foretrukket løsning blant publikum av gruppen ville vært å lage en nettleserbasert løsning istedet for app. På det grunnlag at det antas at ikke majoriteten av befolkningen ville innstallert en Politi app på telefonen sin, i frykt for innsyn eller andre ting. Ikke at dette ville vært tilfellet, men tatt i betrakning at folk generelt ikke leser om programmers retningslinjer og vilkår for bruk så antas det at flere ville dratt denne konklusjonen automatisk. Derfor ville det vært bedre med en nettleserbasert løsning fordi man ville da sluppet denne problematikken og kunne fungert på tilsvarende måte som en app. Dette ville i tillegg økt bruksområde fra kun mobile enheter til alle produkter med en nettleser og nettilgang.

5 – Design

5.1 – Grunnleggende struktur app

Appen er utviklet med hensikt å være brukervennlig og intuitiv slik ved at det enkelt forståes hvordan det skal navigeres. Derfor vil man kunne kjenne seg igjen i de forskjellige undermenyene ved at disse er bygget opp på tilsvarende måte og er like i design. Alle undermenyene er enkelt tilgjengelig fra en oversiktlig hovedmeny med knapper kun for det nødvendige uten forstyrrende elementer eller farger. For fullverdig forklaring av funksjoner og ikoner se brukermanualen vedlagt som *vedlegg F*. Tidlig designskisse av appens utseendet ligger vedlagt som *vedlegg I.*



Figur 11:Viser hovedmenyen

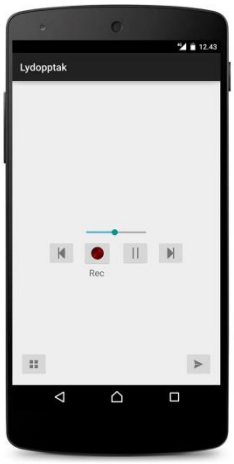
5.1.1 - Hovedmenyen

Hovedmenyen i appen inneholder kun tre hovedknapper som fører til hver sin undermeny for tilhørende funksjonalitet. Fra topp til bunn: Bilde/video, lydopptak og dekningsstatus. I tillegg er det en liten knapp øverst som fører til innstillinger, men som ikke vil være relevant for vanlig brukerinteraktivitet.

Figur 12: Viser undermeny bilde/video

5.1.2 – Undermeny Bilde/video

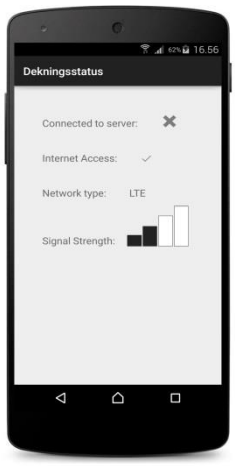
Bildet representerer den første undermenyen og preges av fire valg. Hvor send knappen er deaktivert inntil media er valgt og kan sendes.



5.1.3 – Undermeny Lydopptak

Bildet representerer skjermbildet hvor lydopptak er gjort, og preges totalt av seks knapper pluss en egen glidebryter for spoling. Ble designet spoling både med bruk av knapper og glidebryter.

Figur 13: Viser undermenyen lyd.

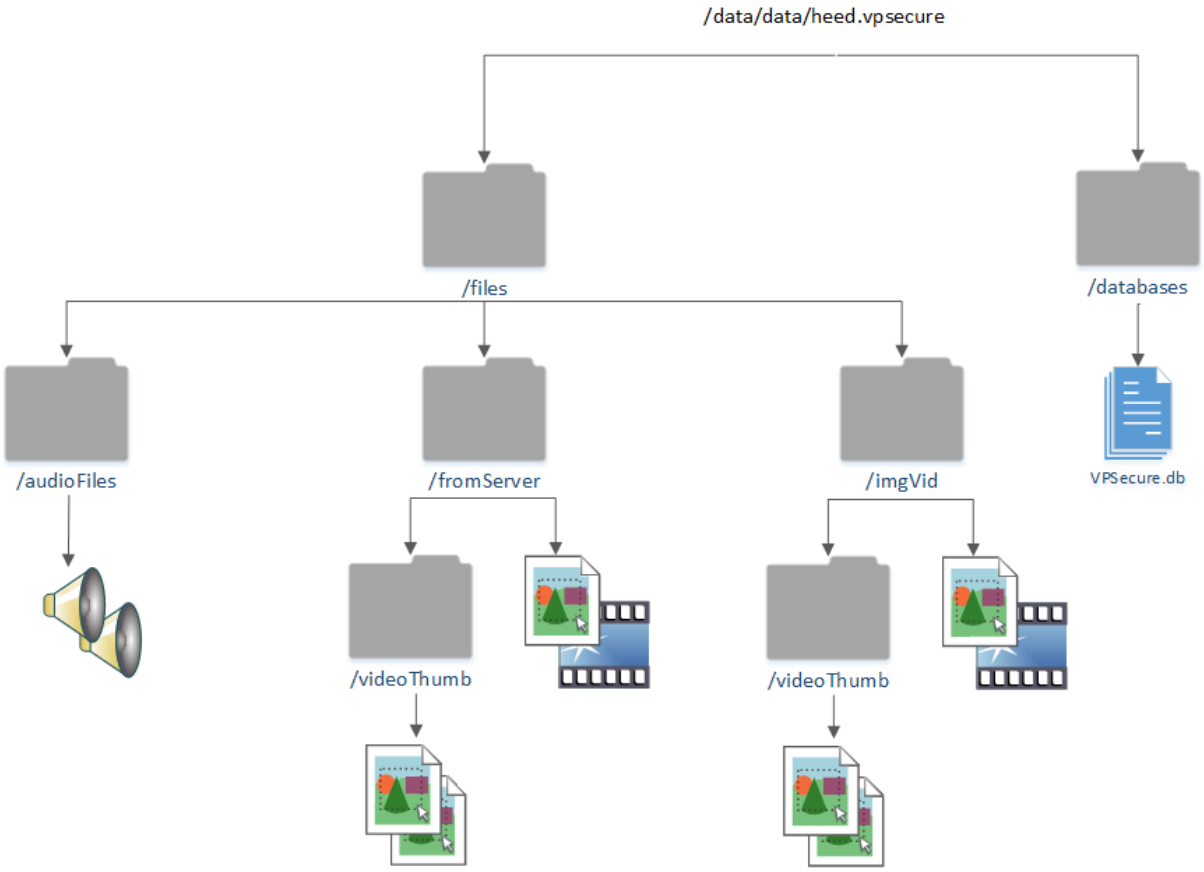


5.1.4 – Dekningsstatus

Bildet representrer det første og eneste skjermbilde i denne undermenyen og preges av kun info uten noen form for knapper til interaksjon.

Figur 14: Viser undermenyen dekningsstatus

5.1.5 – Filstruktur



Figur 15: Viser filstrukturen for appen med tekstfeltene som indikerer de forskjellige filstiene. Utarbeidet i Microsoft Visio 2013

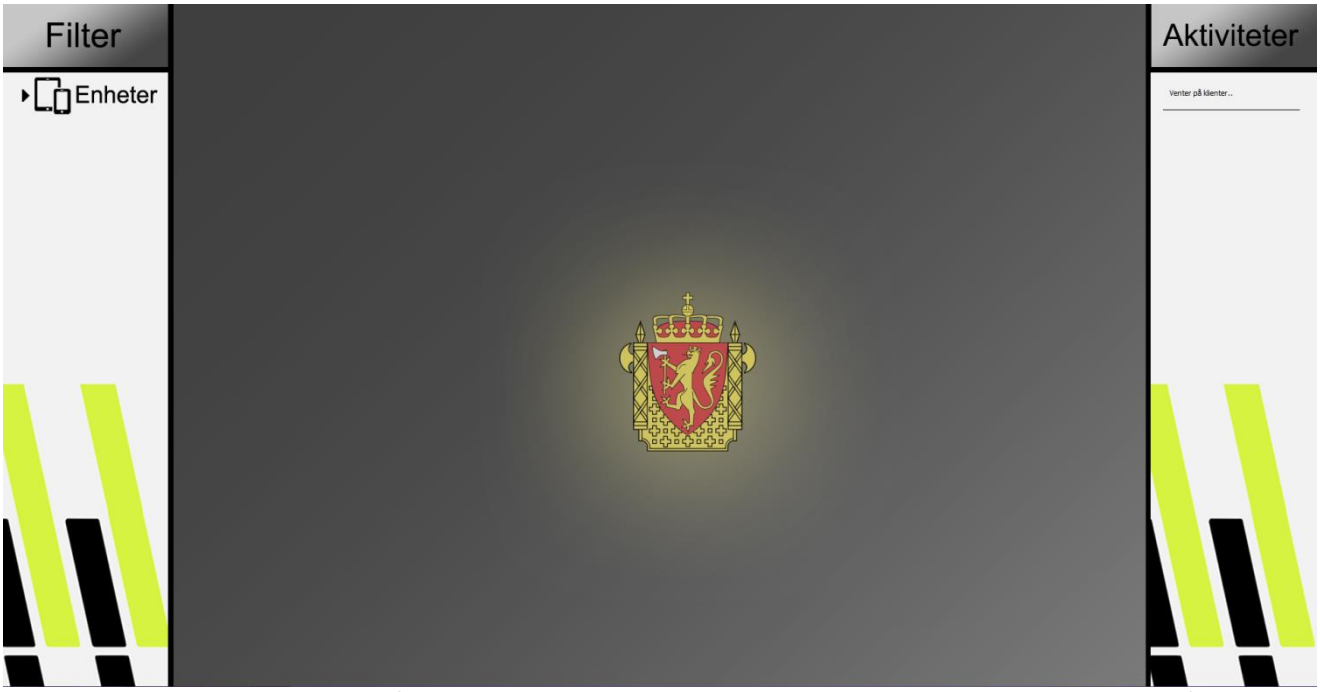
I figur 15 vises filstrukturen i appen og hvor de forskjellige elementene lagres. All media som behandles av appen vil bli lagret privat, sammen med app filene og vil derfor ikke være synlige i enhetens vanlige galleri. Dette ble gjort etter ønske fra oppdragsgiver, men også for å øke konfidensialiteten i hele systemet.

Designet av filstrukturen ble gjort på grunnlag av at utviklerne enkelt kan se hvor de forskjellige elementer skal legges, men også med tanke på ryddighet og at filer med tilhørlighet ligger sammen. Eksempelvis så vil filene i */imgVID* -mappen inneholde både video –og bildefiler siden disse tilhører samme funksjonalitet og det er derfor naturlig at disse vil lagres i felles mappe. Databasefilene som er nødvendig for appen ligger i egen mappe rett under toppnivået fordi disse filene ikke har noe med media å gjøre. Punktet 6.4 – Databaser utdyper bruken av databasen i appen.

5.2 – Grunnleggende struktur serverapplikasjon

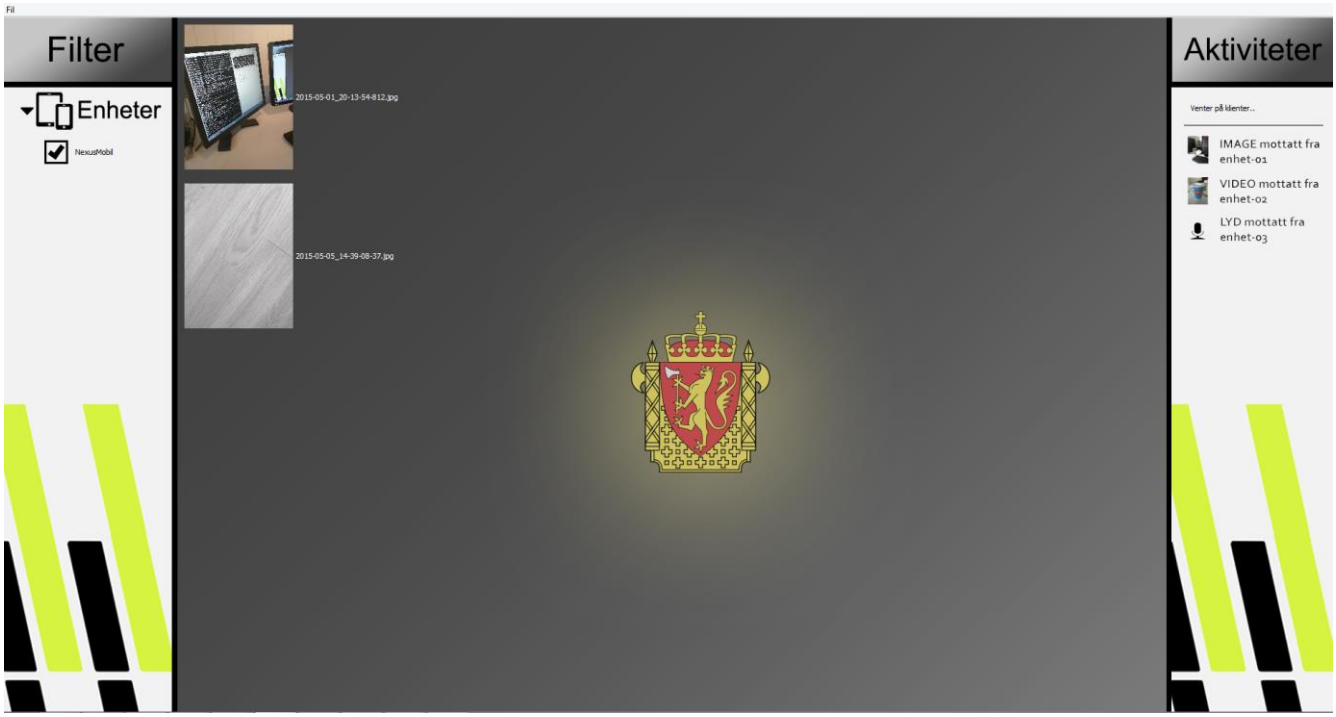
Serverapplikasjonen er designet med de samme premissene som appen og vil tilby et ryddig brukergrensesnitt hvor kun det aller nødvendigste er synlig. Dette gjøres for at man raskt og enkelt skal forstå grensesnittet som presenteres og hva de forskjellige elementene er. Både å se innkommende aktivitet, filtrere frem ønskelig media og videresende vil foregå i samme skjermbilde. Som skjermbildet i figur 16 viser presenteres serverapplikasjonen som et tredelt grensesnitt bestående av filter på venstre siden, hovedområdet i midten og aktivitetsliste til høyre som er en «live feed».

5.2.1– Hovedmenyen



Figur 16: Viser hovedmenyen som også er det eneste skjermbilde i serverapplikasjonen siden all interaktivitet foregår i dette bildet. Serverapplikasjonen blir presentert på denne måten når den er startet opp.

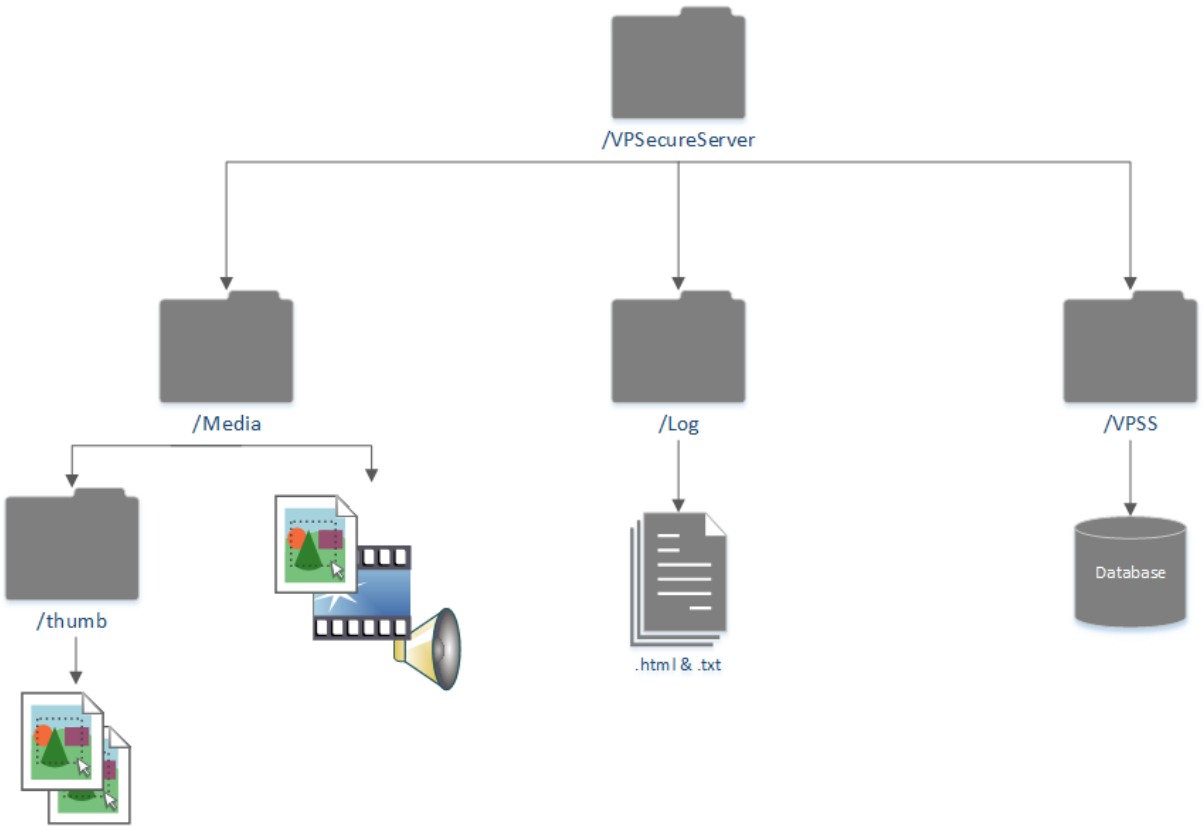
5.2.2 – Aktiviteter og filter



Figur 17: Viser hvordan både aktiviteter, filtre og hvordan serverapplikasjonen vil være i bruk.

Filterlisten på venstre siden i figuren ovenfor viser at man kan velge å filtrere på enheter og vil da få presentert media mottatt fra valgt enhet i hovedbildet i applikasjonen som vist på figur 17. Det er tiltenkt flere filter muligheter, men disse blir ikke implementert. Andre filter muligheter som var tiltenkt å implementeres: dato, type, sakskode og telefonnummer. Aktiviteter på høyresiden vil å vise en oversikt over tilkoblede klienter og en nyhetsmating av innkommende media fra de tilkoblede enhetene. Denne vil oppdateres i takt med det som kommer inn hvor det vil bli presentert et miniatyrbilde av innholdet, typen og hvem det kommer fra.

5.2.3 – Filstruktur



Figur 18:Viser filstrukturen for serverapplikasjonen med filstier til de forskjellige undermappene og toppnivået øverst. Utarbeidet i Microsoft Visio 2013

Filstrukturen for serverapplikasjonen er bygget opp på tilsvarende måte som appens filstruktur, men med noe mindre undermapper av den årsak at databasen her vil organisere all media istedet for mappesortering. Forteller mer om databasens innholdet i punktet 6.4 - Databaser og andre detaljer som er essentielle for å oppnå ønsket virkemåte. Strukturen er designet på den måten at all innkommende media legges i en felles mappe, hvor miniatyrbilder plasseres i en egen undermappe enn selve mediafilene. I tillegg består strukturen av en undermappe fra toppnivå for alle loggfiler som genereres av serverapplikasjonen samt enda en undermappe rett fra toppnivå hvor alle databasefiler er plassert.

5.3 – Kommunikasjonsprotokollen

Kommunikasjonen mellom appen og serverapplikasjonen foregår for det meste gjennom rene strenger. Disse strengene blir delt opp av ett forhåndsbestemt mønster slik at det er mulig å sende forskjellig informasjon i samme streng, noe som vil redusere antall strenger som må sendes. Mønsteret systemet deler på er «*/$BGD%*» ettersom det aldri vil forekomme naturlig i strengene som sendes.

Den første delen av alle strenger vil bestå av en tresifret kode og for strenger som blir sendt fra app til server vil koden starte med én, eksempelvis 101. For strenger som blir sendt fra server til app vil koden starte med to, eksempelvis 201. Dette er ett eksempel på hvordan en slik streng kunne sett ut i systemet:

101/$BGD%**355470061029255**/$BGD%**242888055239887**

Det er gitt forskjellige farger i eksempelet over for å lettere identifisere de forskjellige delene av strengen. De tre første grønne sifrene er selve koden, og denne koden forteller hva mottaker (serverapplikasjonen i dette tilfellet) skal gjøre og hvor mye informasjon som følger. Mønsteret som systemet deler på er markert i blått og indikerer start på data. De to lange strengene med svarte tall er selve dataene som blir overført, i dette tilfelle er de 15 første tallene IMEI nummeret til enheten og de 15 etterfølgende IMSI nummeret til simkortet i enheten. (Fiktive tall brukt i eksempelet).

Videre kommer en kort forklaring til hva de forskjellige kodene systemet bruker er:

- 101 - Registrerer enheten på serveren med sitt IMEI og IMSI nummer.
- 102 - Chat melding fra enhet til server.
- 103 – Enhet har sendt ett bilde.
- 104 – Enhet har sendt ett bilde med geografiske koordinater.
- 105 – Enhet har sendt en video.
- 106 – Enhet har sendt ett lydopptak.
- 201 – Serveren skrur seg av.
- 202 – Chat melding fra server til enhet.
- 203 - Server har sendt ett bilde.
- 204 – Server har sendt en video.
- 205 – Server har sendt en lydfil.

6 – Implementering, koding og produksjon

6.1 – Utviklingsmiljø

Vedrørende valg av programmeringsspråk ble det fort kommet til enighet både blant gruppemedlemmene og med oppdragsgiver at serverapplikasjonen og appen ville bli skrevet i Java. Dette ble besluttet etter et drøftingsmøte hvor forskjellige programmeringsspråk ble diskutert tatt i betrakning med erfaring og hva som var vanlig praksis opp i mot det som skulle utvikles. Valget falt på Java av den årsak at Android rammeverket også er bygget opp med Java [50], men ble også avgjort på grunnlag av ene gruppemedlemmets gode erfaringer innenfor språket.

Ettersom Java ble det valgte programmeringsspråket for prosjektet ble det naturlig å velge «Eclipse» som IDE for utviklingen av serverapplikasjonen ettersom dette er verktøyet gruppen har mest erfaring med. Siden ingen av gruppemedlemmene har erfaring med utvikling av app ble det valgt å gå for å bruke den offisielle IDE [51] fra Google som er «Android Studio».

Oppdragsgiveren anskaffet to mobile enheter for bruk i løpet av utviklingsperioden hvorav disse var en mobiltelefon og ett nettbrett. Mobilen av typen «Motorola Nexus 6» og nettbrettet av typen «HTC Nexus 9», og disse enhetene ble valgt på det grunnlag av at disse var de eneste på markedet da prosjektet startet som ble levert med Android 5.0. Det viste seg dog at denne Android versjonen ble sluppet av andre mobilleverandører bare noen få uker inn i prosjektet. Det ble også benyttet et par private mobile enheter som testtelefoner i løpet av prosjektet hvorav disse var en «Sony Xperia Z3» og en «Samsung Galaxy S5». Tiltross for at appen er testet på denne bredden med enheter kan det ikke garanteres at den vil fungere like feilfritt på enheter fra andre produsenter eller andre modeller.

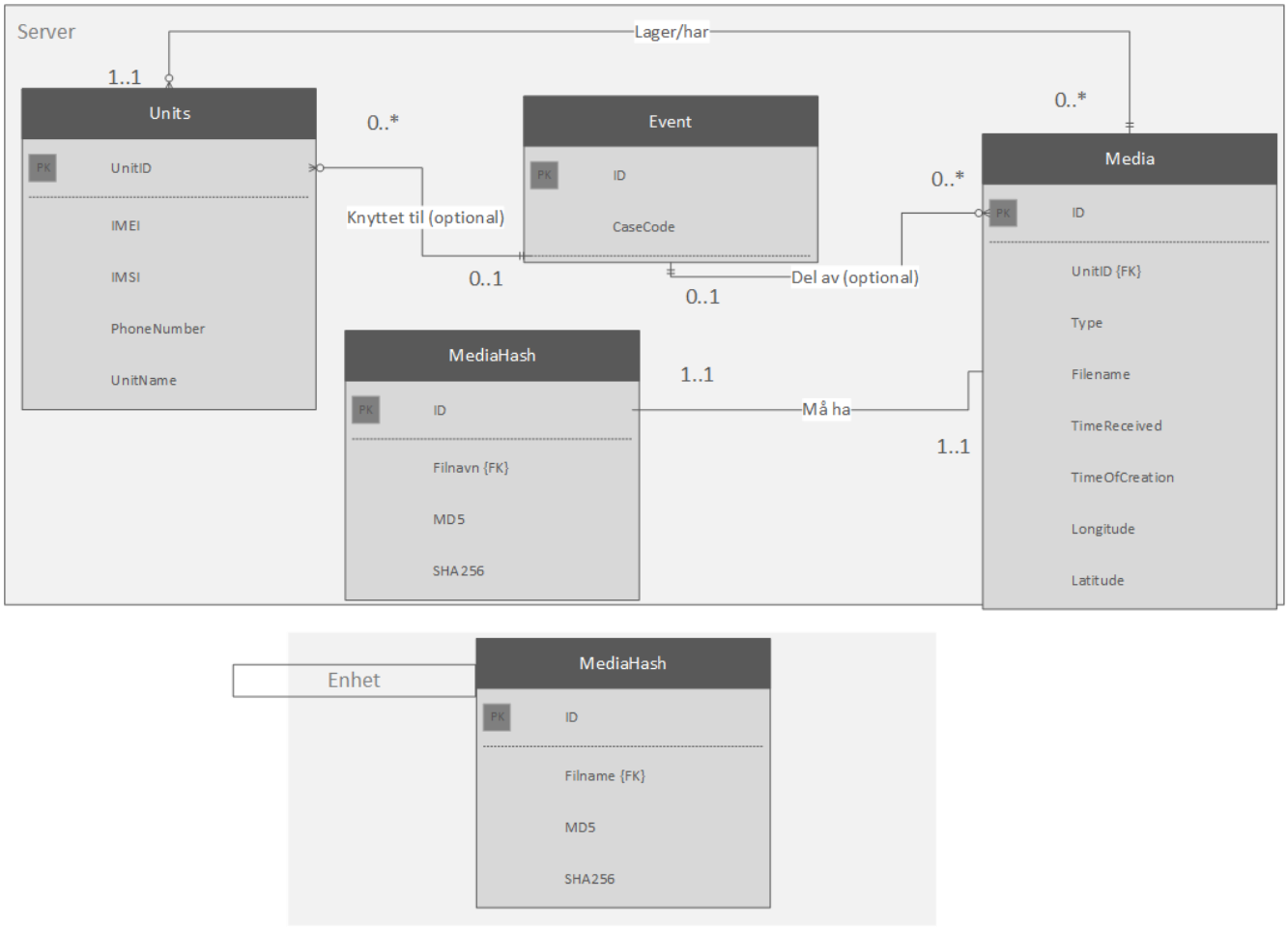
6.3 – Verktøy

I tillegg til de overnenvte utviklingsmiljø verktøyene ble det benytter noen flere verktøy som var nødvendige for utforming av både rapport, illustrasjoner og analysearbeid.
For produsering av rapporten ble det benyttet «Microsoft Word 2013» som genererte leverbar PDF versjon av det endelige utkastet. For illustrasjoner benyttet i rapporten har det blitt

benyttet et annet Microsoft verktøy kalt «Visio 2013» som er et effektivt verktøy for generering av mange type skjemaer og andre illustrasjoner. For produsering av ikoner, elementer for GUI, logo, plakat og diverse ble det benyttet Adobe Photoshop CS6. Ikoner brukt i appen er hentet ned fra internett og den typen ikoner som er brukt er lisensert for all type bruk, inkludert kommersiell bruk og kan derfor benyttes. Overvåking av nettverkstrafikk i forbindelse med kontroll av kommunikasjonen i systemet ble det benyttet Wireshark. Adobe Reader Pro ble benyttet for å koble sammen PDF dokumenter.

6.4 - Databaser

Systemet består totalt av to forskjellige databaser som figur 19 viser og begge er essentielle for at funksjonalitet skal være som ønsket. Databasene er av to forskjellige typer hvor den ene fungerer som kjernen i serverapplikasjonen mens den andre databasen er en mindre tilleggsmodul for appen for å kunne lagre hash.



