LUNDS UNIVERSITET
Ekonomihögskolan

Institutionen för informatik

IT Risk Management vid användning av Open Source Software

Kandidatuppsats 15 hp, kurs SYSK16 i Informatik

Författare: Axel Hellström
Fredrik Moberg

Handledare: Björn Svensson

Examinatorer: Bo Andersson
Magnus Wärja

IT Risk Management vid användning av Open Source Software

FÖRFATTARE: Axel Hellström och Fredrik Moberg

UTGIVARE: Institutionen för informatik, Ekonomihögskolan, Lunds universitet

FRAMLAGD: maj, 2018

DOKUMENTTYP: Kandidatuppsats

ANTAL SIDOR: 80

NYCKELORD: IT Risk Management, Open Source Software, Riskhantering

SAMMANFATTNING (MAX. 200 ORD):
Fenomenet Open Source Software har tagit sig från små slutna kretsar inom hackerkulturen till att förändra hela mjukvaruindustrin. Idag är det ett naturligt och ofta förekommande inslag i organisationers utvecklingsprocess. Dock finns det problem och risker med Open Source Software som skiljer sig från proprietär mjukvara. Dessa risker innefattar bland annat säkerhet, licenser, och support. Samtidigt läggs stora resurser på att reducera risker i form av IT Risk Management men trots det ser vi många exempel på när Open Source Software-riskerna blivit till verklighet. Därför ämnar vi i denna uppsatsen att definiera olika litterära aspekter av IT Risk Management som är relevanta när organisationer använder Open Source Software. Därefter följer en empirisk datainsamling i form av fem intervjuer med organisationer som använder Open Source Software för att identifiera vilka aspekter av IT Risk Management som beaktas när organisationerna använder Open Source Software. Resultatet av undersökningen visar att en majoritet av aspekterna inom IT Risk Management beaktas av de undersökta organisationerna vid användningen av Open Source Software. Däremot konstateras också att det saknas kunskap om begreppet IT Risk Management men att organisationerna ändå har en hög riskmedvetenhet.

Innehåll

Figurer

Tabeller

1 Introduktion

1.1 Bakgrund

År 2016 omsatte mjukvaruindustrin 1038 miljarder euro (IDATE, 2016) och sysselsatte 6.7 miljoner människor i USA (CompTIA, 2016). Det är en global industri där mjukvara kan utvecklas tillsammans över hela världen och säljas över Internet, vilket skiljer den från många andra industrier (Buxmann, Diefenbach, Hess, 2013). Den ständiga transformeringen som sker i industrin möjliggör nya affärsmodeller och innovation som kan leda till signifikanta marknadsandelar och störningar på marknaden (Schief, 2014).

Ett resultat av den globala och föränderliga mjukvaruindustrin är fenomenet *Open Source Software* (OSS). Denna typ av mjukvara har tagit sig från små slutna kretsar inom hackerkulturen och universiteten till att förändra hela mjukvaruindustrin. Tidiga pionjärer inom området gav upphov till några av de mest använda programvarorna i dagens mjukvaruindustri. Idag är Linux det mest populära operativsystemet för webbservrar (W3Techs, 2018-a) och Wordpress det mest använda innehållshanteringssystemet (CMS), till exempel (W3Techs, 2018-b).

En generell definition av OSS är att det är mjukvara, som med hjälp av olika licenser, tillåter utvecklare att använda, modifiera, och distribuera källkod av mjukvara fritt, utan att behöva betala licensavgifter (Open Source Initiative, 2007). Tillgängligheten och öppenheten inom OSS gör det därför möjligt för utvecklare att omforma, förbättra och validera källkoden för att sedan offentliggöra förändringarna för andra användare. Ofta formas ett strukturerat samarbete och ett *community* på Internet runt en OSS-komponent (Linux Foundation, 2018).

