BACHELOROPPGAVE:

PERSONVERN OG DATALEKKASJE

FORFATTERE:

- EIRIK BAE
- KJETIL GARDÅSEN
- DAVID UELAND

Dato: 23.5.2012

SAMMENDRAG AV BACHELOROPPGAVEN

Tittel:	Personvern og datalekkasje	Nr. :
		Dato : 23/5-2012
Deltaker(e):	Eirik Bae	
	Kjetil Gardåsen	
	David Ueland	
Veileder(e):	Nils Kalstad Svendsen	
Oppdragsgiver:	Høgskolen i Gjøvik	
Kontaktperson:	Lasse Øverlier	
Stikkord (4 stk)	Personvern, Android, Analyse, Datalekkasje	

Antall sider: 133	Antall bilag: 18	Tilgjengelighet (åpen/konfidensiell): Åpen

Kort beskrivelse av bacheloroppgaven:

Bacheloroppgaven omhandler personvern ved bruk av Android og applikasjoner på plattformen. Kartleggingen av personvernet på Android ble gjennomført ved å utføre analyser av utvalgte applikasjoner med et egenlagd testmiljø.

Det var nødvendig å utvikle et testmiljø ettersom eksisterende verktøy ikke hadde tilfredsstillende funksjonalitet for å utføre en analyse av applikasjonene. Vi endte opp med å modifisere sslsniff for å kunne dekryptere SSL-trafikk med Wireshark. Først da kunne vi hente ut binære datastrømmer, samt analysere trafikken på en oversiktelig måte.

En applikasjon ble analysert ved å koble en Android-enhet til testmiljøet vårt for å se hva som sendes ut mot Internett og til hvem mens applikasjonen brukes.
Datatrafikken blir registrert i Wireshark og senere brukt som datagrunnlaget i analysen av applikasjonen.

Konklusjonen på oppgaven er at alle applikasjonene, med unntak av Google Sync, sendte sensitiv informasjon fra Android-enheten uten at det var nødvendig for funksjonaliteten til applikasjonen. Det mest uskyldige var attributter som IMEI og telefonnummer, mens det mest graverende var applikasjonen Hooked som samlet informasjon om alle applikasjonene som var installert og ble startet på telefonen.

Utvidet sammendrag

Denne bacheloroppgaven har undersøkt hvordan personvernet blir ivaretatt på smarttelefonplattformen Android, ved bruk av operativsystemet og applikasjoner. Det har blitt gjort en analyse av selve operativsystemet og et sett med utvalgte applikasjoner, for å finne ut om privat informasjon blir sendt vekk fra Android-enheten. Analysene har blitt gjennomført ved å undersøke applikasjonenes datatrafikk, for å se hva den inneholder og hvem den blir sendt til.

For å gjøre analysene har det blitt utviklet et testmiljø som kan fange opp datatrafikken og analysere innholdet i den. Datatrafikken har for det meste vært kryptert, og derfor måtte testmiljøet kunne dekryptere SSL-trafikk. For å dekryptere datatrafikken ble programmet sslsniff brukt. Ett av kravene til testmiljøet var at det hadde muligheten til å hente ut binære datastrømmer fra den dekrypterte datatrafikken, og dette var ikke mulig med sslsniff. Derfor ble det skrevet om for å kunne tilby denne funksjonaliteten i nettverksanalyseverktøyet Wireshark. Dette er så vidt vi vet en unik funksjonalitet som gjør at SSL-trafikk enkelt kan fanges opp for dekryptering og analyse i Wireshark.

I tillegg til analysene av datatrafikken ble verktøyet TaintDroid benyttet. Dette programmet viser all informasjon som blir aksessert på enheten og sendt ut på Internett. Disse dataene har blitt brukt sammen med analysene av datatrafikken, for å se hvordan operativsystemet og de utvalgte applikasjonene ivaretar personvernet.

Konklusjonen på prosjektet er at alle de analyserte applikasjonene, med unntak av Google Sync, sender privat informasjon fra Android-enheten og ut på Internett. Det mest uskyldige var attributter som IMEI og telefonnummer, mens det mest graverende var applikasjonen Hooked som samlet informasjon om alle applikasjonene som var installert og ble startet på telefonen. Alle de utvalgte applikasjonene hadde legitime formål, og var ikke konstruert utelukkende for spionasje eller datainnsamling. Dermed har det fremstått for oss som at det er vanlig praksis for applikasjoner å sende vekk informasjon, slik som IMEI eller lokasjon, fra enheten den kjører på.

Extended abstract

This bachelor thesis have examined how the privacy is being addressed on the smartphone platform Android, when using the operating system and applications. There has been done an analysis of the operating system itself and a selection of applications, to find out if privacy sensitive information is being sent from the Android device. The analysis have been done by examining the applications data traffic, to see what it contains and to whom it is being sent.

To do the analysis there has been developed a test enviroment which can capture data traffic and analyze its content. The data trafffic has mostly been encrypted and therefore the test enviroment had to be able to decrypt SSL-traffic. To decrypt the data traffic the program sslsniff was used. One of the requirements for the test enviroment was that it could get binary data streams from the decrypted traffic, but this was not currently possible with sslsniff. Therefore the program was changed to provide this functionality in the network analysis tool Wireshark. This is a unique functionality that makes SSL-traffic easy to capture for decryption and analysis in Wireshark.

In addition to the analysis of the data traffic, the tool TaintDroid was used. This program shows all the sensitive information that is being accessed on the device and sent onto the Internet. This data have been used together with the analysis of the data traffic, to see how the operating system and the chosen applications maintains the users privacy.

The conclusion of the project is that all the analyzed applications, with the exception of Google Sync, send privacy sensitive information from the Android device and onto the Internet. The most innocent was IMEI and phone number, while the most serious was done by the application Hooked, which collected information about all the installed and every started application. All the chosen applications had legitimate purposes, and was not constructed solely for for spying or data collection. Therefore it has seemed to us that it is usual practice for applications to send away information such as IMEI and location from the device it is being run on.

Forord

Oppgaven "Personvern og datalekkasje" var en bacheloroppgave som ble gitt av førsteamanuensis Lasse Øverlier ved Høgskolen i Gjøvik, våren 2012. Som informasjonssikkerhetsstudenter ønsket vi å ha en oppgave som handlet om informasjonssikkerhet, og det var derfor denne oppgaven ble valgt. Alle gruppemedlemmene er opptatt av personvernsspørsmål og ville gjerne bidra med mer informasjon om temaet. Oppgaven var også av en teknisk art, og vi ønsket å få anledning til å bruke de tekniske ferdighetene vi har tilegnet oss gjennom studiet.

Vi har gjort mye nytt i løpet av tiden vi har drevet med prosjektet, og det har blitt gjennomført analyser av hvordan applikasjoner ivaretar personvernet, hvor vi har lært mer om personvern, analyser av datatrafikk og gjenbruk av kildekode. Det har vært et tidkrevende prosjekt, men vi håper analysene og testmiljøet vi har laget kan komme til nytte for oppdragsgiver og andre som ønsker å vite hvordan personvernet blir ivaretatt på Android-plattformen, ved bruk av applikasjoner.

Vi ønsker å takke oppdragsgiver Lasse Øverlier og veileder Nils Kalstad Svendsen for veiledning og hjelp gjennom hele prosjektet. Takk til alle som har hjulpet oss med forskjellige spørsmål av teknisk art som vi har hatt under prosjektarbeidet.

Kjetil Gardåsen

Eirik Bae

David Ueland

Innholdsfortegnelse

Forkortelser og ordforklaringer

Her følger definisjonen av forkortelser og ord brukt i rapporten.

Forkortelser

- **ADB** - Android Debug Bridge - Kommandolinjeverktøy for å kommunisere med en Android-enhet fra en PC.
- **APK** - Application Package File - Et filformat for å levere og installere applikasjoner på Android.
- **C2DM** - Cloud to Device Messaging - Rammeverk for å sende data fra servere til applikasjoner på Android.
- **CA** - Certificate Authority - Utsteder av digitale sertifikater brukt i SSL.
- **IMEI** - International Mobile Equipment Identity - Unikt identifikasjonsummer for mobiltelefoner.
- **MITM-angrep** - Man-in-the-middle-angrep - En type angrep hvor en angriper passivt lytter eller aktivt endrer datatrafikken mellom to endepunkter.
- **ROM** - Read-only memory - Brukes som betegnelse på fastvaren på Android-enheter.
- **SSL** - Secure Sockets Layer - En protokoll som benyttes for å kryptere kommunikasjon mellom to endepunkter. Vi bruker ordet SSL til å omfatte både TLS og SSL i rapporten.
- **TLS** - Transport Layer Security - Etterfølgeren til SSL.
- **White hat hacking** - Lovlig penetrasjonstesting.
- **Wi-Fi** - Trådløst nettverk.
- **RPC** - Remote Procedure Call.
- **VoIP** - Voice over IP - Telefonsamtaler over Internett.
- **Kernel** - Kjernen/hovedfunksjonaliteten i et operativsystem.

Ordforklaringer

- **Personvern** - "Vern mot misbruk av persondata(system)" [62]
- **Fastvare** - Programvaren på en Android-enhet. Installert versjon av Android på en mobiltelefon.
- **Android-enhet, mobiltelefon, telefon, nettbrett** - For å refere til Android-enhetene som er brukt i prosjektet kalles de både mobiltelefoner, nettbrett og Android-enheter for å få variasjon i språkbruken. Betydningen er fortsatt den samme.
- **Flashing** - Installere ny fastvare på en Android-enhet.
- **Omvendt utvikling** - Fornorsking av det engelske uttrykket reverse engineering. Omgjøringen av programfiler til kildekode som lettere kan leses av mennesker.
- **Sensitiv informasjon** - Omfatter informasjon som telefonnummer, IMEI, kontaktliste, SMS, kamerabilder, GPS-posisjon osv.
- **Android ID** - En Android-enhets unike ID. Forandres ved tilbakestilling til fabrikkinstillinger.

1 Introduksjon

1.1 Innledning

De siste årene har smarttelefoner og nettbrett blitt allemannseie [51], samtidig som de har hatt en eksplosiv vekst i lagringskapasitet og funksjonalitet. En smarttelefon i dag har like mye lagringsplass [61] som en datamaskin hadde for 12 år siden, og dermed kan telefonen vite mye mer om brukerne sine enn tidligere. Når Internett i tillegg har blitt tilgjengelig over alt hvor det er mobildekning gjør det at veien fra en smarttelefon til resten av verden stadig blir kortere.

Personopplysninger blir i personopplysningsloven [38] definert som *"opplysninger og vurderinger som kan knyttes til en enkeltperson"*, og inneholder dermed et bredt spekter av informasjon som må beskyttes. Dette kan være viktig for å både ivareta omdømmet og sikkerheten til enkeltpersoner, og sørge for at eieren av opplysningene har oversikt over hvem som har tilgang til dem.

Personvern på mobile plattformer var tidligere kun et forhold mellom brukeren og produsenten av telefonen, siden det bare var produsenten som leverte programvare til den. Men etter at smarttelefonene gjorde sitt inntog har antallet aktører som kjører programvare på telefonen eksplodert [3]. Når brukerne kan installere og kjøre en nesten ubegrenset mengde med applikasjoner må de også forholde seg til et utall programvareprodusenter som både har tilgang til alt de har lagret på telefonen, samtidig som de har muligheten til å samle informasjon om dem i sanntid. Mange mer eller mindre skruppeløse programvareutviklere har benyttet seg av mulighetene til å samle inn informasjon om brukere, både for reklameformål og statistikk, men også for regelrett spionasje. Det er flere eksempler [44] på applikasjoner som stjeler data fra telefonen og sender dem tilbake til utviklerne eller andre tredjeparter.

Det finnes mekanismer for å begrense rettighetene [24] applikasjonene har til å få tilgang til data og funksjoner på mobiltelefonen, men det kan fremstå som vanskelig å bedømme hva de forskjellige tillatelsene betyr i praksis og hva som trengs for at en applikasjon skal fungere. Når en applikasjon har blitt gitt en tillatelse er det ingenting som forteller hva den gjør innenfor tillatelsen den har fått. Den kan gjøre noe så uskyldig som å varsle om en ny innkommende SMS, eller den kan lese alle SMSene og sende dem tilbake til utvikleren. Det er ingen måte å nyansere på innenfor en gitt tillatelse.

Å forvente at brukerene skal være bevisste på hvordan personvernet blir ivaretatt ved bruk av smarttelefonen er neppe realistisk, og for å faktisk forstå hva applikasjonene gjør med dataene de har tilgang til må det gjøres analyser. I dette prosjektet har vi undersøkt en rekke populære applikasjoner på Android for å finne ut av om de deler informasjonen de har tilgang på med utviklere eller andre tredjeparter.

1.2 Prosjektets bakgrunn

Oppdragsgiver ønsket å gjøre en studie for å finne ut av om det foregår informasjonslekkasjer ved bruken av applikasjoner på mobile enheter. Målet var å finne ut av hvilken informasjon applikasjonene eventuelt samler inn, og hvem den sendes til. Dette prosjektet var noe alle gruppemedlemmene fant interessant, og siden vi studerer informasjonssikkerhet er personvern og personvernsutfordringer noe som interesserer oss. På smarttelefoner er dette en aktuell problemstilling, siden det er mange forskjellige aktører som kan kjøre programvare på disse enhetene. Dermed blir personvernsutfordringene komplekse, og det må utføres analyser for å få klarhet i hvordan personvernet blir ivaretatt.

Oppgaven var veldig fri med tanke på valg av applikasjoner som skulle analyseres, og gruppa stod fritt til å selv velge dem. Etter en diskusjon med oppdragsgiver ble det enighet om at oppgaven burde omhandle applikasjoner på smarttelefonoperativsystemet Android [16]. Android ble valgt fordi det er det største [50] operativsystemet for smarttelefoner, med en markedsandel på over 48%. Gruppa ville gjerne undersøke applikasjoner som har stor utbredelse, og helst noe som finnes installert på mange Android-enheter. Derfor ble det valgt å undersøke hvordan selve Android-operativsystemet kommuniserer med Google, både gjennom "skjult" kommunikasjon når Android kobler til Internett, men også hvordan en innebygd funksjon som Google Sync kommuniserer med Googles servere når den synkroniserer med dem. I tillegg ble det planlagt å undersøke applikasjoner som ikke var laget av Google, og det ble gjort et utvalg av applikasjoner som er populære på plattformen for analyse.

Det gjennomføres stadig analyser av applikasjoner på Android, men siden operativsystemet og maskinvaren endres raskt, så gjør applikasjonene det samme. Dermed er det ikke nødvendigvis slik at en applikasjon oppfører seg på samme måte i dag som da den sist ble analysert. Det må følges med på utviklingen og undersøkes hvordan applikasjonene oppfører seg etter at de har blitt oppdatert. Mange applikasjoner har lukket kildekode og bruker proprietære protokoller for kommunikasjon, og dermed er det enkelt å gjøre forandringer for utviklerne, men desto vanskeligere for utenforstående å gjøre analyser, siden alt kan endre seg på kort tid.

Å utføre analyser har tidligere vært begrenset av mulighetene for å sette opp et testmiljø for å utføre dem. Det finnes per i dag lite tilgjengelig programvare for å dekryptere SSL-trafikk, og de programmene som gjør det har vært beregnet på vanlig web-trafikk med nettsider av tekst. Binærdata, slik som komprimerte filer, blir ødelagt hvis de lagres som tegn i en tekstfil, og de må i stedet hentes ut som en binær datastrøm og lagres som det for å kunne brukes. Dermed måtte det også bli en del av oppgaven å lage et testmiljø som var så robust og funksjonelt at det kunne takle alle datatyper som sendes via SSL.

Tidligere arbeider

I rapporten "Hva vet appen om deg?" [56] fra Datatilsynet [7] blir det konkludert med at *"App-brukeren er ikke i tilfredsstillende grad informert om hvilke opplysninger som samles inn om vedkommende, hva som er formålet med innsamlingen og hvordan opplysningene eventuelt viderebrukes."*. Informasjon som applikasjoner typisk henter ut er navn, adresse, mobilnummer og IMEI.

Jon Oberheide er en uavhengig sikkerhetsanalytiker som har undersøkt [53, 54] hvordan Android Market installerer programvare på Android ved hjelp av GTalkService-protokollen. Dette er Googles mekanisme for å sende kommandoer til Android via en såkalt C2DM-arkitektur. I artikkelen omvendt utviklet Oberheide GTalkService-protokollen for å undersøke hvordan den fungerte.

Forskere ved universitetene Duke og Penn State har i samarbeid med Intel Labs utviklet TaintDroid [8], en modifisert versjon av Android som overvåker hva andre applikasjoner aksesserer av data, og hvem de sendes til. Dette er et verktøy som er nyttig for å undersøke applikasjoner og se om de sender vekk personsensitiv informasjon.

To studenter ved NTNU har laget en prototype til en applikasjon [58] som hindrer andre applikasjoner fra å samle informasjon og spre denne videre.

Et forskningsprosjekt [46] ved Kansas State University har gått ut på å lage et program som via statisk analyse finner frem til applikasjoner som lekker personlig informasjon.

Ved NC State University har man laget et system [70] for å gi Android en egen personvernsmodus som gir bedre kontroll over hvilke rettigheter applikasjoner har tilgang til på Android.

Vår bakgrunn

Alle gruppemedlemmene studerer til en bachelor i informasjonssikkerhet ved Høgskolen i Gjøvik. Ingen av oss har tidligere erfaring med programmering på mobile plattformer, men alle har erfaring med programmering i C++ og Java fra studiene. Gjennom prosjektet har det måttet brukes eksisterende kildekode skrevet i C og C++, og vi har måttet gjøre endringer på den. Kunnskapene vi allerede hadde har stort sett vært tilstrekkelig, og det er lite nytt som har behøvdes læres om programmering for å kunne gjøre endringene vi behøvde.

Fra emnet "Datakommunikasjon og nettverkssikkerhet" har vi lært nettverksteori, og hvordan man bruker Wireshark for å gjøre analyser av nettverkstrafikk. Dette har vært svært nyttig for å undersøke hva applikasjonene sender og mottar av data. For å gjøre analysene og feilsøkingen har vi måttet lære mer om hvordan forskjellige nettverksprotokoller fungerer, eksempelvis SSL og HTTP.

Emnene "Operativsystemer" og "Systemadministrasjon" har gitt oss kunnskaper om

hvordan man kompilerer og bruker mange forskjellige Linux-verktøy som har blitt brukt for å sette opp testmiljøet og gjøre analysene.

1.3 Oppgavebeskrivelse

For å undersøke om applikasjoner sender fra seg sensitiv informasjon ble det gjort et utvalg av dem som skulle undersøkes nærmere:

- Androids kommunikasjon med Google ved oppkobling mot Internett.
  - Det sendes mye data til Google ved oppkobling mot Internett, og derfor er det interessant å vite hva denne inneholder.
- Google Play [17]
  - Google Play har tidligere blitt undersøkt, og det er derfor interessant å vite om applikasjonen fortsatt oppfører seg på samme måte.
- Google Sync [23]
  - Google Sync er en viktig funksjon i Android som synkroniserer innholdet fra Googles nettjenester med mobiltelefonen. Derfor kan man anta at mye personlig informasjon blir sendt når denne funksjonen brukes.
- Facebook for Android [9]
  - Facebook ble valgt fordi det er en populær applikasjon på Android. Ettersom det er en applikasjon, har den tilgang til mer funksjonalitet på enheten enn nettsidene, og derfor er det mulig at mer informasjon samles om brukerene enn det som går ann gjennom nettsidene.
- UberSocial for Twitter [64]
  - Dette er en uoffisiell Twitter-klient, og ble valgt fordi offisielle applikasjoner ofte blir analysert, mens uoffisielle i større grad har unngått søkelyset.
- Hooked - best games for you! [35]
  - Hooked anbefaler nye spill for brukerne, og trenger informasjon for å gjøre valgene sine. Dermed kan det sendes mye informasjon mellom applikasjonen og dem.
- Skype [60]
  - Skype er et populært program for VoIP, og det er derfor interessant å se hvordan personvernet blir ivaretatt ved å bruke denne applikasjonen.
- Gmail [21]

– Gmail gir tilgang til Googles epost-tjeneste. Applikasjonen brukes derfor ofte til å synkronisere med Google, og det er interessant å se om det blir utvekslet mer informasjon enn bare epost.

Hovedmålet med prosjektet var å gjøre en analyse av applikasjonene og finne ut om de sender vekk sensitiv informasjon over Internett. Siden det var datatrafiken som gikk over Internett som skulle undersøkes ble det et krav om å lage et testmiljø for å fange opp denne. De fleste av applikasjonene bruker SSL for å kryptere datatrafikken sin, og testmiljøet måtte derfor også kunne dekryptere SSL.

Dermed fikk prosjektet to hovedmål som måtte nåes for å fullføre det:

1. Lage et testmiljø for å fange opp datatrafikk fra Android-enheter og dekryptere den.
2. Gjøre en analyse av Android-plattformen og applikasjoner som kjører på den.

Mål 2 var avhengig av at mål 1 var oppnådd, for analysene kunne ikke utføres før testmiljøet var ferdig. Hovedmålene kunne igjen deles opp i flere punkter:

- Sette opp et testmiljø for å fange data mellom applikasjonene og Internett.
- Se hva slags informasjon som blir utvekslet mellom Google og Android-enheten når man kobler til Internett.
- Finne ut om applikasjonene samler inn og sender vekk informasjon om brukerne av enheten.
- Finne ut av hva slags informasjon de sender.

Resultatmål

Prosjektet skal levere et testmiljø for å analysere Android-applikasjoner, sammen med en analyse av et sett utvalgte applikasjoner. Testmiljøet skal kunne brukes til å analysere alle typer applikasjoner på Android, samtidig som det er enkelt å sette opp. Analysene skal vise hvordan personvernet blir ivaretatt ved bruken av flere forskjellige applikasjoner på Android.

Effektmål

Det er ønskelig at prosjektet skal øke bevisstheten rundt personvernsspørsmål ved bruken av applikasjoner på Android-plattformen, ved å vise hvilken informasjon applikasjonene samler og sender vekk om brukerne og telefonen. Det skal også bli enklere å utføre analyser av flere applikasjoner ved å bruke testmiljøet som er utviklet.

Prosjektets målgruppe

Det er tre hovedmålgrupper for denne rapporten. Ønsket er at rapporten både skal være opplysende for brukere av smarttelefoner som ønsker å vite hvordan personvernet blir ivaretatt, samtidig som det viser utviklere og andre hvordan informasjon blir sendt vekk fra Android-enheter når applikasjonene brukes.

- Brukere av smarttelefoner som vil vite hvordan personvernet blir ivaretatt.
- Beslutningstagere(arbeidsgivere, politikere o.l.) som bruker smarttelefoner i jobbsammenheng, og som vil vite hvordan forretningskritisk informasjon kan bli kompromittert.
- Utviklere som vil vite hva som skjer når informasjonen blir kompromittert.

Formål

Gjennom prosjektet har det blitt utviklet et testmiljø for å gjennomføre analyser av applikasjoner på Android. Testmiljøet kan hente ut krypteringsnøklene som brukes i SSL-tilkoblingen, slik at den krypterte datatrafikken kan dekrypteres i Wireshark.

1.4 Rammer

Prosjektet skal være levert innen 23.05 kl. 12:00.

Kontaktpersoner

- Nils Kalstad Svendsen - Veileder
- Lasse Øverlier - Oppdragsgiver

1.5 Avgrensninger

Operativsystem

Oppdragsgiver ønsket at analysen skulle gjøres på Android-plattformen på grunn av dens store markedsandel, og det var hovedgrunnen til at denne plattformen ble valgt. Ettersom operativsystemet har åpen kildekode, og dermed kan modifiseres, er det også enklere å analysere siden det kan installeres verktøy som har tilgang på kernel-nivå i operativsystemet.

Applikasjoner

Det er kun applikasjoner som kjører på Android-plattformen som har blitt undersøkt. Undersøkelsene omhandler bare hva applikasjonene sender av data ut på Internett, siden vi mener dette gir de mest kritiske bruddene på personvernet. Applikasjoner som generer personlige data og lagrer dem på telefonen, for eksempel en

bankapplikasjon med kontoinormasjon, har ikke blitt ansett som alvorlige personvernsbrudd før dataene blir sendt til uvedkommende. Applikasjonene har ikke blitt omvendt utviklet, men protokollene de bruker for kommunikasjon har blitt der det har behøvdes.

Dataoverføringsteknologi

For å gjøre innsamlingen av datatrafikk mulig har det kun blitt samlet datatrafikk via Wi-Fi. For å sikre at data ikke blir sendt via mobilnettverket har undersøkelsene blitt gjort uten SIM-kort i enheten. Datatrafikken har blitt samlet inn fra det trådløse aksesspunktet som ble satt opp i testmiljøet.

Etikk

Prosjektet er hva som populært kalles "white hat hacking" [68]. Forsøkene går ut på å angripe Android og applikasjonene for å få tilgang til informasjon som utviklerne ikke vil at utenforstående skal få tak i, men i forskingsøyemed. Angrepene har blitt utført på vårt eget utstyr, og ingen tredjeparter har fått sin informasjonen kompromittert. Det er kun informasjon vi har kontroll over som har blitt brukt.

Lovlighet

Eksperimentene innebærer at kommunikasjonsprotokollene som brukes av applikasjonene og Android har måttet omvendt utvikles for å finne ut hva de inneholder.. Dette er ikke lov i følge flere av programmenes lisensavtale [26] [22], men det er fullt lovlig i følge norsk lov [42], slik vi tolker den.

Fra "Lov om opphavsrett til åndsverk:"
§**39h.** *"Den som har rett til å bruke et eksemplar av et datamaskinprogram kan, i forbindelse med slik lesning, fremvisning på skjerm, kjøring, overføring eller lagring av programmet som brukeren er berettiget til å utføre, iaktta, undersøke eller prøve ut hvordan programmet virker for å fastslå idéene og prinsippene som ligger til grunn for de enkelte deler av programmet."*

"Bestemmelsene i andre, tredje og fjerde ledd kan ikke fravikes ved avtale."

1.6 Organisering av rapporten

Rapporten er delt inn i seks hovedkaptiler, i tillegg til vedlegg.

1. **Introduksjon** - Dette kapittelet inneholder informasjon om selve oppgaven, og avgrensningene rundt den.
2. **Kravspesifikasjon** - Dette kapittelet handler om hvordan oppgaven har vært oppdelt under gjennomføringen.
3. **Metode for analysene** - Dette kapittelet forteller hvordan analysene i prosjektet var planlagt utført.

4. **Plattform og testmiljø** - Dette kapittelet inneholder informasjon om hvordan testmiljøet var planlagt og ble satt opp, sammen med hvilke problemer som oppstod underveis.

5. **Analyse** - Dette kapittelet forteller hvordan analysene ble gjennomført.

6. **Diskusjon** - Dette kapittelet oppsummerer resultatene av analysene og gjennomføringen av prosjektet.

7. **Vedlegg** - Her finnes kildekode, tillatelser for applikasjonene, data fra analysene og andre større vedlegg som ikke fikk plass i rapporten.

2 Kravspesifikasjon

2.1 Hovedinndeling av prosjektet

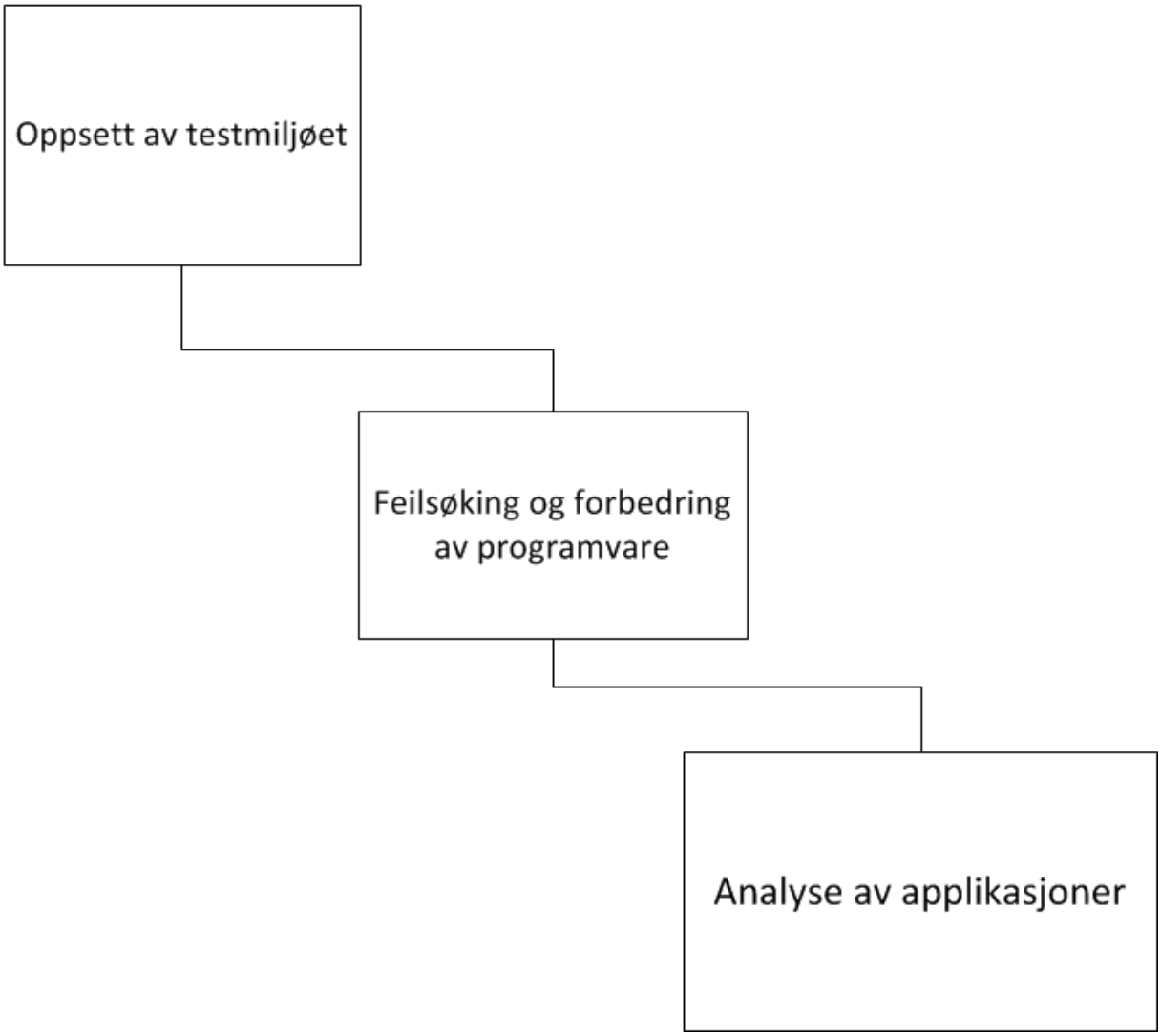
Det ble valgt en fossefallsmodell for prosjektet da det hovedsaklig har bestått av tre faser som er avhengig av hverandre, hvor den foregående fasen har måttet utføres før den neste kunne påbegynnes. I begynnelsen av prosjektet ble det antatt at inndelingen av prosjektet ville ligge nærmere en SCRUM-modell [67] hvor det ble jobbet med oppsett av testmiljø og analyse av applikasjoner i flere kortere faser, hvor det så ville bli gjort forbedringer og nye analyser etterhvert som kravene til testmiljøet endret seg med applikasjonene. I realiteten ble analysene mye likere hverandre, og oppsettet av testmiljøet tok mye lengre tid enn planlagt. Derfor ble det endret til en fossefallsmodell(ref. 1) hvor hver foregående fase måtte gjøres ferdig for å kunne fortsette til den neste.

Fasene har bestått av: oppsett av testmiljøet, forbedring av programvaren brukt i testmiljøet og gjennomføringen av analysene. Det viste seg at det måtte brukes mye tid på å modifisere og forbedre programvaren brukt i testmiljøet, og det resulterte i at dette ble en egen fase.

1. Den første fasen bestod i å sette opp testmiljøet med fysisk infrastruktur og installasjon av nødvendig programvare. Målet med fasen var å kunne koble en Android-enhet til et trådløst aksesspunkt for å fange opp datatrafikk fra den.

2. Den andre fasen gikk ut på å forbedre programvaren brukt i testmiljøet, og det innebærte feilsøking av tilkoblingsproblemer, fiksing av feilene og å legge til ny funksjonalitet i programvaren.

3. Den tredje fasen er selve analysen av de utvalgte applikasjonene. Her ble det samlet inn data om applikasjonene, og etterpå gjort en vurdering av hvordan de ivaretar personvernet til brukerne.



Figur 1: Utviklingsmodellen.

3 Metode for analysene

For å gjøre analysene har det krevdes et testmiljø som kunne koble en Android-enhet til et trådløst nettverk og ut mot Internett. Dette simulerer et vanlig oppsett for hvordan en Android-enhet blir koblet mot Internett og gjør innsamlingen av data så realistisk som mulig. Det var ønskelig å finne ut av hvordan applikasjonene oppførte seg i scenarier som var så like et virkelig bruksscenario som mulig, slik at resultatene ville vise hvordan personvernet ble ivaretatt ved bruk av applikasjonene på en måte som var realistisk.