Figur 19: Viser UML-skjema over begge databasene i systemet og relasjoner mellom ulike tabeller. «Server» og «Enhet» er der for å påpeke hvor de forskjellige tabellene hører til. Utarbeidet i Microsoft Visio 2013.

6.4.1 - Serverapplikasjon database

Databasen som benyttes for serverapplikasjonen er av typen «Java DB» som er bygget på en Apache Derby distribusjon [47]. Dette ble gjort på grunnlag av den tette implementasjonen denne database distribusjonen har i Java, samt kombinasjon med erfaringer fra tidligere prosjekter innad i gruppen, og ble derfor et naturlig valg.

Denne databasen inneholder totalt fire tabeller hvorav disse er: *Units, Event, MediaHash* og *Media.* Disse er essentielle for å oppnå ønsket funksjonalitet av serverapplikasjonen og for oversikt over tilhørende attributer i hver klasse se figur 19. Nevner derfor kun noen av attributene her.

Units: Denne tabellen lagrer informasjon om de forskjellige enhetene som er lagt til i databasen. «IMEI» og «IMSI» er spesielt viktige siden disse attributene må legges inn for at en enhet skal bli godkjent og få muligheten til å koble seg til serveren.

Media: Inneholder informasjon om media som er mottatt på serveren. Koblet sammen med enhet på «UnitID» for at hvert media kan kobles opp i mot en bestemt enhet. Med type menes filformatet og indikerer om det er lyd, video eller bilde. «TimeofCreation» er tidsattributen appen sender med og indikerer når filen ble opprettet.

MediaHash: Denne tabellen tar vare på alle hash verdier som genereres av programmet når media kommer inn og er essentiell for at data integritet skal bli opprettholdt. Inneholder hash verdiene fra begge implementerte hashing algoritmer og knyttes opp til media på attributen «Filnavn». Ikke direkte tilkoblet «MediaHash» tabellen fra databasen i appen, men verdiene som ligger her knyttes til et filnavn og vil være de samme som er tilknyttet samme filnavn i «MediaHash» tabellen i appen. Dette gjøres for at disse skal kunne sammenlignes opp i mot hverandre for å verifisere at filer ikke er modifiserte i ettertid.

6.4.2- App database

Databasen som benyttes for appen er av typen SQLite og er implementert på grunnlag av hashing funksjonalitet. Dette gjøres for at enheten skal inneholde sin kopi av hashing verdier for filene den produserer og serveren skal ha de samme hashing verdiene for de samme filene.

Denne databasen inngår som en del av appen og installeres sammen med appen når den legges inn på en enhet. Databasen inneholder kun en tabell som er: *MediaHash*

MediaHash: Som nevnt i punktet ovenfor om «MediaHash» så inneholder denne databasen en tilsvarende tabell med samme informasjon og struktur som den på serveren. Dette for at man skal ha hashing verdiene for samme fil på to forskjellige steder.

6.5 - Koding

Under dette hovedpunktet vil det nevnes oppnådd kjernefunksjonalitet for både server applikasjonen og appen etter endt utvikling. Noe av kjernefunksjonaliteten vil også bli forklart i mer detalj og ha tilhørende kodeeksempler som vedlegg. Mindre viktig funksjonalitet som for eksempel bildevisning på appen og andre GUI elementer vil ikke bli nevnt her.

6.5.1 – Server

Som nevnt tidligere har vi hatt mest fokus på appen, men utviklet serverapplikasjonen i den grad at den har nok funksjonalitet til å kunne svare opp imot appen. Av oppnådd funksjonalitet inngår:

- Mulighet for å opprette en sikker TLS kommunikasjonskanal med tilkoblende klienter.
- Kunne motta media som bilder, lydopptak og videoer fra tilkoblede klienter.
- Strukturerer mottatt media i database ved å lagre informasjon tilknyttet denne, se punkt 6.4.1 – Serverapplikasjon database.
- Genere hash av media for å opprettholde validitet.
- Filtrere og vise mottatt media for å enkelt kunne finne fram til ønsket informasjon.
- Videresending av media til tilkoblede klienter.

Vedlegg O viser hvordan serveren initialiseres og hvordan den tvinger bruk av protokoll og cipherpakke som vi kom frem til under 4.10 - Implementasjon

6.5.2 – App

For utdypende forklaring av funksjonalitet se *vedlegg F*. For appen har det blitt oppnådd følgende kjernefunksjonalitet:

- Mulighet for å koble seg til serveren ved bruk av TLS.
- Kunne ta bilder, video og lydopptak og vise i appen.

- Kunne sende media som bilder, lydopptak og video til server.
- Generere hash av media og lagre disse i database.
- Kunne se oversikt over tilkoblingsstatus.

Vedlegg P viser metoden for sending av fil fra app til server. Som man kan se i dette vedlegget sendes først størrelsen på filen og deretter filen. Størrelsen på filen må sendes først fordi det brukes TLS kryptering og serveren kan derfor ikke vite hvor mange bytes den skal lese før den har dekryptert dataen. Under punktet 6.8 – Services / handlers forklares det hvordan media blir lagt på kø før sending og at tråden ansvarlig for å sende tar fra denne køen. *Vedlegg Q* viser koden for hvordan media legges på køen og *vedlegg R* viser hvordan media tas fra denne køen.

6.6 – Internasjonalisering

For å gjøre det mulig for ikke norsktalende personer å bruke appen og serverapplikasjonen er de internasjonalisert. Det vil si at alle tekstelementene i systemene ikke er hardkodet i kildekoden, men ligger i egen fil og blir hentet derfra. Slik er det enkelt å opprette flere filer og oversette hver enkelt tekststreng til andre språk.

På serveren er det tre filer som brukes til internasjonalisering, hvor den første er en standard fil som bli brukt hvis applikasjonen ikke har blitt satt til eller ikke klare å finne maskinens språk. Den andre er filen som inneholder de norske tekststrengene og den siste filen inneholder tekstrengene oversatt til engelsk. Filen med de norske tekststrengene heter «I18N_no.properties», *i18n* er en vanlig forkortelse for internasjonalisering og kommer av at det er 18 bokstaver mellom den første og siste bokstaven [1][2].

I likhet med serverapplikasjonen er det på appen også tre filer som brukes til internasjonalisering, disse tre filene heter det samme, «strings.xml», men skilles ved at de ligger i tre forskjellige mapper. Mappene heter «values-» etterfulgt av ISO språkkoden for språket denne mappene gjelder for, bortsett fra mappen for de standard tekststrengene som bare heter «values» [4]. Eksempelvis heter mappen med de norske tekststrengene «values-no».

Ved at tekststrengene ligger i egne filer, og ikke er hardkodet i kildekoden, kan utviklere lett rette skrivefeil, endre på strengene eller legge til nye filer for nye språk uten at applikasjonen må rekompileres [1]. Her er ett eksempel på en av tekststrengene fra serverapplikasjonen og hvordan den ser ut i internasjonaliseringsfilen:

```
addUnit = Legg til enhet.
```

Når applikasjonen trenger teksten «Legg til enhet» i programmet, leter den etter nøkkelen «addUnit» i internasjonaliseringsfilen for det språket som har blitt satt. Det er derfor viktig at alle internasjonaliseringsfilene har de samme nøklene. Når applikasjonen har funnet nøkkelen henter den verdien til denne nøkkelen altså det som er på høyreside av likhetstegnet i eksempelet over [3].

Internasjonaliseringen på appen fungerer nesten helt likt som på serverapplikasjonen. Den største forskjellen er hvordan tekststrengene er lagret, i motsetning til serverapplikasjonen der filene er lagret i plain-text format [3], følger filene på den appen markeringsspråket XML (Extensible Markup Language) [4]. Her er ett eksempel på hvordan en av tekststrengene fra den mobile applikasjonen ser ut:

```
<string name="about">Om</string>
```

På samme måte som ovenfor har man her en nøkkel, «about», og en verdi «Om», hvor nøklene må være like i alle internasjonaliseringsfiler.

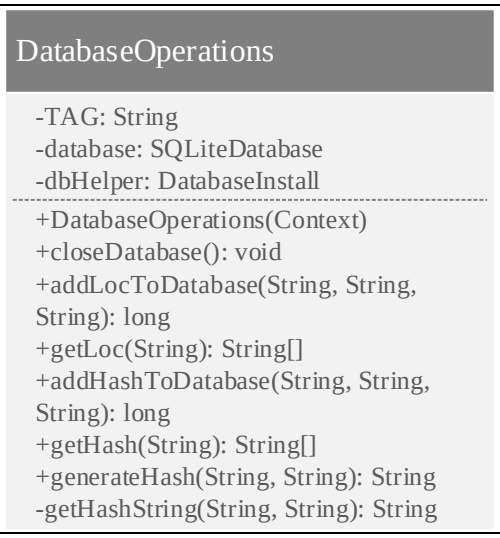
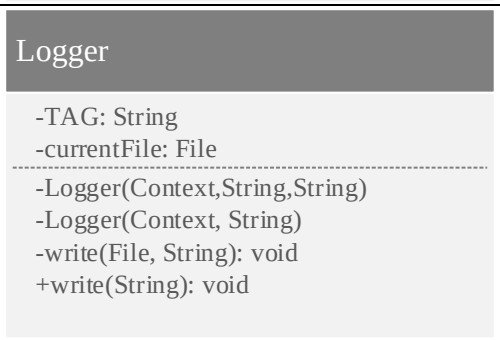
6.7 – Klassediagram

Klassediagrammene for både app og serverapplikasjonen er vedlagt som *vedlegg K*.

6.7.1– Forklaring til klasser app

De fleste klassene som er en del av appen er av typen «ActionBarActivity» som vil si at de er aktiviteter i appen. Alle aktivitetene er beskrevet i brukermanualen, se *vedlegg F* og det vil derfor ikke beskrives i detalj her, men kun gi en forklaring til de øvrige klassene som ikke er aktiviteter. Klassen «ConnectionService» vil heller ikke bli forklart siden denne er forklart under punktet 6.8 – Services / Handlers. Følgende forklaringer kommer her:

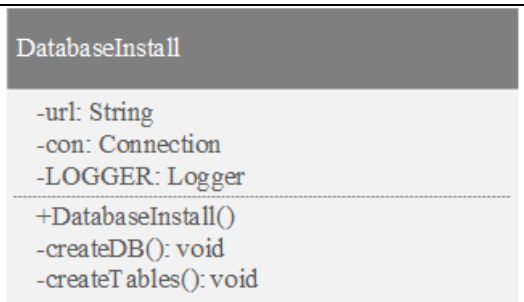
CustomServiceConnection -TAG: String -cx: Context -mIsBound: boolean -mService: Messenger -mMessenger: Messenger -extraMsg: Message --- -doBindService(): void +CustomServiceConnection(Messenger, Context) +CustomServiceConnection(Messenger, Context, Message) +onServiceConnected(ComponentName, IBinder): void +onServiceDisconnected(ComponentName): void +rebind(): void +unbind(): void +send(Message): boolean	Denne klassen brukes for å koble seg til servicen «ConnectionService». Denne klassen brukes av de aktivitetene som må sende informasjon til serveren eller sjekke om appen er koblet til
DatabaseInstall +TEXT_NOT_NULL: String +DATABASE_VERSION: int +DATABASE_NAME: String +TABLE_MEDIALOC: String +COLUMN_ID: String +COLUMN_FILENAME: String +COLUMN_LONGITUDE: String +COLUMN_LATITUDE: String +TABLE_MEDIAHASH: String +COLUMN_MD5: String +COLUMN_SHA256: String -CREATE_TABLE_MEDIALOC: String -CREATE_TABLE_MEDIAHASH: String --- +DatabaseInstall(Context) +onCreate(SQLiteDatabase): void +onUpgrade(SQLiteDatabase, int,int): void	«DatabaseInstall» sjekker om databasen er innstallert og om den ikke er det installeres den med tabellene som er definert i denne klassen.

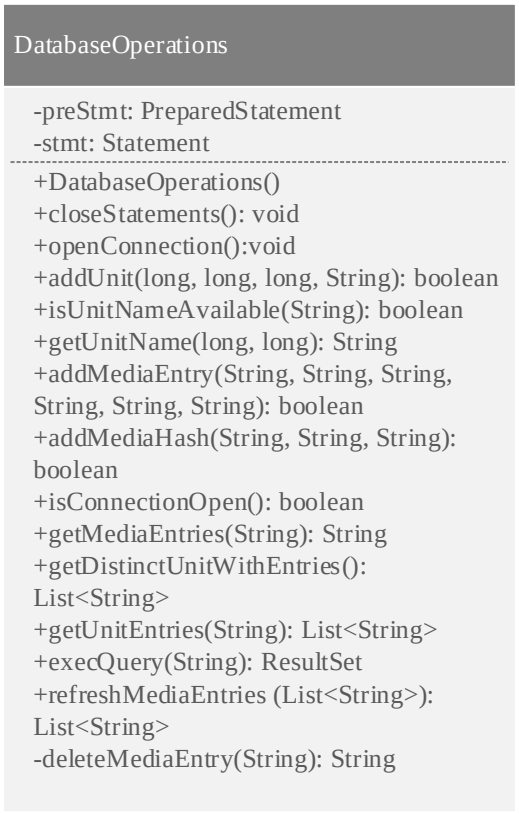
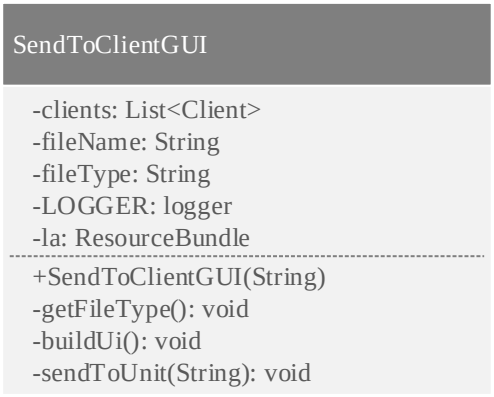
	«DatabaseOperations» brukes for å kjøre spørringer mot den innstallerte SQLite databasen.
	«Logger» klassen brukes til spesial logging, kun for test formål. Denne klassen kan bli brukt for å logge slik at man kan lese loggen direkte i appen.

Tabell 6: Tabell som forklarer de viktigste klassene i databasen for appen. Utarbeidet i Microsoft Visio 2013.

6.7.2– Forklaring til klasser serverapplikasjonen

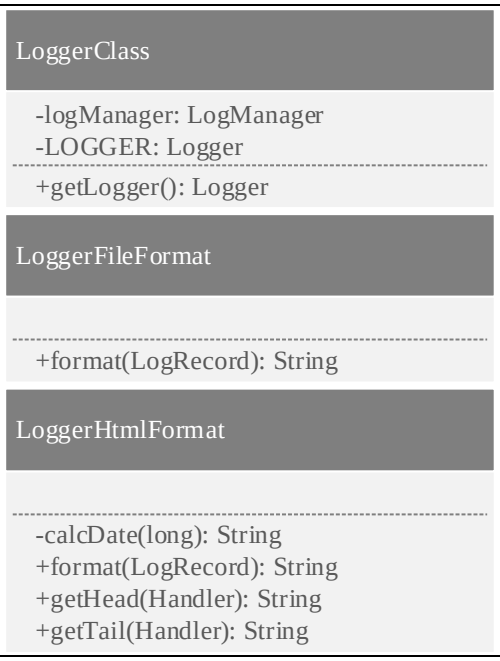
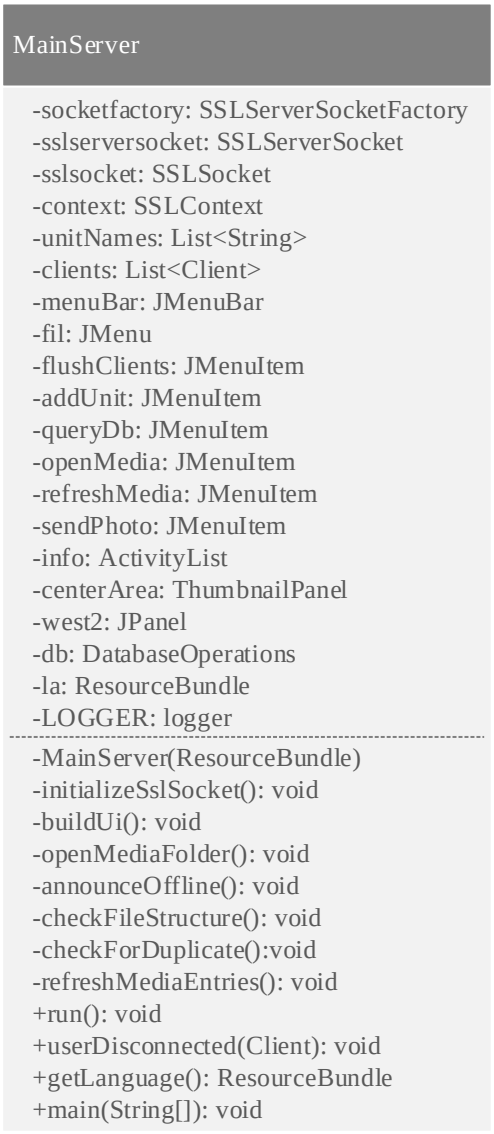
Nedenfor kommer forklaring til klassene i serverapplikasjonen. Noen klasser av mindre betydning vil bli utelatt enten at de er trivielle eller at det ikke er noe poeng å forklare innholdet. Følgende forklaringer av klasser kommer her:

	«DatabaseInstall» sjekker om databasen har blitt installert og installerer den om nødvendig med de angitte tabellene som nevnt i 6.4.1 – Serverapplikasjon database.

	«DatabaseOperations» håndterer operasjoner mot databasen. Klassen oppretter en kobling til databasen og kjører ønskede spørringer.
	«ActivityList» oppretter og håndterer aktivitets listen i serverapplikasjonen. Denne klassen legger til innkommende aktiviteter i listen både med og uten thumbnail.
	Med denne klassen kan du velge hvilke klienter du vil sende fil til.

addUnitForm -db: DatabaseOperations -imei: JTextField -imsi: JTextField -tlfnr: JTextField -enhetsnavn: JTextField -imeiDesc: JLabel -imsiDesc: JLabel -tlfnrDesc: JLabel -enhetsnavnDesc: JLabel -la: ResourceBundle -LOGGER: Logger +AddUnitForm(DatabaseOperations) -buildUi(): void -addUnit(): boolean	«AddUnitForm» klassen brukes for å legge til nye enheter i databasen. Den oppretter et eget GUI hvor det må fylles ut enhetsnavn, imei, imsi og telefon nummer. Derretter sender den denne informasjonen til databasen for lagring.
Const +PORT: int +MEDIAPATH: String +THUMBNAILPATH: String +LOGGERPATH: String +IMGEXT: String +VIDEXT: String +TYPEVIDEO: String +TYPEIMAGE: String +TYPEAUDIO: String +CONNECT: String +TEXTMSG: String +INCPHOTO : String +INCPHOTOWITHLOC: String +INCVIDEO: String +INCAUDIO: String +SERVEROFFLINE: String +MESSAGE: String +SENDPHOTO: String +SENDVIDEO: String +SENDAUDIO: String +PATTERN: String -Const()	«Const» klassen lagrer konstante variabler som blir brukt på serveren.

Client -sslSocket: SSLSocket -in: BufferedReader -out: BufferedWriter -enhetsNavn: String -imsi: String -imei: String -mainFrame: JFrame -messages: JTextArea -toClient JTextField -send:JButton -uiBuilt: boolean -calendar: Calendar -la: ResourceBundle -LOGGER: Logger --- +Client(SSLSocket) -buildUI(): void -receiveFile(String, String, String, String): void -scale(File, double): BufferedImage -getCompatibleImage(int, int): BufferedImage -msgReceived(String): void -connected(String, String): void -splitString(String): String[] -generateHash(String): String[] -getHashString(String, String): String +run(): void -checkUI(): void +sendFile(String, File): void +sendString(String): void +getInputReader(): BufferedReader +getSslSocket(): SSLSocket +getImsi(): String +setImsi(String): void +getImei(): String +setImei(String): void +dispose(): void	For hver enhet som kobler seg til serveren blir det opprettet ett objekt av typen «Client». Denne klassen håndterer alle sendinger til og fra denne enheten, samt informasjon om denne enheten som ligger lagret i minnet.

	Disse tre klassene håndterer logging. «LoggerFileFormat» kan brukes for å overstyre hvordan log linjene vil se ut i fil og i konsoll. «LoggerHtmlFormat» definerer hvordan loggingen er formatert som html, slik at logger kan lese i nettleser.
	Dette er klassen hvor man kan finne «public static void main». «MainServer» initialiserer de forskjellige GUI komponentene, setter opp SSLServerSocket for serveren og venter på at klienter skal koble seg til.

Tabell 7: Tabell for forklaring av de viktigste klassene for databasen for serverapplikasjonen. Utarbeidet i Microsoft Visio 2013.

6.8 – Services / handlers

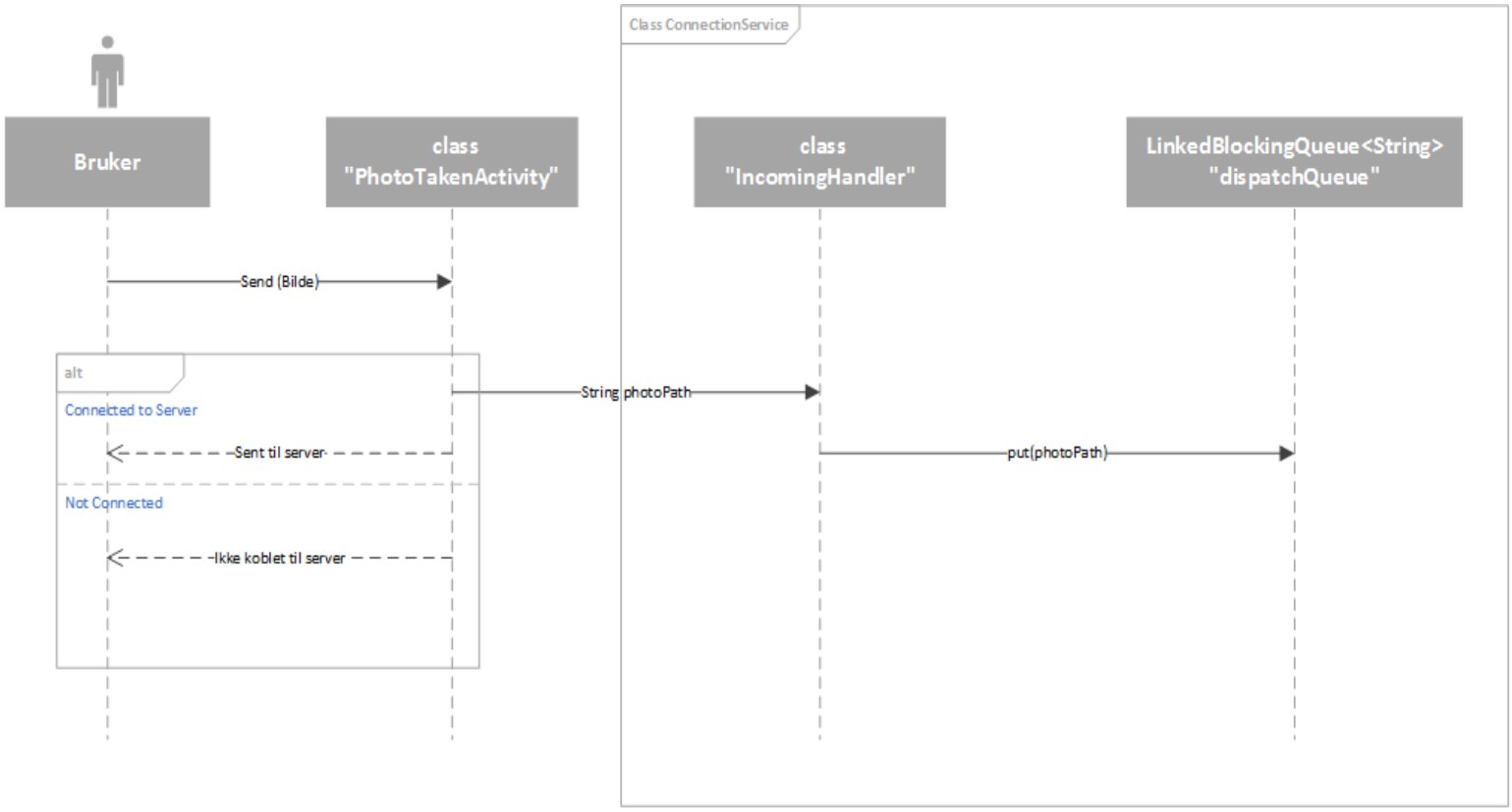
ConnectionService
-Tag: String -protocols: String[] -sLeepTime: long[] -msgHandler: MessageHandler -mClient: Messenger -dispatchQueue: LinkedBlockingQueue<String> -mMessenger: Messenger -reconnectCounter: int -pattern: String -sslSocket: SSLSocket -sslContext: SSLContext -serverAddr: String -serverPort: String -MSG_REGISTER_CLIENT: int -MSG_UNREGISTER_CLIENT: int -MSG_SEND_TO_SERVER: int -MSG_SET_STRING_VALUE: int -MSG_SEND_PHOTO_TO_SERVER:int -MSG_SEND_VIDEO_TO_SERVER:int -MSG_SEND_AUDIO_TO_SERVER:int -MSG_RECONNECT: int -MSG_IS_CONNECTED: int -MSG_CONNECTED_RESPONS: int
-onCreate(): void -onBind(Intent): IBinder -onStartCommand(Intent, int,int): int -setUpTrust(): TrustManagerFactory -isServerConnected(): void -connectSslSocket(): void -setUpSSL(): void -reconnectHandler(): void -identifyUnit(): void -getSocketInfo(): String -reconnect(): void -disconnectSocket(): void

Tabell 8: Klasen for «service» tjenesen som er implementert i systemet. Utarbeidet i Microsoft Visio 2013.

I Android rammeverket er en «Service» en komponent av programmet som gjør operasjoner i bakgrunnen uten interaksjon fra en bruker og fungerer selvom et annet program brukes [49]. Appen som utvikles i dette prosjektet er avhengig av å hele tiden være oppkoblet serveren hos OPS for å fungere er det derfor implementert en «service» som behandler all nettverksrelatert funksjonalitet. Denne vil da «lytte» etter innkommende informasjon, hele tiden gjøre det klart for sending og bistå med at implementert funksjonalitet fungerer hele tiden forutsatt at appen er startet. Figuren til venstre viser denne «service» klassen, dens variabler og metoder.

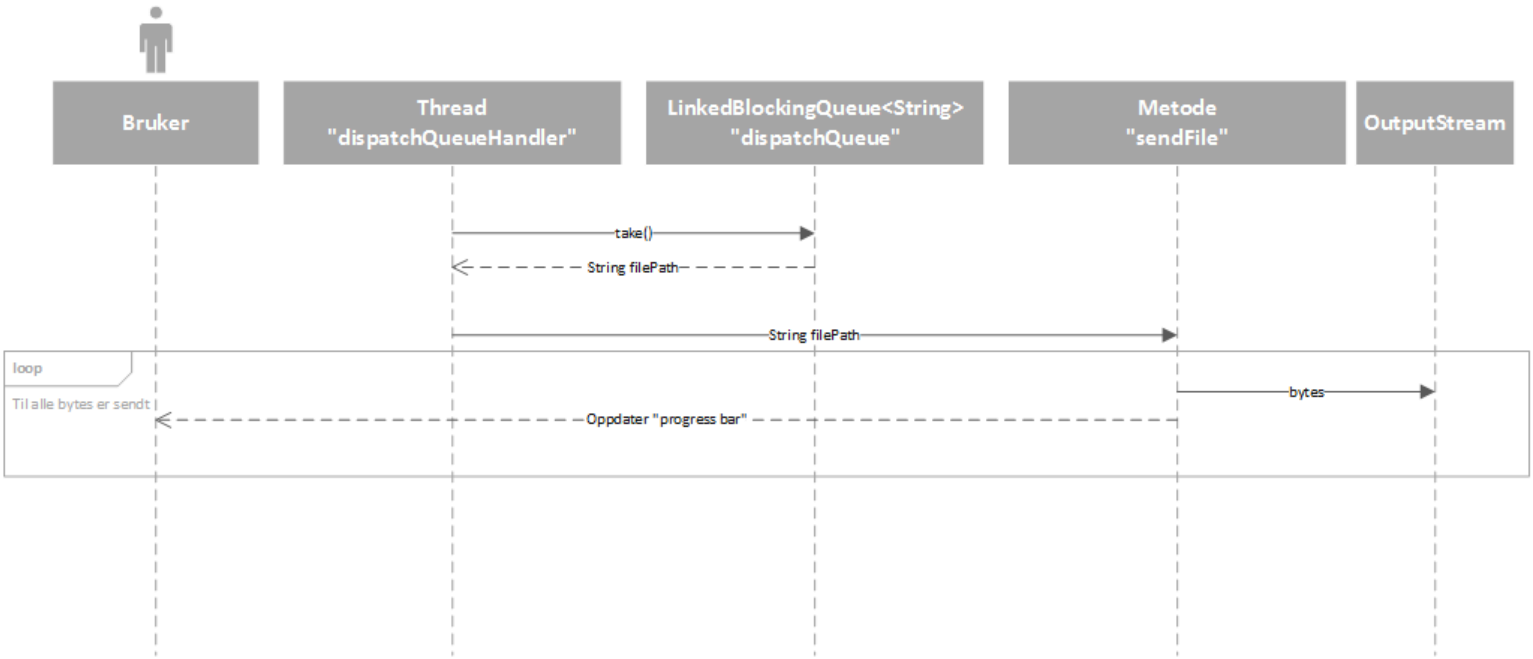
Variabelen «mMessenger» er av typen «Messenger» og er objektet som brukes for å kommunisere med aktivitetene som kobler seg til denne klassen. For å sende og motta informasjon fra serveren har «ConnectionService» en egen privat klasse som kalles «MessageHandler».