Detta tankesätt inom mjukvaruutveckling kan härledas så långt tillbaka som till 1960-talet där forskare inom universitet delade datorprogram med varandra. Vid den tiden ansågs det som praxis att man modifierade varandras program och gav tillbaka de modifierade versionerna, men under 1970-talet gick åsikterna om mjukvarans tillgänglighet och öppenhet isär. Vissa, till exempel IBM och Bill Gates, menade att spridningen av gratis mjukvara skadade mjukvaruutvecklare och att royalties var nödvändigt för att utveckla högkvalitativ mjukvara (Gates, 1976). Detta gav upphov till proprietär mjukvara och sågs som OSS motsats (Androutsellis-Theotokis, Spinellis, Kechagia, Gousios, 2011). Proprietär mjukvara och OSS har sedan dess utmanat varandra där proprietära förespråkares försök att kontrollera mjukvaruindustrin genom royalty-baserade licenser, ofta svarats på av OSS-förespråkare (Androutsellis-Theotokis et al. 2011). Ett exempel på en sådan reaktion var GNU-projektet som startades av Richard Stallman i mitten av 1980-talet, vars mål var att skapa ett fritt och öppet operativsystem. Stallman startade även den ideella organisationen Free Software Foundation där utvecklare anställdes för att driva GNU-projektet framåt och för att sprida åsikten om att mjukvara ska vara fri. Han förklarade att definitionen av fri var densamma som frihet, inte gratis. “When I speak of free software, I’m referring to freedom, not price. So think of free speech, not free beer” (Stallman, 2002). Cirka 10 år senare färdigställdes GNU-

projektet när Linus Torvald skapade operativsystemkärnan Linux som i kombination med GNU-systemet utgjorde ett komplett operativsystem.

Med åren har OSS mognat och blivit ett naturligt och ofta förekommande inslag i organisationers utvecklingsprocess. Rapporter visar att en överväldigande majoritet av organisationer och utvecklare använder OSS (GitLab, 2016; Black Duck Software, 2017). Anledningarna till OSS framgång inom mjukvaruindustrin är många men de mest framstående ur ett verksamhetsperspektiv är de låga utvecklingskostnaderna, den minskade leveranstiden, flexibiliteten och främjandet av innovation (Feller, Fitzgerald, Scacchi, Sillitti, 2007). Det har dock varit en lång och strävsam väg att nå till dagens läge men nu ser vi även tidigare skeptiska giganter bidra till och anamma konceptet. Microsoft, till exempel, ändrade sig från att kalla Linux för "cancer" (Greene, 2001) till att "älska Open Source" (Brodkin, 2010) och sponsra Open Source Initiative (Open Source Initiative, 2017).

Samtidigt som OSS-användandet har ökat har också problemen och riskerna med OSS ökat. Rapporter visar att säkerhetshål är ofta förekommande och att det tar förhållandevis lång tid innan de upptäcks (Black Duck Software, 2017; National Vulnerability Database, 2018; Snyk, 2017). I september förra året råkade kreditupplysningsföretaget Equifax ut för ett gigantiskt dataintrång där hackare lyckades stjäla användardata, som till exempel kreditkortsnummer, tillhörande 145 miljoner kunder (Koskelainen, 2017). Hackarna utnyttjade ett välkänt säkerhetshål i Apache Struts som är ett gratis OSS-ramverk för att skapa webbapplikationer.

Det är inte heller enkelt att navigera sig i djungeln av OSS-licenser (Black Duck Software, 2017). Undersökningar av mjukvara som innehåller OSS-komponenter visar att användandet inte överensstämmer med kraven från en majoritet av OSS-licenserna (Black Duck Software, 2017; OpenLogic, 2011). Ett välkänt exempel är rättegången mellan Free Software Foundation och Cisco Systems år 2008, där Cisco Systems anklagades för att använda OSS vars licenskrav inte uppfylldes från Cisco:s sida (Free Software Foundation, 2008).

I dagsläget missar många organisationer att applicera *risk management* vid implementeringen av OSS (Franch et al. 2013; Franch & Susi, 2016). *Risk management*-processer underlättar identifieringen, bedömningen och hanteringen av både tekniska och verksamhetsstrategiska OSS-relaterade risker (Back & Silic, 2015; Franch & Susi, 2016). Enligt en rapport från Deloitte (2016) hade 73% av alla organisationer de undersökte en *risk management*-process implementerad. *Risk management* kan kort förklaras som alla de aktiviteter som är inblandade när man tar chanser inom en organisation (Hubbard, 2009).

IT-skandalen hos Transportstyrelsen (Svahn & Örstadius, 2017) är ett tydligt exempel på vad som kan hända om en organisation inte aktivt arbetar med att förebygga risker. Organisationer har börjat arbeta med hanteringen av dessa nya IT-relaterade risker genom en anpassning av det tidigare mer strategiska och affärsmässiga området *risk management* (ISACA, 2009). Enligt en rapport från Ernst & Young (2014) angav 92% av de förfrågade organisationerna att de använder sig av en IT-inriktad *risk management*-process utöver deras affärsinriktade *risk management*.