Fordi mange applikasjoner bruker SSL for å kryptere datatrafikken måtte testmiljøet kunne brukes til å fange opp og dekryptere dette. Dermed måtte det utføres et man-in-the-middle-angrep mot tilkoblingen mellom applikasjonene og serverne de kommuniser med, slik at den krypterte trafikken kunne dekrypteres. Det betød igjen at testmiljøet trengte programvare for å gjøre et MITM-angrep og dekryptere dataene. Oppdragsgiver anbefalte oss å bruke sslsniff, siden dette programmet utfører et MITM-angrep og dekrypterer alle data som er kryptert med SSL.

3.1 Tilnærming

I begynnelsen av prosjektet ble sslsniff kjørt med koden slik den ble levert fra utvikleren, men det ble tidlig klart at den ikke hadde all funksjonaliteten som var nødvendig, og at det dermed måtte gjøres endringer i programmet for å legge til mer funksjonalitet. Spesielt var det loggføringen av dekrypterte data som ikke fungerte tilfredsstillende, siden sslsniff kun logget data som tekst, selv om det fantes binærdata. I binærdata trengs hele datastrømmen slik den sendes, for å lagre den riktig. Derfor ble det et mål å få endret sslsniff slik at data kunne dekrypteres i Wireshark, for å få tak i binærdata, samtidig som det ble enklere å holde oversikten i store datamengder.

4 Plattform og testmiljø

I denne seksjonen følger en beskrivelse av hvordan testmiljøet har blitt satt opp, sammen med hvilke problemer som har dukket opp underveis. Først er det en del definisjoner og begreper som er viktige for å forstå resten av rapporten.

4.1 Terminologi

Android

Android er Googles operativsystem for mobile enheter. Operativsystemet er basert på Linux-kjernen, men med Googles egne modifikasjoner [27, 41]. Kildekoden er fritt tilgjengelig via Android Open Source Project [18], men for å få fullt utbytte av operativsystemet trenger man proprietære drivere [20] og applikasjoner for å øke ytelsen og funksjonaliteten. Android er tett knyttet opp mot Google og gir brukerene tilgang til Googles tjenester [23], slik som Gmail, Google Calendar og Google Documents.

Android har muligheten til å kjøre programmer brukeren installerer og har gode muligheter for kommunikasjon over Internett. Operativsystemet har de siste årene blitt svært utbredt og er det største [50] og raskest voksende [15] operativsystemet for smarttelefoner. Dermed er det et interessant økosystem å analysere, siden størrelsen gjør at det tiltrekker seg både mobiltelefonprodusenter og utviklere som lager et utall forskjellige applikasjoner og varianter av Android, hvor både funksjonalitet og graden av personvern kan variere.

Applikasjoner

Brukerne har muligheten til å installere applikasjoner på Android, i tillegg til de som blir levert med enheten. Applikasjoner kan være spill eller programmer som øker funksjonaliteten til telefonen. Applikasjonene distribueres via Google Play(tidligere Android Market) [17], hvor brukerene kan laste ned og installere applikasjoner rett på telefonen, eller via APK-filer som brukerne selv legger over på enheten og installerer applikasjonene fra. Disse applikasjonene kan gis varierende grad av tilgang til informasjon via Androids tillatelser [24], men selv programmer med tilsynelatende uskyldige tillatelser kan få tilgang til mye sensitiv informasjon [33].

Rooting

Å roote en Android-enhet innebærer å skaffe rettighetene til root-brukeren slik at man får tilgang til hele filsystemet og muligheten til å kjøre programmer som trenger root-rettigheter [66]. Alle Android-enhetene som er brukt i prosjektet er rootet for å kunne installere egne SSL-sertifikater og kjøre Wireshark.

Rooting er et tema hvor lovligheten har vært diskutert, men hvor rettsavgjørelser har vedtatt at det er lovlig, både i følge amerikansk [65] og norsk lov, som beskrevet i avsnitt 1.5.

TLS/SSL

SSL [37] blir brukt til å kryptere nettverkstrafikk for å hindre at uvedkommende får tilgang til å lese den. SSL fungerer ved at klienten og serveren oppretter en sikker tilkobling til hverandre ved først å kommunisere via offentlig-nøkkel kryptografi, hvor de blir enige om en symmetrisk nøkkel som skal brukes for resten av kommunikasjonen. Klienten autentiserer serveren ved hjelp av det digitale sertifikatet den har, som er signert av en CA som klienten stoler på, for å verifisere at serveren er den den utgir seg for å være.

De fleste applikasjonene som er analysert bruker SSL for å kryptere kommunikasjonen mellom applikasjonen og serverne sine, og dermed må man først bryte denne krypteringen for å se hva som blir sendt mellom dem og Android-enheten.

Wireshark

Nettverkstrafikken ble fanget opp og analysert ved å bruke Wireshark. Programmet lytter på et nettverksgrensesnitt og tar opp all trafikken som blir sendt via det. Det har et kraftig brukergrensesnitt som lar brukeren se innholdet av hver enkelt datapakke, filtrere i trafikken, og for eksempel bare se kryptert trafikk. Datatrafikken kan også lagres som en .pcap-fil, slik at den kan undersøkes flere ganger, uten å måtte fanges opp på nytt. I tillegg kan SSL-trafikk dekrypteres, og da kan hele SSL-sesjonen følges for å se hva som blir sendt i datapakkene.

Wireshark kan også installeres som en applikasjon [43] på Android og fange opp datatrafikken på enheten. Da kan trafikk som går via mobilnettverket overvåkes sammen med Wi-Fi-trafikk. Wireshark er installert på alle enhetene, men har kun blitt brukt for å feilsøke nettverksproblemer, siden det er enklere å kjøre Wireshark på laptopen.

TaintDroid

TaintDroid [2] er et verktøy for å overvåke hva applikasjoner på en Android-enhet sender av sensitiv informasjon ut på Internett, og til hvem [1]. Dette har vært et essensielt verktøy i analysene, siden det forteller hvilken informasjon som blir aksessert på telefonen. Verktøyet leveres som et sett med moduler som kompileres inn i Android-kjernen, samt en applikasjon for å bruke TaintDroid, vist i figur 2.

En svakhet med TaintDroid er at programmet ikke oppdager mer sofistikerte forsøk på å stjele informasjon, slik som applikasjoner som tar bilder av skjermen. Da kan en applikasjon potensielt unngå å bli oppdaget av TaintDroid, samtidig som den klarer å stjele informasjon.

Figur 2: TaintDroid-applikasjonen gir mulighet til å starte og stoppe overvåking av datalekkasjer.

4.2 Testmiljøet

Før testmiljøet ble satt opp ble det brukt én uke på å undersøke lignende analyser, for å se hvordan de var utført. Ettersom fagfeltet er relativt nytt er det lite tilgjengelig informasjon om hvordan analyser gjennomføres på denne plattformen, men mange teknikker er overførbare fra andre typer digital etterforskning og nettverksovervåkning. Jon Oberheides eksperiment [53] og Blizzhackers guide [5], basert på Jon Oberheides testmiljø, forklarer hvordan et testmiljø som kan dekryptere SSL-trafikk på Android settes opp.

Utstyr

For å fange opp datatrafikken fra Android-enheten trengs et trådløst aksesspunkt den kan koble seg til. Det ble valgt å bruke et eksternt trådløst nettverkskort siden dette kan kobles rett i datamaskinen som brukes, og dermed trengs det ikke eksterne strømforsyninger eller lignende. Rekkevidden for nettverkskortet er heller ikke viktig, siden alle analysene har blitt utført i umiddelbar nærhet til nettverkskortet.

Oppdragsgiver gav oss tre forskjellige Android-enheter som analysene kunne gjøres på. To av enhetene kjørte forskjellige Android-versjoner, slik at forskjeller kunne sammenlignes mellom dem. Utstyret som ble brukt er vist i tabell 1.

Verktøy

Testmiljøet har blitt satt opp ved hjelp av forskjellig programvare som har gjort spesielle oppgaver. Det er disse som har gitt testmiljøet funksjonaliteten som trengs for

Enhet	Programvare	Annet
Samsung Galaxy Tab	Android 2.2	Rootet
HTC Desire HD	Android 2.2	Rootet
Google Nexus One	Android 2.3.4 m/TaintDroid	Rootet
Laptop	Arch Linux	Eksternt trådløst nettverkskort
Trådløst Nettverkskort	Linux RTL-WiFi-drivere	RTL8188CU-chipset

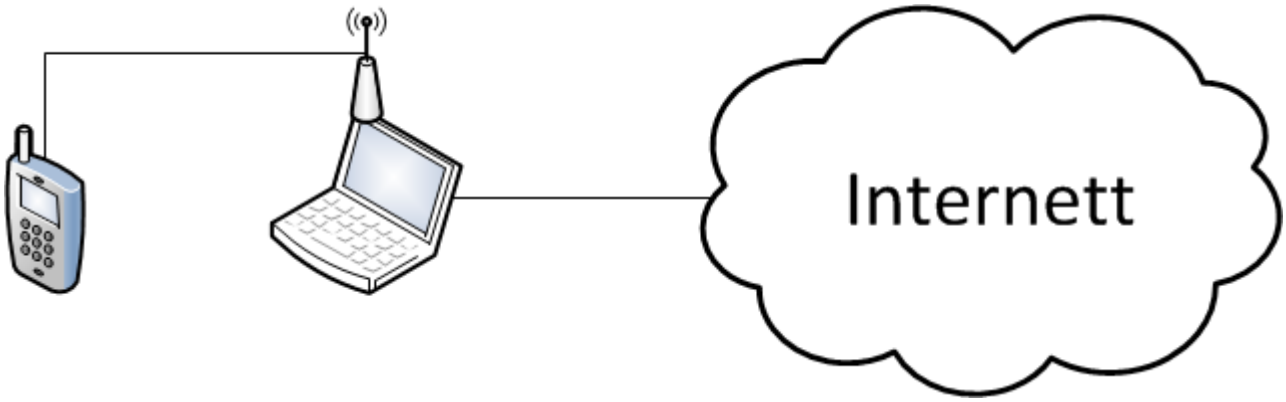
Tabell 1: Utstyr brukt i testmiljøet.

å gjøre analysene. Verktøyene kan deles inn i to hovedkategorier: programmene som brukes for å lage testmiljøet, og programmene som brukes for å gjøre analysene. Verktøyene har blitt valgt fordi de er det eneste av sitt slag, eller fordi de er ansett som industristandard for å løse en bestemt oppgave.

- TaintDroid [8] 2.3.4 - Dette er en modifisert Android-versjon som lar deg undersøke hvilke data andre applikasjoner sender ut på internett og til hvem.
- Wireshark [69] 1.6.7 - Dette programmet kan overvåke nettverkstrafikk og vise innholdet i den.
- Android SDK [19] - Inneholder flere verktøy for å lage og debugge applikasjoner på Android, samt overføre data til enheten.
- SuperOneClick [59] - Verktøy for å roote Android-enheter.
- sslsniff [48] 0.8 - Verktøy for å gjøre et MITM-angrep mot SSL-trafikk og dekryptere den.
- hostapd [47] 1.0 - Program for å sette opp et trådløst aksesspunkt i Linux.
- dnsmasq [39] 2.59 - DHCP-server og DNS-videresender for det trådløse aksesspunktet.
- Linux [45] kernel versjon 3 - Testmiljøet er satt opp på en Linux-maskin.
- OpenSSL [55] 1.0.0g - Samling av biblioteker og verktøy for å lage programmer som bruker SSL, og for å feilsøke SSL-tilkoblinger.

Oppsett av trådløst aksesspunkt

Den første delen av testmiljøet som ble satt opp var det trådløse aksesspunktet. En bærbar datamaskin som kjørte hostapd lagde et trådløst aksesspunkt med det nye nettverkskortet. På laptopen ble trafikken rutet fra nettverkskortet og ut på Internett, slik at Android-enheten hadde Internettilgang. Med dette oppsettet, vist i figur 3, kunne all trafikk som gikk mellom Android-enheten og Internett fanges opp i testmiljøet. Aksesspunktet var ustabilt under hele prosjektet, og den tilkoblede enheten måtte pinges kontinuerlig for å unngå at den koblet seg fra.



Figur 3: Oppsettet med det trådløse aksesspunktet.

Oppsett av sslsniff

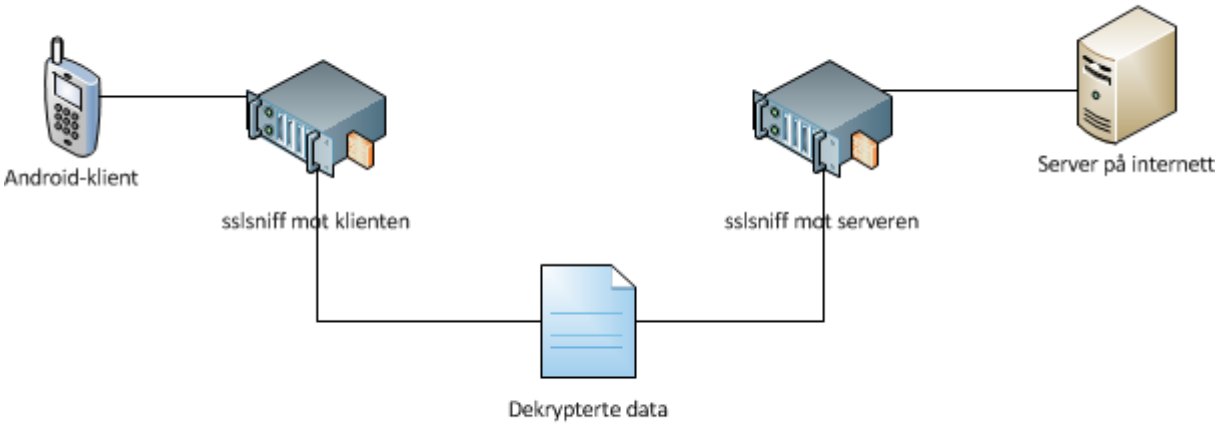
For å dekryptere SSL-trafikken ble sslsniff satt opp mellom Android-enheten og Internett. Programmet kan utføre et MITM-angrep og fungere som et mellomledd mellom mobiltelefonen og Internett, hvor all datatrafikk som er kryptert med SSL blir sendt fra telefonen og via sslsniff, som så dekrypterer den, før den krypterer den igjen og sender den videre ut på Internett. Det omvendte skjer med SSL-trafikk som kommer inn. Grunnen til at sslsniff får dekryptert trafikken er at den presenterer seg som serveren mobiltelefonen prøver å få tak i, og dermed tror mobiltelefonen at den snakker med serveren når den egentlig prater med sslsniff. Det samme skjer mot serveren hvor sslsniff presenterer seg som telefonen. Siden den krypterte tilkoblingen blir gjort mot sslsniff, og ikke serveren, kan sslsniff dekryptere datatrafikken. sslsniff oppretter så en kryptert tilkobling videre mot serveren, slik at kommunikasjonen foregår som normalt. Resultatet blir at sslsniff styrer den krypterte tilkoblingen mellom serveren og klienten, og kan få tilgang til all informasjonen som blir utvekslet, og alt som brukes for å kryptere den.

For å bruke sslsniff ble det genereret et eget CA-sertifikat som programmet bruker til å signere nye sertifikater med. George Notaras guide [52] ble brukt for å lage et CA-sertifikat. Dette sertifikatet ble lagt til i listen over CA-sertifikater som Android-enheten stoler på, og lar den dermed godta alle sertifikater som er signert av denne CAen.

Dekryptering i Wireshark

Ett av målene i prosjektet var å få dekryptert nettverktrafikken i Wireshark, slik at binærdata enkelt kan hentes ut av datatrafikken, samtidig som presentasjonen blir mer oversiktelig. Det ble diskutert flere idéer for å få til denne funksjonen, og flere løsninger ble forsøkt.

Det første alternativet var å laste inn krypteringsnøkkelen fra CA-sertifikatet i Wireshark. Under generering av sertifikatet blir det laget en dekrypteringsnøkkel som er knyttet opp mot sertifkatet. Det ble antatt at denne kunne brukes til å dekryptere SSL



Figur 4: Oppsettet av sslsniff med dekrypteringen.

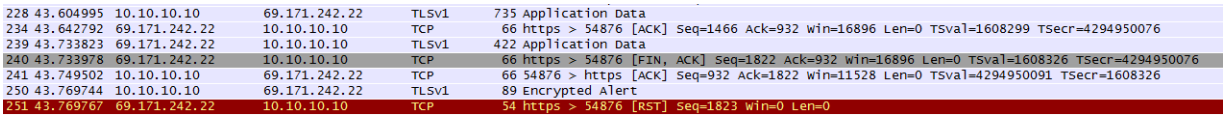
trafikken, men det viste seg å ikke være tilfellet. Ved nærmere undersøkelser ble det oppdaget at Diffie-Hellmann-nøkkelutvekslingen som ble brukt lager en ephemeral-nøkkel [63], det vil si en midlertidig nøkkel, for å tilby et ekstra lag med sikkerhet om privatnøkkelen til en av partene blir kompromittert [4]. Når denne brukes kan ikke SSL-trafikken dekrypteres på denne måten.

Løsningen ble å skrive om sslsniff for å få masternøklene som ble brukt i SSL-sesjonen, og dermed dekryptere SSL-trafikken med disse. Masternøkler fra en SSL-sesjon kan lastes rett inn i Wireshark og dekrypterer da all SSL-trafikk hvor disse har vært brukt. Endringene ble gjort i sslsniff slik at nøklene ble skrevet rett til en loggfil, og det ble lagd et Bash-skript(vedlegg C), som tolket denne filen og skrev ut nøklene på et format(vedlegg L) som Wireshark forstod. Denne løsningen gjorde at hver gang sslsniff opprettet en ny SSL-tilkobling ble det laget en fil med nøkler som kunne brukes til å dekryptere datatrafikken i Wireshark.

Testing av testmiljøet

For å sjekke om testmiljøet fungerte til å dekryptere nettverkstrafikk ble det kjørt tester med Facebook-applikasjonen. Testene bestod av å logge inn på Facebook via applikasjonen, fange opp datatrafikken og dekryptere den. Selv om trafikken ble dekryptert ble innholdet i applikasjonen aldri lastet, og dermed fungerte testmiljøet fortsatt ikke ordentlig. Feilen lå i sslsniff hvor headerne for HTTP-trafikk blir bygd, og versjonsnummeret ble satt til HTTP 1.0(se figur 5), selv om trafikken bruker versjon 1.1. Denne endringen gjorde at Facebook-applikasjonen fungerte og innholdet lastet. Feilen ble oppdaget da datatrafikken som gikk fra applikasjonen ble undersøkt med Wireshark. Ved å sammenligne trafikken som ble generert når sslsniff kjørte, mot den hvor sslsniff

Figur 5: Datapakke med HTTP 1.0.



Figur 6: RST-pakke fra Facebook.

ikke var aktiv, så man at det var unormalt mange RST-pakker i TCP-trafikken, vist i figur 6. Dette var på grunn av det overnevnte problemet med HTTP-versjonen, og med versjon 1.0 ble nye tilkoblinger avsluttet og opprettet hele tiden. En av hovedforskjellene mellom HTTP 1.0 og 1.1 er at kun 1.1 har støtte for vedvarende tilkoblinger [40]. Det betyr at serveren kan sende mer trafikk på et senere tidspunkt, uten å måtte opprette en ny tilkobling. I tillegg blir det mindre belastning på nettverket siden det ikke er nødvendig å sende pakker for oppretting og avslutning av tilkoblinger hver gang data skal sendes.

Problemer med sertifikater

Selv om problemene med HTTP-versjonen ble løst ga nettleseren i Android feilmeldinger om at navnet på sertifikatet ikke stemte med navnet på siden som ble besøkt. Det ble antatt at sslsniff kun fungerte ordentlig hvis alle sertifikater signert av vårt CA-sertifikat ble godtatt på Android-enheten. Alternativene da var å la programmet være slik det var, og prøve å ignorere feilmeldingene når de dukket opp, eller å fikse feilen med sertifikatet slik at det ble riktig. Alternativ en ble forsøkt, men applikasjoner lastet ofte ikke inn innholdet når det var problemer med sertifikatene. Dermed ble alternativ to valgt.

Først ble github-siden [49] for sslsniff ble undersøkt, og der fantes det en foreslått løsning [57] som lå til vurdering for integrasjon i sslsniff. Denne feilfiksen la til all informasjonen som fantes i det originale sertifikatet til det nye som ble generert av sslsniff. Ved å integrere denne feilfiksen i vår kode ble problemet med sertifikatene løst og feilmeldingene forsvant.

Sendte data blir ødelagt

Selv om feilmeldingene med sertifikatene ble løst gav applikasjonene fortsatt andre typer feilmeldinger. Alle var varierende utgaver om ødelagte data som ble sendt fra applikasjonene. Det ble valgt å feilsøke dette problemet videre, og dette avslørte at sslsniff hadde flere feil i seg som slo ut ved visse anledninger og gjorde at data sendt fra Android-enheten ble satt sammen feil igjen etter å ha blitt dekryptert.

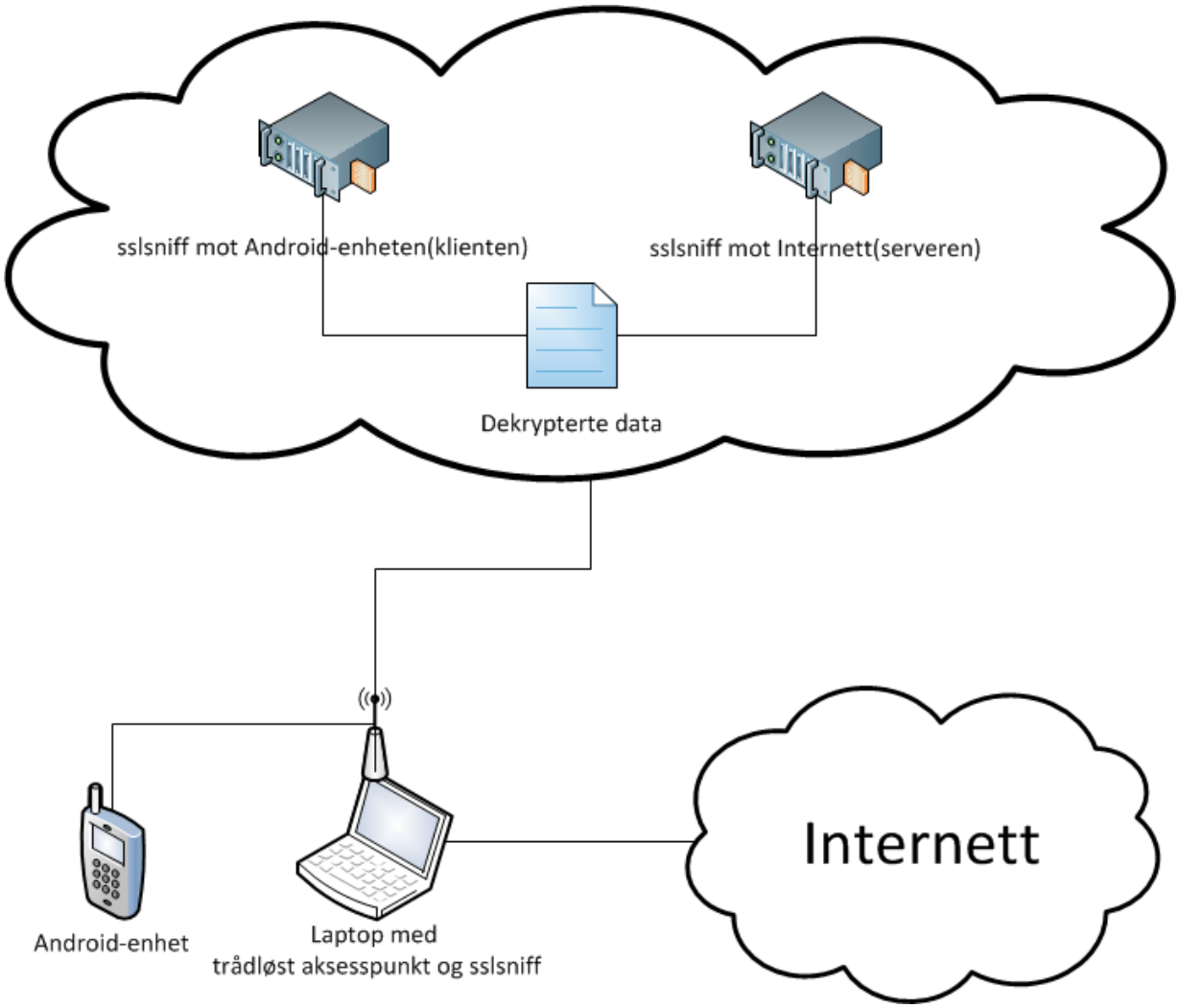
Ved å undersøke loggen med dekrypterte data fra sslsniff så man at en sendt forespørsel ble satt sammen flere ganger til den tilsvarte lengden av en fullverdig forespørsel. Når den ble sendt førte det til at feilmeldinger om ødelagte data ble returnert

tilbake til applikasjonen. Én teori var at dette kunne være forårsaket av problemer med bufferene, eller innlesing til bufferen, hvor for mye data ble lest inn. Det ble forsøkt å endre størrelsene på dem for å tillate at de leste inn mer data, men uansett hvor store de var så ble aldri resten av forespørslen satt sammen riktig. Ved å undersøke innholdet i bufferne kunne man se at det fra klienten kun ble lest 2048 bytes av gangen med ssl_read-funksjonen, og de innholdt kun den første delen av en forespørsel. Resten av dataene kommer senere, men da har sslsniff allerede prosessert den første delen og sendt den videre. Når resten kommer blir det prosessert på tilsvarende måte og sent av sted igjen. Ved å sammenligne med en vanlig nettside som bruker HTTPS, kunne man se at det var mindre data som ble lest inn med ssl_read(), typisk rundt 500 bytes. Til gjengjeld ble disse forespørslene prosessert og satt sammen riktig. Ut i fra forsøkene våre kan det set ut som om sslsniff oppfører seg unormalt når det som leses fra klienten er over en viss størrelse, og hvor det så blir prosessert feil og satt sammen igjen til ubrukelige data.

Det ble brukt mye tid på å feilsøke dette problemet, og forsøkt å finne løsninger ved å endre i koden til sslsniff slik at dataene ble satt sammen riktig. Konklusjonen ble til slutt at det ble vanskelig å endre måten programmet satte sammen dataene igjen, og derfor ble problemet løst ved å la sslsniff videresende alle mottatte data, uten å dekryptere eller modifisere dem(vedlegg B). Siden sslsniff var endret til å gi krypteringsnøklene som ble brukt i SSL-sesjonen var det ikke lenger nødvendig med funksjonaliteten som dekrypterte data, det behøvdes kun å sende den videre mellom serveren og klienten. Da ble alle data sendt korrekt, samtidig som vi hadde tilgang til krypteringsnøklene og kunne dekryptere datatrafikken i Wireshark. Denne endringen gjorde at testmiljøet fungerte tilfredsstillende og kunne brukes til å analysere alle applikasjonene. Testmiljøet ble seende ut som vist i figur 7.

Konklusjon om testmiljøet

Mesteparten av tiden som ble brukt på å sette opp testmiljøet gikk med til å feilsøke sslsniff og rette de feilene som ble oppdaget. Feilsøkingen har tidvis vært komplisert, og det har vært mange teorier om hva som er galt, men som senere har vist seg å være uriktige. Mye av feilsøkingen har blitt gjort ved å endre kildekoden for å fjerne feil som har blitt funnet, og for å eksperimentere med forskjellige innstillinger og nye funksjoner. All feilsøkingen og eksperimenteringen har gitt mye kunnskap om sslsniff og hvordan programmet fungerer, og denne kunnskapen har vært nødvendig for å finne løsningene som har vært gjort på problemene. Siden programmet har manglet dokumentasjon har mye tid blitt brukt på å gjenoppdage og finne ut av hvordan det fungerer. Det ville uten tvil spart mye tid i prosjektet om sslsniff hadde vært bedre dokumentert.



Figur 7: Det ferdige testmiljøet.

5 Analyse

Målet med å analysere applikasjonene har vært å undersøke hvordan de ivaretar brukerens personvern, samt å kartlegge hvilke brudd på personvernet som eventuelt foregår. Applikasjonene har blitt analysert ved å undersøke hver applikasjons datatrafikk for å se hva de har sendt ut på Internett og tilbake til utviklere eller andre tredjeparter. I tillegg har TaintDroid blitt brukt for å se hvilken sensitiv informasjon som blir aksessert og sendt. Disse har sammen blitt brukt for å finne ut av hvilke personsensitive opplysninger applikasjonene deler.

Innsamling av data

Alle data som er brukt i analysen har blitt samlet inn ved å bruke Wireshark og sslsniff, sammen med TaintDroid. Under datainnsamlingen har det blitt utført tre forsøk hvor det samme bruksmønsteret har blitt gjentatt i applikasjonen, samtidig som Wireshark logger datatrafikken. TaintDroid har blitt kjørt på Google Nexus One-enheten for å se hvilke personsensitive data som har blitt brukt av hver applikasjon. Analysene har naturlig nok vært forskjellige for hver applikasjon, men hver analyse har bestått av tre generelle faser:

- Innlogging og visning av startskjerm.
- Applikasjonsspesifikk bruk.
- Utlogging og avslutning av applikasjonen.

I hvert forsøk har hovedfunksjonaliteten for hver applikasjon blitt brukt, for å simulere et vanlig bruksscenario. Dette kan være å sende en melding til en annen bruker, eller laste ned en ny applikasjon. Det har ikke blitt gjort undersøkelser av all funksjonaliteten, men blitt begrenset til det som er det essensielle for applikasjonen. Ved å bruke applikasjonene har det blitt generert nettverkstrafikk som har blitt fanget opp og analysert.

I analysene har disse peronssensitive dataene vært spesielt interessante å se etter, fordi de kan inneholde mye sensitiv informasjon, eller brukes til å unikt identifisere brukerne:

- IMEI
- Telefonnummer
- Kontaktliste
- Lokasjon (GPS-koordinater eller navn på nærliggende trådløse nettverk).
- Lagrede data på minnekort o.l.
- SMS, MMS, epost o.l.

Analyse av data

Alle innsamlede data har blitt analysert ved bruk av Wireshark. Datatrafikken har blitt dekryptert ved mindre noe annet er nevnt. Målet med å kunne dekryptere all SSL-trafikk i Wireshark var å forenkle analysen. Datatrafikken inneholder mange binærfiler som må hentes ut og undersøkes, for å finne ut hva de inneholder.

Vektlegging i analysen

I analysene er det lagt spesielt vekt på applikasjoner som henter brukernes informasjon, uten at brukerne har initiert en funksjon i applikasjonen som bruker denne informasjonen. Det betyr i praksis at det har blitt ansett som mer alvorlig hvis en applikasjon samler opp informasjonen om hvor en bruker befinner seg mens applikasjonen kjører i bakgrunnen, enn at den blir samlet opp hvis brukeren bruker en funksjon i applikasjonen som spør om å få vite brukerens plassering.

Konklusjonen for hver applikasjon forteller noe om hvorvidt vi synes informasjonen applikasjonen henter ut er rimelig med tanke på hva applikasjonen kan bruke den til, og hva informasjonen er. Mye er identifiserende informasjon, som unikt forteller hvem brukeren eller enheten er, og dermed kan den brukes i målrettet reklame o.l. Det har blitt ansett av oss som relativt sett ufarlig, siden det inneholder lite personlig identifiserende informasjon. Det vil si at vi har konkludert med at en applikasjon ivaretar personvernet til brukeren hvis informasjonen kun er identifiserende for enheten eller brukeren, men ikke hvis den innebefatter annen informasjon, eksempelvis lokasjon eller kontaktliste.

Analysenes kompletthet

Alle forsøkene har blitt utført tre ganger. Dette er for å sikre et sammenligningsgrunnlag for analysene og dermed muligheten til å identifisere potensielle avvik i de innsamlede data. Vi mener tre forsøk er nok for å kunne anta at de innsamlede data lar seg reprodusere ved gjenntagelse av forsøket.

Presentasjon av funn

Hver applikasjon er presentert individuelt med tilhørende informasjon om applikasjonen og analysen som er gjort. Presentasjonen fokuserer på beskrivelsen av applikasjonen, hvilke rettigheter den har og hvilke funn som er gjort i analysene. Spesielt interessant er informasjonen som har blitt sendt over Internett.