Figurene på neste side vil illustrere hvordan de forskjellige klassene fungerer og kommuniserer med hverandre når brukeren skal sende ett bilde.



Figur 20: Sekvensdiagram over aktiviteten som omhandler sending av bilde, og hvordan denne blir behandlet. Utarbeidet i Microsoft Visio 2013.

Figur 20 starter når brukeren trykker på send knappen etter at et bilde er tatt eller valgt. Hvis appen er koblet til serveren vil klassen «PhotoTakenActivity» sende bildets filsti til «ConnectionService». Bildets filsti blir deretter plassert på en kø, og brukeren vil få beskjed om sendinger kunne gjennomføres.



Figur 21: Viser sekvensdiagrammet for fullførelsen av aktiviteten sending av bilde. Utarbeidet i Microsoft Visio 2013

Figuren ovenfor, *figur 21*, viser fortsettelsen fra *figur 20* og hvordan sendingen blir gjennomført. I «MessageHandler» er det en metode som starter en ny tråd, hvor denne tråden brukes for å avvente muligheten for å ta elementer fra køen og starter når appen kobles til serveren. I det filstien plasseres på køen vil denne tråden ta den fra køen og kalle på metoden som sender filen.

7 – Testing og kvalitetssikring

7.1 – Testing underveis

Som en del av systemutviklingsmodellen som er valgt blir det spesifisert tydelige punkter i henhold til hvordan det skal gjøres testing av hver ferdige inkrement som er resultat av endt sprint. Se 1.5.2 – Plan for gjennomføring for konkret definisjon for oppbygning av hver sprint. Det defineres at det vil bli gjort testing i to steg: ved implementasjon og ved integrasjon hvorav ved implementasjon skal det gjøres tester på de utvalgte «user storiene» som er blitt utviklet. Dette ble gjort på den måten at etter hvert som funksjonalitet ble utviklet ble den koden testet for å verifisere at ønsket funksjonalitet var tilfredsstillende. Ved integrasjon vil den/de vellykkede implementasjonene bli testet som en helhetlig bit av systemet og dette ble gjort i form av at utviklet funksjonalitet ble integrert som en tilleggsmodul. For deretter at funksjonaliteten som var ønsket ble testet i selve systemet, enten i appen eller i serverapplikasjonen, for å verifisere at den svarte til ønskene.

7.2 – Brukertest

Som planlagt ble det avholdt en brukertest hos Vestoppland Politidistrikt for å få nyttig tilbakemeldinger som skulle virke styrende for resten av utviklingen. Brukertesten ble avholdt på et senere stadiet enn først planlagt, slik at utviklingen var kommet godt i gang innen dette møtet ble avholdt, til tross nyttige tilbakemeldinger. Det oppstod heller ingen problemer eller funksjonalitet som måtte settes til side som resultat av møtet ble flyttet, siden gruppen selv gjorde forutsetninger sammen med oppdragsgiver.

Brukertesten som ble avholdt var av typen «Pluralistic walkthrough» og er en gjennomgang av ulike forhåndsvalgte scenarioer basert på hvordan systemet vil fungere i praksis. I en slik gjennomgang samles brukere av systemet, utviklere og eksperter på brukergrensesnitt for å kunne innsamle viktige opplysninger som brukertesten produserer. Brukerne som deltar i testen bør representere den faktiske målgruppen slik at tilbakemeldingene som kommer under testen er så relevante som mulig. Utviklerne og ekspertene er med for å kunne svare på spørsmål om brukergrensesnittet eller komme med forslag til løsninger om noe viser seg å være uklart. Før gjennomgangen skal utviklerne ha produsert prototyper av systemet, gjerne i form av papirutskrifter av de forskjellige skjermbildene man vil møte [45].

Under gjennomgangen vil alle testpersonene ta for seg rollen som bruker av systemet og utviklerne vil i plenum gjennomgå de unike scenarioene. Vil fungere på den måten at brukeren vil bli representert med et scenario hvor man skal gjøre en bestemt handling, gjerne knyttet opp i mot et use case, se punkt 3.3 – Use case for eksempler på forskjellige scenarioer, og brukeren blir spurt hvor man vil trykke for å oppnå dette. De vil da skrive ned sitt valg og eventuelle problemer eller misforståelser vil diskuteres før man får representert neste skjermbilde innenfor samme scenario. Dette gjøres i dette mønsteret til man har gått gjennom alle planlagte scenarioer i testen og man vil med en slik gjennomgang ha sterkt fokus på brukeren. Derfor er det meget essensielt å teste opp i mot riktig eller relevant målgruppe siden det er disse som vil benytte systemet ved fullførelse og derfor de beste til å finne feil, mangler eller misforståelser ved brukergrensesnittet [46].

Den praktiske gjennomførelsen av brukertesten ble gjennomført i tråd med teorien, med unntak av mangel på eksperter på brukergrensesnitt tilstede. Dette skyldes at gruppen ikke fikk tak i personer med det som fagområde som kunne stille som medhjelpere til denne testen, så gruppen selv fungerte som både utviklere og som eksperter den dagen.
Testen ble gjennomført for fire personer hos Vestoppland Politidistrikt hvor disse var:

- Seksjonsleder Pål Erik Teigen
- IKT konsulent Jan Erik Ødegård
- Patruljerende politibetjent
- Åstedsgransker

De unike scenarioene som ble presentert for brukerne var:

- Scenario 1: Ta bilde, send til OPS
- Scenario 2: Velge 3 bilder fra galleri, sende til OPS
- Scenario 3: Ta lydopptak, sende til OPS
- Scenario 4: Sjekke signal –og tilkoblingsstatus

Det var på forhånd produsert utskrifter av de forskjellige skjermbildene som var nødvendig for de unike scenarioene, samt et skjema som skulle fylles ut av brukeren for hvert scenario hvor man forklarte hvor man trykte og eventuelt hva som ble misforstått. Se *vedlegg E* for de ulike skjemaene som ble produsert i løpet av brukertesten og fullverdig resultat av brukertesten. Mer overordnet produserte brukertesten positive tilbakemeldinger om systemet, og brukerne forstod jevnt over hva de skulle trykke på for å utføre de forskjellige scenarioene. Det som kom frem var at dagens praksis hos Vestoppland Politidistrikt belaget seg på at det

hovedsakelig var en person i deres distrikt som brukte nettbrettet i bilen for å dokumentere åsteder og brukte e-post for å sende dette til OPS. Årsaken for at ikke flere gjorde dette skyldes mangelen på relevant system og rutiner var årsaken samt mengden med utstyr som man hele tiden måtte kontrollere at var oppladet, tilgjengelig og med.

Åstedsgranskeren som var med på brukertesten presiserte at han og personer i tilsvarende stilling ikke vil erstatte kameraet med en mobilenhet, men nevnte tanken for å kunne implementere muligheten for at bildene produsert av åstedsgranskernes kameraer kunne importeres og behandles med dette systemet. Dette skyldes også at disse personene vil måtte dokumentere større og viktigere saker som ulykker, branner, mm. mens vanlige politibetjenter behandler mindre saker som innbrudd, voldssaker og tilsvarende som ikke har egne kameraer til dette. Et system som kunne erstatte behovet for at hver enkelt betjent må ha egen lydopptaker var svært ønskelig og ble derfor satt meget pris på at var implementert i dette systemet. Et forslag som kom ut ifra det scenarioet var muligheten for å spole, og etter diskusjoner om emnet, ble det i ettertid utviklet som en del av funksjonaliteten.
Ble også nevnt at det var ønskelig å kunne legge ved kommentarer til media, som var en funksjonalitet som ble diskutert med oppdragsgiver som mente at det ikke var nødvendig, men som viste seg å være et ønske allikevel. Avslutningsvis ønsket IKT konsulent Jan Erik Ødegård at det skulle utarbeides en kort innføring i sikkerheten i systemet som de skulle motta per e-post i ettertid av brukertesten, se *vedlegg H*.

7.3 – Test sprint

Etter sprint 5 ble utviklingsarbeidet avviklet og ansett som tilfredsstillende nok i henhold til oppgaven. Det ble det da satt av en periode tilsvarende en sprintlengde kalt «testing og feilretting» som ble benyttet for hardkjøring av systemet for å avdekke svakheter og for å rette de svakhetene som eventuelt dukket opp.

Testingen under denne perioden var tiltenkt å gjøres på den måten at gruppen dro ut på forskjellige lokasjoner innenfor Vestoppland Politidistrikt, men etter drøfting ble det avgjort at det ble gjort i mindre grad enn planlagt på grunnlag av at systemet ikke vil bli tatt i bruk før en eventuell videreutvikling i ettertid av bacheloren. Derfor ble testingen foretatt på den måten at de mobile enhetene som gruppen innehar ble tvunget til å kjøre på forskjellige mobiltjenestene i mobilnettet og prøvde ut forskjellige scenarioer i systemet. Ble foretatt

tester på både «EDGE», «HSPA», «HSPA+», «3G» og «4G» for å verifisere at systemet fungerte på alle de forskjellige teknologiene. Resultatet av testen påviste svært lite feil eller problemer, med unntak av en mangel og en svakhet av mindre betydning. Mangelen viste at implementasjonen av løsningen for å filtrere på forskjellige parametere på serverapplikasjonen ikke var så tilstedeværende som i første omgang planlagt. Filteret for å filtrere på «enheter» var det eneste som var implementert med funksjonalitet, men funksjonaliteten for dette fungerte tilfredsstillende. Vedrørende appen ble det påvist en svakhet i sending av video over «EDGDE» forbindelse ved forsinkelse på et tidsspenn fra 10-40 sekunder før opplastingsindikatoren dukket opp som gjør det mulig å følge prosessen for opplastningen. Dette i kombinasjon med at en video på 5 sekunder tok flere minutter å laste opp, som tydelig forsterker behovet for implementering av komprimering av innhold.

7.4 – SonarQube

Det er foretatt en statisk analyse av all kode som er det utviklede systemet for å opprettholde god kvalitet på koden. Det er benyttet «SonarQube» til dette som er en open source plattform som håndterer kodekvalitet ved å nøye analysere og presenterer resultatetene i et nettleser grensesnitt [52]. Rapportene som ble generert ligger vedlagt, se *vedlegg G*. Ble også benyttet en plugin til Eclipse levert av SonarQube som viser problemene analysen finner direkte i koden. SonarQube rapporter kategorieserer problemer i følgende rekkefølge fra mest til minst kritisk:

- Blocker
- Critical
- Major
- Minor
- Info

7.4.1 – Serverapplikasjon

Etter å ordnet opp i de fleste problemene i koden gjensto det ett «Major» problem og 17 «Minor» problemer.

«Major» problemet som dukker opp er fordi analysen mener at det er en ubrukt variabel i koden, men denne variabelen er det første og eneste objektet som blir initialisert av main funksjonen. Dette er årsaken til hvorfor objektet aldri blir brukt, men det er ingen risiko å

overse denne feilmeldingen siden dette objektetet initialiserer resten av applikasjonen og er derfor nødvendig.

«Minor» problemene som gjenstår er irrelevante av den grunn at det påpekes feil som eksempelvis vil være at det er benyttet tabulator for innrykk i koden isteden for mellomrom. Det er derfor ikke gjort noe med de «minor» feilene som gjenstår siden de er av en slik karakter at det ikke er nødvendig.

7.4.2 – App

Etter retting av kode gjensto det ut ifra rapporten totalt 10 problemer hvorav disse var 7 «Major» og tre «minor». To av disse «major» feilene skyldes at SonarQube oppdager klasser den anser som for store, men som det ikke er blitt valgt å gjøre noe med fordi det ikke er noen hensikt å dele opp disse klassene kun på grunn av det. Resterende «major» feil skyldes duplikasjon av kode og er ikke mulig å unngå siden noen av standard funksjonene trenger noe som er likt. Eksempelvis standardkode for Android aktiviteter som er like.

7.5 – Findbug

Findbug er ett open source program som ser etter potensielle feil i koden ved bruk av statisk analyse [53]. Feilene blir gruppert i følgende kategorier fra mest til minst kritisk [54]:

- Scariest
- Scary
- Troubling
- Of concern

7.5.1 – Serverapplikasjon

Etter fullført analyse av Findbug ble det funnet totalt fire feil, hvorav to var innenfor kategorien «scary», en innenfor «troubling» og en innenfor «of concern».
Feilene innenfor «scary» og «troubling» skyldtes variabler som kunne utløse «Null Pointer Exception», og ble fikset ved å legge inn ekstra sjekker som hindret dette.
Feilen under kategorien «of concern» skyldtes skriving til en «static» variabel og siden dette er det første som skjer når appen starter og før de andre klassene initialiserer er det derfor hetl trygt å la den være.

7.5.2 – App

I koden for appen fant FindBug totalt 28 feil, hvor 23 av disse var innenfor «of concern» og 5 innenfor «troubling». Etter feilretting gjensto bare fire av disse og da fra kategorien «of concern». Disse ble ikke rettet på det grunnlag at de gjaldt hvordan bytes leses inn og konverteres til strenger. Det benyttes plattofrm standard så derfor vil ikke det være et problem [55].

7.6 – Kvalitetssikring

Punktet 1.5.3 – Konfigurasjonsstyring går i detalj på hvordan kvalitetssikring foretas i dette prosjektet og utover dette ble det også benyttet en perm hvor alle viktige dokumenter forbundet med prosjektet, samt møtereferater og viktige avgjørelser ble ivaretatt. Dette for at det skulle være enkelt å slå opp om detaljer rundt implementasjonen, endringer påpekt ved møter, eller tilsvarende og vil derfor bidra positivt for å sikre kvaliteten av både systemet og rapporten.

En del av kvalitetssikringen av systemet var å ha en grundig gjennomgang av sikkerheten på implementasjonsnivå av sikkerhetskonsulent Christoffer Vargtass Hallestensen. Dette ble gjort ved at systemet ble presentert for han først ved overliggende arkitetktur og virkemåte deretter med dypdykk i koden for å vise hvordan forskjellige sikkerhetsløsninger ver blitt utviklet. Det ble da gjort en vurdering om sikkerheten var iveratatt og tilfredsstilte kravene og noen generelle råd for videre gang ble også gitt. Møtet var av stor nyttighet for prosjektet og vil være essentielt for å kunne verifisere at kvaliteten i større grad enn utviklerne selv kan verifisere. Ingen store endringer ble påpekt som nødvendige, men det ble foreslått å bytte fra AES128 til AES256. Det ble også nevnt at hashing burde foregå på et tidligere stadiet enn det opprinnelig gjorde. Dette ble utbedret relativt kjapt etter møtet. Generelle råd omhandlet hvordan å best mulig gripe oppgaven om sikkerhetsvurdering av platformer, gjennomgå koden med statisk analyse verktøy og bruke Enisa dokumenter i større grad enn NIST.
For total oversikt over møtet se møtereferat 4 under *vedlegg N*.

Det er også utarbeidet to feiltre i forbindelse med systemet, hvor det ene vil ta for seg risikoene for systemfeil (topphendelse) og er ment hovedsakelig opp mot serverapplikasjonen. Hvorav det andre tar for seg risikoene for sendingsfeil (topphendelse) og er fokusert hovedsakelig mot appen. Feiltreene er vedlagt som vedlegg og finnes under *vedlegg J*.

7.6.1 – Sedvane kode

Kildekoden skrives på engelsk, og med dette menes variabler, kommentarer og andre elementer i koden som må navngis og gjøres på grunnlag av å holde koden konsistent. Alle «public» funksjoner dokumenteres i form av «Javadoc», og i tillegg vil større komplekse funksjoner få kommentarerer i koden. Disse bidragene vil virke positivt for en eventuell videreutvikling ved at utviklerne kan får god dokumentasjon på utviklet kode, samt at norsk navngivning i engelsk programmeringsspråk ikke fungerer sømløst sammen.

Definering av klasser starter alltid med stor forbokstav i kombinasjon med at oppkommende orddelinger også skilles med stor forbokstav. Eksempelvis «AddUnitForm», noe som vil bidra med å øke lesbarheten og ser mer elegant ut i koden enn bruk av bindestrek eller understrek. Denne definereingen av orddelinger gjelder, i tillegg til klasser, for variabler og funksjoner, men disse starter da med liten forbokstav.

Standarden som benyttes i prosjektet for dokumentering av kildekode er «Javadoc», som er et verktøy for å skrive API dokumentasjon, som vil ha tett tilhørlighet til kildekoden ved at den kan ekspanderes og leses om ønskelig. «Javadoc» dokumentet er også mulig å eksportere til HTML format for å kunne leses utenfor kildekoden eller legges med som vedlegg. Siden «Javadoc» dokumentet inneholder flere titallssider vil det kun bli lagt ved eksempler tilhørende kodeeksemplene under vedlegg. Se *vedleggene: O, P, Q, R, S.*

8 – Avslutning

8.1 – Drøftinger

Drøftinger vedrørende valg og beslutninger som er gjort i løpet av prosjektet gjøres for det meste sammen med avsnittene som de omhandler. Drøftinger som ikke kom naturlig med i rapporten gjøres her.

Det er utarbeidet en brukermanual som vil forklare bruk og ikonene i appen. Det er ikke utarbeidet en slik brukermanual for serverapplikasjonen av den grunn at gruppen kom frem til at det ikke var noe stort behov for dette. Årsaken til dette skyldes at utviklingen av serverapplikasjonen kun er på et tidlig proof-of-concept stadie.

Det ble vurdert om muligheten for å implementere frisøk som en mulighet i serverapplikasjonen i tillegg til at man kan filtrere ut media på nevnte attributer. Dette var tenkt som en løsning om man ønsket å skrive søkeord istedet for å filtrere, men denne idèen ble droppet fordi vi kom frem til at det ikke var noen attributer i databasen å gjøre frisøk på. Filterene som er planlagt vil dekke alle disse attributene slik at et frisøk felt ikke ville gjort det mulig å søkt på noe utover hva som allerede ble tilbudt gjennom filtre. Eneste muligheten for implementasjon av dette ville vært å kunne gjort frisøk på samme attributer som filtrene bruker, slik at man hadde valget om å bruke filter eller å skrive søkeord om det var ønskelig. Vi kom frem til at tiltross for at dette ville vært mulig og kanskje kunne effektivisert prosessen med å finne ønsket media om man visste spesifikt hva man så etter så ble det valgt å ikke imlpementere dette siden det kom til å kreve en litt kompleks løsning og ble derfor valgt å prioritere andre oppgaver.

8.1.1– Resultater

Det var satt som ett effektmål å levere et proof-of-concept system som skulle bistå Vestoppland Politidistrikt med nevnte utfordringer oppramset i prosjektbeskrivelsen. Etter presentasjon av systemet for Vestoppland Politidistrikt og oppdragsgiver ble det gitt uttrykk for at systemet fungerte tilfredsstillende opp i mot kravene. Oppdragsgiver har i løpet av hele prosjektperioden vært meget fornøyd med gruppens og systemets kvalitet og gitt tilbakemeldinger om overraskende stor progresjonen vedrørende utviklingen.

Utviklingsperioden ble i starten av prosjektet definert i ett GANTT-skjema hvor det ble satt opp utviklingsperioder på to uker omgangen, noe som viste seg å fungere meget elegant tidlig i prosjektet. Backlogen inneholdt krav til funksjonalitet som var hverken for store eller for små slik at de passet veldig bra opp i mot definerte sprinter. Det resulterte i at ingen krav trengte å deles opp over flere sprinter, noe som viser at planleggingen var godt gjennomtenkt og gjennomført.

Proof-of-concept systemet som leveres tilfredsstiller kravene om kjernefunksjonalitet og svarer på ønsket om å se på hvordan et slikt system kan implementeres i praksis. I tillegg ble det gjort grundig analyse for å se på mulige løsninger og sikkerhetsaspekter som trengs for et slikt system. Appen er den delen av systemet som ble høyest prioritert og har derfor kommet lengst med tanke på funksjonalitet. Det resulterer i at appen anses som ferdig i den grad at resterende arbeid anses som videreutvikling. Serverapplikasjonen har blitt utviklet for å kunne fungere som en del av systemet, men trenger noe arbeid for å komme opp til stadiet som appen.

8.2 – Kritikk av oppgaven

«JUnit» er et «unit testing» verktøy for kodespråket Java og er essentiell i test-dreven utvikling. «Unit testing» baserer seg på konseptet om å teste induviduelle deler av kildekoden enten i så små deler som mulig eller i sett av en eller flere moduler. Dette for å avgjøre om de kvalifiserer til bruk ved at man sjekker om kodet funksjonalitet svarer til forventningene. Det er derfor vanlig å skrive testene på forhånd siden man vet hva slags parametere funksjonen tar i mot og hvilke parametere som kommer ut. I dette prosjektet benyttes det ikke en test-dreven systemutviklingsmodell og anses derfor som en av hovedårsakene til å ikke implementere dette tiltross for fordeler det vil gi. En annen årsak for at dette ikke ble implementert er at gruppen anså planlagt plan for testing, se 1.5.2 – Plan for gjennomføring, som god nok til å verifisere at utviklede funksjoner fungerte som de skulle uten å bruke overdrevent mye tid på testing. Samt at det var ønskelig å bruke mer tid på selve utviklingen enn på å skrive testkode og siden valgt systemutviklingsmodell ikke vektlegger dette ble det ikke gjennomført.

«Source-Tree» er programmet som ble benyttet til versjonshåndtering opp i mot «Bitbucket» og dette ble opplevd som delvis trøblete og rotete å bruke. Dette anses ikke som kritikk mot selve oppgaven, men skapte noen utfordringer vedrørende arbeid på oppgaven ved at det

skjedde litt oftere enn det burde at kode ble sammenslått feil, og at hverandres oppdateringer ikke kom med. Dette skyldes mest sannsynlig feil bruk, men opplevdes som generelt tungvint og ikke veldig trivielt i hvilken rekkefølge man skulle «pushe», «pulle», osv. Derfor kunne det føles som en hindring til tider og skapte tidvis hodebry og ressurser måtte brukes for å rydde opp i dette.

8.3 – Videre arbeid

Hele prosessen med videre arbeid må i første omgang starte med at POD blir instansen som står ansvarlig for at dette prosjektet skal fullføres. Dette siden det er de som styrer alle distriktene og som nevnt på grunnlag av deres retninglinjer må det derfor være de som distribuerer leveringen av et slikt system istedet for et utviklerteam direkte til distriktene. Mer konkret for hva som må gjøres for at serverapplikasjonen skal fungere som et selvstendig program på den måten at den ikke kun svarer opp i mot funksjonaliteten til appen, listes opp her:

- Videreutvikling av server GUI
- Filter muligheten er implementert til den grad at det er mulig å filtrere ut media mottatt fra forskjellige enheter. Denne må utvides til å kunne filtrere på de andre planlagte filtrene som nevnes i punktet 5.2.2 – Aktiviteter og filter.

- Hele funksjonaliteten rundt database tabellen «Event» må implementeres, men dette gjøres i et tettere samarbeid med Politiet siden dette vil kreve samhandling med deres interne systemer. Denne funksjonaliteten vil muliggjøre å legge til sakskode knyttet opp i mot media slik at media fra flere enheter, dager, mm. kan knyttes til samme sak.

- Videreutvikling av server GUI vil bestå av flere småendringer, men hovedsakelig går det på grafisk design av programmet. Sekundært funksjonalitet som muliggjør å tømme filtrene og tømme hovedskjermen. Det kunne også vært mulig å sett på hvordan å løse nevnt funksjonalitet for å gjøre frisøk i databasen.

Det som trengs av videreutvikling i appen er implementering av ønskelig funksjonalitet i forbindelse med brukertesten hvor Vestoppland Politidistrikt uttrykket ønske om å kunne legge til kommentar til media som skal sendes. Utover dette vil det være å fjerne test relaterte funksjoner og gjøre en beslutning vedrørende implementert chat-funksjonalitet, enten ved å fjerne eller å ferdigstille. En annen nødvendighet som bør prioriteres tidlig i videre arbeid er

implementeringen av «push notification», dette for at funksjonaliteten for at OPS kan videresende til appen skal fungere. Det fungerer med dagens implementasjon, men dette krever at appen er åpen for å kunne motta noe. En annen ting vedrørende varslinger omhandler at det må implementeres meldinger om at sending er fullført og mottatt av serveren. Dette bidrar til at det ikke oppstår misforståelser om sendinger er fullført og mottatt.

Det ble tydeliggjort tidlig i prosjektet at Vestoppland Politidistrikt benytter hovedsakelig iOS enheter, men vår oppdragsgiver presiserte at det uten problemer kunne utvikles til Android til tross for dette. I tillegg viste presentasjonen av systemet hos Vestoppland Politidistrikt at de ikke så på dette som en hindring, ved at det alltid utvikles for en type operativsystem først og i kombinasjon med at Politidirektoratet ikke ville tillatt dem å tatt i bruk systemet.
I følge Politidirektoratets rutiner kan ikke enkelte distrikter ta i bruk programvare som ikke er distribuert av direktoratet selv. De følger prinsippet "alle, eller ingen" som vil si at det må kunne distribueres til alle distriktene hvis ikke er det ikke tillatt å ta det i bruk.

I analysearbeidet fremmes det forskjellige løsninger til hvordan deloppgaven «Publikum → OPS» kan bli løst. Det lar seg absolutt gjøre å utvikle noe slikt og det vil fylle et etterlengtet hull i samfunnet for det finnes ingen muligheter for tipsing til Politiet sett bortifra innringing til sentralen eller nødnummeret. Derfor vil det være noe som bør være med i videre arbeid etter at hovedsystemet er ferdig utviklet og implementert.

8.4– Evaluering av gruppas arbeid

8.4.1 - Innledning

Vi startet arbeidet så tidlig det lot seg gjøre ved å drøfte arbeidsmetoder, hvordan oppgaven skulle angripes og drøftet i grove trekk hvilke løsninger vi så for oss som mulige å gjennomføre. Det ble også gjennomført en grov estimering av tiden vi hadde til rådighet for å få et omtrentlig overblikk over hvor mye av oppgaven vi skulle ta for oss.
Gruppen startet veldig motivert og har holdt motivasjonen oppe i løpet av hele prosjektet noe som har bidratt til høyt engasjement og god arbeidsmoral. Samholdet har vært strukturert og vi har planlagt arbeidsoppgavene godt før de ble påbegynt.
Oppdragsgiver har vært meget motivert og samarbeidsvillig under prosjektperioden noe som har bidratt positivt til en allerede spennende oppgave.

8.4.2 - Organisering

Organiseringen av utviklingen fungerte meget godt ved bruk av digitalt SCRUM board og gjorde at det til enhver tid var oversikt over hva som stod på agendaen. Dropbox som fellesportal for oppbevaring av dokumenter og illustrasjoner har fungert uten negative konsekvenser og har bidratt til at gruppen enkelt kunne få innblikk i hverandres arbeid og benytte utarbeidede illustrasjoner. I tillegg til Dropbox for deling av digitale dokumenter ble det benyttet en felles perm for oppbevaring av møte referater, avgjørelser og generelt viktige dokumenter. Det gjorde det enklere å slå opp nødvendig informasjon under møter og fungerte som en bra erstatter for å måtte finne frem dokumentene på en PC.