Brukerstatistikkverktøyet Flurry Analytics

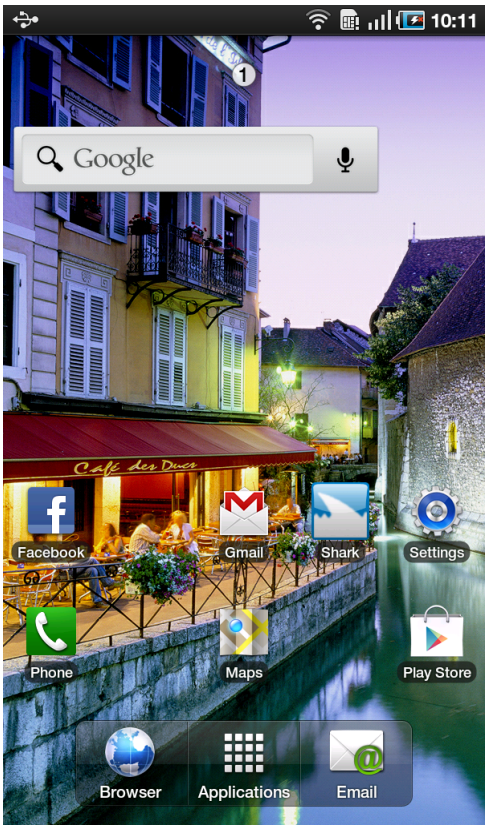
Tre av de analyserte applikasjonene har brukerstatistikkverktøyet Flurry Analytics innebygd. Dette er et sett med biblioteker for å samle inn data om hvordan applikasjonen blir brukt, slik at utviklerne kan bruke dataene for å forbedre applikasjonen. Flurry påstår de innsamlede dataene ikke kan knyttes til et enkelt individ [13], og at bruken av informasjonen er eksplisitt definert. Flurry blir brukt av over 60 000 bedrifter i mer enn

150 000 applikasjoner [12]. Informasjon om brukeren, telefonen og tilhørende maskinvare blir samlet inn og brukt i målrettet reklame [11, 14]. Verktøyet sender statistikken ukryptert tilbake til Flurry, slik at noen som lytter på nettverkstrafikken kan fange den opp.

Oppdateringsfrekvens for applikasjonene

Applikasjoner på Android oppdateres svært hyppig, og flere av applikasjonene ble oppdatert flere ganger i løpet av perioden vi analyserte dem. Trolig er de fleste oppdateringene mindre endringer og feilfikser, men det kan også være større endringer som forandrer hvilke data applikasjonene samler inn. Derfor kan resultatene av analysene være forskjellige fra hvordan applikasjonene fungerer i dag.

5.1 Androids kommunikasjon med Google



En Android-enhet gjør en såkalt "check in" når den kobles til Internett for første gang. Her sender Android en HTTP POST-request med en Protocol Buffer til Google med forskjellig informasjon om enheten. I våre forsøk var dette en tekstfil på ca. 140KB bestående av informasjon om versjonen av operativsystemet som ble kjørt, sammen med en logg over hva som har skjedd med systemet siden Android ble installert.

Protocol Buffers

Protocol Buffers [25] er Googles dataformat for å kode og stukturere data. I følge Google brukes det til nesten alt av interne RPC og filformater.

Loggfilen

Ut i fra hvor mye informasjon som må sendes, blir det sendt en eller flere Protocol Buffers til Google (se vedlegg J), som inneholder alle større endringer som har blitt gjort med systemet. Dette er ting som gjenoppretting av fabrikkinstillinger på telefonen, feilmeldinger om dårlige sektorer i filsystemet og størrelsen på diskpartisjonene. Som svar får telefonen en Protocol Buffer fra Google som inneholder ulike variable, blant annet Android-ID, instillinger for Google sine tjenester, samt lengden på inter-

```
Process: com.google.process.gapps
Flags: 0x8be45
Package: com.google.android.gsf v10 (2.3.3)
Build: generic/full_passion/passion:2.3.4/GRJ22/eng.taintdroid
    .20120207.104554:eng/test-keys
System-App: true
Uptime-Millis: 471375
Loop-Violation-Number: 5
Duration-Millis: 245

android.os.StrictMode$StrictModeDiskWriteViolation: policy=135
    violation=1
        at android.os.StrictMode$AndroidBlockGuardPolicy.
           onWriteToDisk(StrictMode.java:732)
```

Figur 8: Et utsnitt av en feilmelding i loggfilen.

vallet mellom hver "check in" som skal utføres. Se vedlegg K for et eksempel på en slik Protocol Buffer.

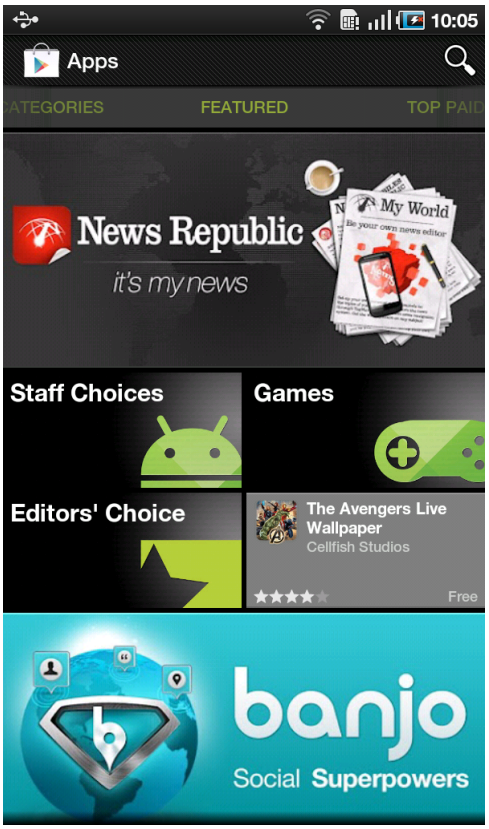
Eksempel på feilmelding

Et eksempel på en feilmelding som finnes i loggfilen vises i figur 8. Den viser en krasjrapport og stack trace fra en prosess. Feilmeldingen er avkortet for å spare plass, men ligger vedlagt i J.

Nytten av informasjonen

Det er uvisst hva Google skal med denne informasjonen, men vi antar det brukes for å føre statistikk og samle feilrapporter. Analysene våre har ikke avdekket noe personlig informasjon i denne loggfilen, men den inneholder flere parametere som dreier seg om telefonen og dens egenskaper og tilstand. Informasjonen kan trolig si noe om hvilke applikasjoner brukeren har brukt og har kjørende, og hvor ofte visse handlinger blir utført.

5.2 Google Play



Navn: Google Play
Versjon: 3.5.16
Utviklet av: Google Inc.

Beskrivelse

Google Play er Googles distribusjonssystem for applikasjoner til Android-enheter. Det brukes for å finne, laste ned og installere applikasjoner på Android.

Forventede resultater av anaylse

Android Market (nå Google Play) har tidligere blitt analysert av Jon Oberheide. Forsøkene hans har vist hvordan Android Market installerer applikasjoner på Android via kommandoer fra Google. Vi forventer at applikasjonen fortsatt fungerer på samme måte.

Det er antatt at applikasjonen sender fra seg mye informasjon om enheten for å vite hvilke applikasjoner som er kompatible med den. Dette er derimot den eneste informasjonen vi mener applikasjonen har bruk for å fungere.

CuECCosCRFFBQUFNQUFBQUFsZl81bVNhU3JmTl9ZWDNsODVqeE1
YRjRRTG9uZlpjbnRDNGFKQmx0NzEwbVVkU055bDAwekxkWWotaT
JTb0QwaDE3TG83THppcTlCUGYybE84eWNGT0w4dzJOcmEwN2xpS
nNvN1RycGEya1MtV0czUkZWVGl2TDM0OHRvb1Z4MjZkQjR2YmM3
aXhQNC1ZOV9fbVhNejBNUlZXZ0hfMll5bzBtcXQ4MnZCTC1EdEZ
rOWZoY1Z0QnU2UXNITTBSREhsYUhhU2pjdzY3QngtaTVTa1pYNm
VxSFpXR0ZBRzFOR0EwLTdmVDNXdi03UWlPaDB6RWJQZ2lYRVhUX
zJrdzE0EAEYiProAyIQMzQwZWYyNWU3Y2Y0N2I1ZSoKcGFzc2lv
bjoxMDICZW46AlVTQgBKAFIAWgBiCmFtLXVua25vd25qEDI5NGU
xMDM3MzQwYThhOTcTmgGvFBL0EwgDEAEYAyACKAAwATjwAUCAgA
hKE2FuZHJvaWQudGVzdC5ydW5uZXJKCmphdmF4Lm9iZXhKIGNvb
S5hbmRyb2lkLmZ1dHVyZS51c2IuYWNjZXNzb3J5Shdjb20uZ29v
Z2xlLmFuZHJvaWQubWFwc0odY29tLmFuZHJvaWQubG9jYXRpb24
ucHJvdmlkZXJSFWFuZHJvaWQuaG

Figur 9: En del av en Base64-kodet Protocol Buffer.

Figur 10: En del av en dekodet protobuffer.

Hvilke sensitive data sender applikasjonen over Internett og hvem blir de sendt til?

I datatrafikken blir det sendt en Protocol Buffer tilbake til Google med generell informasjon om Android-enhetens egenskaper. I figur 9 kan man se et utsnitt av en Protocol Buffer som er kodet i Base64.

Den dekodede delen vises i figur 10. Den første delen av denne dekodede Protocol Bufferen inneholder ulike variable som kan være vanskelige å dekode uten å ha skjemaet den ble opprettet med. Allikevel er det mulig å tyde navnet på telefonen og språket, med mer.

```
com.android.location.provider
android.hardware.wifi
android.hardware.location.network
com.google.android.feature.GOOGLE_BUILDR
android.hardware.telephony
android.hardware.location
android.software.sip
android.hardware.touchscreen.multitouch
android.hardware.sensor.compass
android.hardware.camera
android.hardware.usb.accessory
android.hardware.bluetooth
```

Figur 11: Et utsnitt av funksjonaliteten på Google Nexus One-telefonen.

Det sendes og en liste som inneholder all funksjonaliteten på telefonen tilbake til Google. Eksempelvis kamera, akselerometer, GPS osv. Listen er vist i figur 11.

Som vist i figur 12 viser TaintDroid at Google Play sender fra seg denne informasjonen til Google:

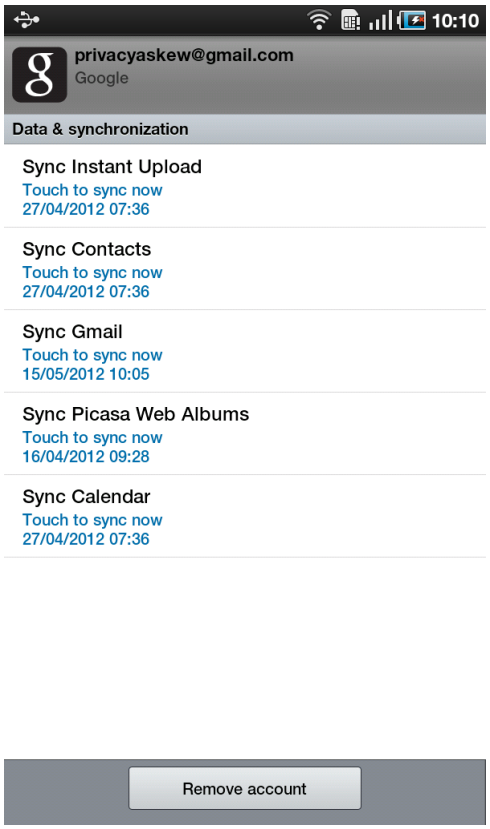
- Lokasjon
- Telefonnummer
- Lokasjon funnet via trådløse nettverk i nærheten
- Kamera

Ivaretas brukerens personvern?

Applikasjonen sender fra seg den informasjonen vi forventet om funksjonaliteten på enheten. Derimot mener vi at informasjonen om lokasjon og telefonnummer ikke er nødvendig for bruken av applikasjonen, siden vi ikke ser hvordan det kan være nødvendig for at den skal fungere. Dermed samler applikasjonen mer informasjon enn det vi synes er nødvendig, og bryter derfor med brukerens personvern ved å fortelle Google om vedkommendes lokasjon.

Figur 12: TaintDroid som har gitt utslag på Google Play.

5.3 Google Sync



Navn: Google Sync
Versjon: 2.2 og 2.3
Utviklet av: Google

Beskrivelse

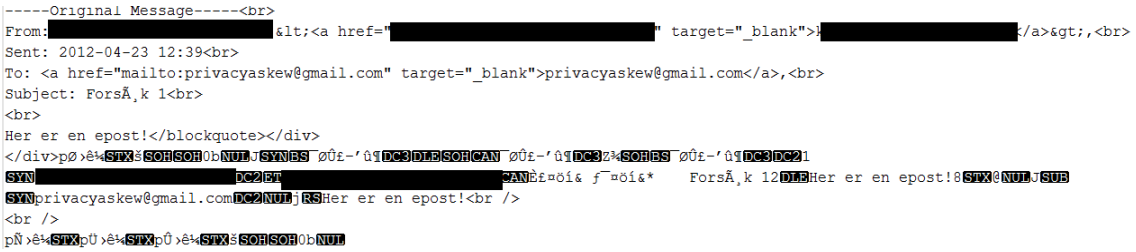
Google Sync er en innebygd funksjon i Android som synkroniserer data på Android-enheten med Googles tjenester. Dette inkluderer synkronisering av epost, kalender og kontaktliste.

Forventede resultater av anaylse

Fordi essensen av denne funksjonen er å sende personlig informasjon til Google er det naturlig å anta at det er akkurat det som skjer. All informasjonen på enheten som er knyttet opp mot Googles tjenester blir trolig overført.

Hvilke sensitive data sender applikasjonen over Internett og hvem blir de sendt til?

Ved å bruke synkroniseringsfunksjonen blir det sendt en forespørsel om å få tilsendt alle epostene fra Gmail-serveren. Alle epostene blir så sendt tilbake i en XML-fil(ref.



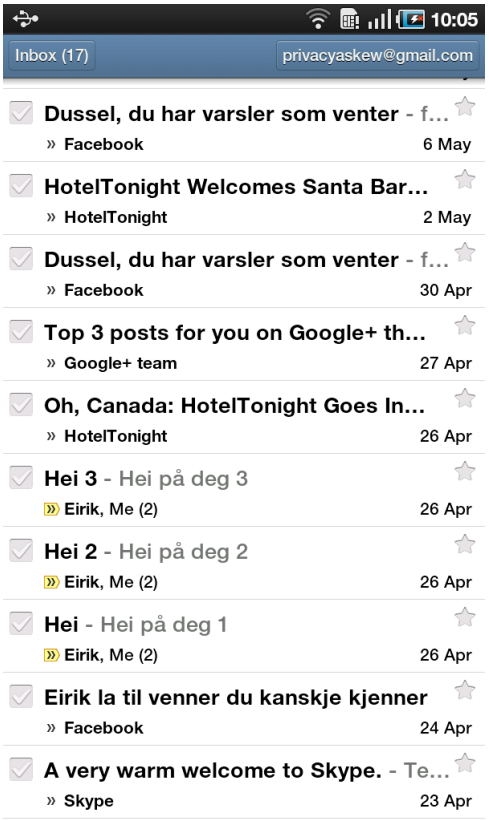
Figur 13: Utdrag fra et dekryptert XML-dokument inneholdende epost.

fig. 13), som ble hentet ut fra datatrafikken med Wireshark. Ingen annen informasjon blir sendt, trolig fordi det ikke var lagret informasjon på Android-enheten om annet enn eposten. TaintDroid ga ingen utslag om sensitiv informasjon som ble aksessert eller overført.

Ivaretas brukerens personvern?

Denne funksjonen synkroniserer kun informasjonen brukeren har bedt om, og ingen annen informasjon har blitt overført i eksperimentet. Derfor er det rimelig å påstå at denne applikasjonen fullt ut ivaretar personvernet til brukeren, ved å kun overføre informasjonen som trengs for at den kan fungere.

5.4 Gmail



Navn: Gmail
Versjon: 2.3.6
Utviklet av: Google Inc.
Antall installasjoner: 100 000 000 - 500 000 000

Beskrivelse

Gmail er Googles applikasjon for å få tilgang til epost-tjenesten deres. Applikasjonen fungerer som en ordinær epost-klient som kan lese og sende epost.

Tillatelser på Android

- Bruk autentiseringsopplysningene for en konto
- Full tilgang til Internett
- Les kontaktdata
- Skriv kontaktdata

- Endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort
- Kjøre automatisk ved oppstart

Se vedlegg H for resten av tillatelsene.

Forventede resultater av anaylsen

Siden Gmail-applikasjonen håndterer synkronisering av epost mellom Android-enheten og epostserverne til Gmail er det naturlig å anta at det kun er epost som blir sendt tilbake til Google.

Trenger applikasjonen de tillatelsene den ber om?

Alle rettighetene Gmail-applikasjonen ber om er relevante for en epostklient. Det er ingenting som tyder på at den ber om flere tillatelser enn den trenger for å fungere.

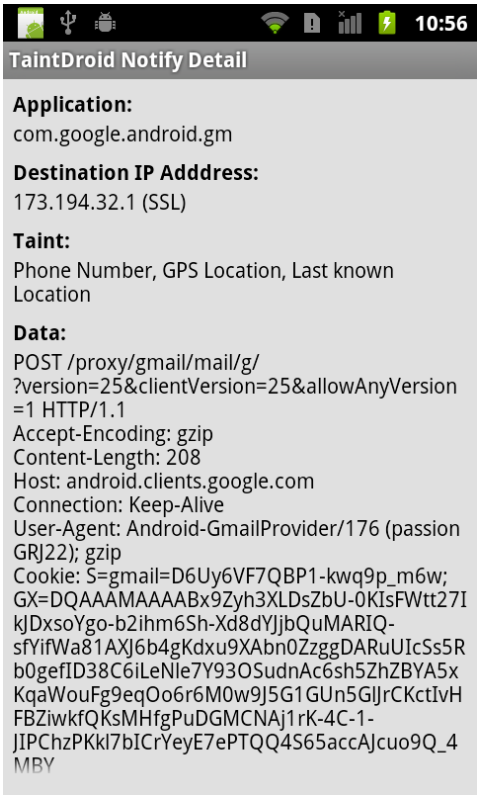
Hvilke sensitive data sender applikasjonen over Internett og hvem blir de sendt til?

Ved å undersøke datatrafikken ser man at Gmail-applikasjonen sender en gzip-fil til Google. Dataene i denne gzip-filen er i XML-format lik figur 13 og inneholder sendte og mottatte eposter, samt kontaktinformasjon. I figur 14 forteller TaintDroid at applikasjonen sender fra seg følgende informasjon:

- Telefonnummer
- GPS-lokasjon
- Sist kjente posisjon

Ivaretas brukerens personvern?

Dataene som sendes tilbake til Google, i følge TaintDroid, er ikke essensielle for bruken av applikasjonen, og de brukes trolig for unik identifisering og lokalisering av brukeren. Brukerens lokasjon burde ikke være nødvendig for at en epost-applikasjon skal fungere, og innsamlingen av denne informasjonen fremstår som unødvendig for bruken av applikasjonen. Dermed vil vi påstå at denne applikasjonen ikke ivaretar brukerens personvern, siden den samler inn mer personlig informasjon enn det som trengs for å fungere.



Figur 14: TaintDroid viser informasjon som sendes til Google.

5.5 Facebook

Navn: Facebook for Android
Versjon: 1.8.4
Utviklet av: Facebook
Antall installasjoner: 100 000 000 - 500 000 000

Beskrivelse

Facebook har en egen applikasjon som brukes til å få tilgang til Facebook fra en Android-enhet. Applikasjonen har samme funksjonalitet som Facebooks nettsider, men er tilpasset skjermstørrlesen og brukergrensesnittet på mobile enheter. Applikasjonen trenger flere forskjellige tillatelser for å fungere, slik som tilgang til Internett og muligheten for å bruke GPS [10]. Nylig har applikasjonen også fått muligheten til å lese SMS og MMS fra telefonen [6], uten at alle brukerne har tilgang til funksjonalitet som trenger denne tillatelsen enda.

Tillatelser på Android

- Ta bilder og videoer
- Ta opp lyd

- Detaljert (GPS) posisjon
- Motta data fra Internett
- Les kontaktdata
- Skriv kontaktdata
- Endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort

Se vedlegg D for resten av tillatelsene.

Forventede resultater av anaylse

Siden Facebook-applikasjonen har tilgang til mange tillatelser på Android, er spekteret av informasjon applikasjonen kan få tak i stort. Både data som ligger lagret på minnekortet og informasjonen om kontaktene kan hentes ut av applikasjonen, og dermed kan vi anta at begge disse tillatelsene også brukes. Det vi forventer å finne er synkronisering mellom kontaktlisten på mobiltelefonen og Facebook, slik at man kan knytte kontaktinformasjonen sammen. Det er ingen grunn til å tro at applikasjonen stjeler informasjonen som finnes på minnekortet, men trolig trengs tilgangen slik at brukerne kan laste opp video og bilder fra minnekortet til Facebook. Applikasjonen har også muligheten til å samle inn mye informasjon om brukeren siden man inviteres til å dele informasjon med Facebook kontinuerlig. Brukere kan benytte "sjekk inn"-funksjonen på Facebook for å fortelle hvor de befinner seg. Da kan GPS-sensoren benyttes for å finne brukerens posisjon.

Trolig er mye av den personsensitive informasjonen som sendes til Facebook avhengig av at brukeren har samtykket til at den sendes, men det er også muligheter for at Facebook samler inn informasjon uten at brukeren har aktivt bedt om det.

Trenger applikasjonen de rettighetene den ber om?

Kontaktlistetillatelsene benyttes for å integrere kontaktlisten på telefonen mot Facebook, mest sannsynlig for å foreslå nye venner å legge til, eller for å synkronisere informasjonen. Applikasjonen har mulighet til å ta bilder og lyd, og ber derfor om de rettighetene. Applikasjonen kan og å laste opp bilder, og da kreves tillatelsen for å lese minnekortet.

{"session_key":"c8e143f79abc45207c00914b.0-100003514246356",
"uid":100003514246356,"secret":"a071f4e77c88bea624e576ae035ba622"
,"access_token":"BAAAAUaZA8jlABADO6ejUfWUCrXlj9TZAn29ePj4CQKkFK7
vnwqsfVWGKzBAOzr9Cv9hjfrwOxMWIVaRZBBFTdpV9WH6SVXc5k45TzCEH
wZDZD","machine_id":"2_piT0J0O1s9W8UmSv5OM8yU"}

Figur 15: Tekststreng med nøkler, token og IDer.

User-Agent: \[FBAN\/FB4A;FBAV\/1.8.3;FBDM\/\{density\=1.5,width\=
480,height\=800\};FBLC\/en_US;FBCR\/;FBPN\/com.facebook.katana;
FBDV\/Full Android on Passion;FBSV\/2.3.4;]

Figur 16: Tekststreng med informasjon om enheten.

Hvilke sensitive data sender applikasjonen over Internett og hvem blir de sendt til?

For analysen har det blitt gjennomført tre forskjellige forsøk som har undersøkt forskjellige funksjoner i applikasjonen:

- Pålogging i applikasjonen.
- Pålogging i applikasjonen, visning av en annen brukers profil og posting på vår egen vegg.
- Kjøring av applikasjonen i bakgrunnen, mens enheten ikke brukes.

Undersøkelsene ble først foretatt ved å analysere datatrafikken som ble sendt til Facebook, for å se hva den inneholder. Innlogging skjer via HTTPS, og når brukeren logger inn på Facebook-applikasjonen blir brukernavn og passord (se figur 17), og en tekststreng med informasjon om enheten sendt til Facebook. I figur 16 kan man se en slik streng med informasjon.

Strengen som er vist i figur 15 inneholder versjonen til applikasjonen, skjermoppløsning, språk, serveren den ønsker å koble til, telefonmodell og Android-versjonsnummer. Tilbake fra serveren får klienten en session key, bruker-ID, hemmelig nøkkel, access token og maskin-ID. Maskin-IDen er forskjellig for hver gang man logger inn på applikasjonen.

Selv om innloggingen på applikasjonen går over HTTPS og dermed er kryptert, så skjer ikke det samme med bilder. De blir sendt ukryptert over vanlig HTTP, og dermed vil noen som fanger opp datatrafikken kunne se hvilke bilder en bruker ser på.

Når sider besøkes i applikasjonen sendes det en lang streng med informasjon for å få tak i siden som skal vises. Denne informasjonen er kodet, men vi har ikke funnet noen måte å dekode den på. Strengen er som vist i 18.

Figur 17: Inlogging på Facebook med applikasjonen. Passord markert med rødt. Maskin-ID markert med gult.

Derfor ble TaintDroid brukt for å følge med på hvilken informasjon som ble sendt fra telefonen og tilbake til Facebook. Som vist i figur 19 ble følgende informasjon sendt:

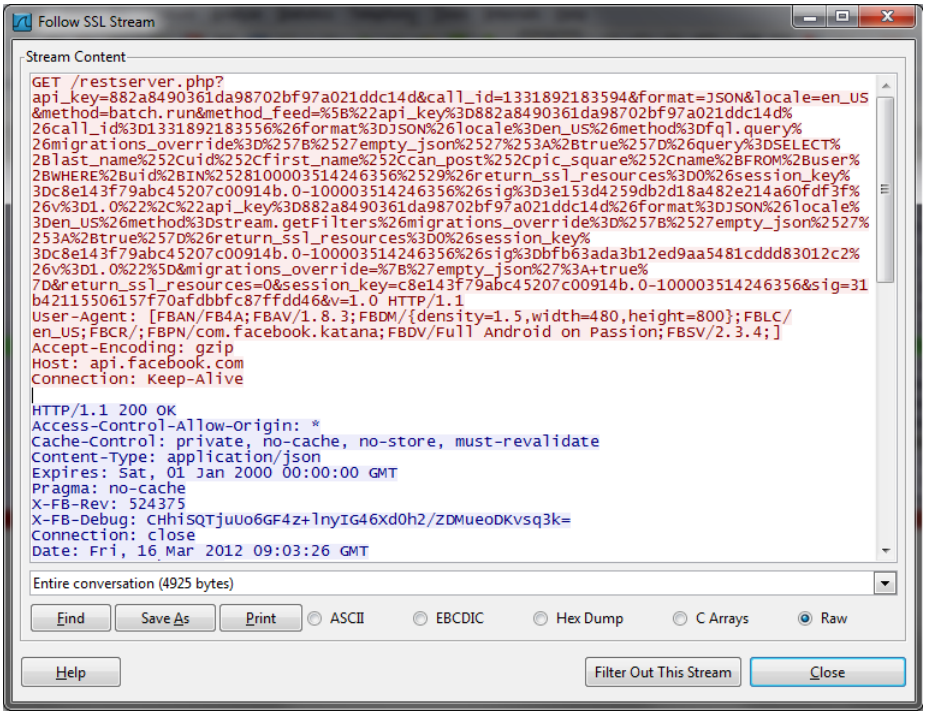
- Telefonnummer
- IMEI
- Sist kjente posisjon(ref. fig. 19(b))
- Kamera
- Akselerometer
- Det innskrevne passordet

Hva skal applikasjonen med informasjonen?

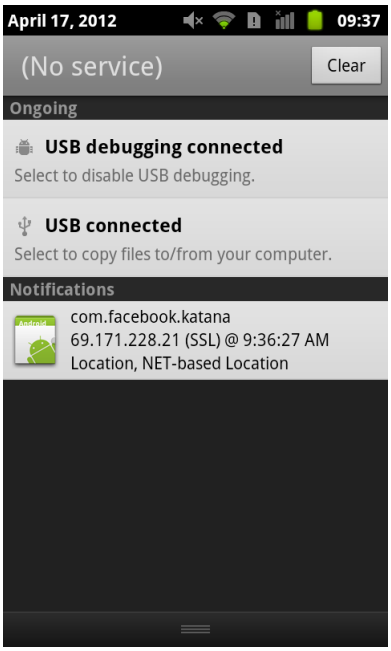
Facebook kan bruke informasjonen til å identifisere enheten brukeren logger seg på, og se hvor brukeren har vært. Trolig brukes dette for en form for statistikkføring, for å se hvor brukerne befinner seg.

Ivaretas brukerens personvern?

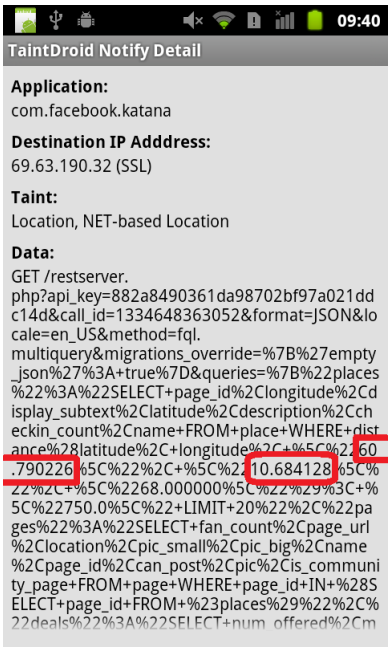
Vi vil påstå at denne applikasjonen stort sett ivaretar personvernet til brukerne, men applikasjonen bør ikke ha behov for å vite brukerens sist kjente posisjon eller telefonnummer. Sikkerheten til applikasjonen forbedres ved å bruke HTTPS for all dataoverføring, ikke bare innlogging.



Figur 18: Innhold som ikke har latt seg dekode.



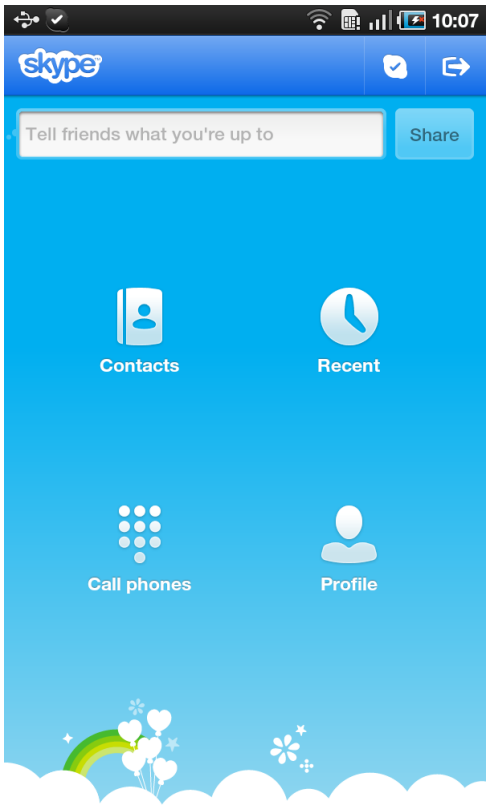
(a) Data Facebook-applikasjonen sender vekk i følge TaintDroid.



(b) Lengde- og breddegrader som Facebook har hentet ut.

Figur 19: Informasjon som TaintDroid har oppdaget, vises i figur (a) og (b).

5.6 Skype



Navn: Skype
Versjon: 2.7.0.907
Utviklet av: Skype
Antall installasjoner: 50 000 000 - 100 000 000

Beskrivelse

Skype er en applikasjon for å få tilgang til Skypes tjenester fra en Android-enhet. Applikasjonen kan ringe og sende meldinger til andre Skype-brukere over Internett. Det er ett av de mest populære VoIP-programmene på Android og har en stor brukermasse.

Tillatelser på Android

- Bruke autentiseringsopplysningene for en konto
- Ringe telefonnumre direkte
- Ta bilder og videoer
- Ta opp lyd

- Les kontaktdata
- Skriv kontaktdata
- Lese telefontilstand og -identitet
- Endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort
- Endre globale systeminnstillinger
- Send fast kringkasting

Se vedlegg F for resten av tillatelsene.

Forventede resultater av anaylse

Skype er kjent for høy sikkerhet [34], og det ble antatt da undersøkelsen ble påbegynt at det ville være et ekstra lag med sikkerhet i tillegg til SSL-protokollen som brukes. Dermed kan dataene som sendes fortsatt ikke leses, selv om SSL-trafikken er dekryptert.

Trenger applikasjonen de tillatelsene den ber om?

Vi mener Skype behøver alle tillatelsene den ber om for å kunne fungere.

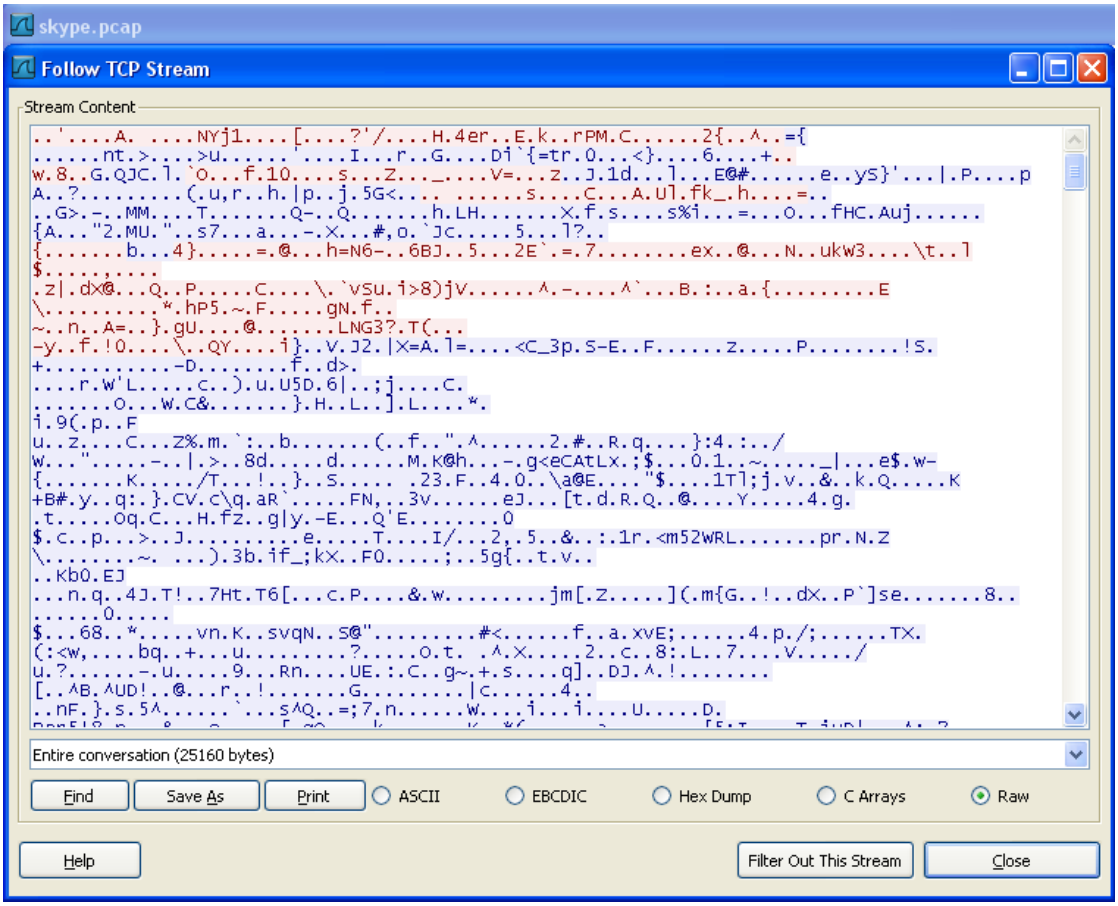
Hvilke sensitive data sender applikasjonen over Internett og hvem blir de sendt til?

Skype bruker kun SSL for å kryptere sendingen av en fil med navn på enheter som er godkjent for å bruke Skype. All annen trafikk(samtaler, meldinger osv.) er kryptert med en annen krypteringsalgoritme som ikke kan dekrypteres med testmiljøet. Figur 20 viser et eksempel på slik trafikk.

TaintDroid opererer uavhengig av datatrafikken og forteller at lokasjon og nettverksbasert lokasjon for brukeren blir sendt tilbake til Skype.

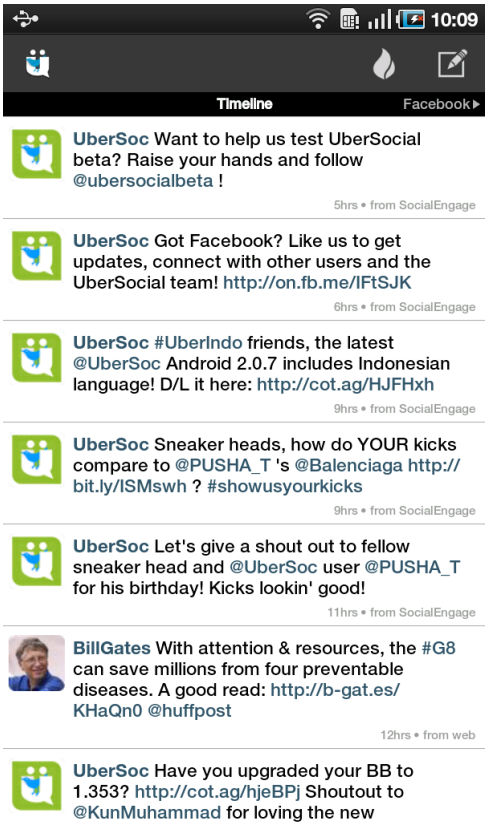
Ivaretas brukerens personvern?

Det har ikke vært mulig å dekryptere all trafikken Skype bruker, og derfor er det ikke mulig å vite om det sendes mer informasjon tilbake til Skypes servere enn det som er oppdaget gjennom TaintDroid. Trolig ville TaintDroid oppdaget mer informasjon hvis det ble sendt, og man kan anta at applikasjonen ikke samler mer informasjon enn det som allerede er kjent. Å kjenne til brukerens posisjon er ikke nødvendig for programmet, og det er derfor en attributt som samles inn unødvendig.



Figur 20: Kryptert trafikk.

5.7 UberSocial for Twitter



Navn: UberSocial for Twitter
Versjon: 2.0.6
Utviklet av: UberMedia Inc.
Antall installasjoner: 1 000 000 - 5 000 000

Beskrivelse

UberSocial for Twitter er en uffosiell Twitter-klient som gir tilgang til Twitters funksjonalitet i et nytt brukergrensesnitt.

Tillatelser på Android

- Detaljert (nettverksbasert) posisjon
- Lese telefontilstand og -identitet
- Endre Wi-Fi-tilstand
- Vis nettverks- og Wi-Fi-tilstand

Se vedlegg E for resten av tillatelsene.

Forventede resultater av anaylse

Denne applikasjonen har tilgang til tillatelser som gjør at den kan se hvor brukeren er, men den kan ikke lese personsensitiv data slik som kontaktlister og meldinger. Trolig brukes lokasjonen kun for lokasjonsspesifikke tjenester, slik som å fortelle hvor brukeren er på Twitter, hvis brukeren ber om det som en funksjon i applikasjonen. De resterende tillatelsene brukes trolig for å styre oppførselen til applikasjonen mens den kjører på telefonen, og vi antar derfor at applikasjonen ikke sender fra seg personsensitiv informasjon.

Trenger applikasjonen de tillatelsene den ber om?

Tillatelsene "Lese telefontilstand og -identitet" og "Endre Wi-Fi-tilstand" øker brukervennligheten i programmet. Lesing av telefontilstanden brukes for å sjekke om eieren er i telefonen og skrur av påtrengende varslinger i applikasjonen for å ikke forstyrre eieren i samtalen. Endring av Wi-Fi-tilstanden trengs når applikasjonen ikke får kontakt over 3G-nettet og forsøker da å koble seg til nærliggende trådløse nettverk. Å kunne skille mellom hvilket type nettverk man er koblet til, gjennom rettigheten "Vis nettverks- og Wi-Fi-tilstand", spiller en viktig rolle i valg av kompresjonsalgoritme ved opplasting av bilder.

Hvilke sensitive data sender applikasjonen over Internett og hvem blir de sendt til?

Analysen av datatrafikken viser at applikasjonen har Flurry innebygd. Figur 21 viser innsamlede data der man kan se at telefonens modell og Android-versjon blir sendt tilbake Flurry når applikasjonen starter. De samme dataene blir også sendt til en AdMarvel reklameserver, som bruker informasjonen i målrettet reklame.

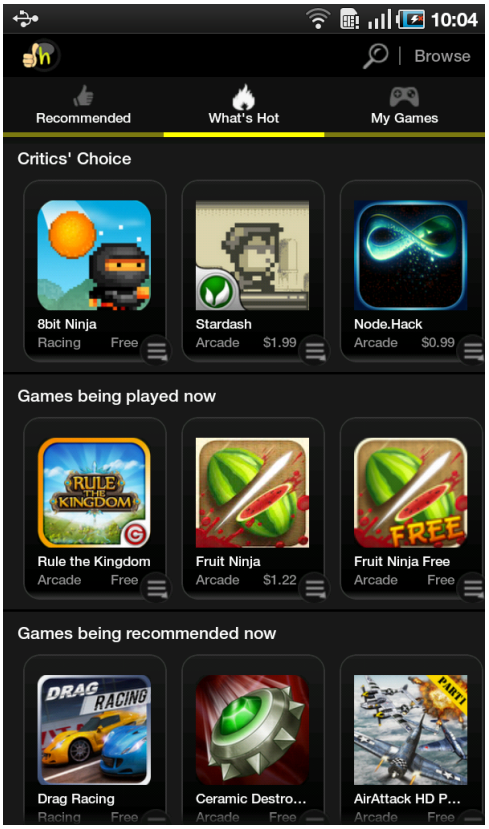
Ivaretas brukerens personvern?

Det som sendes av informasjon kan ikke knyttes til brukeren, og hensikten med informasjonen som sendes er klart definert hos Flurry [13]. Derfor mener vi at denne applikasjonen ivaretar personvernet til brukeren, ved å ikke sende personlig identifiserbar informasjon.

. . . . . u . . . 6 . a . . . RBCZQPBG1DVUXM24G99N . . 2 . 0 . 6
. . . . ID3fe27ec1c13914ee . . . 6 . a . . . . . 6 . a . . . . . . device . model . .
GT–P1000 . . build . brand . . samsung . . build . id . . FROYO . . version
. release . . 2 . 2 . . build . device . . GT–P1000 . build . product . .
GT–P1000 . .

Figur 21: Data som sendes tilbake til Flurry.

5.8 Hooked



Navn: Hooked - best games for you!
Versjon: 1.2.28.0501
Utviklet av: Hooked
Antall installasjoner: 1 000 000 - 5 000 000

Beskrivelse

Hooked er en applikasjon som anbefaler nye spill for brukeren gjennom å analysere brukerens preferanser og spillemønster. Programmet fører statistikk over hva brukeren spiller og hvor mye tid som brukes på hvert spill. Programmet har et brukergrensesitt som lar deg bla i mange typer spill, og se hva som spilles av andre spillere for øyeblikket, hva som er populært og spill som anbefales for deg.

Tillatelser på Android

- Grov (nettverksbasert) posisjon
- Motta data fra Internett

- Endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort
- Kjøre automatisk ved oppstart

Se vedlegg G for resten av tillatelsene.

Forventede resultater av anaylsen

Siden applikasjonen fører statistikk over hva du spiller er det naturlig å anta at disse dataene blir sendt tilbake til utviklerne av applikasjonen. Vi antar at det kun er data som dreier seg om spill, og at spillene som brukes i statistikken er de brukeren selv har valgt at utvikleren skal informeres om. SSL brukes antagelig i overføringen av informasjonen.

Trenger applikasjonen de tillatelsene den ber om?

Hooked ber om flere tillatelser som vi mener er unødvendig for at applikasjonen skal fungere. Det er ikke nødvendig at programmet starter ved oppstart av mobilen, da det er ingenting ved programmet som tilsier at det kreves. Det er ingen funksjon i applikasjonen som krever lokasjonen til brukeren. I tillegg ber applikasjonen om tilgang til minnekortet, og det er ingen funksjon i programmet som krever det.

Hvilke sensitive data sender applikasjonen over Internett og hvem blir de sendt til

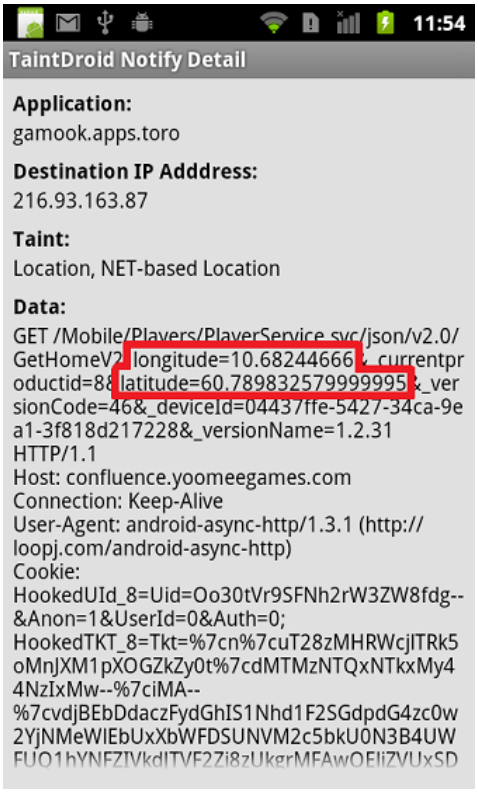
I analysene av datatrafikken ble det oppdaget at når Hooked startes sendes navnet på alle applikasjonene som er installert på telefonen til YooMee Games, en partner av HookedMediaGroup. Hooked sender også navnet på hver eneste applikasjon som blir startet på Android-enheten mens Hooked kjører. Lokasjonen til brukeren blir og sendt tilbake til utviklerne. All informasjon sendes ukryptert.

Hooked har Flurry innebygd, og det ble oppdaget at telefonens Android-ID 23(a), merke, modell og Android-versjon, sammen med brukstatistikk ble sendt tilbake til dem 23(b).

I følge TaintDroid blir lokasjonen(ref. figur 22) sendt vekk fra telefonen til en server på Internett.

Ivaretas brukerens personvern?

Hooked sender mye informasjon fra Android-enheten. Vi mener en applikasjon som samler inn så mye om hvordan enheten blir brukt, og hva som er installert på den, gjør den til en applikasjon som foretar svært grove brudd på brukernes personvern. Brukerne blir informert om at mye data samles inn via "Terms of Use" [36], men dette fremkommer ikke når man installerer eller bruker applikasjonen. Dermed bør informasjonen om hva som samles inn om enheten og brukerne bli mye bedre.



Figur 22: Informasjon som Hooked sender i følge TaintDroid.

```
GET /Mobile/Android/AndroidService.svc/json/v1.0/AreTrackable? [..] deviceId=6237511a-ac81-32fb-a7e7-4c9aa2396e4a
```

(a) Android-id sendes.

```
.................y...6.9:...SEE9RG9KLQL27FYDJVKG..1.2.31....ID3fd96ea3967b0aa0...
6.8=....6.8..device.model..GT-P1000..build.brand..samsung.. [..]
build.product..GT-P1000......1.2.31...6.8=.....................en_GB..GMT.
...............Device......startupTime.....
userAppAction........Device....deviceId.@Manuf: samsung | Model: GT-P1000 |
OS: 2.2 | App Version: 1.2.31..........startupTime....duration..2227
userAppAction....appUserActionType..4..appId..5282.......v.........
userAppAction....appUserActionType..4..appId..3396.........................
```

(b) Brukstatistikk sendes.

Figur 23: Informasjon som blir sendt til Flurry.

5.9 Oversikt over funn i analysene

I tabellen vist i figur 24 vises en oversikt over resultater fra analysen. I venstre kolonne vises navn på applikasjon. Midterste kolonne viser hvilke personsensitive data som sendes av gårde. Til høyre vises mottaker av informasjonen som sendes.

Applikasjon	Informasjon sendt vekk	Mottaker
Androids kommunikasjon med Google	•IMEI •Logg med feilmeldinger og hendelser.	Google
Facebook	•Telefonnummer •IMEI •Sist kjente posisjon •Kamera •Aksellerometer •Det innskrevne passordet	Facebook
Google Play	•Telefonnummer •Lokasjon •Lokasjon funnet via trådløse nettverk i nærheten •Kamera	Google
Google Sync	•Ingenting	
Hooked	•Installerte applikasjoner •Applikasjoner som startes •Telefonens Android-ID, merke og modell •Kjørende versjon av Android •Bruksstatistikk	YooMee Games Flurry
Skype	•Lokasjon •Nettverksbasert lokasjon	Skype
UberSocial for Twitter	•Telefonens merke og modell •Kjørende versjon av Android	Flurry
Gmail	•Telefonnummer •GPS-lokasjon •Sist kjente posisjon	Google

Figur 24: Oversikt over resultater fra analysen

6 Diskusjon

I denne seksjonen er det drøftinger rundt oppnåelsen av målene som var satt i prosjektet, og hva vi tror kan bli de fremtidige effektene av det som har blitt produsert. Det blir også gitt en vurdering av hvordan oppgaven ble utført og punkter vi synes kunne vært gjort bedre. Avslutningsvis nevnes egenvurderingen av gruppearbeidet og en konklusjon om arbeidet.

Resultatmål

Vi mener prosjektet har nådd de ønskede resultatmålene 1.3 om å levere et testmiljø for å gjøre analyser av applikasjoner, sammen med en analyse av de utvalgte applikasjonene. Håpet er at analysene skal være en tankevekker for hvor lett det er for applikasjoner å stjele informasjon fra mobiltelefoner, og dermed øke bevisstheten rundt hva man installerer.

Analysene

Samtlige av de analyserte applikasjonene sender informasjon fra enheten som kan kalles sensitiv. Det mest uskyldige var attributter som IMEI og telefonnummer, mens det mest graverende var applikasjonen som samlet informasjon om alle applikasjonene som var installert og ble startet på telefonen.

Alle applikasjonene i analysen hadde legitime formål og var ikke konstruert utelukkende for spionasje eller datainnsamling. Vi synes derfor at det er et viktig poeng å fremheve at det fremstår som "standard" praksis blant applikasjonsutviklere å samle inn identifiserende informasjon om brukerne og enheten, selv om det ikke er påkrevd for funksjonaliteten i applikasjonen. Uten å kunne vise til statistikk er det vår oppfatning at majoriteten av applikasjonene på Android-plattformen med Internettilgang, på et eller annet vis unikt identifiserer brukerne sine og sender denne informasjonen tilbake til applikasjonsutvikleren. Dette bør være en tankevekker for de som bruker applikasjoner, og som har interesse i å vite hvilken informasjon som blir samlet om dem.

Formålet med de innsamlede dataene er uvisst, men vi tror mesteparten av informasjonen blir brukt for statistikk og reklameformål. Informasjonen samsvarer med hva som ville vært logisk å benytte for målrettet reklame. Unikt identifiserbar informasjon som forteller noe om hvem brukeren er, sammen med informasjonen om hvordan applikasjonen brukes, gjør at det går ann å lage en unik profil for å vise reklame basert på interesser og bruksmønster.

Funnene som er gjort gjennom analysene vil være interessante for både brukere, arbeidsgivere, politikere og andre beslutningstagere som må forholde seg til personvern på mobile plattformer. Det har vært et ønske å informere om hvordan personvernet blir ivaretatt ved å analysere applikasjoner, for å kunne vise til eksempler på hvilken informasjon som blir samlet inn, og hva som kan være konsekvensene av det. Ved å bruke rapporten har man et grunnlag for å vite hva slags informasjon som vanligvis

samles inn, og dermed bli mer bevisst på hva som er problemstillingene rundt bruken av applikasjoner.

Testmiljøet

Vi er spesielt fornøyd med funksjonaliteten som ble lagt til sslsniff for å dekryptere SSL-trafikk med Wireshark. Hverken prosjektruppa eller oppdragsgiver har sett noe lignende gjort tidligere, og dermed tror vi det er en funksjonalitet mange kan ha nytte av. Å forenkle gjennomføringen av lignende analyserer har vært ett av effektmålene 1.3 i prosjektet, og dette mener vi har blitt oppnådd med dette punktet. sslsniff trengte forbedringer for å kunne gjøre analyser på Android, og disse ble laget i løpet av prosjektperioden. For andre som vil gjøre lignende analyser trengs det kun å installere den modifiserte versjonen av sslsniff og lage CA-sertifikater for å få dekryptert all SSL-trafikk på Android.

6.1 Videre arbeid

Testmiljøet som ble laget i prosjektet har gjort det mulig å gjøre flere analyser av applikasjoner hvor man trenger å dekryptere SSL-trafikk. En naturlig fortsettelse på prosjektet er å analysere flere applikasjoner og nyere versjoner av Android. Det kan være interessant å analysere applikasjonene som ble analysert i rapporten om igjen på et senere tidspunkt, for å sammenligne resultatene og se om det har skjedd endringer. Det vil også være interessant å se om testmilmjøet kan brukes på andre plattformer, eksempelvis Apples iPhone eller Microsofts Windows Phone.

6.2 Kritikk av oppgaven

Det ble brukt mye tid på å sette opp testmiljøet, og vi føler at dette har tatt av tiden som var planlagt og ønsket brukt på å gjøre analyser av applikasjoner. Vi budsjetterte med mye kortere tid til oppsett av testmiljøet enn det vi faktisk brukte. Til gjengjeld lot analysene seg utføre raskere enn planlagt, og vi fikk analysert flere applikasjoner enn først forventet.

Kildekoden med forbedringene i sslsniff burde vært ryddigere og blitt plassert andre steder i koden. Endringene bærer preg av å være satt sammen raskt og eksperimentelt, med hardkodede stinavn og et skript som må kjøres for å formatere loggfilene fra programmet. Å få loggingen av krypteringsnøkler til å logge nøklene på et format Wireshark forstår med en gang, uten å måtte kjøre Bash-skriptet først, ser vi som en stor forbedring som kan gjøres. Dette er ting som er planlagt gjort etter at prosjektet er ferdig, med mål om å få koden inkludert i sslsniff.

6.3 Evaluering av gruppearbeidet

Ettersom alle gruppemedlemmene kjenner hverandre og har samarbeidet på prosjekter tidligere, har gruppearbeidet gått meget bra. Det har vært enighet om de fleste

avgjørelser i prosjektet, og det har ikke vært store konflikter eller problemer som ikke har blitt løst med diskusjon eller kompromisser. Prosjektgruppa har møttes jevnlig på høgskolen fire dager i uka og arbeidet sammen på oppgavene. Kommunikasjonen innad i gruppen har fungert godt, og terskelen for å ta opp problemer med andre gruppemedlemmer har vært meget lav.

Arbeidsfordelingen har fungert godt, og gruppa har jobbet sammen om de fleste problemstillingene gjennom hele prosjektet. Det har vært tidspunkter hvor gruppen har syntes det har vært hensiktsmessig å dele opp gruppa hvor én person har jobbet med rapporten, mens de to siste har jobbet med teknisk feilsøking. Arbeidsmengden har stort sett vært jevnt fordelt over hele prosjektet, og arbeids- og møtetidene har vært relativt faste.

Oppdragsgiver og veileder har vært hjelpsomme med tips til rapportskriving og feilsøking under hele prosjektet. Det har vært møter med dem ca. én gang i uken, enten fysisk eller over epost.

6.4 Konklusjon

Hele gruppa er svært fornøyd med å være ferdig med bacheloroppgaven. Selve prosessen med å gjøre et så stort prosjekt har vært svært lærerik, og vi synes vi har fått en forsmak på hvordan det er å drive med et forskingsprosjekt. Vi håper og tror at det som er levert både kan være interessant og nyttig for oppdragsgiver og andre som ønsker vite mer om personvern på Android.

Hovedkonklusjonen for prosjektet er at samtlige undersøkte applikasjoner henter informasjon fra Android-enheten i en eller annen form. Slik vi har oppfattet det har dette ikke vært nødvendig for bruken av applikasjonene, og har dermed blitt brukt for andre formål enn selve funksjonaliteten i den. Vi mener brukerne burde blitt bedre informert om hva informasjonen brukes til og hvorfor den samles inn.

Det ville vært interessant å analysere flere applikasjoner i fremtiden og få et større utvalg av applikasjoner å sammenligne med, slik at man får et bedre bilde av hvordan Android-plattformen som helhet ivaretar personvernet. Vi håper testmiljøet som er laget kan brukes til å analysere flere applikasjoner og at analysene vi har gjort er opplysende om hvordan personvernet blir ivaretatt.

Referanser

[1] APPANALYSIS. Frequently Asked Questions on Our TaintDroid Paper. http://appanalysis.org/faq.html. [Online; sist besøkt 11-februar-2012].

[2] APPANALYSIS. Realtime Privacy Monitoring on Smartphones. http://appanalysis.org/. [Online; sist besøkt 26-januar-2012].

[3] APPSTOREHQ. iOS vs Android? Over 1,000 Developers (Including Some Top Names) Are Having it Both Ways. http://blog.appstorehq.com/post/760323632/ios-vs-android-over-1-000-developers-including-some. [Online; sist besøkt 24-februar-2012].

[4] BERNAT, V. SSL/TLS - Perfect Forward Secrecy. http://vincent.bernat.im/en/blog/2011-ssl-perfect-forward-secrecy.html. [Online; sist besøkt 11-mai-2012].

[5] BLIZZHACKERS. Mitm the gtalk service. "http://www.blizzhackers.cc/viewtopic.php?f=72&t=486999, 2011. [Online; sist besøkt 11-februar-2012].

[6] BRODKIN, J. Facebook testing Android SMS integration, denies "spying" allegations. http://arstechnica.com/gadgets/news/2012/02/facebook-testing-android-sms-integration-denies-spying-allegations.ars. [Online; sist besøkt 25-mars-2012].

[7] DATATILSYNET. Datatilsynet - personvern og informasjonssikkerhet. https://www.datatilsynet.no/. [Online; sist besøkt 10-mai-2012].

[8] ENCK, W., GILBERT, P., CHUN, B.-G., COX, L. P., JUNG, J., MCDANIEL, P., AND SHETH, A. N. Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones. *9th USENIX Symposium on Operating Systems Design and Implementation* (2010), 15.

[9] FACEBOOK. Facebook for Android. https://market.android.com/details?id=com.facebook.katana&feature=search_result#?t=W251bGwsMSwxLDEsImNvbS5mYWNlYm9vay5rYXRhbmEiXQ.. [Online; sist besøkt 25-januar-2012].

[10] FACEBOOK. Facebook for Android - Android-apper på Google Play:. https://play.google.com/store/apps/details?id=com.facebook.katana. [Online; sist besøkt 25-mars-2012].

[11] FARAGO, P. Apple Tablet: The Second Stage Media Booster Rocket. http://blog.flurry.com/bid/30019/Apple-Tablet-The-Second-Stage-Media-Booster-Rocket. [Online; sist besøkt 14-mai-2012].

[12] FLURRY. Flurry Analytics. http://www.flurry.com/product/analytics/index.html. [Online; sist besøkt 8-mai-2012].

[13] FLURRY. Flurry Analytics FAQ. http://www.flurry.com/product/analytics/technical-info.html. [Online; sist besøkt 7-mai-2012].

[14] FLURRY. Flurry Privacy Policy. http://www.flurry.com/about-us/legal/privacy.html. [Online; sist besøkt 14-mai-2012].

[15] GARTNER. Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent. http://www.gartner.com/it/page.jsp?id=1848514. [Online; sist besøkt 12-februar-2012].

[16] GOOGLE. Android. http://www.android.com/. [Online; sist besøkt 9-mai-2012].

[17] GOOGLE. Android-apper på Google Play:. https://play.google.com/store. [Online; sist besøkt 9-mars-2012].

[18] GOOGLE. Android Open Source. http://source.android.com/. [Online; sist besøkt 12-februar-2012].

[19] GOOGLE. Android SDK | Android Developers. http://developer.android.com/sdk/index.html. [Online; sist besøkt 26-januar-2012].

[20] GOOGLE. Building for devices. http://source.android.com/source/building-devices.html. [Online; sist besøkt 9-mars-2012].

[21] GOOGLE. Gmail. https://market.android.com/details?id=com.google.android.gm&hl=no. [Online; sist besøkt 25-januar-2012].

[22] GOOGLE. Google Play Terms of Service. http://play.google.com/about/terms.html. [Online; sist besøkt 25-mars-2012].

[23] GOOGLE. Google Sync for mobiltelefon. http://www.google.com/mobile/sync/. [Online; sist besøkt 25-januar-2012].

[24] GOOGLE. Manifest.permission. http://developer.android.com/reference/android/Manifest.permission.html. [Online; sist besøkt 24-februar-2012].

[25] GOOGLE. protobuf - Protocol Buffers - Google's data interchange format. https://code.google.com/p/protobuf/. [Online; sist besøkt 9-mai-2012].

[26] GOOGLE. Terms and Conditions. http://developer.android.com/sdk/terms.html. [Online; sist besøkt 26-februar-2012].

[27] GOOGLE. What is Android? https://developer.android.com/guide/basics/what-is-android.html. [Online; sist besøkt 9-mars-2012].

[28] GOOGLE INC. Facebook for Android - Android-apper på Google Play. https://play.google.com/store/apps/details?id=com.facebook.katana. [Online; sist besøkt 19-mai-2012].

[29] GOOGLE INC. Gmail - Android-apper på Google Play. https://play.google.com/store/apps/details?id=com.google.android.gm. [Online; sist besøkt 19-mai-2012].

[30] GOOGLE INC. Hooked - best games for you! - Android-apper på Google Play. https://play.google.com/store/apps/details?id=gamook.apps.toro. [Online; sist besøkt 19-mai-2012].

[31] GOOGLE INC. Skype - Android-apper på Google Play. https://play.google.com/store/apps/details?id=com.skype.raider. [Online; sist besøkt 19-mai-2012].

[32] GOOGLE INC. UberSocial for Twitter - Android-apper på Google Play. https://play.google.com/store/apps/details?id=com.twidroid. [Online; sist besøkt 19-mai-2012].

[33] GRACE, M., ZHOU, Y., WANG, Z., AND JIANG, X. Systematic detection of capability leaks in stock android smartphones. 15.

[34] HAYES, B. Skype: A Practical Security Analysis. Tech. rep., SANS Institute, 10 2008.

[35] HOOKED. Hooked - best games for you! https://play.google.com/store/apps/details?id=gamook.apps.toro. [Online; sist besøkt 2-mai-2012].

[36] HOOKED MEDIA GROUP, INC. TERMS OF USE. http://www.hookedmediagroup.com/Mobile/SignUp/Account/Tos. [Online; sist besøkt 18-mai-2012].

[37] INTERNET ENGINEERING TASK FORCE. The Transport Layer Security (TLS) Protocol Version 1.2. https://tools.ietf.org/html/rfc5246. [Online; sist besøkt 9-mars-2012].

[38] JD (JUSTIS- OG BEREDSKAPSDEPARTEMENTET). Lov om behandling av personopplysninger (personopplysningsloven). http://lovdata.no/cgi-wift/wiftldles?doc=/app/gratis/www/docroot/all/nl-20000414-031.html. [Online; sist besøkt 21-mai-2012].

[39] KELLEY, S. Dnsmasq. http://thekelleys.org.uk/dnsmasq/doc.html. [Online; sist besøkt 27-januar-2012].

[40] KRISHNAMURTHY, MOGUL, AND KRISTOL. Key Differences between HTTP/1.0 and HTTP/1.1. http://www8.org/w8-papers/5c-protocols/key/key.html. [Online; sist besøkt 16-mars-2012].

[41] KROAH-HARTMAN, G. Android and the Linux kernel community. http://www.kroah.com/log/linux/android-kernel-problems.html. [Online; sist besøkt 9-mars-2012].

[42] KULTURDEPARTEMENTET. Lov om opphavsrett til åndsverk m.v. (åndsverkloven). http://www.lovdata.no/cgi-wift/wiftldles?doc=/app/gratis/www/docroot/all/tl-19610512-002-037.html. [Online; sist besøkt 24-februar-2012].

[43] KUŠTANS, E. Shark for Root. https://market.android.com/details?id=lv.n3o.shark&feature=search_result#?t=W251bGwsMSwxLDEsImx2Lm4zby5zaGFyayJd. [Online; sist besøkt 26-januar-2012].

[44] LIEBOWITZ, M. Data-stealing malware hits Android Market. http://www.msnbc.msn.com/id/44232800/ns/technology_and_science-security/t/data-stealing-malware-hits-android-market/#.T0NUkPV2P3E. [Online; sist besøkt 21-februar-2012].

[45] LINUX KERNEL ORGANIZATION, INC. The Linux Kernel Archives. https://www.kernel.org/. [Online; sist besøkt 18-mai-2012].

[46] LUO, K. Using static analysis on Android applications to identify private information leaks. Tech. rep., Computing and Information Sciences, Kansas State University, 04 2011.

[47] MALINEN, J. Hostapd. http://w1.fi/hostapd/. [Online; sist besøkt 27-januar-2012].

[48] MARLINSPIKE, M. sslsniff. http://www.thoughtcrime.org/software/sslsniff/. [Online; sist besøkt 26-januar-2012].

[49] MOXIE0. moxie0 / sslsniff - GitHub. https://github.com/moxie0/sslsniff. [Online; sist besøkt 14-mai-2012].

[50] NIELSEN. America's New Mobile Majority: a Look at Smartphone Owners in the U.S. http://blog.nielsen.com/nielsenwire/?p=31688. [Online; sist besøkt 20-mai-2012].

[51] NIELSEN. Mobile & Smartphone Trends. http://www.nielsen.com/us/en/insights/top10s/mobile.html. [Online; sist besøkt 21-februar-2012].

[52] NOTARAS, G. Be your own Certificate Authority (CA). http://www.g-loaded.eu/2005/11/10/be-your-own-ca/. [Online; sist besøkt 9-mars-2012].

[53] OBERHEIDE, J. A peek inside the gtalkservice connection, 2010. [Online; sist besøkt 17-januar-2012].

[54] OBERHEIDE, J. Remote kill and install on google android, 2010. [Online; sist besøkt 17-januar-2012].

[55] OPENSSL. Welcome to the OpenSSL Project. http://www.openssl.org/. [Online; sist besøkt 9-mars-2012].

[56] ÅRNES, A., AND NES, C. Hva vet appen om deg? https://www.datatilsynet.no/Global/04_veiledere/app_rapport_DT2011.pdf. [Online; sist besøkt 24-januar-2012].