Skype og facebook chat ble benyttet i stor grad for kommunikasjon blant gruppemedlemmene. Dette gjorde at gruppemedlemmene til enhver tid var tilgjengelige for hverandre og bidro til at respons ble gitt mye raskere enn ved bruk av annen kommunikasjon.

8.5 – Fordeling av arbeid

Når arbeidet skulle fordeles ble hverandres erfaringer diskutert for å avgjøre hvordan hverandres ressurser best mulig kunne benyttes. Alle gruppemedlemmene har vært delaktige i alt som har blitt gjort, men vi fordelte arbeidet i to hovedområder hvor den ene var analysearbeid og den andre var utvikling. Resultatet av dette var mer effektiv bruk av ressursene og bidro til at gruppemedlemmene kunne fokusere hovedsakelig på sitt tyngdeområde.

8.6 – Konklusjon

Til tross for at systemet ikke vil kunne tas i bruk ved sluttføring av prosjektet grunnet valg av OS og restriksjoner fra POD har det i aller største grad vist seg at det vil være mulig å bistå ikke bare Vestoppland Politidistrikt, men hele Politi Norge med en løsning for deres utfordringer vedrørende overføring av digital informasjon. Helt kontret var oppgavens mål å kunne se på mulige løsninger, helst levere et proof-of-concept og vise implementering av dette i praksis hvor vi har gjort begge deler. Vi har både trukket frem forskjellige løsninger som er på markedet som kan brukes som de er eller med utvikling av tilleggsmoduler kan benyttes for å fylle ønsket behov. Selvom dette er løsninger ment hovedsakelig for andre

formål så kom vi frem til at dette var uten tvil nettopp det oppgaven spør om: mulige løsninger. Siden dette kun var en del av oppgaven og det ble presisert at det helst skulle utvikles et eget system så ble hovedfokuset lagt på nettopp det området. Resultatet av dette ble et proof-of-concept med en del funksjonalitet og en suplerende rapport som spesifierer tydelig veien videre for å få dette til å bli et ferdig operativt system som kan brukes i deres hverdag. Selve definisjonen av et proof-of-concept vil være at det skal lages en liten del for å kunne demonstrere at et konsept eller en teori faktisk har potensialet for å kunne bli brukt. Vårt system demonenstrerer nettopp dette ved å allerede på dette stadiet ha den kvaliteten i sikkerheten som kreves av det fullverdige programmet og funksjonalitet i stor grad for å vise ikke bare en liten del, men nesten hele bruksområdet. Om man for konklusjonensskyld ser bort ifra retningenlinjene fra POD og med et Android nettbrett/mobil tilgjengelig ville det vært mulig å allerede i dag å sette opp systemet hos Vestoppland Politidistrikt slik at de kunne tatt dette i bruk.

I løpet av prosessen har det, som i prosjekter flest, dukket opp noen utfordringer. Dette som eksempelvis feil i vår antagelse som ble gjort sammen med oppdragsgiver om at kommentar til media ikke var nødvendig viste seg å være veldig ønskelig alikevel. Et annet eksempel er muligheten for spoling i lydopptak som vi ikke hadde tenkt på i det hele tatt, men som viste seg å være et stort ønske av betjentene og ble derfor også implementert meget raskt. Så er det den største utfordringen som omhandler Vestoppland Politidistrikts påpeking av PODs retningslinjer og at de ikke var tillatt å ta i bruk systemet som ble utviklet i denne bacheloroppgaven. Selvom dette var en stor motgang i prosessen så resulterte det med at vi har tatt kontakt med POD og er i forhandlinger med de vedrørende videre gang av prosjektet.

Derimot på den positive siden viste brukertesten at behovet var enda større enn først antatt når vi så mottakelsen systemet fikk av Vestoppland Politidistrikt. De likte konseptet veldig godt og kunne ikke sett for seg at det var mulig å løse så mange av deres utfordringer på en slik sømløs måte. Derfor naturlig nok ønsker de veldig å se fullførelse av systemet og at det godtas av POD. Dette virket meget motiverende på oss, og det var en positv bidragsyter å se hvor godt mottatt systemet allerede ble på dette tidlige stadiet.

Uifra oppgaveteksten kommer det tydelig frem at dette var en stor oppgave som ville kreve mye tid, ressurser og risikoen for å bite over for mye var stor. Det gjorde det ikke lettere at oppgavedefinisjonene var såpass bred og la mye av ansvaret opp til oss. Det var derfor veldig

avgjørende for vår suksess å gjøre skarpe avgrensninger og velge ut det som var viktigst uten at det ble for lite eller for mye å gjøre. Vi synes at vi klarte dette i god grad siden resultatet er både relevant, inneholder det som ble planlagt og demonstrerer det viktigste av funksjonaliteter samt litt ekstra. På dette grunnlaget samt all sikkerhetsvurdering som måtte gjøres ble oppgaven delt i to hoveddeler, bestående av analysearbeid og utviklingsarbeid. Vi løste dette veldig godt med å dele gruppen opp i to, slik at fokuset hele tiden foregikk på begge områder som resulterte i at vi i dag har et komplett og bredt analysearbeid og et fungerende proof-of-concept system. Strukturert planlegging, godt samhold, flinke ressurspersoner og høy motivasjon har vært pådrivere til at vi i dag er ved veis ende med de resultatene som vises til og er en kunnskapsrik avslutning på studiene ved Høgskolen i Gjøvik.

Referanser

[1] Oracle. Lesson: Introduction [Internetside]. [Sitert 29.04.2015] Tilgjengelig fra: https://docs.oracle.com/javase/tutorial/i18n/intro/index.html

[2] CCJK Technologies Co., Ltd. What is Internationalization (i18n), Localization (L10n) and Globalization (g11n) [Internetside]. [Sitert 29.04.2015] Tilgjengelig fra: http://www.ccjk.com/what-is-internationalization-i18n-localization-l10n-and-globalization-g11n/

[3] Oracle. Internationalizing the Sample Program [Internetside]. [Sitert 29.04.2015] Tilgjengelig fra: https://docs.oracle.com/javase/tutorial/i18n/intro/steps.html

[4] Google Inc. Supporting different languages [Internetside]. [Sitert 29.04.2015] Tilgjengelig fra: http://developer.android.com/training/basics/supporting-devices/languages.html

[5] Dropbox, Inc. Dropbox for Business security | A Dropbox whitepaper [Internetside]. [Oppdatert 16.01.2015, sitert 22.04.2015] Tilgjengelig fra: https://www.dropbox.com/static/business/resources/dfb_security_whitepaper.pdf

[6] Microsoft Corporation. What is Bitlocker? What does it do? What does it not do? [Internettside]. [Oppdatert 17.03.2010, sitert 28.04.2015] Tilgjengelig fra: http://blogs.technet.com/b/uspartner_ts2team/archive/2010/03/17/what-is-bitlocker-what-does-it-do-what-does-it-not-do.aspx

[7] Dropbox, Inc. Everything you need to work smarter [Internetside]. [Sitert 22.04.2015] Tilgjengelig fra: https://www.dropbox.com/business/pricing

[8] Microsoft. Select a plan [Internetside]. [Sitert 22.04.2015] Tilgjengelig fra: https://products.office.com/en-us/business/compare-more-office-365-for-business-plans

[9] Owncloud. User Management [Internetside]. [Sitert 24.04.2015] Tilgjengelig fra: https://doc.owncloud.org/server/8.0/admin_manual/configuration_user/user_configuration.html

[10] Owncloud. ownCloud Server Administration Manual Release 8.0 [Internettside]. [Oppdatert 23.04.2015, Sitert 24.04.2015] Tilgjengelig fra: https://doc.owncloud.org/server/8.0/ownCloudServerAdminManual.pdf

[11] – The Internet Engineering Task Force (IETF). The Transport Layer Security (TLS) Protocol Version 1.2 [Internettside]. [Oppdatert august 2008, sitert 6.02.2015] Tilgjengelig fra: http://www.ietf.org/rfc/rfc5246.txt

[12] -Wikipedia. Transport Layer Secutiry [Internettside]. [Oppdatert 09.02.2015, sitert: 09.02.2015] Tilgjengelig fra: http://en.wikipedia.org/w/index.php?title=Transport_Layer_Security&oldid=646367385

[13]- NIST. Recommendation for Key Management – Part 1: General (Revision 3) [Internettside]. [Oppdatert: juli 2012, sitert: 12.02.2015] Tilgjenglig fra: http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57_part1_rev3_general.pdf

[14] –Microsoft TechNet. What is TLS/SSL? [Internettside]. [Oppdatert: 28.03.2003, sitert: 12.02.2015] Tilgjengelig fra: https://technet.microsoft.com/en-us/library/cc784450%28v=ws.10%29.aspx

[15] – Codenomicon. The Heartbleed Bug [Internettside]. [Oppdatert: 29.04.2014, sitert: 13.02.2015] Tilgjengelig fra: http://heartbleed.com/

[16] – EKR (Educated Guesswork). Understanding the TLS Renegotiation Attack [Internettside]. [Oppdatert: 5.11.2009, sitert: 15.02.2015] Tilgjengelig fra: http://www.educatedguesswork.org/2009/11/understanding_the_tls_renegoti.html

[17] – Nadhem AlFardan, Dan Bernstein, Kenny Paterson, Bertram Poettering, Jacob Schuldt. On the Security of RC4 in TLS and WPA [Internettside]. [Oppdatert: 28.08.2013, sitert: 17.02.2015] Tilgjengelig fra: http://www.isg.rhul.ac.uk/tls/

[18] – NIST. Recommendation for Key Management – Part 1: General (Revised) [Internettside]. [Oppdatert: 08.03.2007, sitert 17.02.2015] Tilgjengelig fra: http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf

[19] – The Internet Engineering Task Force (IETF). IP Authentication Header. [Internettside]. [Oppdatert: desember 2005, sitert: 06.05.2015] Tilgjengelig fra: https://tools.ietf.org/html/rfc4302

[20] – Android Developer. SSLSocket [Internettside]. [Sitert: 17.02.2015] Tilgjengelig fra: http://developer.android.com/reference/javax/net/ssl/SSLSocket.html

[21] –Poncho (StackExchange Cryptography forum). What is the difference between CBC and GCM mode? [Internettside]. [Oppdatert: 09.04.2012, sitert: 18.02.2015] Tilgjengelig fra: http://crypto.stackexchange.com/questions/2310/what-is-the-difference-between-cbc-and-gcm-mode

[22] – Wikipedia. Forward Secrecy [Internettside]. [Oppdatert: 18.02.2015, sitert: 20.02.2015] Tilgjengelig fra: http://en.wikipedia.org/wiki/Forward_secrecy

[23] – Digicert. Enabling Perfect Forward Secrecy [Internettside]. [Sitert: 20.02.2015] Tilgjengelig fra: https://www.digicert.com/ssl-support/ssl-enabling-perfect-forward-secrecy.htm

[24] – Ricky Donato. PKI-Chain of Trust [Internettside]. [Oppdatert: 19.03.2012, sitert: 20.02.2015] Tilgjengelig fra: https://www.fir3net.com/Security/Concepts-and-Terminology/pki-chain-of-trust.html

[25] –BrightMinded Ltd. SSL Basics (Youtube video) [Internettside]. [Oppdatert: 13.04.2013, sitert: 20.02.2015] Tilgjengelig fra: https://www.youtube.com/watch?v=3p_e00tEZM8

[26] – Mike-Dee. SSL/TLS protocol overview (Part 1) [Internettside]. [Oppdatert: 03.04.2013, sitert: 21.02.2015] Tilgjengelig fra: http://mihalos.gr/2013/04/03/ssltls-protocol-overview-part1/

[27] – Wikipedia. OSI model [Internettside]. [Oppdatert: 19.02.2015, sitert: 20.02.2015] Tilgjengelig fra: http://en.wikipedia.org/wiki/OSI_model

[28] – Wikipedia. Virtual Private Network [Internettside]. [Oppdatert 13.03.2015, sitert 14.03.2015] Tilgjengelig fra: http://en.wikipedia.org/wiki/Virtual_private_network

[29] – Microsoft TechNet. What is VPN? [Internettside]. [Oppdatert 28.03.2003, sitert 14.03.2015] Tilgjengelig fra: https://technet.microsoft.com/en-us/library/cc739294%28v=ws.10%29.aspx

[30] – The Internet Engineering Task Force (IETF). Security Architecture for the Internet Protocol [Internettside]. [Oppdatert desember 2005, sitert 14.03.2015] Tilgjengelig fra: https://tools.ietf.org/html/rfc4301#section-3.1

[31] – The Internet Engineering Task Force (IETF). Cryptographic Algorithm Implementation Rewuirements and Usage Guidance [Internettside]. [Oppdatert august 2014, sitert 15.03.2015] Tilgjengelig fra: http://tools.ietf.org/html/rfc7321#section-2.2

[32] –Wikipedia. Secure Shell [Internettside]. [Oppdatert 04.03.2015, sitert 16.03.2015] Tilgjengelig fra: http://en.wikipedia.org/wiki/Secure_Shell

[33] – The Internet Engineering Task Force (IETF). The Secure Shel (SSH) Transport Layer Protocol [Internettside]. [Oppdatert januar 2006, sitert 16.03.2015] Tilgjengelig fra: https://tools.ietf.org/html/rfc4253

[34] – Henrick Hellström (Forum). Why is TLS susceptible to protocol downgrade attacks? [Internettside]. [Oppdatert 21.09.2013, sitert 06.03.2015] Tilgjengelig fra: http://crypto.stackexchange.com/questions/10493/why-is-tls-susceptible-to-protocol-downgrade-attacks

[35] – The Internet Engineering Task Force (IETF). HMAC: Keyed-Hashing for Message Authentication [Internettside]. [Oppdatet februar 1997, sitert 08.03.2015] Tilgjengelig fra: http://tools.ietf.org/html/rfc2104

[36] – The Internet Engineering Task Force (IETF). The TLS Protocol Version 1.0 [Internettside]. [Oppdatert januar 1999, sitert 09.03.2015] Tilgjengelig fra: http://tools.ietf.org/html/rfc2246

[37] – Cisco TAC Engineers. SSLv3 and TLSv1 Protocol Weak CBC Mode Vulnerability [Internettside]. [Oppdatert 14.10.2014, sitert 10.03.2015] Tilgjengelig fra: http://www.cisco.com/c/en/us/support/docs/security/email-security-appliance/118518-technote-esa-00.html

[38] – The internet Engineering Task Force (IETF). The Transport Layer Security (TLS) Protocol version 1.1 [Internettside]. [Oppdatert april 2006, sitert 10.03.2015] Tilgjengelig fra: http://tools.ietf.org/html/rfc4346

[39] – Robert D. Silverman. Has the RSA algorithm been compromised as a result of Bernstein's paper? [Internettside]. [Oppdatert 08.04.2002, sitert: 24.03.2015] Tilgjengelig fra: http://www.emc.com/emc-plus/rsa-labs/historical/has-the-rsa-algorithm-been-compromised.htm

[40] – NIST. NIST's policy on hash functions [Internettside]. [Oppdatert 28.09.2012, sitter 26.02.2015] Tilgjengelig fra: http://csrc.nist.gov/groups/ST/hash/policy.html

[41] – Enisa. Recommended cryptographic measures – Securing Personal data [Internettside]. [Oppdatert 04.11, sitert 23.03.2015] Tilgjengelig fra: https://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/recommended-cryptographic-measures-securing-personal-data

[42] – The Internet Engineering Task Force (IETF). Diffie-Hellman Key Agreement Method. [Oppdatert: juni 1999, sitert: 05.05.2015] Tilgjengelig fra: https://www.ietf.org/rfc/rfc2631.txt

[43] – The Internet Engineering Task Force (IETF). Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec [Internettside]. [Oppdatert: Mai 2007, sitert: 06.05.2015]. Tilgjengelig fra: https://tools.ietf.org/html/rfc4868

[44] – OpenSSH. OpenSSH [Internettside]. [Sitert 06.05.2015] Tilgjengelig fra: http://www.openssh.org/.

[45] - Rogers, Y. Sharp, H. Preece, J. Interaction Design. 3rd ed. Chichester, United Kingdom: John Wiley & Sons Ltd; 2011

[46] - Lazar, J. Feng, J H. Hochheiser, H. Research Methods In Human-Computer Interaction. Chichester, United Kingdom: John Wiley & Sons Ltd; 2010

[47] – Oracle. Java DB [Internettside]. [Sitert 10.05.2015] Tilgjengelig fra: http://www.oracle.com/technetwork/java/javadb/overview/index.html

[48] – Oracle. Javadoc Tool [Internettside]. [Sitert 06.05.2015] Tilgjengelig fra: http://www.oracle.com/technetwork/articles/java/index-jsp-135444.html

[49] – Google, inc. Services [Internettside]. [Sitert 10.05.2015] Tilgjengelig fra: http://developer.android.com/guide/components/services.html

[50] – Google, inc. Introduction to Android [Internettside]. [Sitert 06.05.2015] Tilgjengelig fra: http://developer.android.com/guide/index.html

[51] – Google, inc. Android Studio [Internettside]. [Sitert 06.05.2015] Tilgjengelig fra: https://developer.android.com/sdk/index.html

[52] – SonarSource Sa. SonarQube™ [Internettside]. [Sitert 02.05.2015]. Tilgjengelig fra: http://www.sonarqube.org

[53] – The University of Maryland. FindBugs™ Find Bugs in Java Programs [Internettside]. [sitert 02.05.2015]. Tilgjengelig fra: http://findbugs.sourceforge.net

[54] – The University of Maryland. FindBugs 2 [Internettside]. [Sitert 02.05.2015] Tilgjengelig fra: http://findbugs.sourceforge.net/findbugs2.html

[55] – Oracle. Class FileReader [Internettside]. [Sitert 05.05.2015] Tilgjengelig fra:

http://docs.oracle.com/javase/7/docs/api/java/io/FileReader.html

[56] – Google, inc. Personvernregler [Internettside]. [Oppdatert 25.02.2015, sitert 23.04.2015] Tilgjengelig fra: https://static.googleusercontent.com/media/www.google.com/no//intl/no/policies/privacy/google_privacy_policy_no.pdf

[57] – Motorola. Devices running Android 4.1 (Jelly Bean) and Android 4.0 (Ice Cream Sandwich) operating systems [Internettside]. [Oppdatert 25.02.2015, sitert 23.042015] Tilgjengelig fra: http://www.motorola.com/us/About_Motorola-Legal-JB-ICS-Device-Policy/About_Motorola-Legal-JB-ICS-Device-Policy.html

[58] – HTC. Personvererklæring [Internettside]. [Oppdatert 29.09.2014, sitert 24.04.2015] Tilgjengelig fra: http://www.htc.com/no/terms/privacy/

[59] – Techtarget Margaret Rouse. Application Sandboxing [Internettside]. [Sitert 24.04.2015] Tilgjengelig fra: http://searchconsumerization.techtarget.com/definition/application-sandboxing

[60] – Lifehacker. Android for Work Explained: How Android Can Take over Your Work Email, Calendar, Apps and More..[Internettside]. [Sitert 24.04.2015] Tilgjengelig fra: http://www.lifehacker.co.uk/2015/02/26/android-work-explained-android-can-handle-w

[61] – Google Play. Capsule VPN Internettside]. [Oppdatert 09.05.2015, sitert 12.05.2015] Tilgjengelig fra: https://play.google.com/store/apps/details?id=com.checkpoint.VPN

Vedlegg A

Prosjektavtale

2 Sider

HØGSKOLEN I GJØVIK

PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

IT-tjenesten (oppdragsgiver), og

Daniel Kristoffersen, Eirik Lintho Bue og Eirik Ellegård

(student(er))

Avtalen angir avtalepartenes plikter vedrørende gjennomføring av prosjektet og rettigheter til anvendelse av de resultater som prosjektet frembringer:

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 05. jan. 15 til 15. mai. 15.

   Studentene skal i denne perioden følge en oppsatt fremdriftsplan der HiG yter veiledning. Oppdragsgiver yter avtalt prosjektbistand til fastsatte tider. Oppdragsgiver stiller til rådighet kunnskap og materiale som er nødvendig for å få gjennomført prosjektet. Det forutsettes at de gitte problemstillinger det arbeides med er aktuelle og på et nivå tilpasset studentenes faglige kunnskaper. Oppdragsgiver plikter på forespørsel fra HiG å gi en vurdering av prosjektet vederlagsfritt.

2. Kostnadene ved gjennomføringen av prosjektet dekkes på følgende måte:
   - Oppdragsgiver dekker selv gjennomføring av prosjektet når det gjelder f.eks. materiell, telefon/fax, reiser og nødvendig overnatting på steder langt fra HiG. Studentene dekker utgifter for trykking og ferdigstillelse av den skriftlige besvarelsen vedrørende prosjektet.
   - Eiendomsretten til eventuell prototyp tilfaller den som har betalt komponenter og materiell mv. som er brukt til prototypen. Dersom det er nødvendig med større og/eller spesielle investeringer for å få gjennomført prosjektet, må det gjøres en egen avtale mellom partene om eventuell kostnadsfordeling og eiendomsrett.

3. HiG står ikke som garantist for at det oppdragsgiver har bestilt fungerer etter hensikten, ei heller at prosjektet blir fullført. Prosjektet må anses som en eksamensrelatert oppgave som blir bedømt av faglærer/veileder og sensor. Likevel er det en forpliktelse for utøverne av prosjektet å fullføre dette til avtalte spesifikasjoner, funksjonsnivå og tider.

4. Den totale besvarelsen med tegninger, modeller og apparatur så vel som programlisting, kildekode, disketter, taper mv. som inngår som del av eller vedlegg til besvarelsen, gis det en kopi av til HiG, som vederlagsfritt kan benyttes til undervisnings- og forskningsformål. Besvarelsen, eller vedlegg til den, må ikke nyttes av HiG til andre formål, og ikke overlates til utenforstående uten etter avtale med de øvrige parter i denne avtalen. Dette gjelder også firmaer hvor ansatte ved HiG og/eller studenter har interesser.

   Besvarelser med karakter C eller bedre registreres og plasseres i skolens bibliotek. Det legges også ut en elektronisk prosjektbesvarelse uten vedlegg på bibliotekets del av skolens internett-sider. Dette avhenger av at studentene skriver under på en egen avtale hvor de gir biblioteket tillatelse til at deres hovedprosjekt blir gjort tilgjengelig i papir og nettutgave (jfr. Lov om opphavsrett). Oppdragsgiver og veileder godtar slik

offentliggjøring når de signerer denne prosjektavtalen, og må evt. gi skriftlig melding til studenter og dekan om de i løpet av prosjektet endrer syn på slik offentliggjøring.

5. Besvarelsens spesifikasjoner og resultat kan anvendes i oppdragsgivers egen virksomhet. Gjør studenten(e) i sin besvarelse, eller under arbeidet med den, en patentbar oppfinnelse, gjelder i forholdet mellom oppdragsgiver og student(er) bestemmelsene i Lov om retten til oppfinnelser av 17. april 1970, §§ 4-10.

6. Ut over den offentliggjøring som er nevnt i punkt 4 har studenten(e) ikke rett til å publisere sin besvarelse, det være seg helt eller delvis eller som del i annet arbeide, uten samtykke fra oppdragsgiver. Tilsvarende samtykke må foreligge i forholdet mellom student(er) og faglærer/veileder for det materialet som faglærer/veileder stiller til disposisjon.

7. Studenten(e) leverer oppgavebesvarelsen med vedlegg (pdf) i Fronter. I tillegg leveres et eksemplar til oppdragsgiver.

8. Denne avtalen utferdiges med et eksemplar til hver av partene. På vegne av HiG er det dekan/prodekan som godkjenner avtalen.

9. I det enkelte tilfelle kan det inngås egen avtale mellom oppdragsgiver, student(er) og HiG som nærmere regulerer forhold vedrørende bl.a. eiendomsrett, videre bruk, konfidensialitet, kostnadsdekning og økonomisk utnyttelse av resultatene.

   Dersom oppdragsgiver og student(er) ønsker en videre eller ny avtale, skjer dette uten HiG som partner.

10. Når HiG også opptrer som oppdragsgiver trer HiG inn i kontrakten både som utdanningsinstitusjon og som oppdragsgiver.

11. Eventuell uenighet vedrørende forståelse av denne avtale løses ved forhandlinger avtalepartene i mellom. Dersom det ikke oppnås enighet, er partene enige om at tvisten løses av voldgift, etter bestemmelsene i tvistemålsloven av 13.8.1915 nr. 6, kapittel 32.

12. Deltakende personer ved prosjektgjennomføringen:

HiGs veileder (navn): Harald Liodden

Oppdragsgivers kontaktperson (navn): Stian Husemoen

Student(er) (signatur): Daniel Kristoffersen dato 28.01.15

Eirik Bue dato 28.01.15

Eirik Ellegård dato 28.01.15

dato

Oppdragsgiver (signatur): [signature] dato 28.01.2015

IMT Dekan/prodekan (signatur): dato

Versjon Januar 2011hb

Vedlegg B
Gruppregler
1 Side

Gruppereregler Bachleroppgave – Heed

1 – Reglenes omfang:

- Beslutninger fattes i flertall. Gruppen består av et oddetall antall personer så det vil aldri oppstå situasjoner hvor man må votere eller at en person har økt bestemmelsesrett.
- Prosjektlederstillingen går ikke på rundgang, og vil være samme person gjennom hele semesteret.
- Prosjektlederen bestemmes ut ifra første møte.
- Dersom det påløper kostnader vil disse fordeles likt på alle gruppemedlemmene, uavnhengig om alle gruppemedlemmene er med.
- Alle skal ha rett til å signere på vegne av gruppen, men ikke hvor det må besluttes avgjørelser.

2 – Prosjektlederens fullmakter:

- Prosjektlederen kan fatte avgjørende prosjektdetaljer på vegne av gruppen om de andre gruppemedlemmene ikke er tilstede.
- Prosjektlederen har hovedansvar for delegering av arbeidsoppgaver ved uoeverenstemmelser.
- Prosjeklederen teller for to om et gruppemedlem er borte.

3 – Sanksjoner og avskjedigelse:

- Hvis et gruppemedlem ikke utfører avtalt arbeid vil man få en advarsel, hvorav man kan få maksimalt tre før det påløper konsekvenser.
- Ved 3 advarsler vil det være et drøftingsmøte med alle gruppemedlemmer inkludert gjeldende person hvor det diskuteres om personen bør avskjediges.
- Hvis et gruppemedlem gjentatte ganger unnlater å møte uten varsel på forhånd, vil det være grunn for et drøftingsmøte som nevnt ovenfor.
- Veileder vil bli varslet om et drøftingsmøte skal finne sted, og vil få anledningen til å delta.
- Ved en eventuell avskjedigelse vil dette konkluderes på møtet, som etterfølgende vil tas opp med veileder.
- Det vil føres en felles logg for alle gruppetimer bidras med i prosjektet, og på induviduelt arbeid utover dette vil føres i egen logg. Disse vil fremgå som bevis om det blir snakk om sanksjoner og avskjedigelse.

_____	_____	_____	_____
Daniel Kristoffersen	Eirik Lintho Bue	Eirik Ellegård	dato/sted

Vedlegg C

Prosjektplan

16 Sider

HEED

Digital overføring Vestoppland Politidistrikt

Høgskolen i Gjøvik

Daniel Kristoffersen, Eirik Lintho Bue & Eirik Ellegård

Vår 2015

Prosjektplanen for bacheloroppgaven for gruppen “Heed” fra dataingeniør studiumet.

Contents

1 – Mål og Rammer

1.1 – Bakgrunn

"Vestoppland Politidistrikt er partner i det nasjonale Center for Cyber –and Information Security (CCIS) og ønsker å finne lokale samarbeidsflater med Høgskolen i Gjøvk i forhold til «vanlige politiets» utfordringer. De ønsker, som nærmeste politimyndighet for HIG, å være en premisseleverandør for å gi innsyn i utfordringer og problemstillinger som politiet står ovenfor. " [1] .

Vestoppland Politidistrikts hverdag er preget av utfordringer i form av mangel på sikker og god infrastruktur mellom enheter ute opp i mot operasjonssentralen, og ønsker i denne anledning en bacheloroppgave som tar for seg denne problematikken. De ønsker at vi skal se se på tekniske løsninger og levere et proof-of-concept for å kunne danne et bilde av hvordan det vil fungere i hverdagen.

1.2 – Prosjektmål

1.2.1 Effektmål

Vi leverer et system som skal bistå Politiet ved å gi de muligheten til å overføre digital informasjon direkte til sentralen på en sikker og verifiserbar måte. Dette vil da innebære at kommunikasjonen mellom sentralen og enhetene vil være kryptert over en sikker TSL linje. Det at informasjonen skal være verifiserbar vil bety at mediet som blir sendt til sentralen kan garanteres å komme fra oppgitt enhet uten at informasjonen har blitt tuklet med iløpet av transaksjonen. Systemet vil også gjøre det enklere for Politiet å samle og sende medie sammenlignet med dagens løsning som de benytter.

1.2.2 Resultatmål

Vi leverer en prosjektrapport som har med drøftingen av arbeidsmetodikk, arbeidsplan, oversikt over informasjonsinnhenting og systemutviklingsmodell. Den vil også inneholde fagligvurdering av tekniske løsninger og begrunnelse for valgene som er blitt gjort. Det leveres også en prototype som inneholder kjernefunksjonalitet for å illustrere praktisk implementasjon.

2 – Omfang

2.1 – Avgrensning

Prosjektet avgrensens til å kun gjelde Android enheter, og ikke versjoner lavere enn versjon 5.0 (Lollipop). Det ble satt krav av oppdragsgiver at vi skulle benytte TLSv1.2 med 256-bits hash funksjon. Til dette benytter vi Android biblioteket SSLEngine som krever API level 20+, laveste Android versjon for mobiler og nettbrett som støtter dette er versjon 5.0.
Vi utvikler en prototype med kjernefunksjonalitet for å illustrere funksjonalitet og kommunikasjon i grove trekk. Hvorav kjernefunksjonalitet omhandler å være muligheten for å sende media kryptert til sentralen og at kilden kan verifiseres.

Vi forutsetter at brukeren har en ubrutt edge-dekning mens sendingen pågår og utvikler ikke muligheten for å gjenoppta sending om forbindelsen blir brutt.

For utvikling av app til mobil enhet vil vi gjenbruke systemfunksjoner som allerede finnes i operativsystemet, så langt dette lar seg gjøre. Dette har vi valgt å gjøre for å spare tid, og unngå å utvikle funksjonalitet som allerede finnes og kan gjenbrukes.

På server siden vil prosjektet avgrensens til at vi utvikler en desktop versjon av programmet som skal brukes på sentral siden. Dette innebærer at det ikke er et program som kjøres fra en egen dedikert server, men at selve programmet utgjør serveren. Derfor vil pc-en med programmet måtte kjøre for at systemet skal fungere.

Ved problemstillingen «Publikum på stedet til operasjonssentralen» velger vi å overlate problematikken rundt det om å benytte tjenesten for å sende skadevare, phising-verktøy, etc. til Vestoppland Pollitidistrikts systemer. Vi avgrenser til å anta at det publikum sender er 100 % legalt og med gode intensjoner, nettopp fordi vi mener at dette er mer opp til infrastrukturen hos Vestoppland Politidistrikt, med tanke på server sikkerhet, brannmurer og annet sikkerhetsutstyr.

2.2 – Oppgavebeskrivelse

Vestoppland Politidistrikt har utfordringer vedrørende overføring av digital informasjon direkte til sentralen når noe skjer. Dette vil særlig gå på muligheten for å utveksle bilder, lydopptak og film fra åsteder til politidistriktets operasjonssentral for videre utbedring. Dette på en slik måte at man kan verifisere hvem som har sendt det, og at det er garanti for at informasjonen ikke er tuklet med på veien.
Selve målet med oppgaven vil være å se på mulige løsninger for denne problematikken og levere et proof-of-concept produkt som viser hvordan dette kan implementeres. Oppgaven i seg selv er veldig åpen fra oppdragsgivers side, men følgende problemstillinger gis [1]:

- **Politi på stedet til Operasjonssentralen**: Denne problemstillingen vil omhandle å løse problematikken rundt nåværende manglende infrastruktur når det kommer til å sending av media mellom enheter ute og sentralen. Dette må være en sikker og

garantert kommunikasjon. Det skal tas hensyn til at Politiet operer i områder med dårlig mobildekning og lave hastigheter.

- **Publikum på stedet til operasjonssentralen:** En egen app for smarttelefoner som utnytter dagens internett samfunn preget av sosiale medier, hvor enkeltpersoner kan rapportere live fra hendelser i god tid før myndigheter og annet media er varslet. Brukere skal kunne sende bilder, filmer, lydklipp og linker fra ting som skjer. Avsender skal verifiseres og GPS koordinater inngår i sendingen om mulig. Det skal også implementeres mulighet for direkte kommunikasjon fra sentralen til vedkommende ved anrop/SMS. Utover dette skal det lagres en hashet kopi på enheten av alt som er sendt slik at det kan brukes for digital etterforskning og brukes som gyldig bevis. Hashen bør basere seg på noe fysisk opp i mot både telefon og bruker av telefon. Personvern aspektet bør også vurderes i en slik sammenheng.
- **Operasjonssentralen til politi/personer på stedet:** Man skal kunne tydelig skille mellom meldingene som kommer fra politiet og de som kommer fra publikum. Gjøre det mulig å enkelt fremheve de mest relevante meldingene, slik at man unngår mye «spam» med irrelevante tips. Gjøre det mulig å enkelt kunne kommunisere med enheten som sender tipset samt å muliggjøre for videresending av media til en gruppe med politienheter.

3 – Prosjektorganisering

3.1 – Ansvarsforhold og roller

Oppdragsgiver er Vestoppland Politidistrikt, hvor Stian Husemoen fra Høgskolen i Gjøvik vil fungere som kontaktperson. Han vil være den vi forholder oss til under prosjektet, men i tillegg vil det være noen møter med Politiet.

Daniel Kristoffersen vil fungere som Scrum Master under prosjektet. Dette kom vi frem til i enighet under et fellesmøte holdt første uka i prosjektperioden. Her ble arbeidsoppgavene og ansvaret til lederen drøftet og lederen ble valgt etter eget ønske og med de andre gruppemedlemmenes godkjennelse.

3.2 – Grupperegler og rutiner

Gruppereglene er lagt med som vedlegg, se vedlegg 1.

Det vil bli holdt møter annenhver torsdag siden det indikerer slutten og starten på en sprint, hvor både avslutningsmøte for foregående og planleggingsmøte for neste vil bli holdt som et møte. Vi kom frem til at det mest tidseffektive var å kjøre disse møtene sammenlagt. Her vil det drøftes helhetlige progresjon, eventuelle problemer underveis og resultatet av den utviklede inkrementen av programmet. Avslutningsvis vil vi føre inn de problemene som eventuelt har oppstått inn i neste sprint, og gjennomgå kravsspesifikasjonen for å avgjøre hvilke user stories som arbeides med på neste sprint.
Denne dagen benyttes også til å ha et møte med oppdragsgiver for å fortelle om progresjon, prosjektets gang og diskutere eventuelle spørsmål eller løsninger om behov.

Hver dag vil det holdes korte scrum møter for å diskutere dagens agenda, påmøtte problemer og progresjonen.

4 – Planlegging, oppfølging og rapportering

4.2 – Systemutviklingsmodell og argumentasjon

Når det kom til valg av SU-modell måtte vi ta hensyn til noen viktige forhold i forbindelse med prosjektet som hadde konsekvenser for hvilken modell vi skulle velge. Disse kravene var blant annet kravene som Høgskolen i Gjøvik har satt i forbindelse med bacheloroppgaven, men også vesentlig konkrete karakteristika ved oppgaven som måtte ligge til grunn for valg av SU-modell. Disse var som følgende:

- Produktet skal utvikles i inkrementer.
- Levere et proof-of-concept/prototype som viser kjernefunksjonalitet og som viser i praksis hvordan det kan implementers.

Vi vurderte forskjellige systemutviklingsmodeller for å avgjøre hvilken eller hvilke som passet vårt prosjekt best ut ifra krav og karakteristika. De modellene vi vurderte for vårt prosjekt var Xtreme Programming, Rup (Rational Unified Process), Scrum og incremental development.

Xtreme programming har gode egenskaper som benyttelsen av story cards hvor disse benyttes senere i prosessen for å dele opp de forskjellige oppgavene og planlegge utviklingen. En annen god egenskap er korte tidsluker mellom releasene som passer vårt prosjekt med tanke på tidsfrister. Derimot egenskaper som skriving av tester før selve koden skrives, bruk av automatiske testingmiljøer og testing av hver oppgave før man går videre passer ikke godt. Årsaken til dette er den korte tidsperioden og at utviklingen vil gå saktere om det skal utvikles tester før hver oppgave utvikles. En annen negativ egenskap er mangel på dokumentasjon, noe som vil være et problem for oss ved en eventuell videreutvikling.

RUP benytter seg av tre forskjellige perspektiver og fire ulike faser for utvikling, hvor disse ikke er teknisk vinklet. De ulike perspektivene kunne vi implementert til en viss grad i vårt prosjekt for å få forskjellige overblikk for prosjektets gang, men derimot de ulike fasene passer ikke like godt. Dette skyldes at det da vil involvere veldig mye forarbeid som allerede er gjort i forbindelse med at dette er en bacheloroppgave. Det vil gå på ting som forretningscasen for systemet, de ulike entitetene, definering av interaksjonene mellom entitetene og beregne bidraget som systemet vil gjøre. Mye av dette er allerede gjort i oppgavebeskrivelsen og ville vært unødig arbeid. Eloboration og Construction fasene til RUP er faser vi kunne benyttet i vårt prosjekt, men vi vurderer det til å ikke være nok for å velge RUP som SU-modell under prosjektet. Siste fase som omhandler overlevering av systemet fra utviklingsmiljøet til brukermiljøet vil ikke være relevant for vårt tilfelle siden vi kun lager et proof-of-concept.

Basert på forholdene vi måtte ta hensyn til og drøfting internt i gruppa basert på de forskjellige su-modellene kom vi frem til å bruke en kombinasjon av SCRUM og inkrementell.

Dette ble drøftet sammen med veileder og vi kom frem til at dette var en god løsning ved at vi bruker SCRUM som verktøy til konfigurasjonsstyring og det nesten alt fra inkrementell til utvikling og arbeidsmetodikk. Det vi ikke benytter fra inkrementell utvikling er det med hver del som blir ferdig produsert blir satt rett ut i bruk hos kunden. Grunnen til at vi benytter SCRUM til konfigurasjonsstyring er fordi den er meget godt tilpasset vårt prosjekt og våre arbeidsmetoder. Dette ved at oppgaven allerede er skrevet slik at oppgavene lett kan defineres til forskjellige «User stories», som vi da benytter for å bygge backlog-en, som definerer arbeidslisten. Dette vil gjøre det enklere å estimere prosjeket tidsmessig ved å justere størrelsene på user story-ene om vi ser at en arbeidsoppgave blir for stor eller at andre oppgaver bør prioriteres. Vi vil ha sprinter på to uker, dette grunnet at prosjektet foregår over en relativt kort tidsperiode. På denne måten vil vi kunne hyppig estimere tidsbruk, men samtidig få god tid til utvikling innenfor hver sprint. Det vil også forekomme daglige korte møter for at vi skal kunne snakke kort om dagens agenda, eventuelle problemer med utviklingen og prosjektets fremgang. Dette vil bli oppsummert i et møte på slutten av hver sprint hvor resultatet av sprinten diskuteres og eventuelle justeringen gjøres.

Inkrementell utvikling benytter vi siden vi tidlig i prosessen vil levere et utkast til kunden og i den forbindelsen ha en brukertest for å få feedback fra kunden. Dette er nyttig for å se om vi har implementert brukervennlige og nyttige funksjoner, samt at eventuelle endringer/tillegg til kravsspesifikasjonen kan meldes fra. Dette vil gi kunden en bedre «følelse» av systemet vi skal levere og feedbacken vil være mer nøyaktig kontra bare å vurdere prosjektplaner. Dette første utkastet inneholder da kjernefunksjonalitet, utbroderes under «2.2 - oppgavebeskrivelse». En annen begrunnelsen for valg av inkrementell er fordi endringer i kravsspesifikasjonen håndteres lett ved å føre de inn som en del av neste imkrement. Dokumentasjon produseres på slutten av hver sprint, hvor da arbeidsoppgaver som er blitt brutt opp blir plassert på et scrum board for å være med på neste inkrement, og implementerte funksjoner blir dokumentert.

5 – Organisering av kvalitetssikring

5.1 – Dokumentasjon, standardbruk og kildekode

Kildekoden skrives på engelsk, her inngår variabler, kommentarer og andre elementer av koden. Dette gjøres for å holde koden konsistent og gjøre det lettere for videreutvikling. Public funksjoner dokumenteres i form av javadoc, mens større komplekse private funksjoner kommenters. Dette kom vi frem til etter drøfting ved at koden vil holdes atomisk ved en eventuell utvidelse og gjenbruk av funksjoner.
Standarden vi benytter for dokumentering er «Javadoc», og er implementert som en del av programmeringsspråket. Dette bidrar til å generere API dokumentasjon som betaler seg i form av høyere lesbarhet, effektivisering med tanke på videreutvikling samt enkelt å kunne legges ved som vedlegg til prosjektrapporten.

Definering av klasser starter alltid med stor forbokstav, og orddelinger defineres med sammenhengende tekst og stor bokstav for hvert delte ord. Eksempelvis: «On Create» vil bli: «OnCreate». Samme gjelder for variabler og funksjoner, men med liten forbokstav.

Det vil ukentlig på slutten av hver sprint bli skrevet dokumentasjon, som nevnt tidligere. Det føres logg daglig, både i felleskap og individuelt, hvor fellesloggen definerer arbeidet til hver enkelt i arbeidstid, og de individuelle definerer arbeid på egenhånd.

5.2 - Konfigurasjonsstyring

Trello fungerer som scrumboard under prosjektet, og er en nettbasert løsning hvor alle gruppemedlemmene har oversikt over user stories, backlog, pågående arbeid og sluttført arbeid. Dette kom vi frem til at fungerer som en bedre løsning siden det alltid er tilgjengelig for alle til enhver tid, kontra f.eks bruk av papirskrevet scrumboard, og vil fungere som en god erstatter for manglende oppslagstavle.

Til utviklingen benyttes «Bitbucket» som versjonshåndteringssystem, hvorav «Source Tree» er knyttet opp i mot Bitbucket for å få en grafisk presentasjon av kodeendringer, versjoner og hverandres arbeid. Bitbucket vil da fungere som selve «repository» serveren, mens Source Tree foretar selve versjonshåndteringen.
I tillegg brukes SCRUM som verktøy for konfigurasjonsstyring av utviklingsprosessen.

Dropbox brukes for organisering av dokumenter og filer i forbindelse med prosjektet som tillater at alle har anledningen til å lese/skrive til disse.

5.2 – Risikoanalyse

Teknologimessige risikoer:

- Endringer i API-nivåer fra Android (At funksjoner blir utdaterte)
- Bugs i nytt OS (Lollipop)

Forretningsmessige risikoer:

- Produktet tilfredsstiller ikke kravene
- Endringer i kravsspesifikasjonen

Prosjektmessige risikoer:

- Sykdom
- Avskjedigelse
- Uforutsette problemer som forårsaker feil tidsestimering.

Risikoanalyse				
Nr.	Beskrivelse	Sannsynlighet	Konsekvens	Tiltak
1	Endring i API-nivå i forbindelse med ny Android versjon	Lite sannsynlig	Middels	Tiltakene varierer ut ifra endringstype, men generelt sett vil det omhandle å omskrive kode, oppgradere OS på enheter.
2	Bugs i nytt os	Svært sannsynlig	Middels	Avhengig av hva slags type bug, vil løsninger være å vente på utbedring fra Google eller gjøre endringer i implementasjon for å arbeide oss rundt feilen.
3	Produktet tilfredsstiller ikke kravene	Lite sannsynlig	Katastrofalt	Gjøre endringer som tilfredsstiller kravene. Forflytte arbeidsressursene fra evt. Videreutvikling og forstørrelse av systemet til å håndtere endringene som må gjøres
4	Endringer i kravspesifikasjonen enten fra brukertesten eller utenom	Sannsynlig	Middels	Legges inn for å inngå i neste inkrement av produktet. Flytte fokus fra opprinnelig plan, til nye ønsker.
5	Sykdom hos en eller flere av gruppemedlemmene over en sammenhengende periode som overskrider 1 uke	Sannsynlig	Høy	Omdelegering av arbeid, arbeide hjemmefra er en løsning ved bruk av Skype for møter og annen interaktivitet Hvis det overstående ikke kan gjøres må backlog-en justeres og resterende gruppemedlemmer må ta det mest kritiske arbeidet
6	Avskjedighet av et gruppemedlem	Liten sannsynlighet	Katastrofalt	Noe av det samme som ovenfor, gjøre mer avgrensninger
7	Uforutsette problemer som forårsaker feil tidsestimering.	Sannsynlig	Lav	Vurdere basert på release backlog-en og justere backlog-en for å delegere arbeidet til det mest kritiske funksjonen. Følge med på progresjonen

Fig 1. Tabell over de ulike risikoene nummerert for bruk i matrisen nedenfor. Sannsynlighet, konsekvens og tiltak er presentert for å danne risikonivået og løsningene om noen av de skulle forekomme.

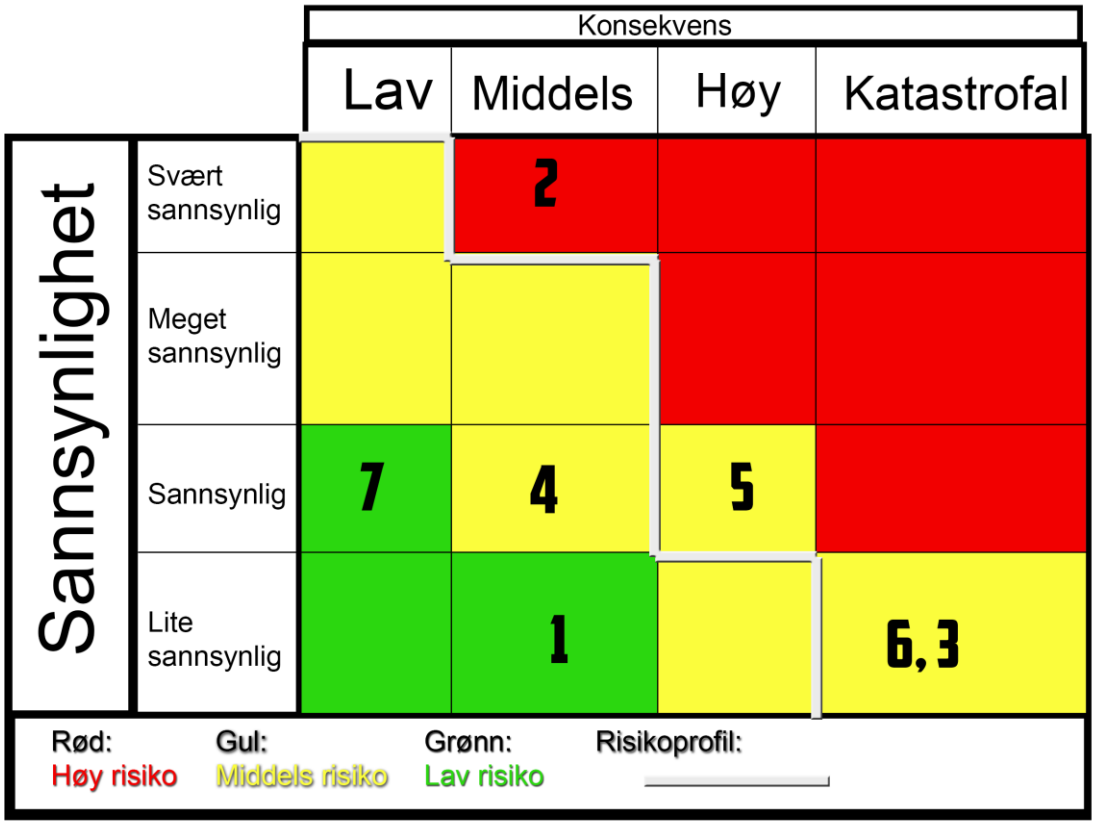


Fig. 2 Illustrerer en grafisk fremstilling av risikoene satt inn i en matrise med representative farger tilhørende graden av risikoen. Alt under risikoprofilstreken markerer hva som vil aksepteres.

6 – Plan for gjennomføring

Se vedlegg 2

6.1 - Kommentar til Gantt –skjemaet

Alle sprinter inngår innunder «Utvikling», hvor hver sprint inneholder:

- System og software design – Som omhandler å designe løsningene vi har kommet frem til opp i mot det faktiske systemet, og skape en overliggende system arkitektur. Involverer også å identifisere og beskrive de fundamentale kravene for kommende sprint og hvordan å utvikle disse.
- Implementasjon og testing – Her vil de utvalgte user stories-ene for den sprint perioden bli utviklet til en fungerende utvidelse av programmet og testet for å verifisere at implementasjonen er vellykket.
- Integrasjon og system testing – Delen fra punktet ovenfor vil bli integrert inn i det helhetlige programmet og testet som en del av programmet, og ikke som en selvstendig del. Ved fullførelse av dette steget vil vi ha en ferdig inkrement klar for levering til kunde.

- Scrum Møte – Vil avslutte nåværende sprint, og indikere starten på neste sprint, i samme møte som foregår annenhver torsdag, noe vi valgte å gjøre siden dette så vi som mest effektiv utnyttelse av tiden. Ved avslutning av hver sprint vil det diskuteres hvordan progresjonen innenfor den sprinten har vært, om det er noe arbeid som ikke ble fullført og dokumentering av den avsluttede sprinten. Planleggingen av neste sprint vil foretas ved at man ser på den helhetlige progresjonen og velger ut user stories som er mest nødvendige. Deretter legger vi de ikke sluttførte oppgavene fra foregående sprint inn i backlog-en, og tar høyde for endringer i kravsspesifikasjonen, om det har kommet noen.

Iløpet av sprint 1 vil det blir holdt en brukertest hos Politiet for å få direkte feedback på første leveranse av programmet. Dette genererer nyttig og relevante tilbakemeldinger og ønsker som brukes til videreutvikling og for å gjøre endringer i systemet. Denne brukertesten vil baseres kun på planlagt desig og virkemåte siden funksjonaliteter ikke er utviklet enda på dette stadiet. Dette gjøres fordi appen og beslektede systemer vil bli utviklet som følge av blant annet resultatene fra denne brukertesten.

Det vil foreligge tre statusrapporter til veileder iløpet av utviklingsfasen, og disse er plassert med tanke på at det vil ha foregått hendelser som er verdt å rapportere. Første statusrapport vil inneholde resultatet av brukertesten, samt retningen av utviklingen som denne feedbacken har gitt. Den andre er plassert nærmere slutten, som en del av nest siste sprinten. Dette fordi vi kom frem til at det da har vært to sprinter med utvikling og vil da være nevneverdig materiale fra utviklingen, generelt om den totalte progresjonen og eventuelle problemer som er verdt å fremme i en statusrapporten. Den siste statusrapporten er plassert helt på slutten, før fokuset rettes helt mot rapport skriving. Denne vil da innholde en oppsummering av hele utviklingsfasen, hvor langt vi kom i utviklingen, samt perioden som er satt av til å foreta en utvidet test i felt og avsatt tid for feilretting av eventuelle problemer vi da måtte oppdage. Vi kom frem til at det falt mest naturlig med statusrapport på dette tidspunktet, siden det her vil være mye relevant og oppgave kritiske hendelser som er verdt å rapportere. Dette for å få rapportert alt som måtte ha oppstått før vi for fullt skriver prosjektrapporten.

«Testing og feilretting» inneholder en stor test av programmet ute i felt for å simulere hvordan programmet vil fungere i virkeligheten i reelle situasjoner. Det er også lagt inn tid til feilrettinger av eventuelle problemer som vi oppdager under felt testen.

Litteraturliste

1 Stian Husemoen, Digital Overføring Politiet, 2014

Vedlegg 1

1 – Reglenes omfang:

- Beslutninger fattes i flertall. Gruppen består av et oddetall antall personer så det vil aldri oppstå situasjoner hvor man må votere eller at en person har økt bestemmelsesrett.
- Prosjektlederstillingen går ikke på rundgang, og vil være samme person gjennom hele semesteret.
- Prosjektlederen bestemmes ut ifra første møte.
- Dersom det påløper kostnader vil disse fordeles likt på alle gruppemedlemmene, uavnhengig om alle gruppemedlemmene er med.
- Alle skal ha rett til å signere på vegne av gruppen, men ikke hvor det må besluttes avgjørelser.

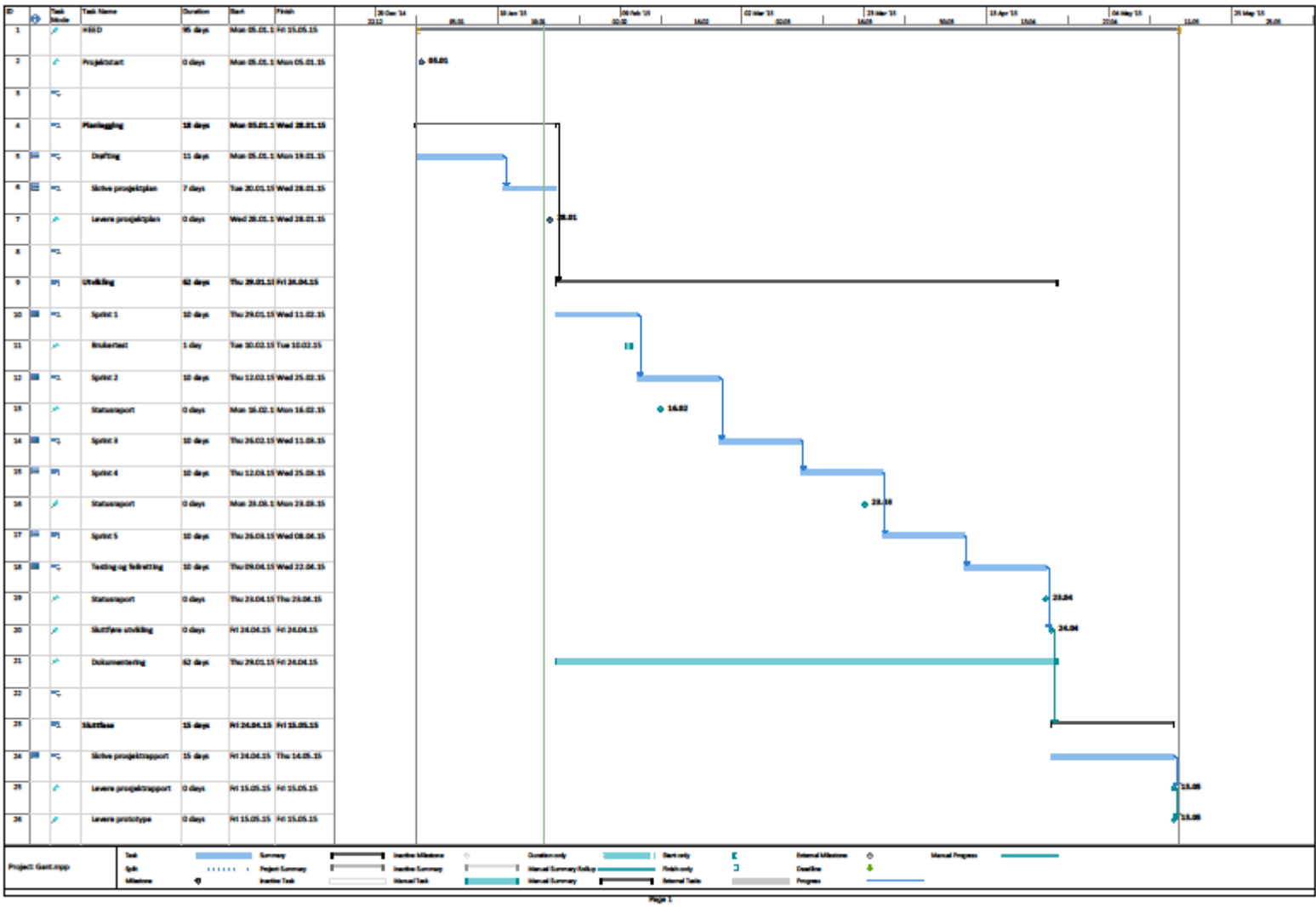
2 – Prosjektlederens fullmakter:

- Prosjektlederen kan fatte avgjørende prosjektdetaljer på vegne av gruppen om de andre gruppemedlemmene ikke er tilstede.
- Prosjektlederen har hovedansvar for delegering av arbeidsoppgaver ved uoeverenstemmelser.
- Prosjeklederen teller for to om et gruppemedlem er borte.