[57] ROETHLISBERGER, D. Copy X509v3 subjectAltName from server certificate . https://github.com/droe/sslsniff/commit/7c3459201d42ed45854d384573a68523a492e458. [Online; sist besøkt 21-mai-2012].

[58] ROSSEN, E. Endelig: En app som passer på de andre, 2011. [Online; sist besøkt 24-januar-2012].

[59] SHORTFUSE. SuperOneClick | shortfuse.org. http://shortfuse.org/?page_id=2. [Online; sist besøkt 26-januar-2012].

[60] SKYPE. Skype. https://play.google.com/store/apps/details?id=com.skype.raider. [Online; sist besøkt 2-mai-2012].

[61] SMITH, I. Cost of Hard Drive Space. http://ns1758.ca/winch/winchest.html. [Online; sist besøkt 24-februar-2012].

[62] SPRÅKRÅDET OG UNIVERSITETET I OSLO. Bokmålsordboka | Nynorskordboka. http://www.nob-ordbok.uio.no/perl/ordbok.cgi?OPP=personvern&bokmaal=+&ordbok=bokmaal. [Online; sist besøkt 20-mai-2012].

[63] THEPLAYER. Wire shark can decrypt SSL trafic, how? http://www.commandlineisking.com/2009/10/wire-shark-can-decrypt-ssl-traffic.html. [Online; sist besøkt 11-mai-2012].

[64] UBERMEDIA INC. UberSocial for Twitter. https://play.google.com/store/apps/details?id=com.twidroid. [Online; sist besøkt 2-mai-2012].

[65] U.S. COPYRIGHT OFFICE. Statement of the Librarian of Congress Relating to Section 1201 Rulemaking. http://www.copyright.gov/1201/2010/Librarian-of-Congress-1201-Statement.html. [Online; sist besøkt 9-mars-2012].

[66] WIKIPEDIA. Rooting (Android OS). https://en.wikipedia.org/wiki/Rooting_%28Android_OS%29. [Online; sist besøkt 9-mars-2012].

[67] WIKIPEDIA. Scrum (development). https://en.wikipedia.org/wiki/Scrum_%28development%29. [Online; sist besøkt 7-mai-2012].

[68] WIKIPEDIA. White hat (computer security). https://en.wikipedia.org/wiki/White_hat_%28computer_security%29. [Online; sist besøkt 9-mai-2012].

[69] WIRESHARK FOUNDATION. Wireshark. https://www.wireshark.org/. [Online; sist besøkt 26-januar-2012].

[70] ZHOU, Y., ZHANG, X., JIANG, X., AND FREEH, V. W. Taming Information-Stealing Smartphone Applications (on Android). Tech. rep., Department of Computer Science, NC State University, 04 2011.

A Forprosjekt

Privacy - datalekkasje

Forprosjekt for bacheloroppgave

Kjetil Gardåsen 080481
David Ueland 090740
Eirik Bae 090741

January 27, 2012

Innholdsfortegnelse

Innledning

De siste årene har smarttelefoner og nettbrett blitt allemannseie, og etterhvert som disse enhetene blir smartere og smartere så lagrer de også mer og mer informasjon om brukerne sine. Det er ikke alltid like lett å vite hva telefonen din faktisk vet om deg eller hvem den deler denne informasjonen med. Dette har vi tenkt å finne ut av ved å undersøke et utvalg av applikasjoner på operativsystemet Android.

Personvernsproblematikken ved bruken av smarttelefoner er et tema som stadig blir diskutert i mediene og folk flest har gjerne en formening om hva de synes om mulighetene for å bli overvåket via smarttelefonen sin. Det er derimot ikke like lett å vite at smarttelefonen din legger igjen informasjon om deg, slik som hvor du har vært og hvem du har kommunisert med, uten at man er klar over det eller blir fortalt at det skjer. Når denne informasjonen blir sendt vekk til utviklere av applikasjoner eller produsenten av mobiltelefonen så har man ikke lenger kontroll på hvem som har tilgang til sensitiv informasjon om deg.

1 Mål og rammer

1.1 Bakgrunn

Prosjektets bakgrunn
Lasse Øverlier ved Høgskolen i Gjøvik ønsket å gjøre en undersøkelse av applikasjoner som kjørte på Android [1], for å finne ut av hva applikasjonene la igjen og sendte av informasjon om brukerene av enheten. Oppgaven var veldig fri med tanke på valg av applikasjoner og protokoller, og vi stod fritt til selv å velge hva vi ville undersøke.

Vi syntes dette var en interessant oppgave som omfavnet et tema som alle på gruppen var interessert i, og vi bestemte oss for å velge denne oppgaven. Etter en diskusjon med oppdragsgiver kom vi frem til at oppgaven burde omhandle applikasjoner på Android, og vi ville gjerne undersøke applikasjoner med medfølgende protokoller som hadde stor utbredelse, helst noe som fantes på alle Android-enheter. Derfor bestemte vi oss for å undersøke hvordan Android kommuniserer med Google, både gjennom "skjult" kommunikasjon når du kobler til internett, men også hvordan en applikasjon som Google Sync [2] faktisk kommuniserer med Googles servere når du synkroniserer med dem. I tillegg ville vi undersøke noe som ikke var laget av Google, så vi valgte Facebook, siden det både er en lukket applikasjon med en egen protokoll, samtidig som den har stor utbredelse. I tillegg vurderte vi mulighetene for å undersøke flere applikasjoner og protokoller hvis vi fant mer som var interessant og fikk tid til å gjøre dette.

Vår bakgrunn
Alle gruppemedlemmene er studenter på en bachelor i informasjonssikkerhet ved Høgskolen i Gjøvik. Ingen av oss har tidligere erfaring med programmering på mobile plattformer, men alle har erfaring med programmering i Java fra studiene. De emnene vi tror vil være mest nyttig i utførelsen av prosjektet er "Datakommunikasjon og nettverkssikkerhet" for å analysere nettverkstrafikk, "Programutvikling (i Java)" for å forstå kode i Android, "Informasjonsstrukturer og databaser" for forståelsen av XML, "Operativsystemer" for forståelsen av Androids virkemåte og "Systemutvikling" for prosjektorganiseringen.

Tidligere arbeider
I rapporten "Hva vet appen om deg" [3] fra Datatilsynet blir det konkludert med at "både norske og utenlandske apper kan hente ut svært mye informasjon fra en mobiltelefon, uten at brukeren blir godt nok informert om dette" [4]. Informasjon som applikasjoner typisk henter ut er posisjonsdata, kontaktliste, kalenderdata og informasjon om selve enheten, slik som telefonnummer og IMEI-nummer. Videre kritiseres applikasjonsutviklere for å gi lite eller ingen informasjon om hva som blir samlet inn, hvorvidt det er nødvendig, til hvilket formål, hvem som er behandlingsansvarlig og hvordan informasjonen sikres. Dette gir grobunn for skepsis blant de som er opptatt av personvern.

Jon Oberheide er en uavhengig sikkerhetsanalytiker som har undersøkt hvordan Android Market installerer programvare på Android ved hjelp av GTalkService-protokollen. [5] [6] Dette er Googles mekanisme for å sende kommandoer til telefonen, en såkalt C2DM(cloud-to-device messaging)-arkitektur. I artikkelen omvendt utviklet Oberheide GTalkService for å undersøke hvordan protokollen fungerte.

Forskere ved universitetene Duke og Penn State har i samarbeid med Intel Labs utviklet TaintDroid, en applikasjon som overvåker hva andre applikasjoner sender av data, og hvem de sendes til. Dette er et verktøy som er nyttig for å undersøke applikasjoner og se om de sender personsensitiv informasjon. [7]

To studenter ved NTNU har laget en prototype til en applikasjon som hindrer andre applikasjoner fra å samle informasjon og spre denne videre. [8]

1.2 Prosjektmål

Målet med prosjektet er å undersøke et utvalg av applikasjoner som kjører på Android-operativsystemet og finne ut av hva de vet om brukerene, og om de deler denne informasjonen. Konkret skal vi undersøke:

- Hva applikasjonene lagrer av informasjon om brukerne av enheten.
- Hvem de sender data til.
- Hva slags informasjon de sender til eksterne servere.
- Når og hvordan data samles inn.

Vi håper at dette fører til økt klarhet rundt brukernes personvern ved bruk av Android og de undersøkte applikasjonene.

1.3 Rammer

Kontaktpersoner

- Nils Kalstad Svendsen - Veileder
- Lasse Øverlier - Oppdragsgiver

Plattform og testmiljø
Vi vil bare ta for oss mobiltelefoner som kjører Android. Testmiljøet vil bestå av et trådløst aksesspunkt som vi kobler Android-enhetene til for å fange opp datatrafikk mellom dem og internett. I tillegg vil vi kjøre forskjellige verktøy på enheten for å fange opp datatrafikk og undersøke filsystemet.

Tidsfrister

- 23/5 2012 kl. 12:00 - Innleveringsfrist for prosjektrapporten.
- 6/6-7/6 2012 - Presentasjon av bacheloroppgaven.

2 Omfang

2.1 Oppgavebeskrivelse

Fra oppdragsgivers oppgavebeskrivelse: *"Hoveddelen av oppgaven blir å se på hvordan Google og Facebooks applikasjoner og protokoller på Android lekker eller legger igjen informasjon om hva brukeren foretar seg. Dette kan for eksempel skje gjennom nettet – lagring til nettjenester, gjennom lokal lagring – på den mobile enheten, gjennom trafikkmønster – hvor og hvem den kommuniserer med, signalering på oppsett av samtaler/sending av meldinger m.m."*

For å definere oppgaven ytterligere gjorde vi et utvalg av applikasjoner og funksjoner i Android som vi ville undersøke grundigere:

- Androids kommunikasjon med Google når enheten kobles til internett
- Google Sync [2]
- Google Talk [9]
- Facebook for Android [10]
- Eventuelt flere applikasjoner som ikke er laget av Google, men som er populære på Android.
- Protokoller som brukes av applikasjonene.

Listen er ikke absolutt, og vi diskuterte med oppdragsgiver og kom frem til at vi burde ha muligheten til å endre eller legge til applikasjoner og protokoller hvis det trengtes.

Ut fra oppgavebeskrivelsen har vi valgt følgende problemstillinger som vi skal ta for oss i prosjektet:

- Hva slags informasjon blir utvekslet mellom Google og Android-enheten når man kobler til internett?
- Hvilken informasjon samler applikasjonen om og fra enheten?
- Sender den valgte applikasjonen personsensitiv informasjon tilbake til utvikleren eller andre?
- Lagrer eller genererer den valgte applikasjonen personsensitiv informasjon på enheten?

2.2 Avgrensning

Operativsystem
Valget av operativsystem var avhengig av to ting: hvor åpent det var for å installere verktøyene vi trengte, og hvor utbredt det var. Dermed var det bare Android som var en reell kandidat blant smarttelefonoperativsystemene som har en stor markedsandel,

siden det er det eneste som tillater å gjøre store endringer og installere verktøy som har tilgang på kernel-nivå.

For å oppdage eventuelle endringer i hvordan operativsystemet kommuniserer med Google så bør vi undersøke to forskjellige versjoner av Android. Vi valgte 2.2 og 2.3 siden de er de mest utbredte versjonene. [11]

Applikasjoner
Vi har planlagt å undersøke et utvalg av applikasjoner og protokoller som kjører på Android. Undersøkelsene skal kun omhandle hva applikasjonene sender og lagrer av data. Vi skal ikke omvendt utvikle applikasjonene for å undersøke dem, men protokollene de bruker må trolig omvendt utvikles.
Dataoverføringsteknologi
For å analysere data som blir sendt ut på internett har vi kun tenkt å undersøke data som går via Wi-Fi. For å sikre at data ikke blir sendt via mobilnettverket vil vi gjøre testene uten SIM-kort i enheten. Vi vil fange opp data ved å sette opp et eget trådløst aksesspunkt, men også ved å fange opp trafikken mens den er på selve Android-enheten. Et eget trådløst nettverk er nødvendig for å undersøke kryptert trafikk.

3 Prosjektorganisering

3.1 Ansvarsforhold og roller

- Kjetil Gardåsen (gruppeleder)
- Eirik Bae (utstyr- og dokumentansvarlig)
- David Ueland (sekretær, webmaster)

Gruppelederen har ansvaret for å delegere arbeidsoppgaver og sørge for at tidsfrister blir overholdt. Sekretær og webmaster er ansvarlig for å sende møteinnkallelser, ta notater under møter og skrive møtereferater. Vedkommende skal også oppdatere gruppens hjemmeside med relevant informasjon. Utstyrsansvarlig tar vare på og bestiller utstyr gruppen trenger, I tillegg til å være ansvarlig for å opprette og administrere LaTeX-dokumenter.

3.2 Rutiner og regler i gruppa

- Ved mindre annet er avtalt på forhånd skal alle gruppemedlemmer møte på skolen 4 dager i uken.
- Alle gruppemedlemmer skal møte på alle møter med oppdragsgiver og veileder, så fremt det lar seg gjøre.
- Alle gruppemedlemmer skal føre en timeliste, som vil inneholde arbeidsoppgavene gjort og antall timer brukt daglig.
- Hvert gruppemedlem skal ha et fungerende arbeidsmiljø med LaTeX, Android SDK samt andre programmer som trengs.
  - LaTeX skal brukes til å skrive hovedrapporten for bacheloroppgaven.
  - Før LaTeX-dokumenter lastes opp på SVN skal de kompilere.
- Hvert gruppemedlem er ansvarlig for å medbringe nødvendig utstyr når gruppen møtes.
- Hvis et gruppemedlem ikke rekker å bli ferdig med sine arbeidsoppgaver innen tidsfristen skal resten av gruppen informeres i god tid.
- Konflikter løses ved avstemning hvor det trengs $\frac{2}{3}$ -flertall for gjennomslag.
- Gruppeleder har rett til å signere på vegne av eventuelle gruppemedlemmer som ikke er til stede grunnet gyldig fravær.
- Dersom eventuelle utgifter påløper i forbindelse med bacheloroppgaven skal utgiftene fordeles jevnt mellom gruppemedlemmene.

- Advarsler og eventuell avskjedigelse fra gruppen:
  - Dersom et gruppemedlem ikke utfører avtalt arbeid vil gruppemedlemmet få en advarsel. Flere advarsler vil kunne ende i avskjedigelse fra gruppen.
  - For at et gruppemedlem skal avskjediges fra gruppen vil gruppemedlemmet gjentatte ganger ha hatt ugyldig fravær og/eller ha fått flere advarsler grunnet ikke utført arbeid.

4 Planlegging, oppfølgning og rapportering

4.1 Hovedinndeling av prosjektet

Prosjektet vil bestå av flere lignende faser hvor hver applikasjon og dens protokoller vil bli utsatt for en analyse. Dermed blir det naturlig å velge en systemutviklingsmodell hvor man har like faser som man går igjennom for hver applikasjon. Vi har tenkt på noe lignende Scrum hvor man har sprinter og kan gjøre analysen i hver av disse. Vi vil trolig trenge en del fleksibilitet i lengden på hver sprint siden analysene kan ha forskjellig omfang og tidsforbruk. Dermed bør man ha muligheten til å variere lengden på hver sprint for å tilpasse seg arbeidsmengden for en analyse.

4.2 Plan for statusmøter og beslutningspunkter

- Gruppen skal holde et internt statusmøte minst én gang i uken, der alle orienterer hverandre om status på sitt arbeid i prosjektet.
- Det skal holdes et møte med oppdragsgiver og veileder hver torsdag klokken 09:00, med mindre annet er avtalt.
- Ved alle møter skal det sendes en møteinnkalling minst to dager i forveien.
- Møteinnkallelser skal inneholde agenda for møtet.

5 Organisering av kvalitetssikring

5.1 Dokumentasjon

Alt arbeid og viktige beslutninger gruppen foretar seg skal noteres i fremdriftsloggen. Hvert gruppemedlem skal føre en timeliste over hva de har gjort og hvor mye tid de har brukt på det.

5.2 Konfigurasjonsstyring

- Subversion skal brukes for versjonskontroll av dokumenter skrevet i LaTeX.
- Mindre dokumenter skal skrives i Google Documents [12] og deles med resten av gruppen ved behov.
- Nettsider som det refereres til skal lastes ned som en PDF og refereres til i BiBTeX.

5.3 Risikoanalyse

Risiko	Konsekvens	Sannsynlighet	Tiltak
Protokollene som applikasjonene bruker endres underveis i prosjektet.	Middels	Høy	Ikke oppdatere applikasjoner eller Android.
Finner ingen personvernlekkasjer.	Middels	Lav	Bytt applikasjon som undersøkes.
Klarer ikke å dekryptere krypterte data.	Middels	Lav	Prøv å bytte krypteringsprotokoll til en som lettere kan dekrypteres. Bytt applikasjon som undersøkes.
Klarer ikke å sette opp et fungerende testmiljø for å samle inn data fra telefonene.	Middels	Lav	Finn alternative testmiljø som lettere kan settes opp.
Undersøke og reverse-engineere protokoller og applikasjoner tar for lang tid til at man rekker å analysere og lære noe fra dataene.	Lav	Middels	Begrense omfanget av undersøkelsene.
Får for mye data til at vi rekker å analysere alt.	Lav	Middels	Gjøre et utvalg av data og analysere dem.

6 Plan for gjennomføring

6.1 Utstyr og verktøy

Utstyr

Android-enhet	Versjon	Er rooted	Skal rootes
Samsung Galaxy Tab	2.2	Nei	Ja
HTC Desire HD	2.2	Nei	Ja
Google Nexus One	2.3.7	Ja	Ja

Verktøy

Verktøyene er et utvalg av spesialverktøy for Android og verktøy som anses som industristandard for å løse spesielle oppgaver. Vi har enten valgt et verktøy fordi det er det eneste av sitt slag, eller fordi det er ansett som det beste alternativet for å løse en bestemt oppgave.

- TaintDroid - Dette er en applikasjon som lar deg undersøke hva slags data andre applikasjoner sender ut på internett. [7]
- Wireshark - Dette programmet lar oss overvåke nettverkstrafikk og se innholdet i datapakkene. [13] [14]
- logcat - Verktøy innebygd i Android for å vise logger. [15]
- GTalkService Monitor - Overvåker statusen på GTalkService og viser tilkoblingsstatus. [16]
- Android SDK - Inneholder flere verktøy for å lage og debugge applikasjoner på Android, samt overføre data til enheten. [17]
- SuperOneClick - Verktøy for å roote Android-enheter. [18]
- sslsniff - Verktøy for å gjøre et MITM-angrep mot SSL-trafikk og overvåke den. [19]
- Android Commander - Filsystemutforsker for Windows som lar deg se systemfilene på Android. [20]
- hostapd - Trådløst aksesspunkt i Linux. [21]
- dnsmasq - DHCP server og DNS videresender som vi har kjørende på det trådløse aksesspunktet. [22]
- trådløst nettverkskort - Brukes for å koble Android-enheten til en datamaskin.
- Linux - Flere av verktøyene vi trenger finnes kun på Linux.

Rooting

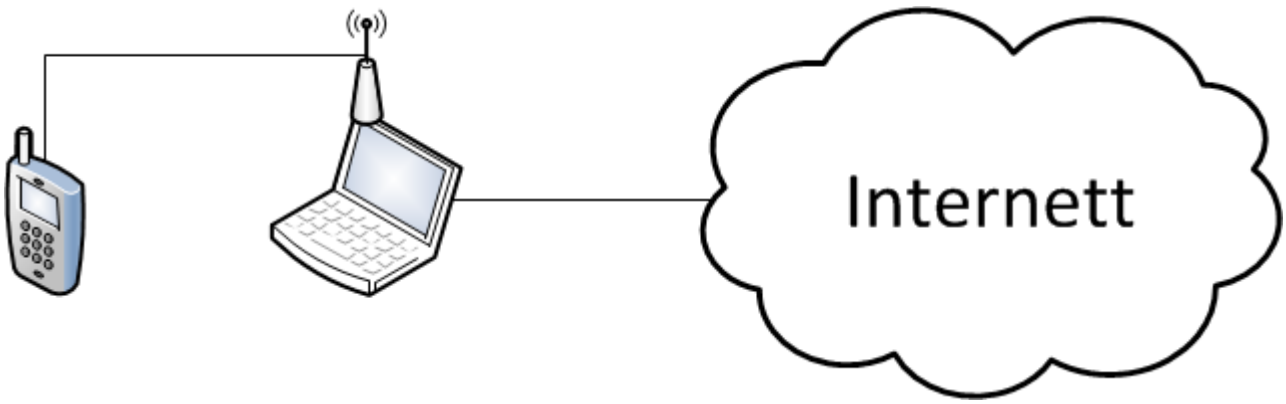
Å roote en Android-enhet innebærer å skaffe rettighetene til root-brukeren slik at man får tilgang til hele filsystemet og muligheten til å kjøre programmer som trenger root-rettigheter for å fungere.

Vi har rootet alle Android-enhetene fordi vi trenger muligheten til å installere våre egne SSL-sertifikater og for å kjøre programmer slik som Wireshark.

6.2 Testmiljø

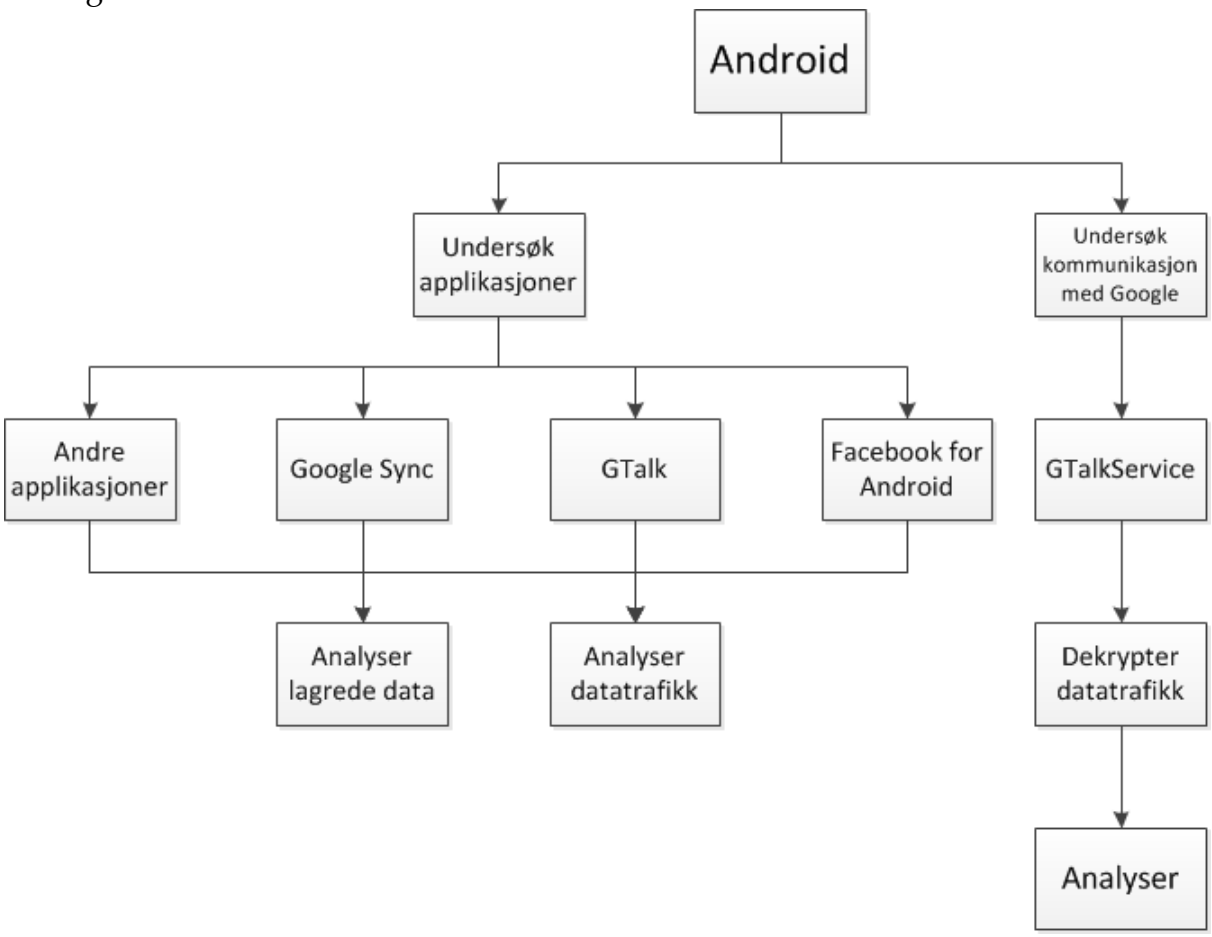
For å undersøke datatrafikken som går mellom Android-enheten og internett må vi sette opp et testmiljø som lar oss fange opp datatrafikken før den blir sendt ut på internett. Vi har tenkt å sette opp et ekstra trådløst nettverkskort på en laptop, slik at vi kan koble Android-enheten til dette og overvåke trafikken som blir sendt via dette til internett.

For å undersøke kryptert trafikk må vi lage vårt eget CA-sertifikat og installere det på enheten, for etterpå å kunne undersøke trafikken med sslsniff.

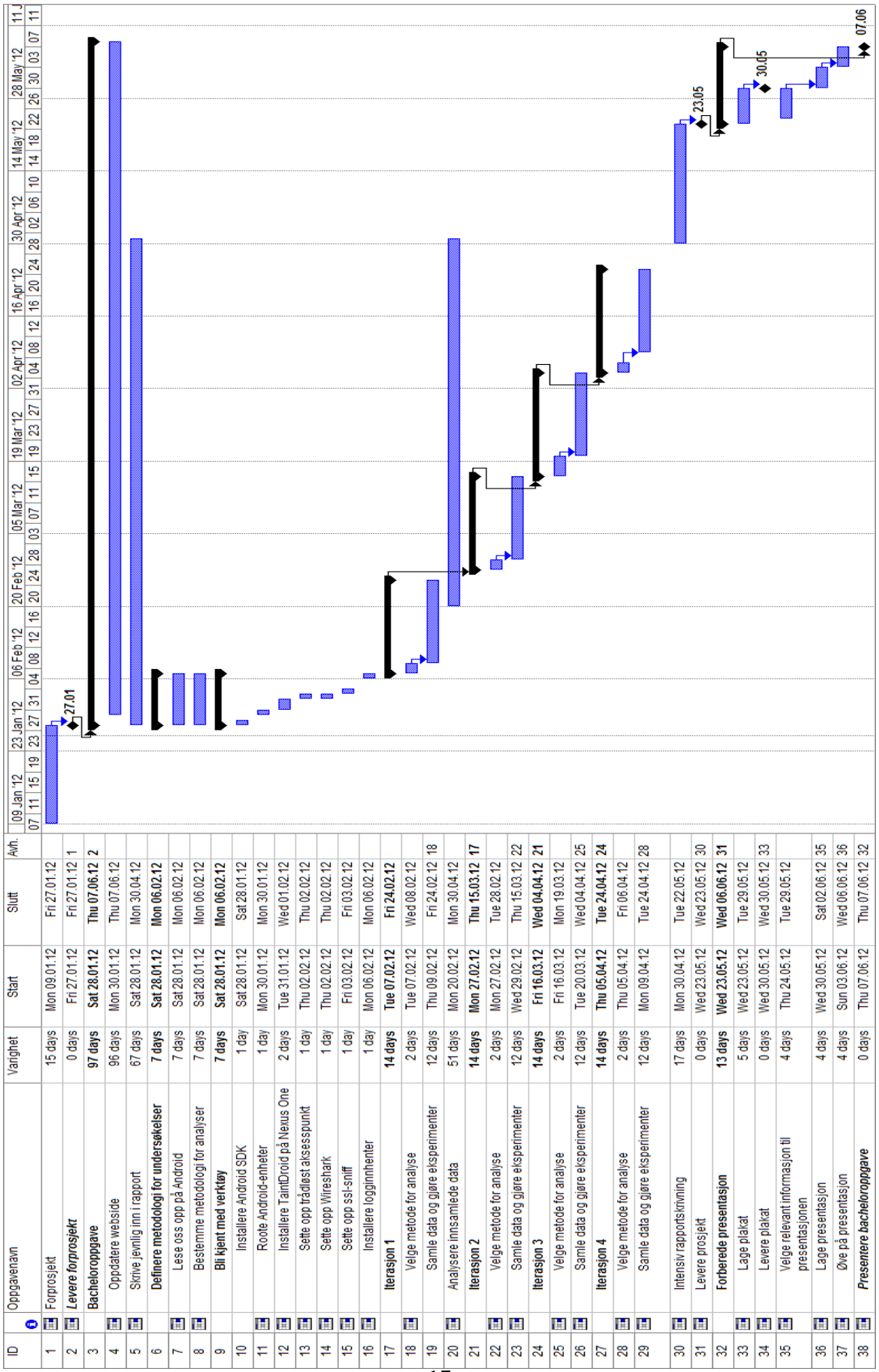


6.3 Liste over aktiviteter

Vi har brutt arbeidet ned i mindre deler hvor vi analyserer hver applikasjon for seg selv. Hver applikasjon blir utsatt for de samme analysene hvor vi undersøker lagrede filer og datatrafikk mot internett.



6.4 Tids- og ressursplan



ID	Oppgavenavn	Varighet	Start	Slutt	Avh.
1	Forprosjekt	15 days	Mon 09.01.12	Fri 27.01.12	
2	*Levere forprosjekt*	0 days	Fri 27.01.12	Fri 27.01.12	1
3	**Bacheloroppgave**	**97 days**	**Sat 28.01.12**	**Thu 07.06.12**	**2**
4	Oppdatere webside	96 days	Mon 30.01.12	Thu 07.06.12	
5	Skrive jevnlig inn i rapport	67 days	Sat 28.01.12	Mon 30.04.12	
6	**Definere metodologi for undersøkelser**	**7 days**	**Sat 28.01.12**	**Mon 06.02.12**	
7	Lese oss opp på Android	7 days	Sat 28.01.12	Mon 06.02.12	
8	Bestemme metodologi for analyser	7 days	Sat 28.01.12	Mon 06.02.12	
9	**Bli kjent med verktøy**	**7 days**	**Sat 28.01.12**	**Mon 06.02.12**	
10	Installere Android SDK	1 day	Sat 28.01.12	Sat 28.01.12	
11	Roote Android-enheter	1 day	Mon 30.01.12	Mon 30.01.12	
12	Installere TaintDroid på Nexus One	2 days	Tue 31.01.12	Wed 01.02.12	
13	Sette opp trådløst aksesspunkt	1 day	Thu 02.02.12	Thu 02.02.12	
14	Sette opp Wireshark	1 day	Thu 02.02.12	Thu 02.02.12	
15	Sette opp ssl-sniff	1 day	Fri 03.02.12	Fri 03.02.12	
16	Installere logginnhenter	1 day	Mon 06.02.12	Mon 06.02.12	
17	**Iterasjon 1**	**14 days**	**Tue 07.02.12**	**Fri 24.02.12**	
18	Velge metode for analyse	2 days	Tue 07.02.12	Wed 08.02.12	
19	Samle data og gjøre eksperimenter	12 days	Thu 09.02.12	Fri 24.02.12	18
20	Analysere innsamlede data	51 days	Mon 20.02.12	Mon 30.04.12	
21	**Iterasjon 2**	**14 days**	**Mon 27.02.12**	**Thu 15.03.12**	**17**
22	Velge metode for analyse	2 days	Mon 27.02.12	Tue 28.02.12	
23	Samle data og gjøre eksperimenter	12 days	Wed 29.02.12	Thu 15.03.12	22
24	**Iterasjon 3**	**14 days**	**Fri 16.03.12**	**Wed 04.04.12**	**21**
25	Velge metode for analyse	2 days	Fri 16.03.12	Mon 19.03.12	
26	Samle data og gjøre eksperimenter	12 days	Tue 20.03.12	Wed 04.04.12	25
27	**Iterasjon 4**	**14 days**	**Thu 05.04.12**	**Tue 24.04.12**	**24**
28	Velge metode for analyse	2 days	Thu 05.04.12	Fri 06.04.12	
29	Samle data og gjøre eksperimenter	12 days	Mon 09.04.12	Tue 24.04.12	28
30	Intensiv rapportskriving	17 days	Mon 30.04.12	Tue 22.05.12	
31	Levere prosjekt	0 days	Wed 23.05.12	Wed 23.05.12	30
32	**Forberede presentasjon**	**13 days**	**Wed 23.05.12**	**Wed 06.06.12**	**31**
33	Lage plakat	5 days	Wed 23.05.12	Tue 29.05.12	
34	Levere plakat	0 days	Wed 30.05.12	Wed 30.05.12	33
35	Velge relevant informasjon til presentasjonen	4 days	Thu 24.05.12	Tue 29.05.12	
36	Lage presentasjon	4 days	Wed 30.05.12	Sat 02.06.12	35
37	Øve på presentasjon	4 days	Sun 03.06.12	Wed 06.06.12	36
38	*Presentere bacheloroppgave*	0 days	Thu 07.06.12	Thu 07.06.12	32

Referanser

[1] Google. Android. http://www.android.com/. [Online; accessed 25-January-2012].