3 – Sanksjoner og avskjedigelse:

- Hvis et gruppemedlem ikke utfører avtalt arbeid vil man få en advarsel, hvorav man kan få maksimalt tre før det påløper konsekvenser.
- Ved 3 advarsler vil det være et drøftingsmøte med alle gruppemedlemmer inkludert gjeldende person hvor det diskuteres om personen bør avskjediges.
- Hvis et gruppemedlem gjentatte ganger unnlater å møte uten varsel på forhånd, vil det være grunn for et drøftingsmøte som nevnt ovenfor.
- Veileder vil bli varslet om et drøftingsmøte skal finne sted, og vil få anledningen til å delta.
- Ved en eventuell avskjedigelse vil dette konkluderes på møtet, som etterfølgende vil tas opp med veileder.
- Det vil føres en felles logg for alle gruppetimer bidras med i prosjektet, og på induviduelt arbeid utover dette vil føres i egen logg. Disse vil fremgå som bevis om det blir snakk om sanksjoner og avskjedigelse.

Vedlegg 2



Vedlegg 2, Gantt-skjema.

Vedlegg D

Statusrapporter

5 Sider

Statusrapport

1 – Oppsmmering av hittil arbeid
2 – Fremgangen
3 - Oganisering av arbeidsoppgaver
4 – Kontakt opp mot veileder og oppdragsgiver(e)

1 – Oppsummering av hittil arbeid

Per dags dato, 18.02.2015, har vi fått gjennomført den første sprinten og er halveis inn i sprint 2. Arbeidsoppgavene vi hadde planalgt å gjennomføre ble fullført og i korthet omhandlet de: TLS kommunikasjon, APP GUI, SQL syntax og opprettelse av hjemmesiden for prosjektet. Forekom ingen problemer eller arbeidsoppgaver som måtte utvides til neste sprint.

Hittil i denne sprinten foretar vi research for å verifisere at valgt teknologi samsvarer med tekniske krav og at det vil være holdbar i årene fremover. Sluttfører utviklingen av overførings muligheten fra enheter til server og definerer server GUI.

2 – Fremgangen

Den totale progresjonen er innenfor tidsskjemaet, og vi har klare mål vi skal rekke i løpet av prosjektperioden. Hele gruppen jobber godt, er motiverte og fungerer godt sammen som et team. Mest selvstendig arbeid for tiden grunnet forskjellige arbeidsoppgaver som ikke krever felles arbeid. Daglige scrum møter foretas derfor over Skype.

3 – Organisering av arbeidsoppgaver

Som nevnt i forrige avsnitt arbeider vi separat innunder denne sprinten , hvorav Daniel foretar research og annet administrativt arbeid Eirk Bue foretar sluttføring av sending av digitalt medie mens Eirik Ellegård utvikler server GUI. Vil fortsette å arbeide separat på denne måten så langt det lar seg gjøre og kommunikasjonen enkelt kan foretas over Skype. Design av GUI for server ble bestemt i et felles møte vi holdt tirsdag 17.02.2015.

4 – Kontakt opp mot veileder og oppdragsgiver(e)

Kontakten med veileder fungerer godt og praksisen med møte annenhver uke viste seg å være en effektiv ordning. Det som derimot ikke fungerer like godt er kontakten med oppdragsgiver, og da herunder Vestoppland Politidistrikt. Vi møter utfordringer ved å utvikle et produkt uten innblanding av kunden. Stian Husemoen responderer ikke per e-post, kun ved personlig oppmøte, og har hittil ikke fått organisert kontakten vår opp mot Vestoppland

Politidistrikt. Han skal fungere som vårt mellomledd, og vi avhenger derfor av at han får til dette. Siden kontakten med Vestoppland Politidistrikt ikke har oppstått enda har vi ikke fått arrangert brukertesten vi planla og ytret ønske om å foreta.

18.02.2015, Daniel Kristoffersen

Statusrapport 2

1 – Oppsummering av hittil arbeid
2 – Fremgangen
3 – Organisering av arbeidsboppgaver
4 – Kontakt opp mot veileder og oppdragsgiver(e)

1 – Oppsummering av hittil arbeid
Nå er vi kommet godt inn i sprint 4 og nærmer oss en avslutning på utviklingsfasen. Dette kommer ikke sjokkerende på oss siden oppsatte arbeidsoppgaver har frem til nå blitt utført etter tidsskjemaet og backlog-en har kun igjen et par-tre elementer som skal fylle resten av den gjenværende tiden.

Mye arbeid er blitt gjort parallellt, både utvikling, møter, vurderinger og research av de forsjellige gruppemedlemmene.
App-en har blitt fylt opp med kjernefunksjonalitet, optimalisert og testet etter hver implementasjon. Research på TLS er blitt gjennomført og det har blitt skrevet et dokument på omlag 5000 ord som et resultat av dette.

Researchen viser til og har hjulpet oss med å fatte korrekte beslutninger og vi hadde i dag, 23.03.2015, et møte med Christoffer Vargtass som sammen med oss gikk gjennom den kryptografiske sikkerheten for å verifisere at vi fulgte standarderer, krav og ga oss noen tips. Dette møtet avdekket noen programmerings endringer som må til og litt reformuleringer i TLS researchen , blant annet med tips om å se i dokumentet for den Europeiske sikkerhetsstandarden.

2 – Fremgangen
Vi har overholdt tidsskjemaet sålangt og eneste vi ikke har fått begynt med er selve rapporten, samt noe dårlig oppdateringer av prosjeketswebside. Derimot har vi fått drøftet og gjort oss noen tanker om innholdet til none av punktene i rapporten så vi står ikke helt på bar bakke.

Fredag 27. mars skal vi treffe Vestoppland Politidistrikt (fra nå; Politiet), for en gjennomgang av utviklingen samt en brukertest for å avdekke eventuelle endringer som må til samt en kartlegging av Politiets arbeidsmetoder.

3 – Organisering av arbeid
Eirik Lintho Bue og Eirik Ellegård foretar hovedparten av tiden til utvikling mens Daniel Kristoffersen har gjort TLS research, ordnet med møteaktiviteter og fått i orden møtet med Politiet.

4 – Kontakt opp mot veileder og oppdragsgiver(e)
Klaget på dårlig kontakt med Stian Husemoen i forrige statusrapport og denne situasjonen har nå , med rask virkning av forrige statusrapport, endret seg veldig til det bedre. Opplever ikke nå de problemene vi nevnte forrige gang.

Kontakten opp i mot Politiet har, etter oppnådd kontaktinformasjon, vært vanskelig, men har gått i orden.

23.03.2015, Daniel Kristoffersen

Statusrapport 3

1 – Oppsummering av hittil arbeid
2 – Fremgangen
3 – Organisering av arbeidsboppgaver
4 – Kontakt opp mot veileder og oppdragsgiver(e)

1 – Oppsummering av hittil arbeid
Har avsluttet utviklingen, og er fornøyd med resultatet av det. Har vist seg at serverer ikke ble så mye utviklet som vi hadde ønsket, men vi så oss nødt til å flytte ressursene over til rapport skriving i stedet og siden serveren fungerer tilfredsstillende nok anså vi det som greit.

Har gjort ferdig alt analysearbeid og har fått dette med inn i rapporten, samt definert kravspesifikasjonen ferdig som en mer helhetlig tekst i stedet for punkter.

Anser oss som ferdige med innledningen på prosjektrapporten og antar å bli ferdig med hovedkapitlene denne uken og tar fatt på avslutningen i løpet av neste uke. Har hele tiden underveis i rapport skrivingen foretatt språkvask og fått flere personer til å lese gjennom det som hittil har blitt skrevet.

Utover dette har vi på kode siden foretatt statisk analyse av koden med to forskjellige verktøy; Sonar og FindBug og vil vedlegge omfattende rapporter som forteller om kvaliteten til koden samt en oversikt over kodelinjer, antall klasser, funksjoner, mm.
2 – Fremgangen
Fremgangen er god i den grad at utviklingen ble avsluttet som planlagt, men retting av kode tok noe lenger tid enn først planlagt og har derfor blitt videreført som oppgave videre ved siden av skriving av rapport.

Rapporten begynner å ta form, og er omtrentlig ¾ på vei med den. Mangler stort sett bare å legge inn de forskjellige avsnittene fra kladd til endelig tekst, og lage innholdsfortegnelse for tabell/figurer.
3 – Organisering av arbeid
Hovedsakelig fokusert på rapportskriving, med unntak av at Eirik Lintho Bue har fokusert noe av sin tid på å ordne struktur i koden for å gjøre den klar for levering.
4 – Kontakt opp mot veileder og oppdragsgiver(e)
Ikke noe å utsette på kontakten i denne omgang.

03.05.2015, Daniel Kristoffersen

Vedlegg E

Skjema for brukertest

16 Sider

Politibetjent:

Scenario 1 – Ta bilde, sende til OPS

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Foto knappen (Lett forståelig)

2 . Ark – Til OPS (Tomt bilde):

Hvor trykker du:

Foto knappen (Lett forståelig)

3 . Ark – Kamerafunksjon

Hvor trykker du:

Trykker på knapper, tar bilde

4 . Ark - Til OPS (med bilde)

Hvor trykker du:

Pila! Vanskeligste, men forståelig. Gir mening at det er den, men kanskje et annet ikon?

Scenario 2 – Forhåndslagret bilde

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

ingen av de tre hovedknappen, men etterhvert foto knappen. Kan skje endringen må til?

2 . Ark – Til OPS (Tomt bilde):

Hvor trykker du:

Forhandler (Lett forståelig)

3 . Ark – Bildegalleri

Hvor trykker du:

markeres bildet, fullfør (Lett forståelig)

4 . Ark - Til OPS (med listevisning)

Hvor trykker du:

Pila!

Lett forståelig, spesielt nå som han visste at det pila mente etter forrige scenario.

Scenario 3 – Ta opp lyd

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

mikrofonen (Lett forståelig)

2 . Ark – Lydopptak (nytt opptak)

Hvor trykker du:

Sa pila først, lurte på hva den røde var. Skriv rec under

3 . Ark – Tar opp lyd

Hvor trykker du:

Stopp knappen (Lett forståelig)

4 . Ark - Tatt opp lyd, klar til sending

Hvor trykker du:

Pilen!

Lange opptakah, må kunne spole. Tidslinje for å spole. Enkel og nøyaktig 3 spoling.

Scenario 4 – Dekningsstaus

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen. Svar så nøye som mulig.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Nederste trodde man vart volum, men siden vi spurte om dekning endre ikon?

2 . Ark – Hva forteller dette skjermbilde deg?

Har middels dekning ikke oppkoblet

(Lett forståelig)

2 . Ark – Er det mulig å sende noe nå?

Nei, ikke nå

2 . Ark – Begrunn overnevnte svar:

Åstedsgransker

Scenario 1 – Ta bilde, sende til OPS

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Total kommentar:

Hvor trykker du:

Foto knapper (lett forståelig)

2 . Ark – Til OPS (Tomt bilde):

Hvor trykker du:

foto knapper

3 . Ark – Kamerafunksjon

Hvor trykker du:

Kamera knapper

4 . Ark - Til OPS (med bilde)

Hvor trykker du:

Pila!

Scenario 2 – Forhåndslagret bilde

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Vil trykke på meny/hjem på telefonen

Endringer i ikon!

2 . Ark – Til OPS (Tomt bilde):

Hvor trykker du:

firkanter

3 . Ark – Bildegalleri

Hvor trykker du:

velger bilde, deretter fullfør

4 . Ark - Til OPS (med listevisning)

Hvor trykker du:

Pila!

Scenario 3 – Ta opp lyd

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Mikrofon

2 . Ark – Lydopptak (nytt opptak)

Hvor trykker du:

Rød, stopper

3 . Ark – Tar opp lyd

Hvor trykker du:

Stopper

4 . Ark - Tatt opp lyd, klar til sending

Hvor trykker du:

Pil

Scenario 4 – Dekningsstaus

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen. Svar så nøye som mulig.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Siste ikonet

2 . Ark – Hva forteller dette skjermbilde deg?

ikke sendeklart,
fordi ikke oppkoblet

2 . Ark – Er det mulig å sende noe nå?

Nei

2 . Ark – Begrunn overnevnte svar:

IKT-konsulent

Scenario 1 – Ta bilde, sende til OPS

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Knappen med kamera, enkelt

2 . Ark – Til OPS (Tomt bilde):

Hvor trykker du:

Knappen med kamera, enkelt

3 . Ark – Kamerafunksjon

Hvor trykker du:

Trykker på knipse knappen

4 . Ark - Til OPS (med bilde)

Hvor trykker du:

Pil for sending

Scenario 2 – Forhåndslagret bilde

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Kamera

2 . Ark – Til OPS (Tomt bilde):

Hvor trykker du:

Forhaster

3 . Ark – Bildegalleri

Hvor trykker du:

Markerer, fullfør

4 . Ark - Til OPS (med listevisning)

Hvor trykker du:

Pil!

Scenario 3 – Ta opp lyd

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Mikrofon

2 . Ark – Lydopptak (nytt opptak)

Hvor trykker du:

Røde ihuret, men fint om det stod rec

3 . Ark – Tar opp lyd

Hvor trykker du:

Stopper med stopp

4 . Ark - Tatt opp lyd, klar til sending

Hvor trykker du:

pil.

Scenario 4 – Dekningsstaus

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen. Svar så nøye som mulig.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Nederste, dekning logo

2 . Ark – Hva forteller dette skjermbilde deg?

Delvis dekning,
ikke koblet til server

2 . Ark – Er det mulig å sende noe nå?

Nei

2 . Ark – Begrunn overnevnte svar:

Pål-Erik seksjonsleder

Scenario 1 – Ta bilde, sende til OPS

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Kamera knappen (Enkelt)

2 . Ark – Til OPS (Tomt bilde):

Hvor trykker du:

Kamera knappen (Enkelt)

3 . Ark – Kamerafunksjon

Hvor trykker du:

Trykker på knappen for å ta bilde

4 . Ark - Til OPS (med bilde)

Hvor trykker du:

Pila for å sende

Scenario 2 – Forhåndslagret bilde

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Litt usikker, men gjetter kamera ikonet

2 . Ark – Til OPS (Tomt bilde):

Hvor trykker du:

Ikonet med firkanter

3 . Ark – Bildegalleri

Hvor trykker du:

markerer ønsket bilde avslutter med fullfør

4 . Ark - Til OPS (med listevisning)

Hvor trykker du:

Pil for sending,

samme som istad

Scenario 3 – Ta opp lyd

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Mic. ikon

2 . Ark – Lydopptak (nytt opptak)

Hvor trykker du:

Rødt ikon

3 . Ark – Tar opp lyd

Hvor trykker du:

Stopper

4 . Ark - Tatt opp lyd, klar til sending

Hvor trykker du:

Pil

Scenario 4 – Dekningsstaus

Noter hvor du ville trykket, eventuelt hvilke problemer som du opplever. Kom gjerne med andre bemerkelser om det er noen. Svar så nøye som mulig.

1. **Ark – Hovedmenyen:**

Hvor trykker du:

Skjønte det var nederste siden denne ikke er brukt

2 . Ark – Hva forteller dette skjermbilde deg?

middels dekning, ikke oppholdet

2 . Ark – Er det mulig å sende noe nå?

Nei

2 . Ark – Begrunn overnevnte svar:

Vedlegg F

Brukermanual

8 Sider

VPSECURE

Brukermanual v1.0

Innhold

1.0 Oversikt

1.1 Ofte brukte ikoner

Ikonet leder til kamerafunksjonalitet.

Ikonet brukes for send knapper.

Ikonet leder til opptak av lyd.

Ikonet leder til dekningsstatus.

Ikonet vises i statuslinjen hvis du mottar noe fra serveren.

Ikonet vises i statuslinjen hvis du sender noe til serveren.

Ikonet vises i statuslinjen hvis du har en ulest melding fra serveren.

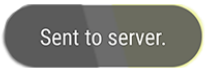
Ikonet leder til galleri.

2.0 App gjennomgang

2.1 Hovedmeny

Dette er appens hovedmeny, herfra har brukeren tre valg.

- Knapp 1 tar brukeren til kamerafunksjonalitet.
- Knapp 2 tar brukeren til opptak av lyd.
- Knapp 3 tar brukeren til dekningsstatus.



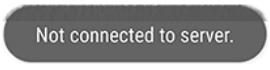
Denne meldingen vil vises i hovedmenyen etter brukeren har sendt media til serveren.

2.2 Til OPS (Kamerafunksjonalitet)

Fra denne skjermen i appen kan brukeren ta bilde og video, samt se bilder og video han har tatt før eller se de han har mottatt fra serveren.

Knappene er forklart fra venstre til høyre.

- Knapp 1 tar brukeren til galleriet hvor han kan se bilder/video tatt før, og bilder/video mottatt fra server.
- Knapp 2 tar brukeren til videoopptak.
- Knapp 3 tar brukeren til billedtagning.
- Knapp 4 vil sende, for å bruke denne må brukeren ha tatt bilde eller video, eller valgt elementer fra galleriet.



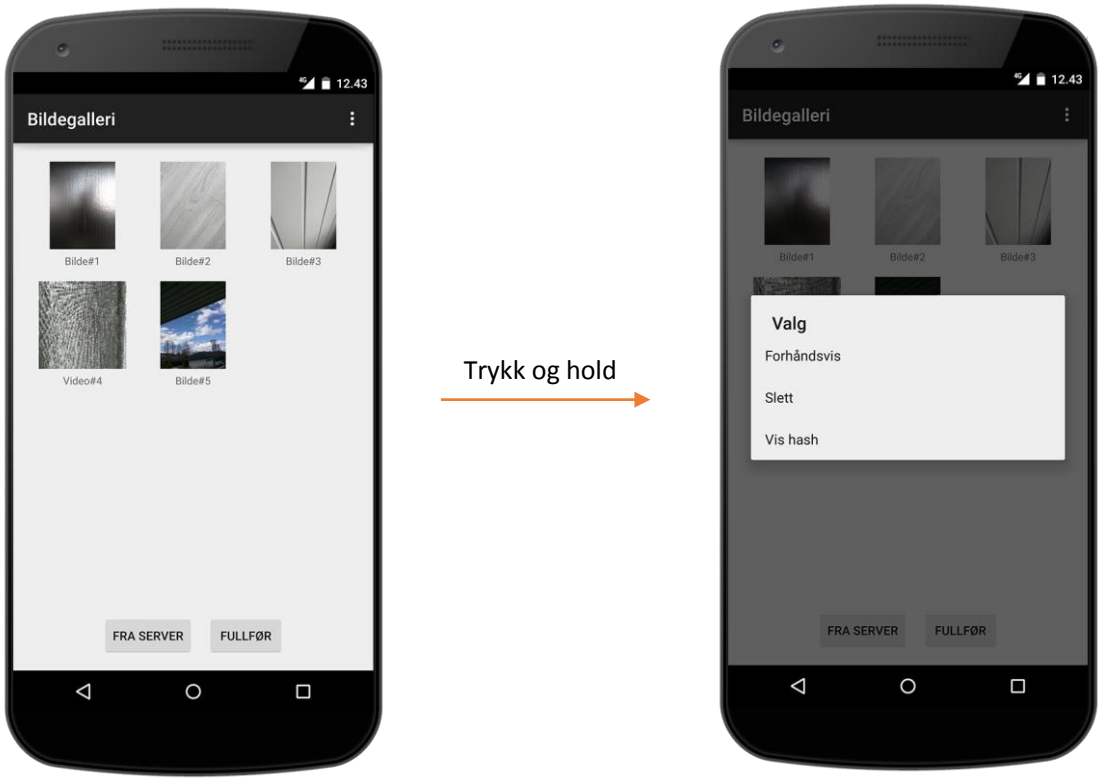
Denne meldingen vil vises hvis brukeren trykker send og ikke er tilkoblet til serveren.

2.3 Galleri

Her kan brukeren se bilder/videoer han har tatt før. Her er det to knapper, men det er også mulig å trykke og holde på ett element i galleriet for å få opp en undermeny. Det er også mulig å velge flere elementer ved å trykke på de.

Under kan man se undermenyen brukeren får opp ved å trykke og holde på ett av elementene i galleriet. Fra undermenyen har man følgende valg:

- Forhåndsvise bildet eller videoen.
- Slette elementet.
- Vise filens hash (Vil ikke bli brukt av vanlig bruker).

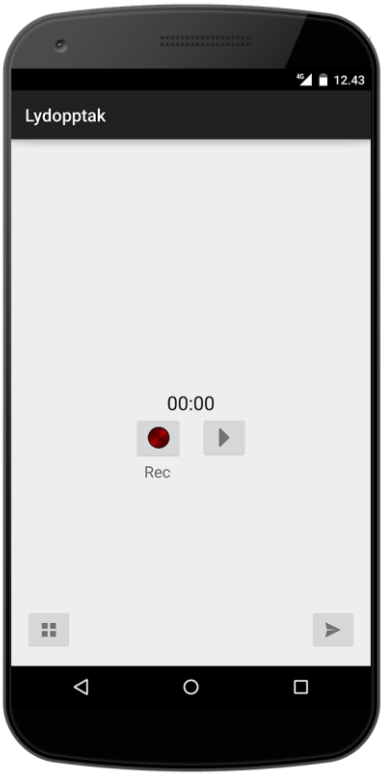


På dette skjermbildet har de to første elementene blitt valgt. Brukeren kan enten trykke på fullfør, eller trykke på knappen som vises som tre prikker på linjen med tittel. Trykker brukeren på denne knappen vil han få opp en undermeny med valg om å slette alle valgte elementer.

«Fullfør» vil sende brukeren til «Til OPS (Kamerafunksjonalitet)» med listen over de elementene som ble valgt. «Fra server» vil bytte innholdet til media mottatt fra serveren.

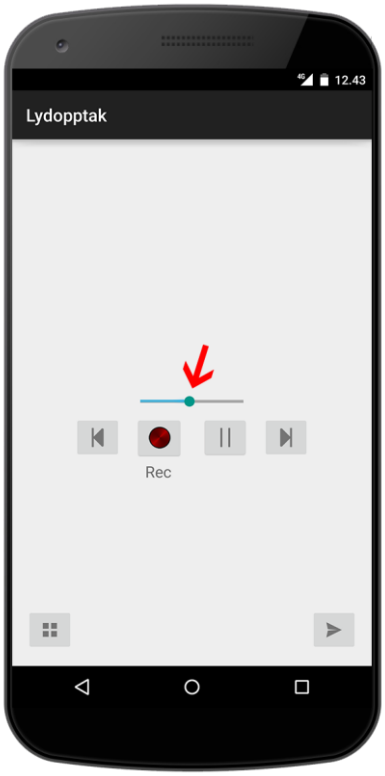
2.4 Lydopptak

Hovedmenyens andre valg tar brukeren til lydopptak.



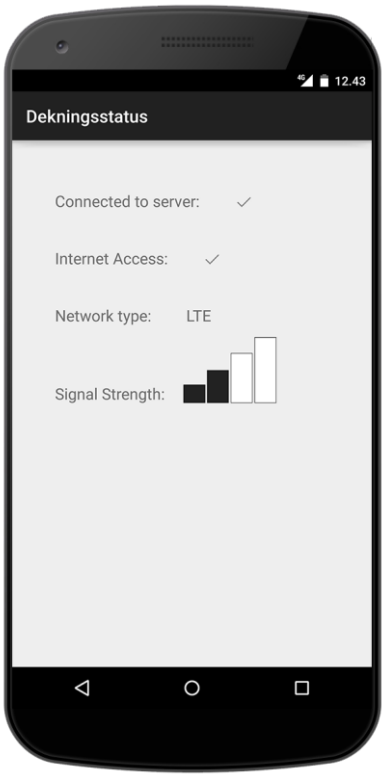
Fra venstre til høyre har knappen denne funksjonaliteten:

- Knapp 1 tar deg til lyd galleri, dette galleriet fungere akkurat slik som bildegalleriet.
- Knapp 2 starter lyd opptak.
- Knapp 3 brukes til avspilling av lyd etter endt opptak.
- Knapp 4 sender lyd, enten lyd valgt fra galleriet eller lydopptaket som nettopp ble tatt.



Knapp 2 fra skjermbildet over starter avspilling av lydopptak. Da får brukeren tilgang til to nye knapper, disse spoler sakte gjennom lydopptaket. For å spole fortere kan man holde og dra sirkelen markert med rød pil (pilen vises ikke i appen).

2.5 Dekningsstatus



Hovedmenyens tredje og siste valg tar brukeren til dekningsstatus. Her får brukeren følgende informasjon:

- Status for tilkobling til server.
- Status for tilkobling til internett.
- Deknings type.
- Signal styrke.

Vises hvis brukeren ikke er koblet til server eller internett.

Vedlegg G

SonarQube rapport for serverapplikasjon og app

2 Sider

VP Secure

Version 1.0 / 2 May 2015 13:33

Overview Components Issues Settings More

Time Machine

Debt
1h 11min

Blocker	0
Critical	0
Major	1
Minor	17
Info	0

Issues
18

	30 Apr 2015	02 May 2015 1.0
Issues	191	18
Blocker issues	0	0
Critical issues	61	0
Major issues	47	1
Minor issues	81	17
Info issues	2	0
Technical Debt	4d 1h	1h 11min

Most Violated Rules Any severity More

Tabulation characters should not be used	13
String literals should not be duplicated	4
Unused local variables should be removed	1

Documentation
54.7%

Public API 53

Pub. Undoc. API 24

Comments
13.0%

Comment Lines 243

Lines Of Code
1,628

Java

Files
13

Directories 2

Lines 2,350

Functions
90

Classes 23

Statements 810

Accessors 16

VPSecureAPP

Version 1.0 / 5 May 2015 16:15

Overview Components Issues Settings More

Time Machine

Debt
3h 5min

Blocker	0
Critical	0
Major	7
Minor	3
Info	0

Issues
10

	02 May 2015	05 May 2015 1.0
Issues	216	10
Blocker issues	0	0
Critical issues	87	0
Major issues	83	7
Minor issues	46	3
Info issues	0	0
Technical Debt	4d 2h	3h 5min

Most Violated Rules Any severity More

Source files should not have any duplicated blocks	5
"switch" statements should have at least 3 "case" clauses	3
Lambdas and anonymous classes should not have too many lines	2

Documentation
73.2%

Public API 71

Pub. Undoc. API 19

Comments
14.5%

Comment Lines 482

Lines Of Code
2,843

Java

Files
17

Directories 2

Lines 4,072

Functions
190

Classes 41

Statements 1,429

Accessors 5

Vedlegg H

Innføring i sikkerhet

2 Sider

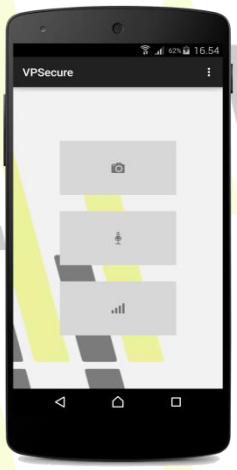
Innføring i sikkerhet – Heed Bacheloroppgave

Hovedaspekter:

TLSv1.2	Nyeste versjon som er tilgjenglig av TLS (Transport Layer Security Protocol) som brukes for å sikre kommunikasjonen mellom patruljenes enheter og serveren hos OPS.
Cipher Suit	TLS_ECHDE_RSA_WITH_AES256_GCM_SHA256. Elliptic Curve Diffie-Hellman brukes til nøkkelutveksling. Vi har vurdert det til tryggest å bruke symmetriske nøkler som byttes ut ved faste mellomrom når enheten er fysisk inne på stasjonen. RSA for verifisering av klientene i handshake protokollen (for at man skal stole på at man er den man utgir seg for å være, brukes som sertifikater). Valgt 2048-bit kryptering som er anbefalt av både NIST og ENISA. AES256 bit kryptering av innhold ved hjelp av GCM. SHA256 (SHA-2) og MD5 for hashing.