[2] Google. Google Sync for mobiltelefon. http://www.google.com/mobile/sync/. [Online; accessed 25-January-2012].

[3] Årnes og Catharina Nes, D. A. 2011. How to monitor google talk service in android. [Online; accessed 24-January-2012].

[4] Datatilsynet. 2011. Hvilke personopplysninger henter de ulike applikasjonene ut fra mobiltelefonen din – og hva får du vite om dette? svaret er at mye kan hentes ut, men at du som bruker får vite lite. [Online; accessed 16-January-2012].

[5] Oberheide, J. 2010. A peek inside the gtalkservice connection. [Online; accessed 17-January-2012].

[6] Oberheide, J. 2010. Remote kill and install on google android. [Online; accessed 17-January-2012].

[7] et al., W. E. 2010. Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones. *9th USENIX Symposium on Operating Systems Design and Implementation*, 15.

[8] Rossen, D. E. 2011. Endelig: En app som passer på de andre. [Online; accessed 24-January-2012].

[9] Google. Gmail. https://market.android.com/details?id=com.google.android.gm&hl=no. [Online; accessed 25-January-2012].

[10] Facebook. Facebook for Android. https://market.android.com/details?id=com.facebook.katana&feature=search_result#?t=W251bGwsMSwxLDEsImNvbS5mYWNlYm9vay5rYXRhbmEiXQ.. [Online; accessed 25-January-2012].

[11] Google. Platform Versions. http://developer.android.com/resources/dashboard/platform-versions.html. [Online; accessed 27-January-2012].

[12] Google. Google Documents. https://docs.google.com/. [Online; accessed 27-January-2012].

[13] Foundation, W. Wireshark. https://www.wireshark.org/. [Online; accessed 26-January-2012].

[14] Kuštans, E. Shark for Root. https://market.android.com/details?id=lv.n3o.shark&feature=search_result#?t=W251bGwsMSwxLDEsImx2Lm4zby5zaGFyayJd. [Online; accessed 26-January-2012].

[15] Android. Android Developers Guide - logcat. http://developer.android.com/guide/developing/tools/logcat.html. [Online; accessed 27-January-2012].

[16] Singh, H. 2010. How to monitor google talk service in android. [Online; accessed 19-January-2012].

[17] Google. Android SDK | Android Developers. http://developer.android.com/sdk/index.html. [Online; accessed 26-January-2012].

[18] shortfuse. SuperOneClick | shortfuse.org. http://shortfuse.org/?page_id=2. [Online; accessed 26-January-2012].

[19] Marlinspike, M. sslsniff. http://www.thoughtcrime.org/software/sslsniff/. [Online; accessed 26-January-2012].

[20] PanPiotr. Android Commander. http://androidcommander.com/. [Online; accessed 26-January-2012].

[21] Malinen, J. Hostapd. http://w1.fi/hostapd/. [Online; accessed 27-January-2012].

[22] Kelley, S. Dnsmasq. http://thekelleys.org.uk/dnsmasq/doc.html. [Online; accessed 27-January-2012].

B Endrede kodefiler i sslsniff

Listing 1: HTTPSBridge.cpp

```
void HTTPSBridge::buildRequestFromHeaders(HttpHeaders &headers,
    std::string &request) {
  std::ostringstream requestStream;
  requestStream << headers.getMethod()  << "␣"
          << headers.getRequest() << "␣"
/*-        << "HTTP/1.0\r\n"; */
+         << "HTTP/1.1\r\n";

  std::map<std::string,std::string>::iterator iter;
  std::map<std::string,std::string,ci_less>& headersMap =
     headers.getHeaders();
  for( iter = headersMap.begin(); iter != headersMap.end(); ++
     iter ) {
    std::string key   = iter->first;
    std::string value = iter->second;

    Util::trimString(key);
    Util::trimString(value);

    if (key != "Accept-Encoding" && key != "Connection" && key
       != "Keep-Alive")
      requestStream << key << ":␣" << value << "\r\n";
  }

  requestStream << "Connection:␣Close" << "\r\n\r\n";

  if (headers.isPost()) requestStream << headers.getPostData();

  request = requestStream.str();
}

bool HTTPSBridge::readFromClient() {
   char buf[8192];
   int bytesRead;

/*
-  int bytesWritten;
-
-  do {
-    if ((bytesRead = SSL_read(clientSession, buf, sizeof(buf))
   ) <= 0)
```

```
-      return SSL_get_error(clientSession, bytesRead) ==
   SSL_ERROR_WANT_READ ? true : false;
-
-    Logger::logFromClient(serverName, buf, bytesRead);
-
-    if (headers.process(buf, bytesRead)) {
-      std::string request;
-
-      buildRequestFromHeaders(headers, request);
-      SSL_write(serverSession, request.c_str(), request.length
   ());
-
-      if (headers.isPost()) Logger::logFromClient(serverName,
   headers);
-    }
-  } while (SSL_pending(serverSession));
*/


// Leser inn SSL-data fra klienten.
+ bytesRead = SSL_read(clientSession, buf, sizeof(buf));

// Sender leste SSL-data videre til serveren.
+ SSL_write(serverSession, buf, bytesRead);

  return true;
}
```

Listing 2: SSLBridge.cpp

```
 void parseCommonName(X509 *cert) {
    // *** Kodefiks av Daniel Roethlisberger ***
/*
-    std::string distinguishedName(cert->name);
-    std::string::size_type cnIndex = distinguishedName.find("
   CN=");
-
-    if (cnIndex == std::string::npos) throw
   BadCertificateException();
-
-    std::string commonName           = distinguishedName.
   substr(cnIndex+3);
-    std::string::size_type nullIndex = commonName.find("\\x00
   ");
-
-    if (nullIndex != std::string::npos) this->name =
   commonName.substr(0, nullIndex);
-    else                                this->name =
   commonName;
*/
+   X509_NAME *ptr = X509_get_subject_name(cert);
+   int sz = X509_NAME_get_text_by_NID(ptr, NID_commonName,
   NULL, 0) + 1;
+   char *commonNameStr = (char*)malloc(sz);
+   X509_NAME_get_text_by_NID(ptr, NID_commonName,
   commonNameStr, sz);
+   this->name = std::string((const char*)commonNameStr);
+   free(commonNameStr);
  }
```

Listing 3: SSLBridge.cpp

```
void SSLBridge::setServerName() {
    // *** Kodefiks av Daniel Roethlisberger ***
/*
-   X509 *serverCertificate     = getServerCertificate();
-   X509_NAME *serverNameField = X509_get_subject_name(
    serverCertificate);
-   char *serverNameStr         = X509_NAME_oneline(
    serverNameField, NULL, 0);

-   this->serverName = std::string((const char*)serverNameStr);
-   int commonNameIndex;

-   if ((commonNameIndex = this->serverName.find("CN=")) != std
    ::string::npos)
-     this->serverName = this->serverName.substr(commonNameIndex
    +3);

-   free(serverNameStr);
*/
+   X509 *serverCertificate = getServerCertificate();
+   X509_NAME *ptr = X509_get_subject_name(serverCertificate);
+   int sz = X509_NAME_get_text_by_NID(ptr, NID_commonName, NULL
    , 0) + 1;
+   char *commonNameStr = (char*)malloc(sz);
+   X509_NAME_get_text_by_NID(ptr, NID_commonName, commonNameStr
    , sz);
+   this->serverName = std::string((const char*)commonNameStr);
+   free(commonNameStr);
}

void SSLBridge::handshakeWithClient(CertificateManager &manager
    , bool wildcardOK) {
  Certificate *leaf;
  std::list<Certificate*> *chain;

+   bool opprettet;
+   BIO *stmp;
+   char* mode;

  ip::tcp::endpoint endpoint = getRemoteEndpoint();
  manager.getCertificateForTarget(endpoint, wildcardOK,
      getServerCertificate(), &leaf, &chain);
```

```
  setServerName();

  SSL_CTX *clientContext = SSL_CTX_new(SSLv23_server_method());
  buildClientContext(clientContext, leaf, chain);

  SSL *clientSession = SSL_new(clientContext);
  SSL_set_fd(clientSession, clientSocket->native());

  if (SSL_accept(clientSession) == 0) {
    Logger::logError("SSL Accept Failed!");
    throw SSLConnectionError();
  }

  // *** Endringer gjort i bacheloroppgaven //

  // Hent en paagaaende SSL-sesjon.
+  SSL_SESSION *session = SSL_get_session(clientSession);

  // Sjekk om loggfila for noklene finnes,
  // og opprett den hvis ikke.
+  if(!opprettet) { mode = "ab"; opprettet = true; }
+  else mode = "wb";

  // Hent peker til fila.
+  stmp = BIO_new_file("/home/temp/client", mode);

+  if (stmp)
+  {
        // Skriv noklene til fil og
        // lukk fila naar det er ferdig.
+       SSL_SESSION_print (stmp, session);
+    BIO_free(stmp);
+  }

  // ********************************************

  this->clientSession = clientSession;
}

void SSLBridge::handshakeWithServer() {
  int bogus;

+  bool opprettet;
+  BIO *stmp;
```

```
+   char* mode;

    ip::address_v4 serverAddress = serverSocket->remote_endpoint
       ().address().to_v4();
    SSL_CTX *serverCtx = SSL_CTX_new(SSLv23_client_method());;
    SSL *serverSession = SSL_new(serverCtx);;
    SSL_SESSION *sessionId = cache->getSessionId(serverSession,
serverAddress.to_bytes().data(),
serverAddress.to_bytes().size(),
&bogus);

    if (sessionId != NULL) {
      SSL_set_session(serverSession, sessionId);
      SSL_SESSION_free(sessionId);
    }

    SSL_set_connect_state(serverSession);
    SSL_set_fd(serverSession, serverSocket->native());
    SSL_set_options(serverSession, SSL_OP_ALL);

    if (SSL_connect(serverSession) < 0) {
      Logger::logError("Error on SSL Connect.");
      throw SSLConnectionError();
    }

    cache->setNewSessionId(serverSession, SSL_get1_session(
       serverSession),
serverAddress.to_bytes().data(),
serverAddress.to_bytes().size());

    // *** Endringer gjort i bacheloroppgaven //

    // Hent en paagaaende SSL-sesjon.
+   SSL_SESSION *session = SSL_get_session(serverSession);

    // Sjekk om loggfila for noklene finnes,
    // og opprett den hvis ikke.
+   if(!opprettet) { mode = "ab"; opprettet = true; }
+   else mode = "wb";

    // Hent peker til fila.
+   stmp = BIO_new_file("/home/temp/server", mode);

+ if (stmp)
```

```
+   {
     // Skriv noklene til fil og
          // lukk fila naar det er ferdig.
+         SSL_SESSION_print (stmp, session);
+    BIO_free(stmp);
   }

  // ********************************************

   this->serverSession = serverSession;
}
```

Listing 4: SSLBridge.cpp

```
void AuthorityCertificateManager::getCertificateForTarget(boost
    ::asio::ip::tcp::endpoint &endpoint,
bool wildcardOK,
X509 *serverCertificate,
Certificate **cert,
std::list<Certificate*> **chainList)
{
  X509_NAME *serverName = X509_get_subject_name(
     serverCertificate);
  X509_NAME *issuerName = X509_get_subject_name(authority->
     getCert());
  X509 *request = X509_new();

    /* *** Kodefiks av Daniel Roethlisberger ***
- X509_set_version(request, 3); */
+ X509_set_version(request, 2); /* 2 is X509v3 */
  X509_set_subject_name(request, serverName);
  X509_set_issuer_name(request, issuerName);

  ASN1_INTEGER_set(X509_get_serialNumber(request),
     generateRandomSerial());
  X509_gmtime_adj(X509_get_notBefore(request), -365);
  X509_gmtime_adj(X509_get_notAfter(request), (long)
     60*60*24*365);
  X509_set_pubkey(request, this->leafPair);

+  X509V3_CTX ctx;
+  X509_EXTENSION *ext;
+  int extpos;
+  X509V3_set_ctx(&ctx, authority->getCert(), request, NULL,
   NULL, 0);
+  X509_add_ext(request, ext = X509V3_EXT_conf(NULL, &ctx,
+(char*)"basicConstraints",
+(char*)"critical,CA:FALSE"), -1);
+  X509_EXTENSION_free(ext);
+  X509_add_ext(request, ext = X509V3_EXT_conf(NULL, &ctx,
+(char*)"keyUsage",
+(char*)"digitalSignature,keyEncipherment"), -1);
+  X509_EXTENSION_free(ext);
+  X509_add_ext(request, ext = X509V3_EXT_conf(NULL, &ctx,
+(char*)"extendedKeyUsage",
+(char*)"serverAuth"), -1);
+  X509_EXTENSION_free(ext);
```

```
+   X509_add_ext(request, ext = X509V3_EXT_conf(NULL, &ctx,
+(char*)"subjectKeyIdentifier",
+(char*)"hash"), -1);
+   X509_EXTENSION_free(ext);
+   X509_add_ext(request, ext = X509V3_EXT_conf(NULL, &ctx,
+(char*)"authorityKeyIdentifier",
+(char*)"keyid,issuer:always"), -1);
+   X509_EXTENSION_free(ext);
+   extpos = X509_get_ext_by_NID(serverCertificate,
    NID_subject_alt_name, -1);
+   if (extpos != -1) {
+     X509_add_ext(request, X509_get_ext(serverCertificate,
    extpos), -1);
+   }

   X509_sign(request, authority->getKey(), EVP_sha1());

   Certificate *leaf = new Certificate();
   leaf->setCert(request);
   leaf->setKey(this->leafPair);

   *cert = leaf;
   *chainList = &(this->chainList);
   // *chain = this->authority;
}
```

C Bash-skript for parsing av loggfiler med krypteringsnøkler

Listing 5: nokkelhenter2.sh

```
#!/bin/bash
# Dette skriptet leser inn to filer med SSL-nokler fra sslsniff
# og henter ut Session ID og Master Key fra dem, og presenterer
# dem paa et format Wireshark forstaar.

# Gaar gjennom filene server og client.
for filename in server client
do
        cat $filename | while read line; # Leser en linje i
           fila.
        do
                # Greper Session ID og Master Key
                # fra fila og awker dem ut sammen med
                # parameterne som Wireshark trenger for aa
                   laste fila.
                echo $line | grep Session-ID: | awk '{printf "
                   RSA Session-ID:%s", $2}' >> tmp.txt
                echo $line | grep Master-Key: | awk '{printf "
                   Master-Key:%s\n", $2}' >> tmp.txt
        done
done

uniq tmp.txt >> masternokkel.txt # Fjerner alle dupliserte
   linjer og skriver til en ny fil.
rm -rf tmp.txt # Fjerner tmp-fila.
```

D Tillatelser for Facebook

Oversikt over tillatelser er hentet fra applikasjonssiden til Facebook for Android på Google Play. [28]

Dine kontoer

- Fungere som en kontogodkjenner
- Administrere kontolisten
- Oppdag kjente kontoer

Maskinvarekontroller

- Ta bilder og videoer
- Ta opp lyd
- Kontroller vibrator

Din posisjon

- Detaljert (GPS) posisjon

Nettverkskommunikasjon

- Full tilgang til Internett
- Vis Wi-Fi-tilstand
- Motta data fra Internett
- Vis nettverkstilstand

Dine personlige opplysninger

- Les kontaktdata
- Skriv kontaktdata

Lagring

- Endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort

Systemverktøy

- Skrive synkroniseringsinnst.
- Hindre nettbrettet i å gå over i hvilemodus hindre telefonen i å gå over i hvilemodus
- Lese synkroniseringsinnst.

E Tillatelser for UberSocial for Twitter

Oversikt over tillatelser er hentet fra applikasjonssiden til UberSocial for Twitter på Google Play. [32]

Din posisjon

- Grov (nettverksbasert) posisjon
- Detaljert (GPS) posisjon

Nettverkskommunikasjon

- Full tilgang til Internett
- Vis nettverkstilstand
- Vis Wi-Fi-tilstand

Telefonsamtaler

- Lese telefontilstand og -identitet

Lagring

- Endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort

Systemverktøy

- Endre Wi-Fi-status
- Hindre nettbrettet i å gå over i hvilemodus hindre telefonen i å gå over i hvilemodus
- Lese synkroniseringsinnst.
- Les synkroniseringsstatistikk
- Kjøre automatisk ved oppstart

Dine kontoer

- Oppdag kjente kontoer

Maskinvarekontroller

- Kontroller vibrator

F Tillatelser for Skype

Oversikt over tillatelser er hentet fra applikasjonssiden til Skype på Google Play. [31]

Dine kontoer

- fungere som en kontogodkjenner
- administrere kontolisten
- bruk autentiseringsopplysningene for en konto
- oppdag kjente kontoer

Tjenester som koster penger

- ringe telefonnumre direkte

Maskinvarekontroller

- ta bilder og videoer
- endre lydinnstillingene
- ta opp lyd
- kontroller vibrator

Din posisjon

- grov (nettverksbasert) posisjon

Nettverkskommunikasjon

- full tilgang til Internett
- opprette Bluetooth-tilkoblinger
- vis nettverkstilstand
- vis Wi-Fi-tilstand

Dine personlige opplysninger

- les kontaktdata
- skriv kontaktdata

Telefonsamtaler

- lese telefontilstand og -identitet

Lagring

- endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort

Systemverktøy

- deaktiver tastelås
- endre Wi-Fi-status
- hindre nettbrettet i å gå over i hvilemodus hindre telefonen i å gå over i hvilemodus
- hente apper som kjører
- skrive synkroniseringsinnst.
- endre globale systeminnstillinger
- lese synkroniseringsinnst.
- les synkroniseringsstatistikk
- send fast kringkasting

Standard

- ringe telefonnumre direkte

G Tillatelser for Hooked

Oversikt over tillatelser er hentet fra applikasjonssiden til Hooked på Google Play. [30]

Dine kontoer

- Bruk autentiseringsopplysningene for en konto
- Oppdag kjente kontoer

Din posisjon

- Grov (nettverksbasert) posisjon

Nettverkskommunikasjon

- Full tilgang til Internett
- Vis nettverkstilstand
- Vis Wi-Fi-tilstand
- Motta data fra Internett

Lagring

- Endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort

Systemverktøy

- Kjøre automatisk ved oppstart

H Tillatelser for Gmail

Oversikt over tillatelser er hentet fra applikasjonssiden til Gmail på Google Play. [29]

Dine kontoer

- Administrere kontolisten
- Bruk autentiseringsopplysningene for en konto
- Google Mail
- Oppdag kjente kontoer
- Vis konfigurerte kontoer
- Les Googles tjenestekonfigurering

Dine Meldinger

- Endre Gmail
- Lese e-poster i Gmail-kontoen
- Send Gmail-e-poster

Nettverkskommunikasjon

- Kontroller overføring av data med NFC-teknologi
- Full tilgang til Internett
- Vis nettverkstilstand
- Last ned filer uten varsling

Dine personlige opplysninger

- Les kontaktdata
- Skriv kontaktdata

Lagring

- Endre eller slette innhold på USB-lagringsenhet endre eller slette innhold på SD-kort

Systemverktøy

- Hindre nettbrettet i å gå over i hvilemodus hindre telefonen i å gå over i hvilemodus
- Skrive synkroniseringsinnst.
- Skriv abonnerte innmatinger
- Kjøre automatisk ved oppstart
- Les abonnerte innmatinger
- Les synkroniseringsstatistikk
- Lese synkroniseringsinnst.

Maskinvarekontroller

- Kontroller vibrator

Standard

- Tilgang til innspilt lyd

I Base64-kodet Protocul Buffer fra Google Play

Kodet:

CuECCosCRFFBQUFNQUFBQUFsZl81bVNhU3JmTl9ZWDNsODVqeE1YRjRRTG9uZlpjbnRDNGFKQ
mx0NzEwbVVkU055bDAwekxkWWotaTJTb0QwaDE3TG83THppcTlCUGYybE84eWNGT0w4dzJOcmE
wN2xpSnNvN1RycGEya1MtV0czUkZWVGl2TDM0OHRvb1Z4MjZkQjR2YmM3aXhQNC1ZOV9fbVhN
ejBNUlZXZ0hfMll5bzBtcXQ4MnZCTC1EdEZrOWZoY1Z0QnU2UXNITTBSREhsYUhhU2pjdzY3Qngta
TVTa1pYNmVxSFpXR0ZBRzFOR0EwLTdmVDNXdi03UWlPaDB6RWJQZ2lYRVhUXzJrdzE0EAEYiPr
oAyIQMzQwZWYyNWU3Y2Y0N2I1ZSoKcGFzc2lvbjoxMDICZW46AlVTQgBKAFIAWgBiCmFtLXVua
25vd25qEDI5NGUxMDM3MzQwYThhOTcTmgGvFBL0EwgDEAEYAyACKAAwATjwAUCAgAhKE2Fu
ZHJvaWQudGVzdC5ydW5uZXJKCmphdmF4Lm9iZXhKIGNvbS5hbmRyb2lkLmZ1dHVyZS51c2IuYWNj
ZXNzb3J5Shdjb20uZ29vZ2xlLmFuZHJvaWQubWFwc0odY29tLmFuZHJvaWQubG9jYXRpb24ucHJvdml
kZXJSFWFuZHJvaWQuaGFyZHdhcmUud2lmaVIhYW5kcm9pZC5oYXJkd2FyZS5sb2NhdGlvbi5uZXR3
b3JrUidjb20uZ29vZ2xlLmFuZHJvaWQuZmVhdHVyZS5HT09HTEVfQlVJTERSGmFuZHJvaWQuaGFyZ
HdhcmUudGVsZXBob255UhlhbmRyb2lkLmhhcmR3YXJlLmxvY2F0aW9uUhRhbmRyb2lkLnNvZnR3YX
JlLnNpcFInYW5kcm9pZC5oYXJkd2FyZS50b3VjaHNjcmVlbi5tdWx0aXRvdWNoUh9hbmRyb2lkLmhhc
mR3YXJlLnNlbnNvci5jb21wYXNzUhdhbmRyb2lkLmhhcmR3YXJlLmNhbWVyYVIeYW5kcm9pZC5oY
XJkd2FyZS51c2IuYWNjZXNzb3J5UhphbmRyb2lkLmhhcmR3YXJlLmJsdWV0b290aFIZYW5kcm9pZC5
zb2Z0d2FyZS5zaXAudm9pcFIhYW5kcm9pZC5oYXJkd2FyZS5zZW5zb3IucHJveGltaXR5Uh1hbmRyb2lk
LmhhcmR3YXJlLnNlbnNvci5saWdodFIbYW5kcm9pZC5oYXJkd2FyZS5taWNyb3Bob25lUh1hbmRyb2lk
LmhhcmR3YXJlLmxvY2F0aW9uLmdwc1IeYW5kcm9pZC5oYXJkd2FyZS50ZWxlcGhvbnkuZ3NtUiFhb
mRyb2lkLmhhcmR3YXJlLmNhbWVyYS5hdXRvZm9jdXNSH2FuZHJvaWQuc29mdHdhcmUubGl2ZV93
YWxscGFwZXJSJWFuZHJvaWQuaGFyZHdhcmUuc2Vuc29yLmFjY2VsZXJvbWV0ZXJSHGFuZHJvaWQ
uaGFyZHdhcmUudG91Y2hzY3JlZW5SHWFuZHJvaWQuaGFyZHdhcmUuY2FtZXJhLmZsYXNoWgthc
m1lYWJpLXY3YVoHYXJtZWFiaWDgA2igBnIAcgJjYXICZGFyAmphcgJuYnICZGVyAmJncgJmaXICd
mlyAnNrcgJ1a3ICZWxyAm5scgJwbHICc2xyAnRscgJybXICaW5yAmtvcgJyb3ICZnJyAmhycgJzcnICdHJ
yAmNzcgJlc3ICaXRyAmx0cgJwdHICaHVyAnJ1cgJsdnICc3ZyAml3cgV1a19VQXIFZW5fR0JyBWluX0l
EcgViZ19CR3IFZmlfRklyBXNsX1NJcgVza19TS3IFemhfQ05yBXZpX1ZOcgVyb19ST3IFaHJfSFJyBWN
hX0VTcgVzcl9SU3IFZW5fVVNyBWVzX1VTcgVsdF9MVHIFcHRfUFRyBWh1X0hVcgVsdl9MVnIFem
hfVFdyBWRlX0RFcgVzdl9TRXIFcm1fQ0hyBXRsX1BIcgVkYV9ES3IFaXdfSUxyBW5sX05McgVwbF9
QTHIFbmJfTk9yBWphX0pQcgVwdF9CUnIFZnJfRlJyBWVsX0dScgVrb19LUnIFdHJfVFJyBWVzX0VTc
gVpdF9JVHIFcnVfUlVyBWNzX0NacgJlbnICZmFyAmJlcgJhZnICdGhyAmhpcgJhbXICYXJyAm1zcgJld
HICenVyAnN3eh1HTF9BTURfY29tcHJlc3NlZF8zRENfdGV4dHVyZXodR0xfQU1EX2NvbXByZXNzZ
WRfQVRDX3RleHR1cmV6GkdMX0FNRF9wZXJmb3JtYW5jZV9tb25pdG9yehpHTF9BTURfcHJvZ3Jhb
V9iaW5hcnlfWjQwMHohR0xfRVhUX3RleHR1cmVfZmlsdGVyX2FuaXNvdHJvcGljeh5HTF9FWFRfdGV
4dHVyZV9mb3JtYXRfQkdSQTg4ODh6IkdMX0VYVF90ZXh0dXJlX3R5cGVfMl8xMF8xMF8xMF9SRV
Z6C0dMX05WX2ZlbmNlehBHTF9PRVNfRUdMX2ltYWdleh5HTF9PRVNfYmxlbmRfZXF1YXRpb25fc2
VwYXJhdGV6GkdMX09FU19ibGVuZF9mdW5jX3NlcGFyYXRlehVHTF9PRVNfYmxlbmRfc3VidHJhY3
R6I0dMX09FU19jb21wcmVzc2VkX0VUQzFfUkdCOF90ZXh0dXJleiJHTF9PRVNfY29tcHJlc3NlZF9wY
WxldHRlZF90ZXh0dXJleg5HTF9PRVNfZGVwdGgyNHoUR0xfT0VTX2RlcHRoX3RleHR1cmV6E0dMX
09FU19kcmF3X3RleHR1cmV6GUdMX09FU19lbGVtZW50X2luZGV4X3VpbnR6GEdMX09FU19mYm9
fcmVuZGVyX21pcG1hcHoeR0xfT0VTX2ZyYWdtZW50X3ByZWNpc2lvbl9oaWdoehlHTF9PRVNfZnJhb
WVidWZmZXJfb2JqZWN0ehlHTF9PRVNfZ2V0X3Byb2dyYW1fYmluYXJ5ehVHTF9PRVNfbWF0cml4
X3BhbGV0dGV6G0dMX09FU19wYWNrZWRfZGVwdGhfc3RlbmNpbHoXR0xfT0VTX3BvaW50X3Npe
mVfYXJyYXl6E0dMX09FU19wb2ludF9zcHJpdGV6EkdMX09FU19yZWFkX2Zvcm1hdHoRR0xfT0VTX
3JnYjhfcmdiYTh6G0dMX09FU19zdGFuZGFyZF9kZXJpdmF0aXZlc3oTR0xfT0VTX3N0ZW5jaWxfd3Jh
cHoRR0xfT0VTX3RleHR1cmVfM0R6F0dMX09FU190ZXh0dXJlX2N1YmVfbWFwehtHTF9PRVNfdGV
4dHVyZV9lbnZfY3Jvc3NiYXJ6FEdMX09FU190ZXh0dXJlX2Zsb2F0ehlHTF9PRVNfdGV4dHVyZV9oY
WxmX2Zsb2F0eiBHTF9PRVNfdGV4dHVyZV9oYWxmX2Zsb2F0X2xpbmVhcnoeR0xfT0VTX3RleHR1c
mVfbWlycm9yZWRfcmVwZWF0ehNHTF9PRVNfdGV4dHVyZV9ucG90ehhHTF9PRVNfdmVydGV4X2h
hbGZfZmxvYXR6HUdMX09FU192ZXJ0ZXhfdHlwZV8xMF8xMF8xMF8yehZHTF9RQ09NX2RyaXZlcl
9jb250cm9sehRHTF9RQ09NX2V4dGVuZGVkX2dldHoVR0xfUUNPTV9leHRlbmRlZF9nZXQyehZHTF9
RQ09NX21lbW9yeV9tb25pdG9yehtHTF9RQ09NX3BlcmZtb25fZ2xvYmFsX21vZGV6F0dMX1FDT01fd
GlsZWRfcmVuZGVyaW5nehtHTF9RQ09NX3dyaXRlb25seV9yZW5kZXJpbmcYACIWemVuMklJMW5L
MVN4MnN3TGNDbjE2dzIXRnVsbCBBbmRyb2lkIG9uIFBhc3Npb246A0hUQxQ&

1/2

Dekodet:

á#
‹#DQAAAMAAAAAlf_5mSaSrfN_YX3l85jxMXF4QLonfZcntC4aJBlt710mUdSNyl00zLdYj-i2SoD0h17Lo7Lziq9BPf2lO8ycFOL8w2Nra07liJso7Trpa2kS-WG3RFVTivL348tooVx26dB4vbc7ixP4-Y9__mXMz0MRVWgH_2Yyo0mqt82vBL-DtFk9fhcVtBu6QsHM0RDHlaHaSjcw67Bx-i5SkZX6eqHZWGFAG1NGA0-7fT3Wv-7QiOh0zEbPgiXEXT_2kw14###ˆúè#"#340ef25e7cf47b5e*
passion:102#en:#USB?J?R?Z?b
am-unknownj#294e1037340a8a97#š#¯##ô####### #(?0#8ð#@€€#J#android.test.runnerJ
javax.obexJ
com.android.future.usb.accessoryJ#com.google.android.mapsJ#com.android.location.providerR#android.hardware.wifiR!
android.hardware.location.networkR'com.google.android.feature.GOOGLE_BUILDR#android.hardware.telephonyR#android.hardware.locationR#android.software.sipR'android.hardware.touchscreen.multitouchR#android.hardware.sensor.compassR#android.hardware.cameraR#android.hardware.usb.accessoryR#android.hardware.bluetoothR#android.software.sip.voipR!
android.hardware.sensor.proximityR#android.hardware.sensor.lightR#android.hardware.microphoneR#android.hardware.location.gpsR#android.hardware.telephony.gsmR!
android.hardware.camera.autofocusR#android.software.live_wallpaperR
%android.hardware.sensor.accelerometerR#android.hardware.touchscreenR#android.hardware.camera.flashZ#armeabi-v7aZ#armeabi`à#h #r?
r#car#dar#jar#nbr#der#bgr#fir#vir#skr#ukr#elr#nlr#plr#slr#tlr#rmr#inr#kor#ror#frr#hrr#srr#trr#csr#esr#itr#ltr#ptr#hur#rur#lvr#svr#iwr#uk_UAr#en_GBr#in_IDr#bg_BGr#fi_FIr#sl_SIr#sk_SKr#zh_CNr#vi_VNr#ro_ROr#hr_HRr#ca_ESr#sr_RSr#en_USr#es_USr#lt_LTr#pt_PTr#hu_HUr#lv_LVr#zh_TWr#de_DEr#sv_SEr#rm_CHr#tl_PHr#da_DKr#iw_ILr#nl_NLr#pl_PLr#nb_NOr#ja_JPr#pt_BRr#fr_FRr#el_GRr#ko_KRr#tr_TRr#es_ESr#it_ITr#ru_RUr#cs_CZr#enr#far#ber#afr#thr#hir#amr#arr#msr#etr#zur#swz#GL_AMD_compressed_3DC_texturez#GL_AMD_compressed_ATC_texturez#GL_AMD_performance_monitorz#GL_AMD_program_binary_Z400z!
GL_EXT_texture_filter_anisotropicz#GL_EXT_texture_format_BGRA8888z"GL_EXT_texture_type_2_10_10_10_REVz#GL_NV_fencez#GL_OES_EGL_imagez#GL_OES_blend_equation_separatez#GL_OES_blend_func_separatez#GL_OES_blend_subtractz#GL_OES_compressed_ETC1_RGB8_texturez"GL_OES_compressed_paletted_texturez#GL_OES_depth24z#GL_OES_depth_texturez#GL_OES_draw_texturez#GL_OES_element_index_uintz#GL_OES_fbo_render_mipmapz#GL_OES_fragment_precision_highz#GL_OES_framebuffer_objectz#GL_OES_get_program_binaryz#GL_OES_matrix_palettez#GL_OES_packed_depth_stencilz#GL_OES_point_size_arrayz#GL_OES_point_spritez#GL_OES_read_formatz#GL_OES_rgb8_rgba8z#GL_OES_standard_derivativesz#GL_OES_stencil_wrapz#GL_OES_texture_3Dz#GL_OES_texture_cube_mapz#GL_OES_texture_env_crossbarz#GL_OES_texture_floatz#GL_OES_texture_half_floatz
GL_OES_texture_half_float_linearz#GL_OES_texture_mirrored_repeatz#GL_OES_texture_npotz#GL_OES_vertex_half_floatz#GL_OES_vertex_type_10_10_10_2z#GL_QCOM_driver_controlz#GL_QCOM_extended_getz#GL_QCOM_extended_get2z#GL_QCOM_memory_monitorz#GL_QCOM_perfmon_global_modez#GL_QCOM_tiled_renderingz#GL_QCOM_writeonly_rendering#?"#zen2II1nK1Sx2swLcCn16w2#Full Android on Passion:#HT