Overblikk:

APP



- Alle filer lagres på egen privat lokasjon og er helt separert fra galleriet, og betyr i praksis at galleriet ikke finner filene appen har produsert og motsatt i appen.
- Hver enhet med innstallert app inneholder en database hvor hashen for hver fil lagres. Dette for å kunne sammenligne med hash som produseres for filer som OPS mottar og kan være bevismateriale i en eventuell rettsak
- Må være forhåndsgodkjent på OPS sin side for å kunne koble seg til. Dette gjøres ved hjelp av IMSI og IMEI nr.

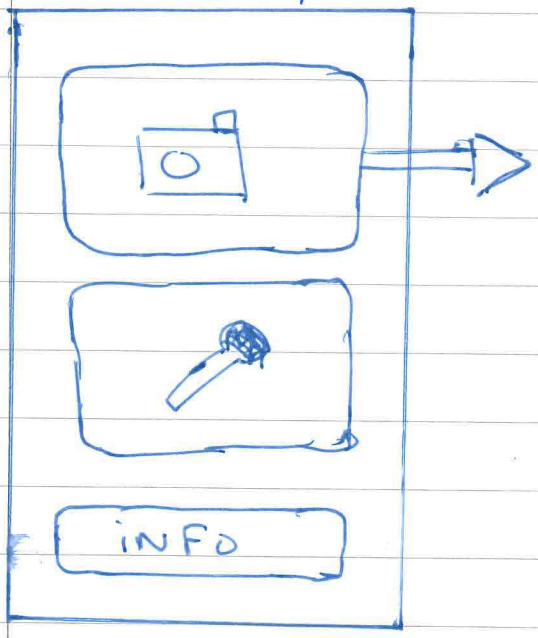
Server

- 1 database, 4 tabeller: «Enheter», «Hash», «Media», & «Hendelsen».
- «Enheter» inneholder en oversikt over alle enhetene som kan koble seg til serveren.
- «Media» inneholder en oversikt over koordinater, filnavn, type, tidsstempler, mm for hver fil i systemet.
- «Hendelsen» kan brukes for å knytte sammen med f.eks en sakskode eller lignende.
- «Hash» brukes for å lagre alle hasher som produseres med innkommende filer.
- Sikkerhetsmessig har vi i denne bacheloroppgaven valgt å avgrense og tatt som forutsetning at sikkerheten på baksiden av brannmuren (hvor serveren opererer) er opp til Politiet.

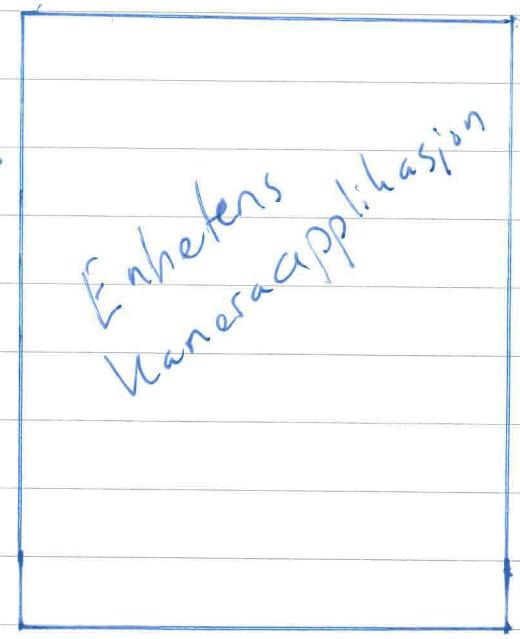
Vedlegg I
Tidlig design
1 Side

Desing Forslag (Tidlig):

Hovedskjerm
Aktivitet 1



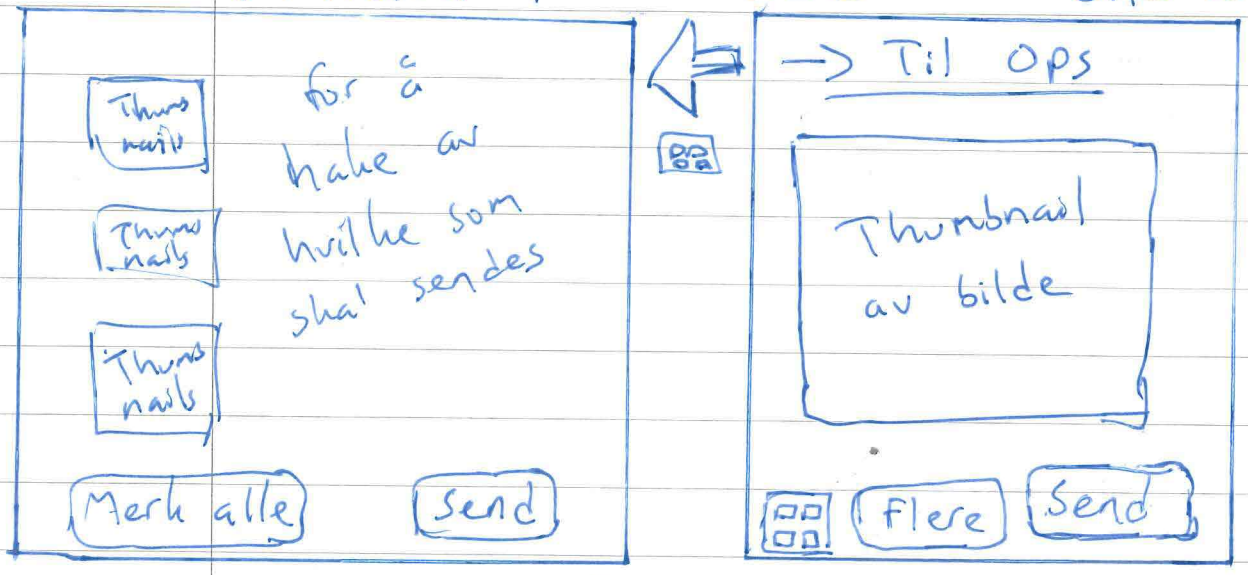
Aktivitet 2 - kanera

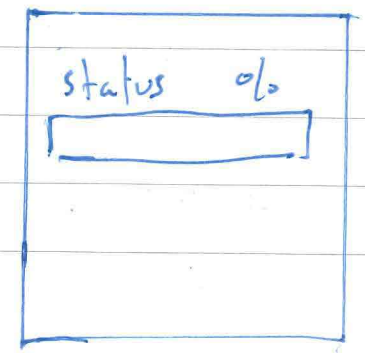


Sending/
Aktivitet 4

Flere

Aktivitet 3 -
etter knipset bilde

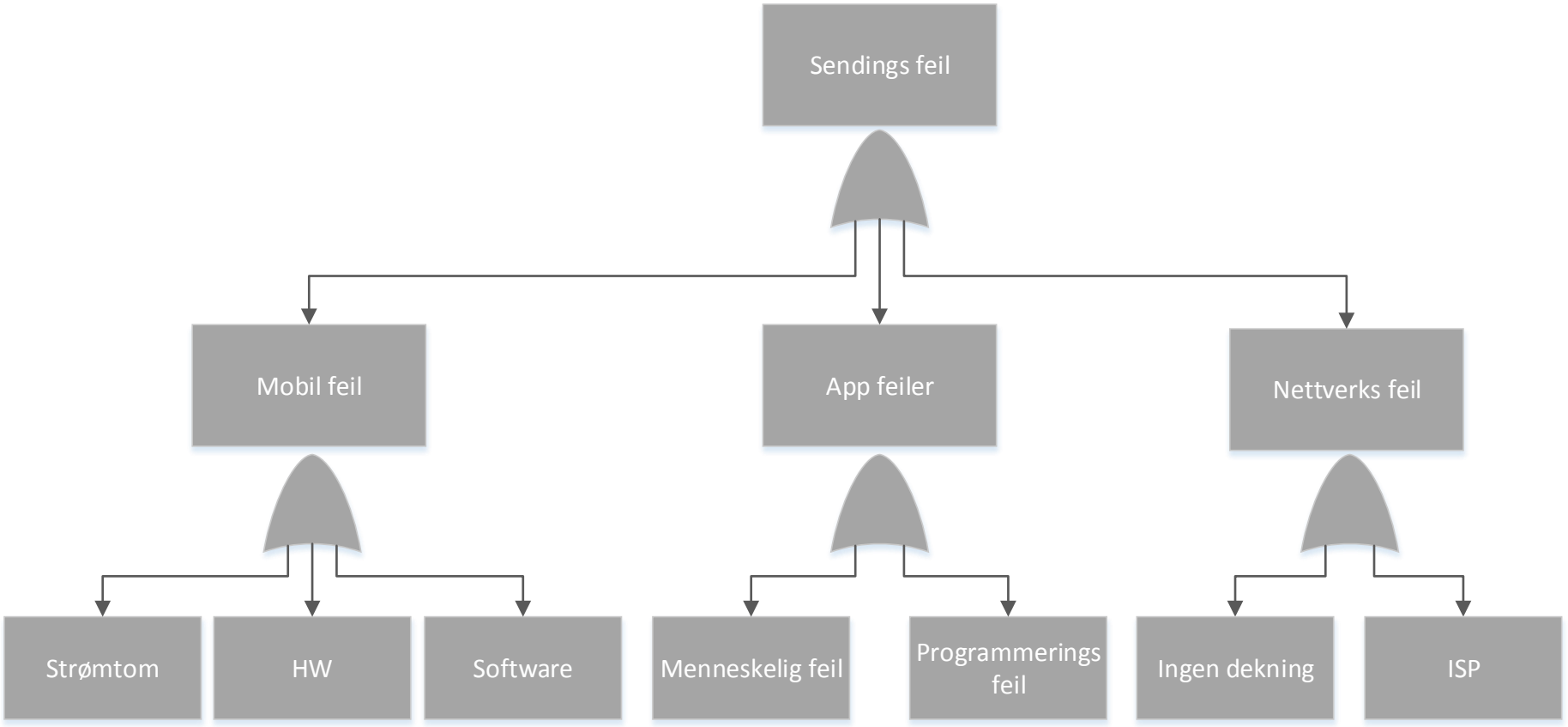


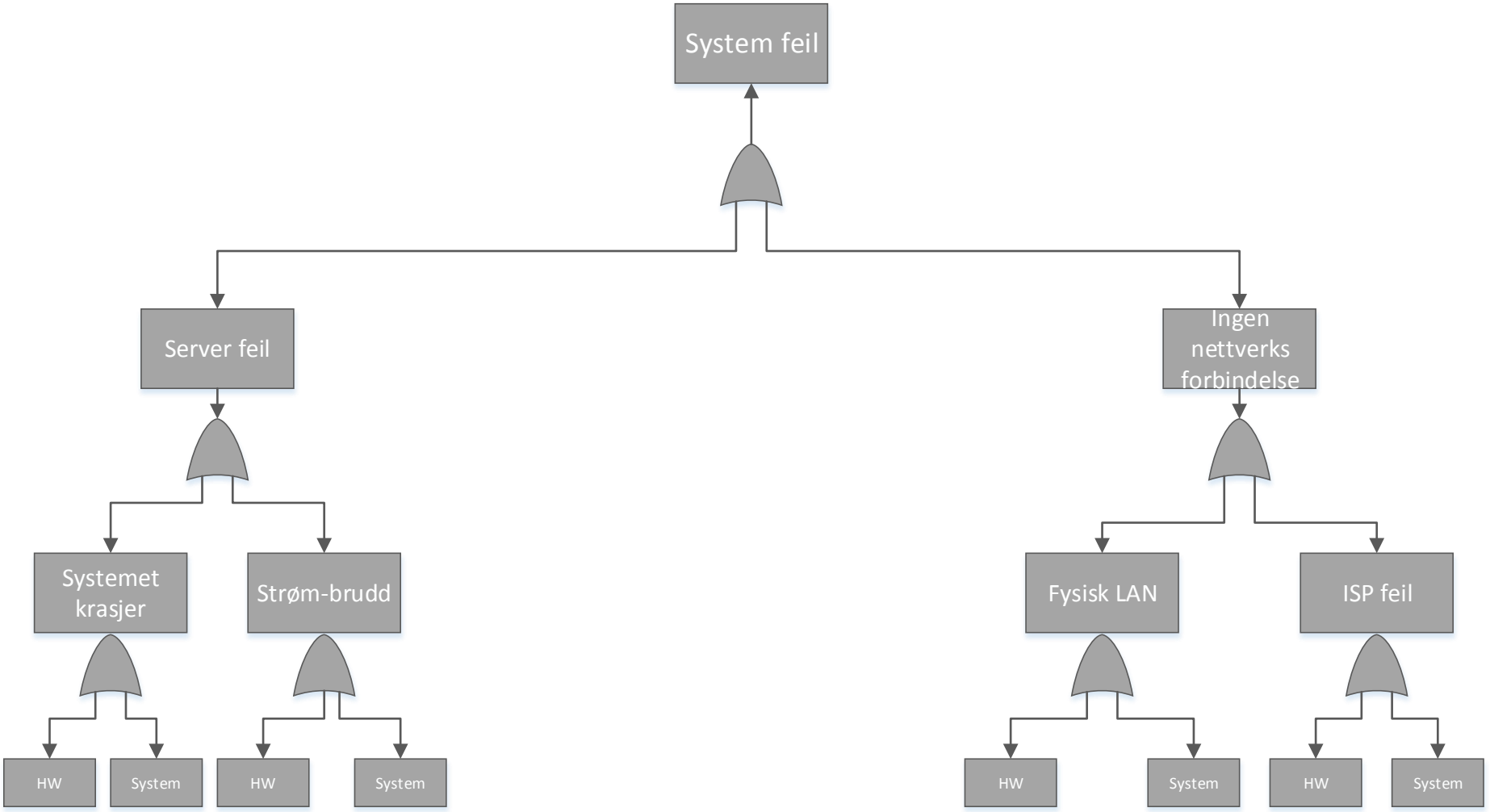


Aktivitet 5 - Sending

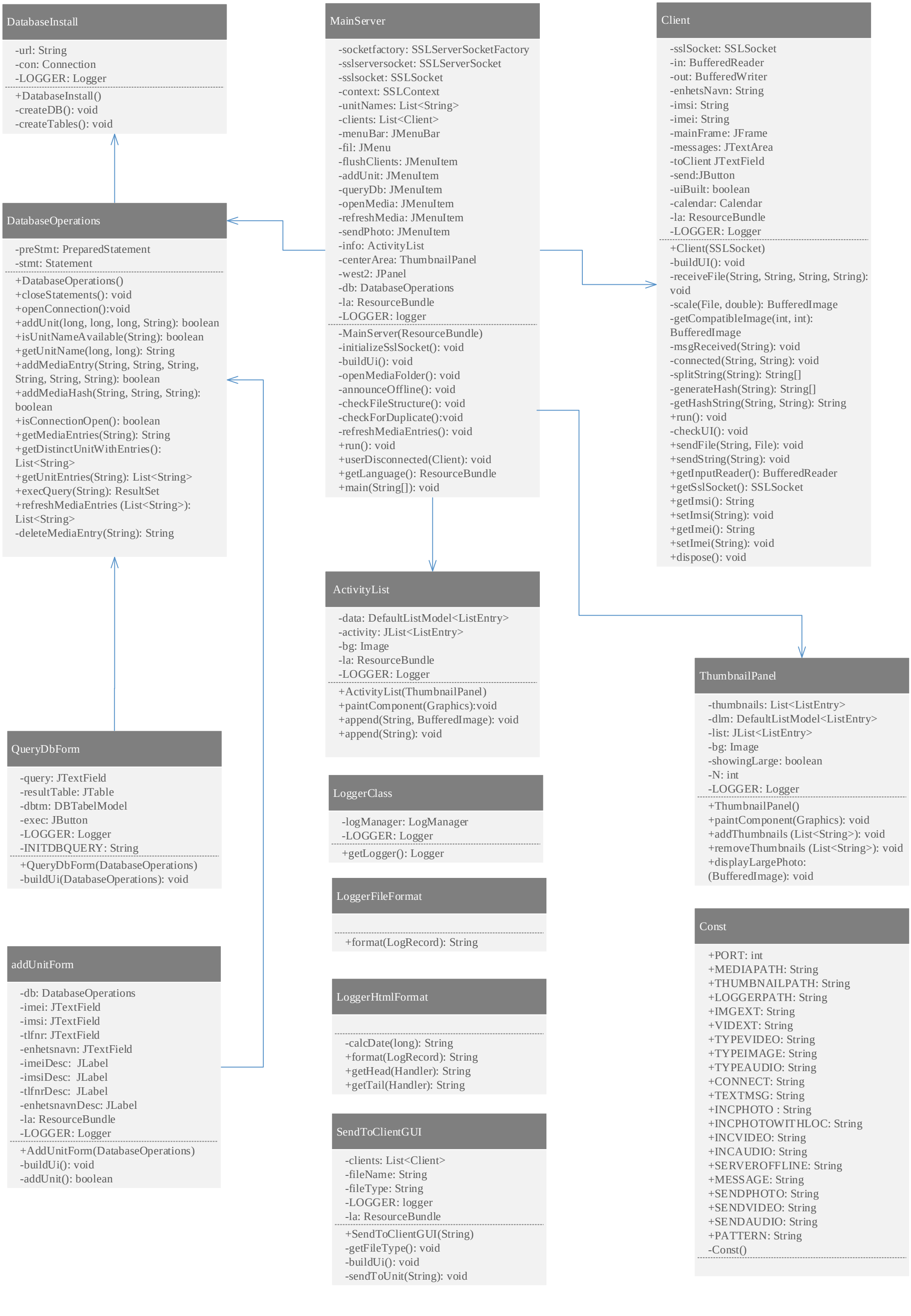
Hva skjer når man trykker til sake?

Vedlegg J
Feiltrær
2 Sider


Sendings feil
Mobil feil
App feiler
Nettverks feil
Strømtom
HW
Software
Menneskelig feil
Programmerings feil
Ingen dekning
ISP


System feil
Server feil
Ingen nettverks forbindelse
Systemet krasjer
Strøm-brudd
Fysisk LAN
ISP feil
HW
System
HW
System
HW
System
HW
System

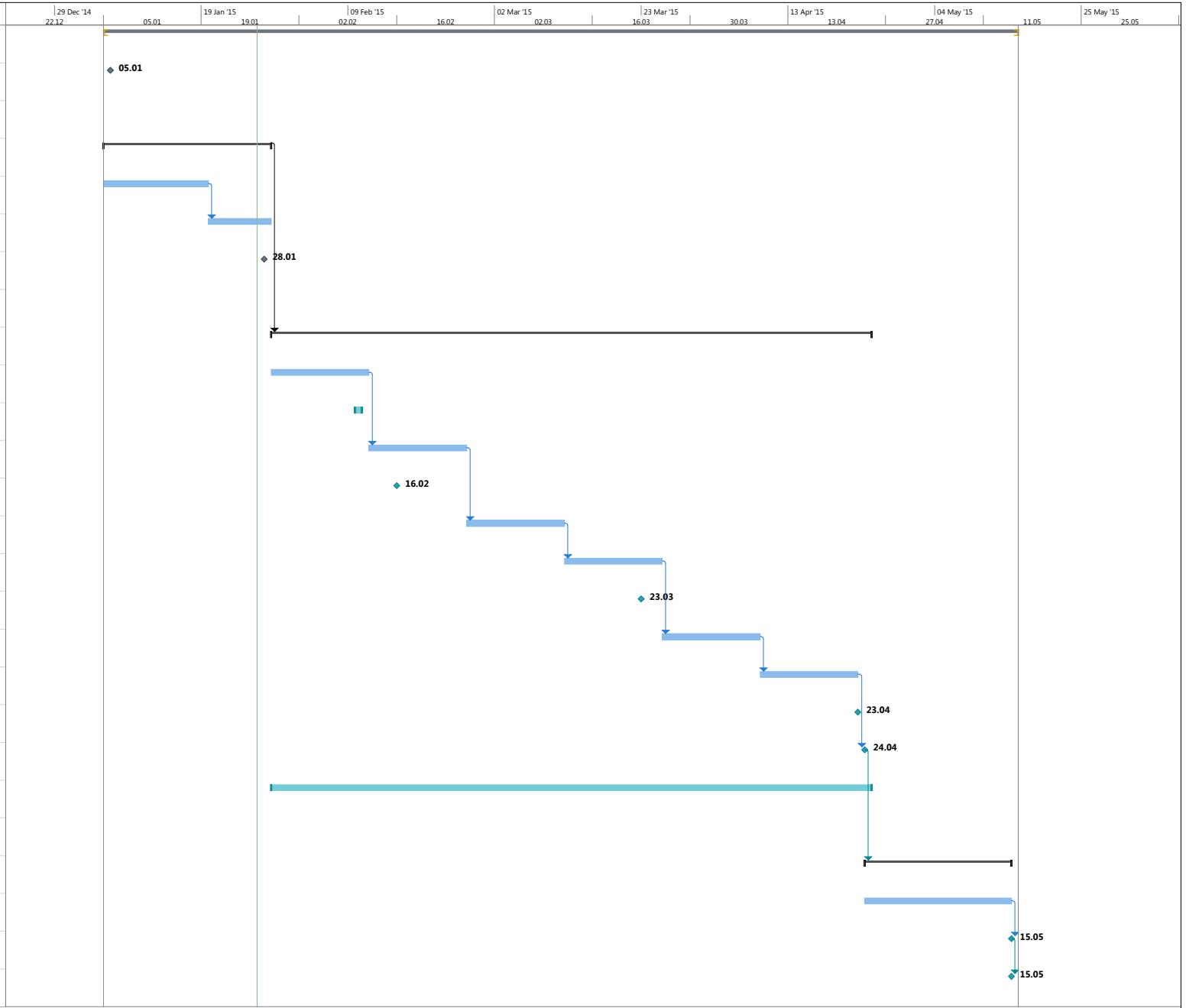
Vedlegg K
Klassediagrammer
2 Sider


DatabaseInstall
-url: String
-con: Connection
-LOGGER: Logger
+DatabaseInstall()
-createDB(): void
-createTables(): void
DatabaseOperations
-preStmt: PreparedStatement
-stmt: Statement
+DatabaseOperations()
+closeStatements(): void
+openConnection():void
+addUnit(long, long, long, String): boolean
+isUnitNameAvailable(String): boolean
+getUnitName(long, long): String
+addMediaEntry(String, String, String, String, String, String): boolean
+addMediaHash(String, String, String): boolean
+isConnectionOpen(): boolean
+getMediaEntries(String): String
+getDistinctUnitWithEntries(): List<String>
+getUnitEntries(String): List<String>
+execQuery(String): ResultSet
+refreshMediaEntries (List<String>): List<String>
-deleteMediaEntry(String): String
QueryDbForm
-query: JTextField
-resultTable: JTable
-dbtm: DBTabelModel
-exec: JButton
-LOGGER: Logger
-INITDBQUERY: String
+QueryDbForm(DatabaseOperations)
-buildUi(DatabaseOperations): void
addUnitForm
-db: DatabaseOperations
-imei: JTextField
-imsi: JTextField
-tlfnr: JTextField
-enhetsnavn: JTextField
-imeiDesc: JLabel
-imsiDesc: JLabel
-tlfnrDesc: JLabel
-enhetsnavnDesc: JLabel
-la: ResourceBundle
-LOGGER: Logger
+AddUnitForm(DatabaseOperations)
-buildUi(): void
-addUnit(): boolean
MainServer
-socketfactory: SSLServerSocketFactory
-sslserversocket: SSLServerSocket
-sslsocket: SSLSocket
-context: SSLContext
-unitNames: List<String>
-clients: List<Client>
-menuBar: JMenuBar
-fil: JMenu
-flushClients: JMenuItem
-addUnit: JMenuItem
-queryDb: JMenuItem
-openMedia: JMenuItem
-refreshMedia: JMenuItem
-sendPhoto: JMenuItem
-info: ActivityList
-centerArea: ThumbnailPanel
-west2: JPanel
-db: DatabaseOperations
-la: ResourceBundle
-LOGGER: logger
-MainServer(ResourceBundle)
-initializeSslSocket(): void
-buildUi(): void
-openMediaFolder(): void
-announceOffline(): void
-checkFileStructure(): void
-checkForDuplicate():void
-refreshMediaEntries(): void
+run(): void
+userDisconnected(Client): void
+getLanguage(): ResourceBundle
+main(String[]): void
ActivityList
-data: DefaultListModel<ListEntry>
-activity: JList<ListEntry>
-bg: Image
-la: ResourceBundle
-LOGGER: Logger
+ActivityList(ThumbnailPanel)
+paintComponent(Graphics):void
+append(String, BufferedImage): void
+append(String): void
LoggerClass
-logManager: LogManager
-LOGGER: Logger
+getLogger(): Logger
LoggerFileFormat
+format(LogRecord): String
LoggerHtmlFormat
-calcDate(long): String
+format(LogRecord): String
+getHead(Handler): String
+getTail(Handler): String
SendToClientGUI
-clients: List<Client>
-fileName: String
-fileType: String
-LOGGER: logger
-la: ResourceBundle
+SendToClientGUI(String)
-getFileType(): void
-buildUi(): void
-sendToUnit(String): void
Client
-sslSocket: SSLSocket
-in: BufferedReader
-out: BufferedWriter
-enhetsNavn: String
-imsi: String
-imei: String
-mainFrame: JFrame
-messages: JTextArea
-toClient JTextField
-send:JButton
-uiBuilt: boolean
-calendar: Calendar
-la: ResourceBundle
-LOGGER: Logger
+Client(SSLSocket)
-buildUI(): void
-receiveFile(String, String, String, String): void
-scale(File, double): BufferedImage
-getCompatibleImage(int, int): BufferedImage
-msgReceived(String): void
-connected(String, String): void
-splitString(String): String[]
-generateHash(String): String[]
-getHashString(String, String): String
+run(): void
-checkUI(): void
+sendFile(String, File): void
+sendString(String): void
+getInputReader(): BufferedReader
+getSslSocket(): SSLSocket
+getImsi(): String
+setImsi(String): void
+getImei(): String
+setImei(String): void
+dispose(): void
ThumbnailPanel
-thumbnails: List<ListEntry>
-dlm: DefaultListModel<ListEntry>
-list: JList<ListEntry>
-bg: Image
-showingLarge: boolean
-N: int
-LOGGER: Logger
+ThumbnailPanel()
+paintComponent(Graphics): void
+addThumbnails (List<String>): void
+removeThumbnails (List<String>): void
+displayLargePhoto: (BufferedImage): void
Const
+PORT: int
+MEDIAPATH: String
+THUMBNAILPATH: String
+LOGGERPATH: String
+IMGEXT: String
+VIDEXT: String
+TYPEVIDEO: String
+TYPEIMAGE: String
+TYPEAUDIO: String
+CONNECT: String
+TEXTMSG: String
+INCPHOTO : String
+INCPHOTOWITHLOC: String
+INCVIDEO: String
+INCAUDIO: String
+SERVEROFFLINE: String
+MESSAGE: String
+SENDPHOTO: String
+SENDVIDEO: String
+SENDAUDIO: String
+PATTERN: String
-Const()