J Protocol Buffer sendt ved første tilkobling til Internett

```
STS354957030935491DLENULSUBNUL"x84x8A
xC6SOH
Ugeneric/full_passion/passion:2.3.4/GRJ22/eng.taintdroid.20120207.104554:eng/test-keysDC2BSmah
imahiSUBBELgeneric"BELunknown*    0.35.00172BELunknown8xCAxE6xC3xF9E0T@
JBELpassionP
ZFFTBFull Android on PassionbFFTXHTCjFFFFfull_passionDLENULSUB'STX
NAKsystem_app_strictmodeDC2xAAx93STXProcess: system_server
Build: generic/full_passion/passion:2.3.4/GRJ22/eng.taintdroid.20120207.104554:eng/test-keys
System-App: true
Uptime-Millis: 45247
Loop-Violation-Number: 1
Duration-Millis: 112

android.os.StrictMode$StrictModeDiskReadViolation: policy=135 violation=2
    at android.os.StrictMode$AndroidBlockGuardPolicy.onReadFromDisk(StrictMode.java:745)
    at dalvik.system.BlockGuard$WrappedFileSystem.open(BlockGuard.java:228)
    at java.io.RandomAccessFile.<init>(RandomAccessFile.java:132)
    at java.io.RandomAccessFile.<init>(RandomAccessFile.java:173)
    at java.util.zip.ZipFile.<init>(ZipFile.java:133)
    at java.util.zip.ZipFile.<init>(ZipFile.java:99)
    at dalvik.system.PathClassLoader.<init>(PathClassLoader.java:131)
    at android.app.ApplicationLoaders.getClassLoader(ApplicationLoaders.java:57)
    at android.app.LoadedApk.getClassLoader(LoadedApk.java:288)
    at android.app.ActivityThread.handleCreateService(ActivityThread.java:1925)
    at android.app.ActivityThread.access$2500(ActivityThread.java:117)
    at android.app.ActivityThread$H.handleMessage(ActivityThread.java:985)
    at android.os.Handler.dispatchMessage(Handler.java:99)
    at android.os.Looper.loop(Looper.java:130)
    at com.android.server.ServerThread.run(SystemServer.java:542)



(Utelatt av plasshensyn)



CANxD1xE4HxEE&SUBxC2'
DC3SYSTEM_RECOVERY_LOGDC2xA3'Build:
generic/full_passion/passion:2.3.4/GRJ22/eng.taintdroid.20120207.104554:eng/test-keys
Hardware: mahimahi
Bootloader: 0.35.0017
Radio: unknown
Kernel: Linux version 2.6.35.7-59463-g52630a3 (android-build@apa28.mtv.corp.google.com) (gcc
version 4.4.3 (GCC) ) #1 PREEMPT Tue Mar 29 11:49:12 PDT 2011

Starting recovery on Tue Apr 24 08:56:41 2012
can't open /dev/tty0: No such file or directory
framebuffer: fd 3 (480 x 800)
ClockworkMod Recovery v5.0.2.0
recovery filesystem table
=========================
  0 /tmp ramdisk (null) (null)
  1 /boot mtd boot (null)
  2 /cache yaffs2 cache (null)
  3 /data yaffs2 userdata (null)
  4 /misc mtd misc (null)
  5 /recovery mtd recovery (null)
```

```
  6 /sdcard vfat /dev/block/mmcblk0p1 /dev/block/mmcblk0
  7 /system yaffs2 system (null)
  8 /sd-ext ext4 /dev/block/mmcblk0p2 (null)

I:Completed outputting fstab.
I:Processing arguments.
Fixing execute permissions for /cache
mtd: successfully wrote block at 0
I:Set boot command "boot-recovery"
I:Checking arguments.
I:device_recovery_start()
Command: "/sbin/recovery"

ro.secure=0
ro.allow.mock.location=1
ro.debuggable=1
persist.service.adb.enable=1
ro.build.id=GRJ22
ro.build.display.id=GRJ90
ro.build.version.incremental=eng.koush.20110905.103150
ro.build.version.sdk=10
ro.build.version.codename=REL
ro.build.version.release=2.3.5
ro.build.date=Mon Sep  5 10:32:20 PDT 2011
ro.build.date.utc=0
ro.build.type=eng
ro.build.user=koush
ro.build.host=Koushik-Lion.local
ro.build.tags=test-keys
ro.product.model=Nexus One
ro.product.brand=google
ro.product.name=passion
ro.product.device=passion
ro.product.board=mahimahi
ro.product.cpu.abi=armeabi-v7a
ro.product.cpu.abi2=armeabi
ro.product.manufacturer=HTC
ro.product.locale.language=en
ro.product.locale.region=US
ro.wifi.channels=
ro.board.platform=qsd8k
ro.build.product=passion
ro.build.description=passion-user 2.3.4 GRJ22 121341 release-keys
ro.build.fingerprint=google/passion/passion:2.3.4/GRJ22/121341:user/release-keys
keyguard.no_require_sim=true
ro.sf.lcd_density=240
rild.libpath=/system/lib/libhtc_ril.so
wifi.interface=eth0
wifi.supplicant_scan_interval=15
ro.ril.hsxpa=2
ro.ril.gprsclass=10
ro.telephony.default_network=0
ro.opengles.version=131072
dalvik.vm.heapsize=32m
media.a1026.nsForVoiceRec=0
media.a1026.enableA1026=1
ro.rommanager.developerid=cyanogenmod
ro.url.legal=http://www.google.com/intl/%s/mobile/android/basic/phone-legal.html
```

```
ro.url.legal.android_privacy=http://www.google.com/intl/%s/mobile/android/basic/privacy.html
ro.com.google.clientidbase=android-google
ro.com.android.wifi-watchlist=GoogleGuest
ro.setupwizard.enterprise_mode=1
ro.com.android.dateformat=MM-dd-yyyy
ro.com.android.dataroaming=false
ro.config.ringtone=Playa.ogg
ro.config.notification_sound=regulus.ogg
ro.config.alarm_alert=Alarm_Beep_03.ogg
ro.ril.enable.managed.roaming=1
ro.ril.oem.nosim.ecclist=911,112,999,000,08,118,120,122,110,119,995
ro.ril.emc.mode=2
ro.modversion=CyanogenMod-7.1.0-RC1-N1-KANG
ro.kernel.android.checkjni=1
ro.setupwizard.mode=OPTIONAL
dalvik.vm.dexopt-flags=m=y
net.bt.name=Android
net.change=net.bt.name
dalvik.vm.stack-trace-file=/data/anr/traces.txt
ro.factorytest=0
ro.serialno=HT9CSP815349
ro.bootmode=recovery
ro.baseband=5.08.00.04
ro.carrier=GOOGLE
ro.bootloader=0.35.0017
ro.hardware=mahimahi
ro.revision=129
init.svc.recovery=running
init.svc.adbd=running

I:Checking for extendedcommand...
I:Skipping execution of extendedcommand, file not found...
mtd: successfully wrote block at 0
I:Set boot command ""

-- Wiping data...
Formatting /data...
mtd: not erasing bad block at 0x00620000
mtd: not erasing bad block at 0x01100000
mtd: not erasing bad block at 0x01920000
mtd: not erasing bad block at 0x06620000
mtd: not erasing bad block at 0x08880000
Formatting /cache...
mtd: not erasing bad block at 0x027c0000
mtd: not erasing bad block at 0x02900000
mtd: not erasing bad block at 0x02c60000
mtd: not erasing bad block at 0x03900000
mtd: not erasing bad block at 0x03ce0000
mtd: not erasing bad block at 0x04580000
Formatting /sd-ext...
Need size of filesystem
E:format_volume: make_extf4fs failed on /dev/block/mmcblk0p2
Formatting /sdcard/.android_secure...
I:Formatting unknown device.
rm: can't remove '.' or '..'
rm: can't remove '.' or '..'
Data wipe complete.
mtd: successfully wrote block at 0
```

```
I:Set boot command ""

-- Wiping cache...
Formatting /cache...
mtd: not erasing bad block at 0x027c0000
mtd: not erasing bad block at 0x02900000
mtd: not erasing bad block at 0x02c60000
mtd: not erasing bad block at 0x03900000
mtd: not erasing bad block at 0x03ce0000
mtd: not erasing bad block at 0x04580000
Cache wipe complete.
mtd: successfully wrote block at 0
I:Set boot command ""
W:failed to mount /dev/block/mmcblk0p2 (No such file or directory)
rm: can't remove '/data/dalvik-cache': No such file or directory
rm: can't remove '/cache/dalvik-cache': No such file or directory
rm: can't remove '/sd-ext/dalvik-cache': No such file or directory
Dalvik Cache wiped.
mtd: successfully wrote block at 0
I:Set boot command ""
CANx82xA0xEE&SUBxC8STX
VTSYSTEM_BOOTDC2xB1STXBuild:
generic/full_passion/passion:2.3.4/GRJ22/eng.taintdroid.20120207.104554:eng/test-keys
Hardware: mahimahi
Bootloader: 0.35.0017
Radio: unknown
Kernel: Linux version 2.6.35.7-59463-g52630a3 (android-build@apa28.mtv.corp.google.com) (gcc
version 4.4.3 (GCC) ) #1 PREEMPT Tue Mar 29 11:49:12 PDT 2011

CANx8CxA0xEE&SUBxB0x8BSTX
DLESYSTEM_LAST_KMSGDC2x93x8BSTXBuild:
generic/full_passion/passion:2.3.4/GRJ22/eng.taintdroid.20120207.104554:eng/test-keys
Hardware: mahimahi
Bootloader: 0.35.0017
Radio: unknown
Kernel: Linux version 2.6.35.7-59463-g52630a3 (android-build@apa28.mtv.corp.google.com) (gcc
version 4.4.3 (GCC) ) #1 PREEMPT Tue Mar 29 11:49:12 PDT 2011

[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 2.6.37.6-cyanogenmod-01509-g8913be8 (shade@toxygene) (gcc
version 4.4.3 (GCC) ) #1 PREEMPT Wed Jul 27 21:31:24 EDT 2011
[    0.000000] CPU: ARMv7 Processor [510f00f2] revision 2 (ARMv7), cr=10c53c7d
[    0.000000] CPU: VIPT nonaliasing data cache, VIVT ASID tagged instruction cache
[    0.000000] Machine: mahimahi
[    0.000000] Ignoring unrecognised tag 0x4d534d76
[    0.000000] Ignoring unrecognised tag 0x5441000a
[    0.000000] CAM_AWB_CAL Data size = 514 , 0x59504550, size = 2048
[    0.000000] Ignoring unrecognised tag 0x41387898
[    0.000000] Memory policy: ECC disabled, Data cache writeback
[    0.000000] Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 105362
[    0.000000] Kernel command line: board_mahimahi.disable_uart3=0
board_mahimahi.usb_h2w_sw=0 board_mahimahi.disable_sdcard=0 diag.enabled=0
board_mahimahi.debug_uart=0 smisize=0  androidboot.baseband=5.08.00.04
androidboot.cid=GOOGL001 androidboot.carrier=GOOGLE androidboot.mid=PB9910000
androidboot.keycaps=qwerty androidboot.mode=recovery androidboot.serialno=HT9CSP815349
androidboot.bootloader=0.35.0017 no_console_suspend=1 wire.search_count=5
[    0.000000] PID hash table entries: 2048 (order: 1, 8192 bytes)
[    0.000000] Dentry cache hash table entries: 65536 (order: 6, 262144 bytes)
```

```
[    0.000000] Inode-cache hash table entries: 32768 (order: 5, 131072 bytes)
[    0.000000] Memory: 232MB 183MB = 415MB total
[    0.000000] Memory: 413852k/413852k available, 11108k reserved, 0K highmem
[    0.000000] Virtual kernel memory layout:
[    0.000000]     vector  : 0xffff0000 - 0xffff1000   (   4 kB)
[    0.000000]     fixmap  : 0xfff00000 - 0xfffe0000   ( 896 kB)
[    0.000000]     DMA     : 0xffa00000 - 0xffe00000   (   4 MB)
[    0.000000]     vmalloc : 0xdb800000 - 0xf8000000   ( 456 MB)
[    0.000000]     lowmem  : 0xc0000000 - 0xdb700000   ( 439 MB)
[    0.000000]     modules : 0xbf000000 - 0xc0000000   (  16 MB)
[    0.000000]       .init : 0xc0008000 - 0xc002a000   ( 136 kB)
[    0.000000]       .text : 0xc002a000 - 0xc04b1000   (4636 kB)
[    0.000000]       .data : 0xc04b2000 - 0xc04e1940   ( 191 kB)
[    0.000000] NR_IRQS:316
[    0.000000] Calibrating delay loop... 509.54 BogoMIPS (lpj=2547712)
[    0.229492] pid_max: default: 32768 minimum: 301
[    0.229705] Mount-cache hash table entries: 512


(Utelatt av plasshensyn)


[    8.083923] cryptomgr_test used greatest stack depth: 6936 bytes left
[    8.084381] cryptomgr_test used greatest stack depth: 6896 bytes left
[    8.085540] io scheduler noop registered
[    8.085693] io scheduler deadline registered
[    8.085906] io scheduler cfq registered
[    8.086212] io scheduler bfq registered (default)
[    8.087463] msmfb_probe() installing 480 x 800 panel
[    8.087951] mdp_lcdc_probe: initialized
[    8.088256] kgsl kgsl: |kgsl_platform_probe| no grp_pclk, continuing
[    8.089111] msm_serial: detected port #0
[    8.089385] uartclk = 19200000
[    8.089569] msm_serial.0: ttyMSM0 at MMIO 0xa9a00000 (irq = 64) is a MSM
[    8.089843] msm_serial: console setup on port #0
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 2.6.37.6-cyanogenmod-01509-g8913be8 (shade@toxygene) (gcc
version 4.4.3 (GCC) ) #1 PREEMPT Wed Jul 27 21:31:24 EDT 2011
[    0.000000] CPU: ARMv7 Processor [510f00f2] revision 2 (ARMv7), cr=10c53c7d
[    0.000000] CPU: VIPT nonaliasing data cache, VIVT ASID tagged instruction cache
[    0.000000] Machine: mahimahi
[    0.000000] find the smi tag
[    0.000000] parse_tag_smi: smi size = 0
[    0.000000] find the hwid tag
[    0.000000] parse_tag_hwid: hwid = 0x0
[    0.000000] find the skuid tag
[    0.000000] parse_tag_skuid: hwid = 0x21f01
[    0.000000] tag_panel_parsing: panel type = 0
[    0.000000] find the engineer tag
[    0.000000] parse_tag_engineerid: 0x0
[    0.000000] Ignoring unrecognised tag 0x4d534d76
[    0.000000] Ignoring unrecognised tag 0x5441000a
[    0.000000] CAM_AWB_CAL Data size = 514 , 0x59504550, size = 2048
[    0.000000] Ignoring unrecognised tag 0x41387898
[    0.000000] Memory policy: ECC disabled, Data cache writeback
[    0.000000] On node 0 totalpages: 106240
```

```
[    0.000000] free_area_init_node: node 0, pgdat c04e12ac, node_mem_map c05ee000
[    0.000000]   Normal zone: 878 pages used for memmap
[    0.000000]   Normal zone: 0 pages reserved
[    0.000000]   Normal zone: 105362 pages, LIFO batch:31
[    0.000000] pcpu-alloc: s0 r0 d32768 u32768 alloc=1*32768
[    0.000000] pcpu-alloc: [0] 0
[    0.000000] Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 105362
[    0.000000] Kernel command line: board_mahimahi.disable_uart3=0
board_mahimahi.usb_h2w_sw=0 board_mahimahi.disable_sdcard=0 diag.enabled=0
board_mahimahi.debug_uart=0 smisize=0  androidboot.baseband=5.08.00.04
androidboot.cid=GOOGL001 androidboot.carrier=GOOGLE androidboot.mid=PB9910000
androidboot.keycaps=qwerty androidboot.mode=recovery androidboot.serialno=HT9CSP815349
androidboot.bootloader=0.35.0017 no_console_suspend=1 wire.search_count=5
[    0.000000] PID hash table entries: 2048 (order: 1, 8192 bytes)
[    0.000000] Dentry cache hash table entries: 65536 (order: 6, 262144 bytes)
[    0.000000] Inode-cache hash table entries: 32768 (order: 5, 131072 bytes)
[    0.000000] Memory: 232MB 183MB = 415MB total
[    0.000000] Memory: 413852k/413852k available, 11108k reserved, 0K highmem
[    0.000000] Virtual kernel memory layout:
[    0.000000]     vector  : 0xffff0000 - 0xffff1000   (   4 kB)
[    0.000000]     fixmap  : 0xfff00000 - 0xfffe0000   ( 896 kB)
[    0.000000]     DMA     : 0xffa00000 - 0xffe00000   (   4 MB)
[    0.000000]     vmalloc : 0xdb800000 - 0xf8000000   ( 456 MB)
[    0.000000]     lowmem  : 0xc0000000 - 0xdb700000   ( 439 MB)
[    0.000000]     modules : 0xbf000000 - 0xc0000000   (  16 MB)
[    0.000000]       .init : 0xc0008000 - 0xc002a000   ( 136 kB)
[    0.000000]       .text : 0xc002a000 - 0xc04b1000   (4636 kB)
[    0.000000]       .data : 0xc04b2000 - 0xc04e1940   ( 191 kB)
[    0.000000] NR_IRQS:316
[    0.000000] Calibrating delay loop... 509.54 BogoMIPS (lpj=2547712)
[    0.229492] pid_max: default: 32768 minimum: 301
[    0.229705] Mount-cache hash table entries: 512
```

(Utelatt av plasshensyn)

```
[    9.578094] rndis_function_bind_config MAC: 02:2C:55:08:76:60
[    9.584045] android_usb gadget: using random self ethernet address
[    9.589996] android_usb gadget: using random host ethernet address
[    9.596618] usb0: MAC 7e:47:4e:9b:d6:97
[    9.599975] usb0: HOST MAC 1e:f5:bd:da:cb:17
[    9.604278] msm_hsusb_phy_reset: success
[    9.608459] acc_bind_config
[    9.611419] android_usb gadget: Mass Storage Function, version: 2009/09/11
[    9.617919] android_usb gadget: Number of LUNs=1
[    9.622406]  lun0: LUN: removable file: (no medium)
[    9.627349] adb_bind_config
[    9.630432] diag_bind_config
[    9.632965] SMD: ch 40 1 -> 2
[    9.635803] SMD: ch 1 1 -> 2
[    9.639282] mahimahi_ts_power: power 1
[    9.792419] usb: notify offline
[    9.792602] msm_hsusb: enable pullup
[    9.792785] msm72k_udc: ONLINE -> OFFLINE
[    9.793090] msm_hsusb: disable pullup
```

```
[    9.793273] ### notify_usb_connected(0) ###
[    9.793457] usb: notify offline
[    9.794433] msm_i2c msm_i2c.0: error, status 43c8 (40)
[    9.794647] msm_i2c msm_i2c.0: Error during data xfer (-5) @40
[    9.832427] msm_hsusb_phy_reset: success
[    9.903045] synaptics_ts_probe: Product Major Version 1


(Utelatt av plasshensyn)


[   31.066864] yaffs: block 488 is bad
[   31.086639] yaffs: block 557 is bad
[   31.146575] yaffs: yaffs_read_super: is_checkpointed 0
[   35.330413] yaffs: dev is 32505861 name is "mtdblock5" rw
[   35.330627] yaffs: passed flags ""
[   35.330810] yaffs: Attempting MTD mount of 31.5,"mtdblock5"
[   35.573852] yaffs: block 50 is bad
[   35.598541] yaffs: block 137 is bad
[   35.617218] yaffs: block 202 is bad
[   35.792724] yaffs: block 818 is bad
[   35.871154] yaffs: block 1093 is bad
[   36.006988] yaffs: yaffs_read_super: is_checkpointed 0
[   43.745666] msmfb_shutdown
[   43.745849] lcdc_blank: ()
[   43.746032] samsung_oled_panel_blank: +()
[   43.953094] samsung_oled_panel_blank: -()
[   43.953277] lcdc_shutdown: ()
[   43.953552] Restarting system.
[   43.953857]
[   43.954010] Restarting Linux version 2.6.37.6-cyanogenmod-01509-g8913be8 (shade@toxygene)
(gcc version 4.4.3 (GCC) ) #1 PREEMPT Wed Jul 27 21:31:24 EDT 2011
[   43.954040]

No errors detected
CANxFFxA0xFF&SUBxF8o
NAKsystem_app_strictmodeDC2xC7oProcess: com.android.inputmethod.latin
Flags: 0x8be45
Package: com.android.inputmethod.latin v10 (2.3.4)
Build: generic/full_passion/passion:2.3.4/GRJ22/eng.taintdroid.20120207.104554:eng/test-keys
System-App: true
Uptime-Millis: 51562
Loop-Violation-Number: 8
Duration-Millis: 4061

android.os.StrictMode$StrictModeDiskReadViolation: policy=135 violation=2
    at android.os.StrictMode$AndroidBlockGuardPolicy.onReadFromDisk(StrictMode.java:745)
    at dalvik.system.BlockGuard$WrappedFileSystem.read(BlockGuard.java:164)
    at java.io.RandomAccessFile.read(RandomAccessFile.java:348)
    at java.util.zip.ZipFile$RAFStream.read(ZipFile.java:427)
    at java.io.InputStream.read(InputStream.java:157)
    at java.io.BufferedInputStream.fillbuf(BufferedInputStream.java:140)
    at java.io.BufferedInputStream.read(BufferedInputStream.java:324)
    at java.util.zip.ZipEntry.myReadFully(ZipEntry.java:417)
    at java.util.zip.ZipEntry.<init>(ZipEntry.java:389)
    at java.util.zip.ZipFile.readCentralDir(ZipFile.java:380)
```

```
    at java.util.zip.ZipFile.<init>(ZipFile.java:135)
    at java.util.zip.ZipFile.<init>(ZipFile.java:99)
    at dalvik.system.PathClassLoader.<init>(PathClassLoader.java:131)
    at android.app.ApplicationLoaders.getClassLoader(ApplicationLoaders.java:57)
    at android.app.LoadedApk.getClassLoader(LoadedApk.java:288)
    at android.app.LoadedApk.makeApplication(LoadedApk.java:458)
    at android.app.ActivityThread.handleBindApplication(ActivityThread.java:3260)
    at android.app.ActivityThread.access$2200(ActivityThread.java:117)
    at android.app.ActivityThread$H.handleMessage(ActivityThread.java:969)
    at android.os.Handler.dispatchMessage(Handler.java:99)
    at android.os.Looper.loop(Looper.java:130)
    at android.app.ActivityThread.main(ActivityThread.java:3683)
    at java.lang.reflect.Method.invokeNative(Native Method)
    at java.lang.reflect.Method.invoke(Method.java:507)
    at com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:839)
    at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:597)
    at dalvik.system.NativeStart.main(Native Method)


(Utelatt av plasshensyn)


CAN ¸ xEE&BDC1mobile-notroaming2ENOen_US8xF7x80xA2xAFxA0ŠxF8x92SOHJFF002376986525ZNULbETXGMTp
STXzFS71Q6Rn2DDZl1zPDVaaeEHItd+Yg=x82SOHFFHT9CSP815349x92SOHxF3 BSETXDLESOHCANETX
STX(NUL0SOH8xE0SOH@x80x80BSJDC3android.test.runnerJ
com.android.future.usb.accessoryJGScom.android.location.providerJETBcom.google.android.mapsJ
javax.obexRSUBandroid.hardware.bluetoothRETBandroid.hardware.cameraR!android.hardware.camera.a
utofocusRGSandroid.hardware.camera.flashREMandroid.hardware.locationRGSandroid.hardware.locati
on.gpsR!android.hardware.location.networkRESCandroid.hardware.microphoneR%android.hardware.sen
sor.accelerometerRUSandroid.hardware.sensor.compassRGSandroid.hardware.sensor.lightR!android.h
ardware.sensor.proximityRSUBandroid.hardware.telephonyRRSandroid.hardware.telephony.gsmRFSandr
oid.hardware.touchscreenR'android.hardware.touchscreen.multitouchRRSandroid.hardware.usb.acces
soryRNAKandroid.hardware.wifiRUSandroid.software.live_wallpaperRDC4android.software.sipREMandr
oid.software.sip.voipR'com.google.android.feature.GOOGLE_BUILDZVTarmeabi-v7aZBELarmeabi`XE0ETX
hXA0ACKrSTXbgrENObg_BGrSTXcarENOca_ESrSTXcsrENOcs_CZrSTXdarENOda_DKrSTXderENOde_DErSTXelrENOe
l_GRrSTXenrENOen_GBrENOen_USrSTXesrENOes_ESrENOes_USrSTXfirENOfi_FIrSTXfrrENOfr_FRrSTXhrrENOh
r_HRrSTXhurENOhu_HUrSTXinrENOin_IDrSTXitrENOit_ITrSTXiwrENOiw_ILrSTXjarENOja_JPrSTXkorENOko_K
RrSTXltrENOlt_LTrSTXlvrENOlv_LVrSTXnbrENOnb_NOrSTXnlrENOnl_NLrSTXplrENOpl_PLrSTXptrENOpt_BRr
ENOpt_PTrSTXrmrENOrm_CHrSTXrorENOro_ROrSTXrurENOru_RUrSTXskrENOsk_SKrSTXslrENOsl_SIrSTXsrrENO
sr_RSrSTXsvrENOsv_SErSTXtlrENOtl_PHrSTXtrrENOtr_TRrSTXukrENOuk_UArSTXvirENOvi_VNrENOzh_CNrENO
zh_TW
```

K Protocol Buffer med variabler fra Google

```
BSSOHCAN-ö“™ï&"CANNS0Glc3KqumKopFIDHoZ3Q==*GS
NAKallow_enterprise_modeDC2EOTtrue*!

android_idDC2DC3459189438715514161O*DC4
STc2dm_auth_tokenDC2SOH1*RS
SYNcheckin_dropbox_uploadDC2EOTtrue*2
*checkin_dropbox_upload:SYSTEM_RECOVERY_LOGDC2EOTtrue*(
 checkin_dropbox_upload:event_logDC2EOTtrue*SUB
DLEcheckin_intervalDC2ACK589331*DC4
SOdevice_countryDC2STXno*SUB
DC3event:system_updateDC2ETXlog*US
CANevent:system_update_userDC2ETXlog*Z
ETBfeedback_tos_pretty_urlDC2?https://tools.google.com/userfeedback/external/android/tos.html*
S
DLEfeedback_tos_urlDC2?https://tools.google.com/userfeedback/external/android/tos.html*%
GSfinsky.content_rating_enabledDC2EOTtrue**
!finsky.purchase_status_timeout_msDC2ENQ60000*"
GSgmail_discard_error_uphill_opDC2SOH1*&
!gmail_discard_error_uphill_op_newDC2SOH1*ETB

gmail_hostDC2   gmail.com*'
FSgmail_max_preview_image_sizeDC2BEL2097152*US
EMgmail_num_retry_uphill_opDC2STX45*!
FSgmail_use_multipart_protobufDC2SOH1*)
USgmail_wait_time_retry_uphill_opDC2ACK129600*,
USgms_disable:com.android.vendingDC2    0:enabled*6
)gms_disable:com.google.android.apps.booksDC2   0:enabled*<
/gms_disable:com.google.android.apps.googlevoiceDC2 0:enabled*5
(gms_disable:com.google.android.apps.mapsDC2    0:enabled*=
0gms_disable:com.google.android.apps.walletnfcrelDC2    0:enabled*2
%gms_disable:com.google.android.videosDC2   0:enabled*7
*gms_disable:com.google.android.voicesearchDC2  0:enabled*3
&gms_disable:com.google.android.youtubeDC2  0:enabled*)
ESgms_disable:com.google.earthDC2   0:enabled*FSC
SYNgms_disable:lock_countDC2SOH4*)
!google_login_generic_auth_serviceDC2EOTmail*ØSOH
ETBgoogle_login_public_keyDC2¼SOHAAAAgMom/1a/v0lblO2Ubrt60J2gcuXSljGFQXgcyZWveWLEwo6prwgi3iJIZ
dodyhKZQrNWp5nKJ3srRXcUW+F1BD3baEVGcmEgqaLZUNBjm057pKRI16kB0YppeGx5qIQ5QjKzsR8ETQbKLNWgRY0QRNV
z34kMJR3P/LgHax/6rmf5AAAAAwEAAQ==*{
DC2google_services:ahDC2eGoogle Apps||Allows applications to sign in to Google Apps using
the account(s) stored on this phone.*•SOH
CANgoogle_services:doraemonDC2yGoogle Catalogs||Allows applications to sign in to the Google
Catalogs service using the account(s) stored on this phone.*†SOH
ETBgoogle_services:financeDC2kGoogle Finance||Allows applications to sign in to Google
Finance using the account(s) stored on this phone.*–SOH
ETBgoogle_services:geowikiDC2{Google Map maker||Allows applications to sign in to the Google
Map maker service using the account(s) stored on this phone.*”SOH
GSgoogle_services:goanna_mobileDC2sGoogle Tasks||Allows applications to sign in to the
Google Tasks service using the account(s) stored on this phone.*‡SOH
FSgoogle_services:grandcentralDC2gGoogle Voice||Allows applications to sign in to Google
Voice using the account(s) stored on this phone.*~
NAKgoogle_services:localDC2eGoogle Maps||Allows applications to sign in to Google Maps using
the account(s) stored on this phone.*‰SOH
CANgoogle_services:notebookDC2mGoogle Notebook||Allows applications to sign in to Google
Notebook using the account(s) stored on this phone.*ŠSOH
EMgoogle_services:panoramioDC2mPanoramio||Allows applications to sign in to the Panoramio
service using the account(s) stored on this phone.*SOH
SYNgoogle_services:readerDC2uGoogle Reader||Allows applications to sign in to the Google
```

Reader service using the account(s) stored on this phone.*ÃSOH
%google_services:speechpersonalizationDC2™SOHPersonalized Speech Recognition||Allows applications to sign in to the Personalized Speech Recognition service using the account(s) stored on this phone.*2
'gtalk_active_heartbeat_ping_interval_msDC2BEL1680000*CAN
STgtalk_auth_saslDC2ENOfalse*GS
NAKgtalk_binary_protocolDC2EOTtrue*'
GSgtalk_clock_skew_threshold_msDC2ACK300000*ETB
SOgtalk_compressDC2ENOfalse*w
ESCgtalk_flickr_photo_info_urlDC2Xhttp://api.flickr.com/services/rest/?method=flickr.photos.getinfo&photo_id=%s&api_key=%s*H
SYNgtalk_flickr_photo_urlDC2.http://farm%s.static.flickr.com/%s/%s_%s_m.jpg*+
 gtalk_heartbeat_ping_interval_msDC2BEL1680000*"
SOgtalk_hostnameDC2DLEmtalk.google.com*RS
NAKgtalk_idle_timeout_msDC2ENO30000*)
USgtalk_max_server_heartbeat_timeDC2ACK900000*2
'gtalk_nosync_heartbeat_ping_interval_msDC2BEL1680000*T
SYNgtalk_picasa_album_urlDC2:http://picasaweb.google.com/data/feed/api/user/%s/album/%s*FS
SYNgtalk_rmq_ack_intervalDC2STX10*EM
DC1gtalk_secure_portDC2EOT5228*DC1
    gtalk_sslDC2EOTtrue*0
%gtalk_sync_heartbeat_ping_interval_msDC2BEL1680000*<
SUBgtalk_terms_of_service_urlDC2RSwww.google.com/talk/terms.html*"
SUBgtalk_url_scraping_for_jpgDC2EOTtrue*G
ETBgtalk_youtube_video_urlDC2,http://gdata.youtube.com/feeds/api/videos/%s*êETX
'latin_ime_voice_input_supported_localesDC2¾ETXen en_US en_GB en_AU en_CA en_IE en_IN en_NZ en_SG en_ZA ar_AE ar_EG ar_JO ar_IL ar_LB ar_QA ar_KW ar_SA zh zh_CN zh_TW zh_HK zh_SG ja ja_JP de de_CH de_DE de_AT de_LI iw_IL es es_ES es_US fr fr_FR fr_BE fr_CH fr_CA it it_IT it_CH ko ko_KR pl pl_PL cs cs_CZ ru ru_RU tr tr_TR pt pt_BR nl nl_NL af af_ZA zu zu_ZA id id_ID in in_ID es_BO es_CL es_CR es_CO es_DO es_EC es_GT es_HN es_NI es_PA es_PE es_PR es_PY es_SV es_VE es_AR es_UY es_MX ms ms_MY*RS
EMmaps_enable_friend_finderDC2SOH1*ESC
SYNmaps_enable_navigationDC2SOH1*EM
DC4market_force_checkinDC2SOH1*"
CANmms_maximum_message_sizeDC2ACK307200*J
NAKmms_x_wap_profile_urlDC21http://www.google.com/oha/rdf/ua-profile-kila.xml*9
NAKmobile_transcoder_urlDC2 http://www.google.com/gwt/n?u=%s*'
USmusic_enable_track_stats_upsyncDC2EOTtrue*"
SUBmusic_enable_tracks_upsyncDC2EOTtrue*+
RSparental_control_timeout_in_msDC2 604800000*RS
SYNperform_market_checkinDC2EOTtrue*!
EMqsb:online_help_availableDC2EOTtrue*(
USsearch_allow_voice_search_hintsDC2ENOfalse*V
FSsearch_voice_app_install_uriDC26market://search?q=pname:com.google.android.voicesearch*'
ESCsecure:dropbox:data_app_anrDC2BSdisabled*)
GSsecure:dropbox:data_app_crashDC2BSdisabled*'
ESCsecure:dropbox:data_app_wtfDC2BSdisabled*ñETX
.secure:latin_ime_voice_input_supported_localesDC2¾ETXen en_US en_GB en_AU en_CA en_IE en_IN en_NZ en_SG en_ZA ar_AE ar_EG ar_JO ar_IL ar_LB ar_QA ar_KW ar_SA zh zh_CN zh_TW zh_HK zh_SG ja ja_JP de de_CH de_DE de_AT de_LI iw_IL es es_ES es_US fr fr_FR fr_BE fr_CH fr_CA it it_IT it_CH ko ko_KR pl pl_PL cs cs_CZ ru ru_RU tr tr_TR pt pt_BR nl nl_NL af af_ZA zu zu_ZA id id_ID in in_ID es_BO es_CL es_CR es_CO es_DO es_EC es_GT es_HN es_NI es_PA es_PE es_PR es_PY es_SV es_VE es_AR es_UY es_MX ms ms_MY*!
FSsecure:send_action_app_errorDC2SOH1*
CANsecure:ssl_session_cacheDC2EOTfile*SUB
NAKsend_action_app_errorDC2SOH1*g

 settings_contributors_pretty_urlDC2Chttps://www.google.com/mobile/android/basic/phone-contrib

utors.html*`
EMsettings_contributors_urlDC2Chttps://www.google.com/mobile/android/basic/phone-contributors.
html*_
ETBsettings_tos_pretty_urlDC2Dhttps://www.google.com/intl/%s/mobile/android/basic/phone-legal.
html*X
DLEsettings_tos_urlDC2Dhttps://www.google.com/intl/%s/mobile/android/basic/phone-legal.html*EM
DC1ssl_session_cacheDC2EOTfile*H
ETBurl:block_crash_reportsDC2-http://android.clients.google.com/crash block*„SOH
RSurl:bookmarks_sync_https_proxyDC2bhttps://clients4.google.com/chrome-sync rewrite
https://android.clients.google.com/proxy/bookmarks*x
FSurl:calendar_sync_http_proxyDC2Xhttp://www.google.com/calendar rewrite
https://android.clients.google.com/proxy/calendar*z
GSurl:calendar_sync_https_proxyDC2Yhttps://www.google.com/calendar rewrite
https://android.clients.google.com/proxy/calendar*x
FSurl:contacts_sync_http_proxyDC2Xhttp://www.google.com/m8/feeds rewrite
https://android.clients.google.com/proxy/contacts*z
GSurl:contacts_sync_https_proxyDC2Yhttps://www.google.com/m8/feeds rewrite
https://android.clients.google.com/proxy/contacts*‡SOH
DLEurl:feedback_urlDC2shttps://www.google.com/tools/feedback/android/submit rewrite
https://www.google.com/tools/feedback/android/__submit*i
EMurl:gmail_sync_http_proxyDC2Lhttp://mail.google.com rewrite
http://android.clients.google.com/proxy/gmail*l
SUBurl:gmail_sync_https_proxyDC2Nhttps://mail.google.com rewrite
https://android.clients.google.com/proxy/gmail*f
SUBurl:google_location_serverDC2Hhttp://www.google.com/loc/m/api rewrite
https://www.google.com/loc/m/api*x
SYNurl:music_sharepreviewDC2^http://music.google.com/music/sharepreview rewrite
https://music.google.com/music/sharepreview*SOH
&url:redirect_youtube_from_sandbox_ytbsDC2Whttp://jmt17.google.com/proxy/ytbs rewrite
http://android.clients.google.com/proxy/ytbs*SOH
&url:redirect_youtube_from_sandbox_ytbtDC2Whttp://jmt17.google.com/proxy/ytbt rewrite
http://android.clients.google.com/proxy/ytbt*•SOH
#url:vending_machine_billing_ssl_urlDC2nhttps://android.clients.google.com/vending/billing/
rewrite https://android.clients.google.com/market/billing/*ŒSOH
ESCurl:vending_machine_efe_urlDC2mhttps://android_efe.clients.google.com/vending/api/efe
rewrite https://android.clients.google.com/market/efe/*}
ESCurl:vending_machine_ssl_urlDC2^https://android.clients.google.com/vending/ rewrite
https://android.clients.google.com/market/*w
ETBurl:vending_machine_urlDC2\http://android.clients.google.com/vending/ rewrite
http://android.clients.google.com/market/*EM
DLEuse_msisdn_tokenDC2ENQfalse*&
ESCvending_carrier_cred_buf_msDC2BEL1800000*)
ESCvending_carrier_ref_freq_msDC2
1209600000*$
ESCvending_dcb_poll_timeout_msDC2ENQ20000*RS
NAKvending_hide_ad_prefsDC2ENQfalse*$
ESCvending_hide_content_ratingDC2ENQfalse*$
ESvending_hide_warning_messageDC2EOTtrue*(
vending_require_sim_for_purchaseDC2EOTtrue*.
)vending_show_similar_tab_in_app_info_pageDC2SOH1*@
DC3vending_support_urlDC2)https://support.google.com/androidmarket/*&
EMvending_sync_frequency_msDC2 172800000*V
STvending_tos_urlDC2Chttps://www.google.com/intl/%locale%/mobile/android/market-tos.html*ES
DC3vending_tos_versionDC2ENQ1.0.0*(
USvending_use_checkout_qa_serviceDC2ENQfalse*US
SUBvideos:vss_sampling_weightDC2SOH1*3
+voice_search:Pig_Latin_action_call_businessDC2EOTCall*2
*voice_search:Pig_Latin_action_call_contactDC2EOTCall*1

)voice_search:Pig_Latin_action_call_numberDC2EOTCall*<
+voice_search:Pig_Latin_action_directions_toDC2
Directions to*,
#voice_search:Pig_Latin_action_go_toDC2ENOGo to*4
'voice_search:Pig_Latin_action_listen_toDC2 Listen to*.
$voice_search:Pig_Latin_action_map_ofDC2ACKMap of*8
)voice_search:Pig_Latin_action_navigate_toDC2VTNavigate to*:
*voice_search:Pig_Latin_action_note_to_selfDC2FFNote to self*6
(voice_search:Pig_Latin_action_send_emailDC2
Send email*3
&voice_search:Pig_Latin_action_send_smsDC2 Send text*4
'voice_search:Pig_Latin_action_set_alarmDC2 Set alarm*+
&voice_search:advanced_features_enabledDC2SOH1*™EOT
(voice_search:alternate_backoff_languagesDC2ìETXaf:af-ZA ar:ar-EG ar-LY:ar-EG ar-PS:ar-IL
ar-BH:ar-QA ar-OM:ar-AE zh-CN:cmn-Hans-CN zh-TW:cmn-Hant-TW zh-HK:yue-Hant-HK
zh-SG:cmn-Hans-CN zh:cmn-Hans-CN ja:ja-JP de-CH:de-DE de-AT:de-DE de-LI:de-DE de:de-DE
en:en-001 es:es-ES fr-BE:fr-FR fr-CH:fr-FR fr:fr-FR fr-CA:fr-FR he:iw-IL he-IL:iw-IL
it-CH:it-IT it:it-IT iw:iw-IL ko:ko-KR pl:pl-PL cs:cs-CZ ru:ru-RU tr:tr-TR pt-PT:pt-BR
pt:pt-BR pt-AO:pt-BR nl:nl-NL nl-BE:nl-NL id:id-ID in:id-ID in-ID:id-ID ms:ms-MY ms-SG:ms-MY
ms-BN:ms-MY zu:zu-ZA*+
USvoice_search:de_DE_action_go_toDC2BSAnzeigen*2
%voice_search:de_DE_action_navigate_toDC2 Ansteuern*:
GSvoice_search:de_DE_hint_go_toDC2EMwikipedia anzeigen*I
#voice_search:de_DE_hint_navigate_toDC2"brandenburger tor ansteuern*C
 voice_search:de_DE_hint_send_smsDC2USsms senden an john smith*3
*voice_search:extra_total_result_timeout_msDC2ENQ20000*I
ESCvoice_search:help_video_urlDC2*http://www.youtube.com/watch?v=tPPcTN5sdX4*A
GSvoice_search:it_IT_hint_go_toDC2 vai su wikipedia</string>*E
 voice_search:it_IT_hint_send_smsDC2!invia sms a giulia bianchi*5
(voice_search:it_IT_slot_send_sms_messageDC2   messaggio*F
USvoice_search:mobile_privacy_urlDC2#https://www.google.com/privacy.html**
&voice_search:personalization_countriesDC2NUL*H
)voice_search:personalization_v2_countriesDC2ESC310 311 312 313 314 315 316*éSTX
RSvoice_search:supported_actionsDC2ÆSTXen-US:0,1,2,3,4 en-CA:0,1,2,3,4 en-GB:0,1,2,3,4
en-AU:0,1,2,3,4 en-NZ:0,1,2,3,4 en-IN:0,1,2,3,4 en-001:0,1,2,3,4 cmn-Hans-CN: cmn-Hant-TW:
ja-JP: de-DE: es-ES: fr-FR: it-IT: ko-KR: pl-PL: cs-CZ: ru-RU: tr-TR: pt-BR: nl-NL: af-ZA:
en-ZA: zu-ZA: yue-Hant-HK: id-ID: ar-EG: ar-IL: ar-JO: ar-KW: ar-LB: ar-QA: ar-AE: ar-SA:
iw-IL:*ÚETX
3voice_search:supported_actions_new_numbering_schemeDC2¢ETXen-US:14,18,2,12,13,15,4,17,1,6,3
en-CA:2,3,4,1 en-GB:14,2,12,4,17,1,3 en-AU:2,3,4,1 en-NZ:2,3,4,1 en-IN:2,3,4,1
en-001:2,3,4,1 cmn-Hans-CN: cmn-Hant-TW: cmn-Hans-SG: ja-JP: de-DE:14,2,12,4,17,1,3
es-ES:14,2,12,4,17,1,3 fr-FR:14,2,12,4,17,1,3 it-IT:14,2,12,4,17,1,3 ko-KR: pl-PL: cs-CZ:
ru-RU: tr-TR: pt-BR: nl-NL: af-ZA: en-ZA: zu-ZA: yue-Hant-HK: id-ID: ar-EG: ar-IL: ar-JO:
ar-KW: ar-LB: ar-QA: ar-AE: ar-SA: iw-IL:*ETX
 voice_search:supported_languagesDC2øSTXaf-ZA ar-EG ar-IL ar-JO ar-KW ar-LB ar-QA ar-SA
 ar-AE cmn-Hans-CN cmn-Hans-HK cmn-Hant-TW yue-Hant-HK cs-CZ nl-NL en-AU en-CA en-IN en-NZ
 en-ZA en-GB en-US en-001 fr-FR de-DE iw-IL id-ID zu-ZA it-IT ja-JP ko-KR ms-MY pl-PL pt-BR
 ru-RU es-AR es-BO es-CL es-CO es-CR es-DO es-EC es-SV es-GT es-HN es-MX es-NI es-PA es-PY
 es-PE es-PR es-ES es-US es-UY es-VE tr-TR Latin Pig-Latin*x
4voice_search:unsupported_action_market_url_set_alarmDC2@http://www.google.com/support/mobile/
bin/answer.py?answer=187559*#
ESCyoutube:enable_awful_playerDC2EOTtrue*
ESCyoutube:vss_sampling_weightDC2SOH1*EM
DC1youtube_use_proxyDC2EOTtrue9êETX~Ã¯¹?A§6CANÄEMB^a

L Eksempel på fil med mesternøkler og sesjons-IDer

```
RSA Session-ID:0D5DD83F5FAAD1BCDA64B335F7DC0C5911B536DCC0C159B9A18941AF83919F16
Master-Key:939318F5D8229D3212753F45E1AC0D86EF0FEA1CD03D24142B68BF9A16F40EFC6D984D88749B403E508A68C33BDA73EA
RSA Session-ID:20C70F54C6666EAB262546C4E053F36E174104661BB92F612B65AEA9ED69509F
Master-Key:F818A497D3981BF7E5BD9613AFE8CC7D2092640A0AFFBE23BBA942ED28B5CB286EBDB327BC5BAF0FCE39351AC60FFE7D
RSA Session-ID:0D5DD83F5FAAD1BCDA64B335F7DC0C5911B536DCC0C159B9A18941AF83919F16
Master-Key:939318F5D8229D3212753F45E1AC0D86EF0FEA1CD03D24142B68BF9A16F40EFC6D984D88749B403E508A68C33BDA73EA
RSA Session-ID:2B500823ABA52919E903590AE5BC18E527884803A140E84C020A3C7369658C65
Master-Key:0B010D35CC089116A0F7F03082E80D25A8F28443C2EF800F761F64D10BC2E3DB1B4160692494C61C0059C7CC1B64F798
RSA Session-ID:0D5DD83F5FAAD1BCDA64B335F7DC0C5911B536DCC0C159B9A18941AF83919F16
Master-Key:939318F5D8229D3212753F45E1AC0D86EF0FEA1CD03D24142B68BF9A16F40EFC6D984D88749B403E508A68C33BDA73EA
RSA Session-ID:4F96941F50E9F73694496 4DD698F78141075CD9322D2C390D61F43EB72CE57EF
Master-Key:AFAB3301F15A6898AA071F86F27D10329BC58ABB029516F1A7C75FDCA4B61F22CEA8A0058D815DE4407151D00BFB2010
RSA Session-ID:03A6E31FDD0B31FD0BD77D577FBDACBAAF279D508D6EF3732C5B5CEB964631D3
Master-Key:1C389F682C5C98FA553DF2EA5D6EA712D57D0C291B20781ED3C2025FBB262020FE196BAB1F4CDAE0275D7634A9B802E8
RSA Session-ID:4F96941F50E9F73694496 4DD698F78141075CD9322D2C390D61F43EB72CE57EF
Master-Key:AFAB3301F15A6898AA071F86F27D10329BC58ABB029516F1A7C75FDCA4B61F22CEA8A0058D815DE4407151D00BFB2010
RSA Session-ID:03A6E31FDD0B31FD0BD77D577FBDACBAAF279D508D6EF3732C5B5CEB964631D3
Master-Key:1C389F682C5C98FA553DF2EA5D6EA712D57D0C291B20781ED3C2025FBB262020FE196BAB1F4CDAE0275D7634A9B802E8
RSA Session-ID:4F96941F50E9F73694496 4DD698F78141075CD9322D2C390D61F43EB72CE57EF
Master-Key:AFAB3301F15A6898AA071F86F27D10329BC58ABB029516F1A7C75FDCA4B61F22CEA8A0058D815DE4407151D00BFB2010
```

M Statusrapport og møteinnkallelse

Dette er et eksempel på en kombinert statusrapport og møteinnkallelse sendt fra prosjektgruppa til oppdragsgiver og veilder.

Statusrapport og møteinnkallelse 19/4

Hei!

Vi har skrevet litt mer om hvordan vi har tenkt å gjøre analysene i rapporten. Vi begynte å sjekke om det var applikasjoner som lot seg analysere med testmiljøet slik det fungerer nå, men vi fant ingen som brukte SSL og fungerte. Dermed fortsatte vi å feilsøke sslsniff. Vi tror vi har funnet ut hvor feilen som ødelegger dataene ligger. Det blir bare lest 2048 bytes fra Android-klienten med ssl_read-funksjonen fra OpenSSL, selv om det skulle ha kommet mer derfra. Derfor leser sslsniff de samme 2048 bytene om og om igjen til den har fylt opp bufferet for "Content-Length" i HTTPS, som f.eks. kan være på 4174. Da blir altså bufferet på 2048 bytes lest så mange ganger at man har fått 4174 bytes i et annet buffer. Vi vet ikke hvor bufferet som er på 2048 bytes er, og vi finner ingen buffere på den størrelsen i sslsniff.

Møteinkallelse:
Dato: 20/4-2012
Tid: 11:00
Sted: Lasse sitt kontor

Agenda:

- Tilbakemelding på rapportutkastet.
- Analyser og feilfiksing.

Mvh.
David, Eirik og Kjetil

N Møtereferat

Dette er et møtereferat fra ett av møtene med oppdragsgiver og veilder.

Møtereferat 22/3-2012

Dato: 22/3-2012
Tid: 09:00-09:30
Sted: Lasses kontor
Deltagere:

- Nils Kalstad Svendsen
- Lasse Øverlier
- David Ueland
- Kjetil Gardåsen
- Eirik Bae

Notater: Vi nevnte probleme med sslsniff og Google-applikasjonene. Pakkerekkefølgeproblemene i sslsniff kunne være grunn til problemene. Bufferet mellom telefonen og serveren var muligens for lite, og måtte endres. Til rapporten må vi finne ut av hvordan vi skal analysere datamengden vi samler inn. Hvis applikasjonene vi analyser er for enkle bør vi analysere flere applikasjoner, slik som Twitter, elle uoffisielle varianter av dem. Burde planlegge hva appene kunne ha gjort med tillatelsene de har, og hva de faktisk gjør. Skrive hva vi leter etter.

O Timelister

Figur 25 viser et utdrag fra de individuelle timelistene som har blitt ført i prosjektet. Timelistene inneholder arbeid gjort og antall timer brukt totalt den dagen. Fullstendige timelister for hvert gruppemedlem finnes her.

Uke	Mandag		Tirsdag		Onsdag		Torsdag		Fredag		Lørdag		Søndag		Totalt for uke
	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	
10	- Begynt analyser - Fikset testmiljø. - Møte med Nils og Lasse	6	- Dekryptert Facebook	7	- Møte med Frode	1	- Testmiljø	6	- Satt opp testmiljø. - Skrevet rapport.	5	- Skrevet rapport	1	- Skrevet rapport - Lest om TCP	4	30

(a) Kjetil

Uke	Mandag		Tirsdag		Onsdag		Torsdag		Fredag		Lørdag		Søndag		Totalt for uke
	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	
4	-Laget ferdig utkast til forprosjekt. -Laget gantt-skjema -Begynt å føre inn i LaTeX -Ført ferdig i LaTeX	7	-Satt opp referanser i LaTeX og feilsøkt LaTeX referanser. -Sett etter nettverkskort med RTL8188 chipset.	6	Sett gjennom LaTeX-rapport og lest administrative e-poster.	1	-Skrevet møtereferat. -Ordnet Gantt-diagram. -Korrekturlest forprosjektrapporten.	7	-Finpusset rapport. -Laget logo for websiden. -Levert forprosjekt	8	Prøvd ut litt flashing av Android.	1			30

(b) Eirik

Uke	Mandag		Tirsdag		Onsdag		Torsdag		Fredag		Lørdag		Søndag		Total for uke
	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	Hva	Timer	
3	Definere forprosjektsplanen nøyere. Sett på om det er en ide å overvåke systemkall fra applikasjoner og Google. Funnet relevant litteratur.	5	Setter opp testmiljø med Android SDK og LaTeX. Funnet tidligere arbeid.	5			Hatt et møte med Nils og Lasse. Gått igjennom rapporten og forbedret ihht. tilbakemelding fra møte. Skrevet om tidligere arbeid.	6	Lært meg CSS. Påbegynt arbeidet med å lage en webside.	6			Jobbet med websiden	2	24

(c) David

Figur 25: Timelister for gruppemedlemmene vises i figur (a), (b) og (c)

P Signert prosjektavtale

HØGSKOLEN I GJØVIK

PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

Lasse Øverlier (oppdragsgiver), og

Kjetil Gardaisen, Eirik Bae og David Veland

(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 10.01.12 til 23.05.12.

   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): Nils Kalstad Svendsen

Oppdragsgivers kontaktperson (navn): Lasse Øverlier

Student(er) (signatur): David Velaml dato 26.01.12

Kjetil T. Gundersen dato 26.01.2

Eirik Bue dato 26/1-12

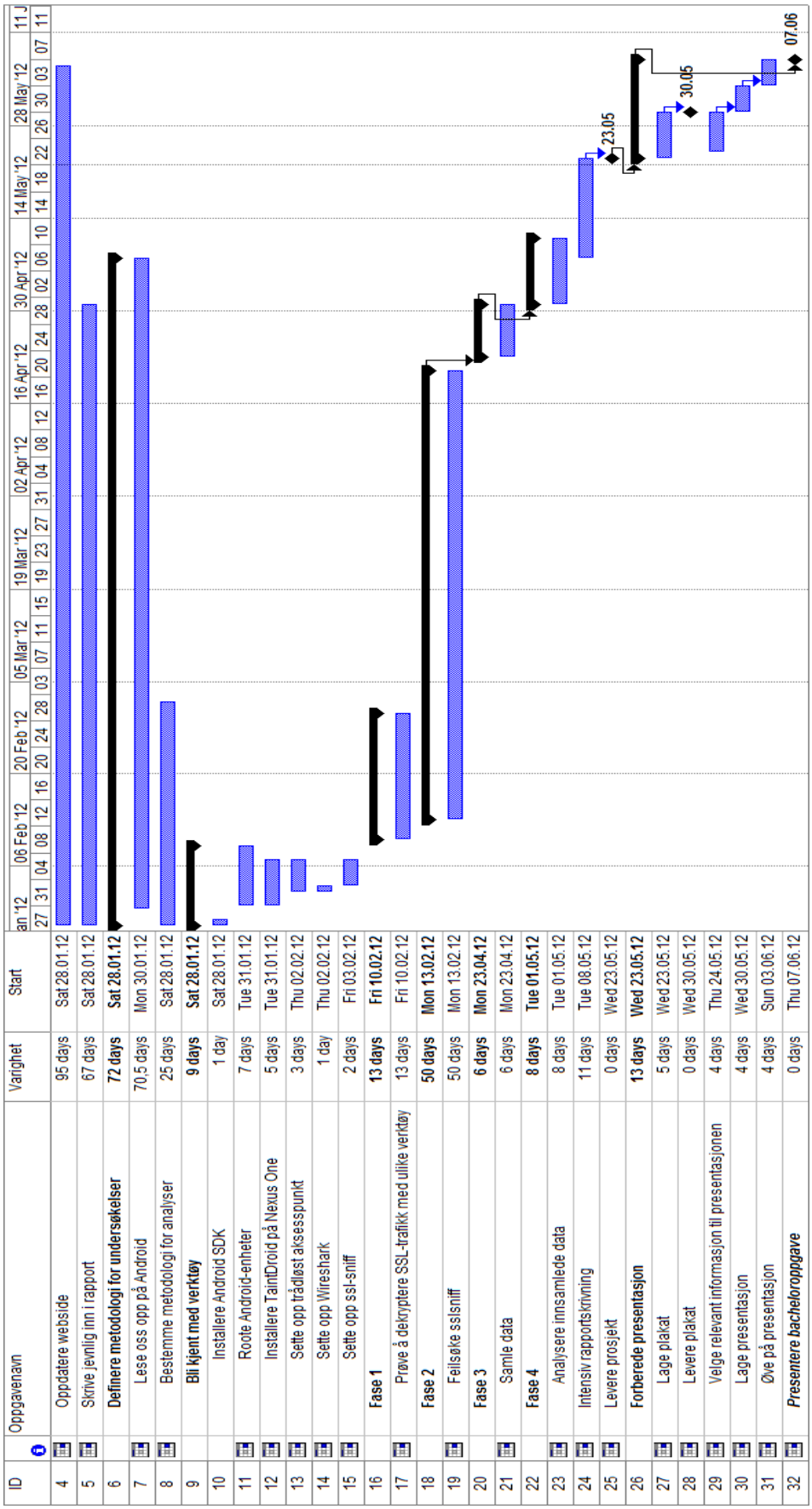
_____ dato _____

Oppdragsgiver (signatur): Lasse Øverlier dato 26/1-12

IMT Dekan/prodekan (signatur): _____ dato 15/2-2012

Versjon Januar 2011hb

Q Oversikt over faktisk tidsbruk



ID	Oppgavenavn	Varighet	Start
4	Oppdatere webside	95 days	Sat 28.01.12
5	Skrive jevnlig inn i rapport	67 days	Sat 28.01.12
6	**Definere metodologi for undersøkelser**	**72 days**	**Sat 28.01.12**
7	Lese oss opp på Android	70,5 days	Mon 30.01.12
8	Bestemme metodologi for analyser	25 days	Sat 28.01.12
9	**Bli kjent med verktøy**	**9 days**	**Sat 28.01.12**
10	Installere Android SDK	1 day	Sat 28.01.12
11	Roote Android-enheter	7 days	Tue 31.01.12
12	Installere TaintDroid på Nexus One	5 days	Tue 31.01.12
13	Sette opp trådløst aksesspunkt	3 days	Thu 02.02.12
14	Sette opp Wireshark	1 day	Thu 02.02.12
15	Sette opp ssl-sniff	2 days	Fri 03.02.12
16	**Fase 1**	**13 days**	**Fri 10.02.12**
17	Prøve å dekryptere SSL-trafikk med ulike verktøy	13 days	Fri 10.02.12
18	**Fase 2**	**50 days**	**Mon 13.02.12**
19	Feilsøke ssisniff	50 days	Mon 13.02.12
20	**Fase 3**	**6 days**	**Mon 23.04.12**
21	Samle data	6 days	Mon 23.04.12
22	**Fase 4**	**8 days**	**Tue 01.05.12**
23	Analysere innsamlede data	8 days	Tue 01.05.12
24	Intensiv rapportskrivning	11 days	Tue 08.05.12
25	Levere prosjekt	0 days	Wed 23.05.12
26	**Forberede presentasjon**	**13 days**	**Wed 23.05.12**
27	Lage plakat	5 days	Wed 23.05.12
28	Levere plakat	0 days	Wed 30.05.12
29	Velge relevant informasjon til presentasjonen	4 days	Thu 24.05.12
30	Lage presentasjon	4 days	Wed 30.05.12
31	Øve på presentasjon	4 days	Sun 03.06.12
32	***Presentere bachelooppgave***	0 days	Thu 07.06.12

R Fremgangslogg

10. januar

- Gruppen har vært på informasjonsforelesning med Tom Røise.
- Satt opp dokumenter i Google Docs for logg, fremdriftsplan og forprosjekt.
- Har begynt å fylle ut prosjektmalen.

12. Januar

- Har hatt BSc kick-off møte med Nils og Lasse.
    - Definert bacheloroppgavene litt nærmere, og sett på hvordan vi skal angripe forprosjektplanen.

13. januar

- Sett på ulike muligheter for å fange opp datatrafikk mellom telefon og Google.
    - Startet med å sette opp et testmiljø der vi sniffer trafikk sendt fra mobilen.
    - Testet oppfangning av datatrafikk med en laptop som "router".
        - To måter å gjøre det på:
            - kjøre tcpdump på mobiltelefonen
            - Rute trafikk gjennom et AP og sniffe det der
                - Tilkobling med ad-hoc (fungerte ikke)
                - Connectify-me(Brukte en demoversjon og fungerte ikke)
                - Virtual Router (fungerer + er gratis, men har ikke vært vellykket enda).
- Vi fikk til å kjøre tcpdump på mobilen
- Fant først nå relevant litteratur til tidligere forskning. (jon.oberheide.org).

16. januar

- Vi har undersøkt hva som finnes av lignende prosjekter og nyttige verktøy for bacheloroppgaven.
    - Vi oppdaget TaintDroid og lest om det.
- Vi ble usikre på om TaintDroid svarte på problemstillingene vi allerede hadde, og begynte å tenke på nye problemstillinger.
- Forprosjektet har blitt forbedret og skrevet på.
- Møteinnkallelse og første utkast til rapporten ble sendt.

17. januar

- Vi har diskutert problemstillinger for oppgaven for å unngå å gjøre det TaintDroid gjør.
- Vi har skrevet mer på forprosjektet.

19. januar

- Møte med Nils og Lasse.
    - Vi fortalte at vi syntes TaintDroid allerede hadde “løst” problemstillingen vår.
    - Vi ble fortalt at vi burde bruke det som et verktøy og bygge på det, og heller vinkle problemstillingen litt annerledes.
    - Vi burde ta et utvalg av applikasjoner som vi undersøker og evnt. bekrefte/ avkrefte tidligere arbeider gjort av andre.
- Forprosjektet ble endret på og vi prøvde å gjøre problemstillingene mer robuste.

20. januar

- Skrev mer på forprosjektet og forbedret planen generelt.
    - Forbedret problemstillinger
    - Diskutert oppsett for testmiljø

21. - 22. januar

- Alle gruppemedlemmene har gjort individuelt arbeid
    - Laget grafikk
    - Påbegynt arbeidet med webside
    - Lest om Android
    - Forbedret forprosjektrapporten

23. januar

- Forprosjektet har blitt forbedret med flere begrunnelser og Gantt-skjema.
    - Forprosjektet ble flyttet fra Google Docs til LaTeX
- Sendt 2. utkast til Nils og Lasse

24. januar

- Send møteinkallelse til Nils og Lasse
- Undersøkt innkjøp av nettverkskort
- Lagt til referanser i BiBTeX
- Ferdigstillit nettside.

26. januar

- Hatt møte med Lasse og fått tilbakemelding på forprosjektet.
- Levert prosjektavtale til Studenttorget.
- Bestillt nettverkskort med RTL8188CU chipset.
- Gjort finpussinger på forprosjektet.
- Satt opp testmiljøet. Kan koble til Internett.

27. januar

- Lagd logo til nettsiden
- Levert forprosjektet

29. januar

- Rooted Samsung Galaxy Tab med SuperOneClick

2. februar

- Sendt Nils og Lasse mail om torsdagsmøte.

3. februar

- Lastet ned og kompilert Android.

6. februar

- Bygd Android og TaintDroid for å undersøke at det fungerer

7. februar
- Oppdatert hBoot på Nexus One til 0.3.5. Hboot var ikke oppdatert, men trengtes å oppdateres for å installere stock 2.3.3
- TaintDroid ble installert og fungerte. Android-versjonen er 2.3.4

10. februar
- Satt opp sslsniff og installert egne CA-sertifikater på Nexus one.
- Sendt statusrapport og møteinnkallelse til Nils og Lasse

13. februar
- sslsniff fungerer. Vi kan fange opp kryptert trafikk.

14. februar
- Rootet Nexus One.
- Begynt å fange opp datatrafikk mellom Android og Google.

17. februar
- Sendt møteinkallelse og første rapportutkast til Nils og Lasse.

20. februar
- Møte med Nils om rapporten

23. februar
- Sendt epost til Jon Oberheide ang. protobuf og sslsniff
- Startet på arbeidet med å få sslsniff til å skrive til .pcap.

27. februar
- Sendt statusrapport og møteinkallelse.
- Undersøkt mer rundt sslsniff og .pcap.

28. februar
- Lagt ideen om å skrive om sslsniff død.
- Fått dekryptert nettverkstrafikken i sslsniff. Kjørte nedgraderingsangrep med dette valget i OpenSSL: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)

1. mars
- Skrev om sslsniff alikevel.
- Dekryptere data i Wireshark fungerer.
- Laget script for å parse nøkkelfiler fra sslsniff.

5. mars
- Møte med Nils og Lasse.

6. mars

- Fått dekryptert en ordentlig Facebook-sessjon.

15. mars
- Møte med Lasse og Nils.
- Endret HTTP 1.0 til 1.1 i sslsniff. Nå fungerer Facebook-applikasjonen hele tiden.

22. mars
- Møte med Nils og Lasse.

27. mars
- Forbedret sslsniff ved å legge til extensions for sertifikatene for å fjerne feilmeldinger.

12. april
- Møte med Nils og Lasse.

20. april
- Møte med Nils.

24. april
- Fikset sslsniff ved å sende all trafikk videre.

25. april
- Begynt med analyser av applikasjoner.

30. april
- Ferdig med datainnsamlingen.

1. mai
- Begynt analyse av innsamlede data.
- Begynt med mer rapportskriving.

7. mai
- Møte med Lasse.

10. mai
- Fedig med dataanalyse.
- Begynt intens rapportskriving.

11. mai
- Møte med Nils.

21. mai
- Møte med Nils.

23. mai

- Innlevering av rapporten.