PhotoTakenActivity
-TAG: String
-REQUEST_IMAGE_CAPTURE: int
-REQUEST_VIDEO_CAPTURE: int
-mCurrentPhotoPath: String
-videoPath: String
-videoThumbPath: String
-photoFileNames: List<String>
-photoPaths: List<String>
-imImageView: ImageView
-noSelectedTextView
-trToSend: TextView
-mVideoView: VideoView
-listview: ListView
-isConnected: boolean
-mMessenger: Messenger
-mConnection: CustomServiceConnection
-VIDEOPATH: String
-onCreate(Bundle): void
-onSaveInstanceState(Bundle): void
-onPause(): void
-onResume(): void
-onBackPressed(): void
-onActivityResult(int, int, Intent): void
-getListRowItem(int): ListRowItem
-showFileNameOnly(List<String>: List<String>
-createMediaFile(String): File
-generateHash(String, DatabaseOperations): void
-copyFile(String): String
-cleanRemainingPath(): void
-setVideo(String): void
-setPic(): void
-setOrientation(ImageView, String): void
-dispatchTakeVideoIntent(View): void
-dispatchTakePictureIntent(View): void
-openGallery(View): void
-sendMedia(View): void
-sendImage(): void
-sendMultipleFile(): void
-sendVideo(): void
-getExtension(String): String
AboutActivity
#onCreate(Bundle): void
+onCreateOptionsMenu(Menu): boolean
+onOptionsItemSelected(MenuItem): boolean
+onCreateContextMenu(ContextMenu, View,ContextMenu.ContextMenuInfo): void
+onContextItemSelected(MenuItem): boolean
ListItem
-title: String
-description: String
+ListItem(String, String)
ListRowItem
-image: byte[ ]
-text: String
+ListRowItem(byte[ ], String)
CustomListAdapter
+CustomListAdapter(Activity, List<ListRowItem>)
+getView(int, View, ViewGroup): View
CheckConnectivity
-conToServer: ImageView
-intAcc: ImageView
-networkType: TextView
-mMessenger: Messenger
-mConnection: CustomServiceConnection
-tm: TelephonyManager
-tiType: int
-onCreate(Bundle): void
-onPause(): void
-onResume(): void
-getNetworkType(): void
-checkInternetState(): void
-runOnUi(ImageView, boolean): void
-isConnectToServer(): void
-setSignalLevel(int): void
-setSignalLevelLine(int): void
CustomServiceConnection
-TAG: String
-cx: Context
-mIsBound: boolean
-mService: Messenger
-mMessenger: Messenger
-extraMsg: Message
-doBindService(): void
+CustomServiceConnection(Messenger, Context)
+CustomServiceConnection(Messenger, Context, Message)
+onServiceConnected(ComponentName, IBinder): void
+onServiceDisconnected(ComponentName): void
+rebind(): void
+unbind(): void
+send(Message): boolean
MainActivity
-TAG: String
-mConnection: CustomServiceConnection
-onCreate(Bundle): void
-onPause(): void
-onResume(): void
-checkFileStructure(): void
-onCreateOptionsMenu(Menu): boolean
-onOptionsItemSelected(MenuItem): boolean
-createDir(File, String): File
-dispatchTakePictureIntent(View): void
-startAudioRecord(View): void
-openConnectivity(View): void
ChatActivity
-mMessenger: Messenger
-mConnection: CustomServiceConnection
-messageText: EditText
-textInView: TextView
-isConnected: boolean
-onCreate(Bundle): void
-onPause(): void
-onResume(): void
-onCreateOptionsMenu(Menu): boolean
-onOptionsItemSelected(MenuItem): boolean
-showMsg(String): void
-sendMessage(View): void
SignalStrengthListener
-onSignalStrengthsChanged(SignalStrength): void
RecordAudio
-TAG: String
-selectedPaths: List<String>
-mFileName: String
-audioDirectory: String
-mTimer: TextView
-mRecordButton: ImageButton
-mRecorder: MediaRecorder
-mPlayButton: ImageButton
-mPlayer: MediaPlayer
-mStartRecording: boolean
-mStartPlaying: boolean
-isStateRecording: boolean
-isStatePlaying: boolean
-seekBar: SeekBar
-mMessenger: Messenger
-mConnection: CustomServiceConnection
-isConnected: boolean
-startTime: long
-timerH: Handler
-timer: Runnable
-seekBarHandler: Handler
-updateThread: Runnable
-onCreate(Bundle): void
-onPause(): void
-onResume(): void
-onBackPressed(): void
-getFileDir(): void
-showFileNameOnly(List<String>): List<String>
-onRecord(boolean): void
-onPlay(boolean): void
-startPlaying(): void
-stopPlaying(): void
-startRecording(): void
-stopRecording(): void
-generateHash(String, DatabaseOperations): void
-reset(): void
-record(View): void
-play(View): void
-sendAudioView(): void
-showErrorToast(): void
-openAudioGallery(View): void
-rewind(View): void
-fastForward(View): void
AudioGallery
-TAG: String
-gridView: GridView
-customGridAdapter: GridViewAdapter
-pathMap: Map<String, String>
-selectedPaths: List<String>
#onCreate(Bundle): void
+onCreateOptionsMenu(Menu): boolean
+onOptionsItemSelected(MenuItem): boolean
+onCreateContextMenu(ContextMenu, View, ContextMenuInfo): void
+onContextItemSelected(MenuItem): boolean
+send(View): void
-showHash(File): void
-deleteFiles(List<String>): void
-getData(): List
CompletionListener
-onCompletion(MediaPlayer): void
CustomSeekBarListener
-onProgressChanged(SeekBar, int, boolean): void
-onStartTrackingTouch(SeekBar): void
-onStopTrackingTouch(SeekBar): void
DatabaseOperations
-TAG: String
-database: SQLiteDatabase
-dbHelper: DatabaseInstall
+DatabaseOperations(Context)
+closeDatabase(): void
+addLocToDatabase(String, String, String): long
+getLoc(String): String[ ]
+addHashToDatabase(String, String, String): long
+getHash(String): String[ ]
+generateHash(String, String): String
-getHashString(String, String): String
DatabaseInstall
+TEXT_NOT_NULL: String
+DATABASE_VERSION: int
+DATABASE_NAME: String
+TABLE_MEDIALOC: String
+COLUMN_ID: String
+COLUMN_FILENAME: String
+COLUMN_LONGITUDE: String
+COLUMN_LATITUDE: String
+TABLE_MEDIAHASH: String
+COLUMN_MD5: String
+COLUMN_SHA256: String
-CREATE_TABLE_MEDIALOC: String
-CREATE_TABLE_MEDIAHASH: String
+DatabaseInstall(Context)
+onCreate(SQLiteDatabase): void
+onUpgrade(SQLiteDatabase, int,int): void
ImageGallery
-TAG: String
-PHOTOTITLE: String
-VIDEOTITLE: String
-gridView: GridView
-customGridAdapter: GridViewAdapter
-switchContent: Button
-pathMap: Map<String, String>
-paths: List<String>
-thumbCreation: boolean
-thumbsToCreate: LinkedBlockingQueue<String[ ]>
-onCreate(Bundle): void
-onCreateOptionsMenu(Menu): boolean
-onOptionsItemSelected(MenuItem): boolean
-onCreateContextMenu(ContextMenu, View, ContextMenu.ContextMenuInfo): void
-onContextItemSelected(MenuItem): boolean
-showHash(File): void
-deleteFiles(List<String>: void
-List getData(File): void
-ImageItem getImage(File, int): void
-File[ ] removeDirectories(File[ ]: void
-handleThumbnailCreation(List): void
-Bitmap getBitmap(String): void
-String getFileName(File): void
-String getExtension(File): void
-rotateThumb(Bitmap, float): Bitmap
-sendMultiple(View): void
-switchContent(View): void
ImageItem
-image: Bitmap
-title: String
ImageItem(Bitmap, String): void
-getImage(): Bitmap
-getTitle(): String
-setTitle(String): void
ConnectionService
-Tag: String
-protocols: String[ ]
-sLeepTime: long[ ]
-msgHandler: MessageHandler
-mClient: Messenger
-dispatchQueue: LinkedBlockingQueue<String>
-mMessenger: Messenger
-reconnectCounter: int
-pattern: String
-sslSocket: SSLSocket
-sslContext: SSLContext
-serverAddr: String
-serverPort: String
-MSG_REGISTER_CLIENT: int
-MSG_UNREGISTER_CLIENT: int
-MSG_SEND_TO_SERVER: int
-MSG_SET_STRING_VALUE: int
-MSG_SEND_PHOTO_TO_SERVER:int
-MSG_SEND_VIDEO_TO_SERVER:int
-MSG_SEND_AUDIO_TO_SERVER:int
-MSG_RECONNECT: int
-MSG_IS_CONNECTED: int
-MSG_CONNECTED_RESPONS: int
-onCreate(): void
-onBind(Intent): IBinder
-onStartCommand(Intent, int,int): int
-setUpTrust(): TrustManagerFactory
-isServerConnected(): void
-connectSslSocket(): void
-setUpSSL(): void
-reconnectHandler(): void
-identifyUnit( ): void
-getSocketInfo(): String
-reconnect(): void
-disconnectSocket( ): void
MessageHandler
-TAG: String
-in: BufferedReader
-out: BufferedWriter
-byteStream: DataOutputStream
-pendingMsgs: List<String>
-SERVEROFFLINE: String
-RECEIVEMESSAGE: String
-RECEIVEPHOTO: String
-RECEIVEVIDEO: String
-RECEIVEAUDIO: String
-VIDEO: String
-AUDIO: String
-MessageHandler(SSLSocket): void
-run(): void
-receive(String, String): void
-createMediaFile(String): File
-createThumbFile(File): File
-getFileName(File): String
-receiveTxtMsg(String): void
-splitString(String): String[ ]
-startDispatchQueueHandler(): void
-sendMessage(final): void
-sendPhoto(String): void
-sendFile(String, String): void
-showErrorNoti(int): void
-passMsgToClient(String): void
-close(): void
SettingsActivity
-ALWAYS_SIMPLE_PREFS: boolean
-sBindPreferenceSummaryToValueListener: OnPrefChangeListener
-onCreate(Bundle): void
-onIsMultiPane(): boolean
-onBuildHeaders(List<Header>): void
-isValidFragment(String): boolean
-isXLargeTablet(Context): boolean
-isSimplePreferences(Context): boolean
-bindPreferenceSummaryToValue (Preference): void
Logger
-TAG: String
-currentFile: File
-Logger(Context,String,String)
-Logger(Context, String)
-write(File, String): void
+write(String): void
ShowLogActivity
-TAG: String
-onCreate(Bundle): void
-onCreateOptionsMenu(Menu): boolean
-onOptionsItemSelected(MenuItem): boolean
SplashActivity
-TAG: String
-onCreate(Bundle): void

Vedlegg L
GANTT-skjema
1 Side

ID	Task Mode	Task Name	Duration	Start	Finish
1		HEED	95 days	Mon 05.01.15	Fri 15.05.15
2		Projektstart	0 days	Mon 05.01.15	Mon 05.01.15
3					
4		**Planlegging**	**18 days**	**Mon 05.01.15**	**Wed 28.01.15**
5		Drøfting	11 days	Mon 05.01.15	Mon 19.01.15
6		Skrive prosjektplan	7 days	Tue 20.01.15	Wed 28.01.15
7		Levere prosjektplan	0 days	Wed 28.01.15	Wed 28.01.15
8					
9		**Utvikling**	**62 days**	**Thu 29.01.15**	**Fri 24.04.15**
10		Sprint 1	10 days	Thu 29.01.15	Wed 11.02.15
11		Brukertest	1 day	Tue 10.02.15	Tue 10.02.15
12		Sprint 2	10 days	Thu 12.02.15	Wed 25.02.15
13		Statusraport	0 days	Mon 16.02.15	Mon 16.02.15
14		Sprint 3	10 days	Thu 26.02.15	Wed 11.03.15
15		Sprint 4	10 days	Thu 12.03.15	Wed 25.03.15
16		Statusraport	0 days	Mon 23.03.15	Mon 23.03.15
17		Sprint 5	10 days	Thu 26.03.15	Wed 08.04.15
18		Testing og feilretting	10 days	Thu 09.04.15	Wed 22.04.15
19		Statusraport	0 days	Thu 23.04.15	Thu 23.04.15
20		Sluttføre utvikling	0 days	Fri 24.04.15	Fri 24.04.15
21		Dokumentering	62 days	Thu 29.01.15	Fri 24.04.15
22					
23		**Sluttfase**	**15 days**	**Fri 24.04.15**	**Fri 15.05.15**
24		Skrive prosjektrapport	15 days	Fri 24.04.15	Thu 14.05.15
25		Levere prosjektrapport	0 days	Fri 15.05.15	Fri 15.05.15
26		Levere prototype	0 days	Fri 15.05.15	Fri 15.05.15



Project: Gant.mpp

Task | Split | Milestone | Summary | Project Summary | Inactive Task | Inactive Milestone | Inactive Summary | Manual Task | Duration-only | Manual Summary Rollup | Manual Summary | Start-only | Finish-only | External Tasks | External Milestone | Deadline | Progress | Manual Progress

Vedlegg M
Definisjonaliste
1 Side

Definisjonsliste

Media – Bilder, lyd og video. Alt interaktivt innhold som kan produserer fra en mobil enhet.

Public funksjon (Java programmering) – En funksjon som kan kalles på av andre funksjoner/klasser og ikke bare innenfor en bestemt klasse.

Javadoc – Er en del av Java Eclipse og andre Java baserte programmeringsspråk og er laget for å generere et HTML dokument med dokumentasjon av Java kode.

MITM – «Man in The Middle attack» , når angriperen kobler seg i kommunikasjonen mellom server/klient.

Flytchiffer – Cipher stream

Last – Nyttelast eller «Payload»

«IV» - Initialisation vectors er et vilkårlig nummer som brukes sammen med en hemmelig nøkkel for å kryptere data.

IDE – Integrated Development Environment

Hash/hasing – er en metode for å transformere forskjellige tegn til, som regel en kortere nøkkel som representerer den originale strengen.

Tellere og Anti-tellere- er en underprotokoll av IPsec.

Internet Key Exchange (IKE) – Er en protokoll som brukes av Internet Protocol Security for å sikre virtual private network.

«CBC»- Cipher block chaining er en metode for å kryptere sekvens av bit som en stor blokk.

Topphendelse- Er en uønsked hendelse

Cipherpakke – «Cipher suite» er en navngitt kombinasjon med nøkkelutvekslingsalgoritmer.

Kortsiktig langtidsbruk – Opp til 10 år

Langsiktigbruk – Over 10 år

GSSAPI – General Security Services Application Program Interface. API for sikkerhetstjenester.

Kerberos – Nettverks autentiseringsprotokoll basert på «tickets».

Vedlegg N
Møtereferater
7 Sider

Møtereferat

Møte nummer: 1

Agenda: Tilbakemeldinger på prosjektplanen og tanker om tilleggskontrakt prosjektavtalen.

Dato: 27. januar 2015

Sted: Høgskolen i Gjøvik, Veileders kontor

Tid anvendt: 30 minutter

Deltakere: Daniel Kristoffersen, Eirik Ellegård, Eirik Lintho Bue, Harald Liodden

---

Prosjektplanen inneholdt en del ufullstendige setninger. Dette må rettes opp i, men ellers ga prosjektplanen et godt inntrykk. Mye godt arbeid.
Prosjektmålene: SMART
S: Spesifikk
M: Målbar
A: Alltid oppnålig
R: Relevant
T: Tidsbestemt

Generelt er det litt rundt skrevet, må bli mer spesifikk og tenke at man skal evaluere opp mot de målene man setter seg.

Organisering: Gjøre noe med organiseringen med tanke på SCRUM master og utviklingsleder. Strukturere om dette i form av smelte det sammen til en

SU-modellene: Drøfte andre modeller før det konkluderes med valget som er gjort. Nevn litt egenskaper til de forskjellige og vei de opp i mot hverandre.

Konfigurasjonsstyring: Ikke noe å utsette på her

Risikoanalyse: Veldig bra at vi har tatt det med. Kommentere hva som er greit å og ikke greit. Lage en risikoprofil.

GANTT-skjema: Bedre nå med nye sprint-størrelser. Dokumentering må med som aktivitet.

Grupperegler: Timelogg, noter det innenfor gruppereglene. En felles og en induviduell.

Møtereferat

Møte nummer: 2

Agenda: Progresjonen og prosjektes tilstand. Statusrapport og Trello.

Dato: 10. februar 2015

Sted: Høgskolen i Gjøvik, Veileders kontor

Tid anvendt: 30 minutter

Deltakere: Daniel Kristoffersen, Eirik Ellegård, Eirik Lintho Bue, Harald Liodden

---

Statusrapport på halv –til en helside hvor man skriver om fremgangen. Ikke noe spesiell struktur, men kun fortelle om tilstanden til prosjektet og hvordan arbeidsforhold og tilsvarende er. Oppsummering av hittil arbeid, nevne eventuelle problemer og fortelle om kontakten med oppdragsgiver. Ikke teknisk detaljert.
Et overblikk bare og skal kun inngå som vedlegg i rapporten.

Skrive hvilke verktøy som benyttes for illustrasjonene.

Kanselerer neste møte siden det ikke er behov.

Møtereferat

Møte nummer: 3

Agenda: Intro møte sikkerhet med sikkerhetskonsulent.

Dato: 18. mars 2015

Sted: Høgskolen i Gjøvik, IT-tjenesten

Tid anvendt: 30 minutter

Deltakere: Daniel Kristoffersen, Eirik Ellegård, Eirik Lintho Bue, Christoffer Vargtass Hallestensen

---

Påpekt hvordan vi bør løse hashing opp i mot hvordan vi sender filer.
Ha to hasher, en sterk og en svak. Dette for å få redundans.

Sikkerhetsvurdering av platformer, ved å se på Samsung Knox. Han viste oss sin implementasjon av det.

TLS er en standard på sikkerhet i slike systemer og derfor vanligst å bruke.

Gjøre klart til neste møte med Christoffer og forberede:

- Hvordan vi lagrer ting. Vise koden av dette
- Koden for overføring
- 2 use case diagrammer (bilde og lyd)

Møtereferat

Møte nummer: 4

Agenda: Kontroll og gjennomgang av sikkerheten i systemet med sikkerhetskonsulent.

Dato: 23. mars 2015

Sted: Høgskolen i Gjøvik, IT-tjenesten

Tid anvendt: 60 minutter

Deltakere: Daniel Kristoffersen, Eirik Ellegård, Eirik Lintho Bue, Christoffer Vargtass Hallestensen

---

Utbrodering angående Samsung Knox siden dette kun er for Samsung enheter så derfor se på «Android for Work» som er Google sin videreutvikling av dette for alle enheter.

Endre når det genereres hashing, fra slik det er i dag til at det genereres hash når filen lages og ikke etterpå. Dette for å hindre at det kan være malware eller angrep som skjer under kopiering.
Bytte fra AES128 til AES256.
Unngå «Export» og «Fips» pågrunn av implementerte svakheter av NSA.

Teoretisk sikkerhet: Garantert
Praktisk sikkerhet: Mulig å knekke, men ikke med dagens datakraft.

Foreta statisk analyse av koden (bruk av eksempelvis Sonar), for å sikre kvaliteten.

Svakheter ved asymmetriske nøkler, bruke symmetriske nøkler istedet? Samtidig svakheter ved symmetriske så må ta en vurdering av dette.
Implementere varsling om at sendingen ble brutt og kanskje implementere resend button?
Varsling om sending fullført og at det er mottatt.

Møtereferat

Møte nummer: 5

Agenda: Obligatorisk fremføring av prosjektplanen

Dato: 15. april 2015

Sted: Høgskolen i Gjøvik, Veileders kontor

Tid anvendt: 40 minutter

Deltakere: Daniel Kristoffersen, Eirik Ellegård, Eirik Lintho Bue, Harald Liodden

---

Gjennomførte fremføringen som er et obligatorisk arbeidskrav fra for ingeniører. Gjennomførelsen gikk meget bra, og merka hvor fort tiden går. Satte av omtrentlig 15 minutter, men brukte 30 minutter uten å fått tatt med alt. Derfor må det gjøres sterke avgrensninger til bachelor fremføringen.

1.8 – Selve rapporten kap. i prosjektrapporten:
- Fortelle at vi konsekvent benytter metode for å fortelle om funksjon
- Fortelle om hvordan klassediagrammet vårt er.

Ellers skryt fra veileder om fremgang og arbeidet vårt.

Møtereferat

Møte nummer: 6

Agenda: Siste møte før levering av prosjektrapporten

Dato: 06. mai 2015

Sted: Høgskolen i Gjøvik, Veileders kontor

Tid anvendt: 15 minutter

Deltakere: Daniel Kristoffersen, Eirik Ellegård, Eirik Lintho Bue, Harald Liodden

---

Ikke stort å utsette eller å komme med.

Noe praktiske detaljer rundt fremføringen, angående hvordan vi burde forberede oss og hva man bør ta med. Tips om å ikke være for detaljerte, og kun ta med det generelle.

Vise frem systemet og vise at det virker.

Vedlegg O

Initialisering av server socket

1 Side

```java
/**
 * Sets the properties needed for the {@code ServerSocket} and starts this class'
 * run method.
 */
private void initializeSslSocket() {
   try {
      socketfactory = (SSLServerSocketFactory) SSLServerSocketFactory.getDefault();
      sslserversocket = (SSLServerSocket) socketfactory.createServerSocket(Const.PORT);
      sslserversocket.setUseClientMode(false);

      String[] protocols = {"TLSv1.2"};
      sslserversocket.setEnabledProtocols(protocols);

      String[] cipher = {"TLS_DHE_RSA_WITH_AES_256_CBC_SHA256"};
      sslserversocket.setEnabledCipherSuites(cipher);
      .
      .
      .
```

Vedlegg P

Sending fra app til server

1 Side

```java
/**
 * Gets the date and time a file has been made. Sends the information and
 * the file to the server.
 *
 * @param filePath absolute path of file to send.
 * @param sendCode code that defines what type of file it is.
 */
private void sendFile(String filePath, String sendCode) {
        .
        .
        .
        out.write(stringToSend);
        out.newLine();
        out.flush();

        fileInputStream = new FileInputStream(filePath);
        File fileToSend = new File(filePath);

        byteStream = new DataOutputStream(sslSocket.getOutputStream());
        long fileLength = fileToSend.length();
        byteStream.writeLong(fileLength);
        int nRead;
        int progress = 0;
        byte[] data = new byte[16384];

        while ((nRead = fileInputStream.read(data, 0, data.length)) != -1) {
             byteStream.write(data, 0, nRead);
             progress += nRead;
             mBuilder.setProgress((int)fileLength, progress, false);
             mNotifyManager.notify(notiId, mBuilder.build());
        }
        byteStream.flush();
        .
        .
        .
```

Vedlegg Q

Plassering av media på kø

1 Side

```
/**
 * Used as an inner class that handles connections from other
 * activities. This class puts any data that wants to be transferred
 * to the server on the queue.
 */
class IncomingHandler extends Handler {
        @Override
        public void handleMessage(Message msg) {
            try {
                switch (msg.what) {
                    case MSG_REGISTER_CLIENT:
                        mClient = msg.replyTo;
                        break;
                    case MSG_UNREGISTER_CLIENT:
                        mClient = null;
                        break;
                    case MSG_SEND_TO_SERVER:
                        String textMsg = msg.getData().getString("str1");
                        dispatchQueue.put(msg.what+pattern+textMsg);
                        break;
                    case MSG_SEND_PHOTO_TO_SERVER:
                        String photoPath = msg.getData().getString("photoPath");
                        dispatchQueue.put(msg.what+pattern+photoPath);
                        break;
                     .
                     .
                     .
```

Vedlegg R

Poppe fra kø

1 Side

```
/**
 * Starts a new thread that will continually try to take elements
 * from the {@link #dispatchQueue}. When something is popped from
 * the queue the {@code String} is splitted on a pattern, and the first String is the
 * command that defines what type of file is to be sent.
 */
private void startDispatchQueueHandler() {
   new Thread(new Runnable() {
      @Override
         public void run() {
            String cmd;
            try {
               while ((cmd = dispatchQueue.take()) != null) {
                  String[] commands = cmd.split(pattern);
                  switch(Integer.parseInt(commands[0])) {
                     case MSG_SEND_PHOTO_TO_SERVER:
                          sendPhoto(commands[1]);
                          break;
                     case MSG_SEND_VIDEO_TO_SERVER:
                          sendFile(commands[1], getString(R.string.sendVideo));
                          break;

                       .
                       .
                       .
```

Vedlegg S

Hashing

1 Side

```java
/**
 * Generates two hashes from the file given by the {@code filePath} <br>
 * and save these in the database.
 * @param filePath Of file to create hashes.
 */
public void generatehash(String filePath) {
    String md5Hash = getHashString(filePath, "MD5");
    String sha256Hash = getHashString(filePath, "SHA-256");
    File file = new File(filePath);
    addHashToDatabase(file.getName(), md5Hash, sha256Hash);
}
```

```java
/**
 * Creates a hash using the given algorithm for the file <br>
 * specified by the {@code filePath}
 * @param filePath Of file to create hash.
 * @param algorithm Algorithm to use when calculating hash.
 * @return the hash as a {@code String}
 */
private String getHashString(String filePath, String algorithm){
    FileInputStream fis = null;
    try {
        MessageDigest md = MessageDigest.getInstance(algorithm);
        fis = new FileInputStream(filePath);
        byte[] dataBytes = new byte[1024];

        int nRead;
        while ((nRead = fis.read(dataBytes)) != -1) {
            md.update(dataBytes, 0, nRead);
        }
        byte[] mdbytes = md.digest();

        StringBuilder sb = new StringBuilder("");
        for (byte mdbyte : mdbytes) {
            sb.append(Integer.toString((mdbyte & 0xff) + 0x100, 16).substring(1));
        }

        return sb.toString();
    } catch (IOException | NoSuchAlgorithmException e) {
        Log.e(TAG, "Error while creating hash: " + e.getMessage() + '\n' + e);
    } finally {
        if (fis != null) {
            try {
               fis.close();
            } catch (IOException e) {
               Log.e(TAG, "Error while closing stream: "+ e.getMessage() + '\n' + e);
            }
        }
    }
    return null;
}
```

Vedlegg T
Logg sammendrag
2 Sider

Logg sammendrag

Uke 2
Denne uken startet planleggingsfasen av prosjektet. Grupperegler og gruppeleder ble valgt. Vi drøftet også oppgaven med oppdragsgiver og luftet tekniske løsninger.
15 timer, pr person.

Uke 3
Denne uken drøftet vi prosjektmål, delmål og utviklingsplan. Vi vurderte også mulig avgrensninger.
26 timer, pr person.

Uke 4
Denne uken gjorde vi ferdig prosjektmål, drøftet organisering av kvalitetssikring og startet på risikoanalyse. Begynte også smått å se på utviklingsverktøy.
23 timer, pr person.

Uke 5
Denne uken satt vi opp GANTT-skjema, leverte prosjektplanen hadde møte med veileder og hadde sprint møte.
22 timer, pr person.

Denne uken startet utviklingsfasen og vi vil derfor skrive loggen sprint vis.

Sprint 1 (29.01-12.02)
Satt opp enkel GUI for app og server. Påbegynt TLS research. Definert server – klient kommunikasjon, kryptering for lagring og definering av hva som lagres permanent. Også satt opp enn enkel TLS koblingsmulighet for app og server. I slutten av denne sprinten hadde vi statusmøte 2 med veileder.
60 timer, pr person.

Sprint 2 (12.02-26.02)
Under denne sprinten var Eirik Ellegård syk en ukes tid, derfor ble det noe mindre jobbing men han jobbet fortsatt hjemme fra. Implementert mulighet for å sende bilde og lyd fra app til server og fullført funksjonalitet for å opprettet oppføringer i databasen for disse media.
70 timer, pr person.

Sprint 3 (26.02 – 12.03)
Under denne sprinten utviklet vi funksjonalitet for å kunne sende lyd til server. Det ble også implementert funksjonalitet for å sende media fra server til appen.
65 timer, pr person.

Sprint 4 (12.03 – 26.03)

Under denne sprinten ble det mye fokus på GUI for app, og funksjonalitet knyttet opp mot dette. Første møte med sikkerhetskonsulent Christoffer Vargtass Hallestensen for å avklare detaljer og avtale dato for gjennomgang av sikkerhet og kode. Samt å ha sikkerhetsmøte med Christoffer hvor det ble gjennomgått kode og vurdering av sikkerheten. Skrev statusrapport.
Prosjektrapporten ble påbegynt og kom godt i gang med innledningen. Skrev statusrapport.
50 timer, pr person.

Sprint 5 (26.03 – 09.04)

Gjorde klart for brukertest hos Politiet. Laget skjemaer, printet ut skjermbilder, lagde presentasjon. Brukertest hos Vestoppland Politidistrikt.
Implementerte ønsker de yttret om systemet.
Fullførte implementasjon av sending av lokasjon. Varsel om sending ble brutt og hashen ble opprettet tidligere, etter råd fra sikkerhetskonsulent.
Påskeferie, med noe jobbing, men litt redusert pågrunn av gruppemedlemmer på hyttetur.
Lagde oppsummeringsdokument for innføring i sikkerhet som IKT konsulent Jan Erik Ødegård hos Vestoppland Politidistrikt yttret ønske om å få.
40 timer pr, person

Test Sprint (09.04-23.04)

Gjennomførte større test av systemet for å verifisere at ting fungerte og benyttet avsatt tid til feilretting. Gjennomførte også engelsk fremføring av prosjektplanen, som er et arbeidskrav.
Gira opp rapportskriving, fullførte hovedkapitler.
Sikkerhetsvurdering av platformer, skrev klasseforklaringer. Skrev statusrapport.
Avsluttet utviklingen.
60 timer pr, person

Sluttfase (24.04-15.05)

Fullførte rapportskriving, forberedte kode for levering, kontrollerte Javadoc.
120 timer pr, person.

Totalt omtrentlig: 550 timer per person