Denna IT-inriktade *risk management*-process går under samlingsbegreppet *IT Risk Management* (ITRM) (ISACA, 2009). Techopedia (2018) definierar ITRM som hanteringen av de risker som ägarskap, implementering och användning av IT medför. ITRM är en del av ett företags riskhanteringssystem men omfattar inte bara risker och negativa effekter utan tar också hänsyn till de potentiella fördelarna med riskabla IT-investeringar (ISACA, 2009).

1.2 Problemformulering

Användningen av OSS är, som sagt, inte problemfri. En rapport från Black Duck Software (2017) visar att 96% av över 1000 kommersiella applikationer som undersökts, innehåller OSS-komponenter och att 67% av dem har säkerhetshål som, i genomsnitt, varit kända i över fyra år. Vidare är över 85% av applikationerna inte i överensstämmelse med licenserna för den OSS som används (Black Duck Software, 2017). Liknande resultat hittar vi i en undersökning från Snyk (2017) som visar att 77% av de 430 000 webapplikationerna som undersökts innehåller minst ett OSS-bibliotek med en känd sårbarhet.

Enligt en undersökning av GitHub (2017) är även ofullständig och utdaterad dokumentation ett vanligt förekommande problem i OSS-projekt. Dokumentationen är enligt GitHub (2017) mycket viktig för att inkludera användare i hur OSS-projektet används, hur man bidrar, och vilka användarregler som finns.

Ett annat liknande problem är att OSS oftast inte erbjuder support på samma sätt som en mjukvaruleverantör gör. Enligt Tony Wasserman, professor vid Carnegie Mellon Silicon Valley och styrelseledamot i Open Source Initiative, finns bra support på olika OSS-forum med snabb responstid men går inte att jämföra med ett supportsamtal till en mjukvaruleverantör. "People often respond very quickly to queries posted on the forum pages of widely-used open source projects, but that's not the same thing as a guaranteed vendor response in response to a toll-free telephone call." (Tony Wasserman, 2014).

Eftersom en majoritet av organisationer arbetar med *risk management* (Deloitte, 2016) och mer specifikt ITRM (Ernst & Young, 2014) vet vi att stora resurser läggs på att reducera IT-risker. Trots detta så finns det problem som verkar förbises vid användandet av OSS. Om en organisation har en etablerad hantering av IT-relaterade risker borde dessa OSS-risker identifieras, bedömas, och hanteras. I denna uppsats ämnar vi därför att ställa teorin mot praktiken genom att definiera olika litterära aspekter av ITRM som är relevanta när organisationer använder OSS. Därefter följer en datainsamling för att identifiera vilka aspekter av IT Risk Management som beaktas när organisationerna använder Open Source Software.

1.3 Forskningsfråga

Vilka aspekter av IT Risk Management beaktas när organisationer använder sig av Open Source Software?

1.4 Syfte

OSS är bevisligen en stor del av mjukvaruindustrin men innefattar också problem och risker. Då vi vet att ITRM enligt teorin ska påverka organisationers hantering av IT-relaterade risker, vill vi undersöka om detta korrelerar med deras processer vid användningen av OSS.

1.5 Avgränsningar

Inom ITRM finns många olika aspekter att ta hänsyn till för att hantera IT-risker. Uppsatsens omfattning har därför avgränsats till de aspekter som är mest relevanta i förhållande till OSS. På samma sätt har inte alla risker och problem med OSS beaktats utan ett urval har gjorts utifrån vad som varit återkommande i litteraturen.

2 Litteraturgenomgång

Kapitel två ämnar att presentera uppsatsens undersökningsmodell samt den litteratur och forskning som ligger till grund för modellen. I kapitel 2.1 redogörs begreppet Open Source Software samt risker med OSS som varit återkommande i litteraturen. I kapitel 2.2 förklaras IT Risk Management och de aspekter som varit mest relevanta för vår forskningsfråga. I kapitel 2.3 presenteras litteratur som berör båda begreppen tillsammans.

2.1 Open Source Software

OSS är mjukvara som distribueras med en licens som tillåter fri spridning av källkoden, fri härledning av källkoden i egen distribuerad mjukvara samt helt öppen användning av mjukvaran (Androutsellis-Theotokis, Spinellis, Kechagia, Gousios, 2011; Buxmann, P., Diefenbach, H., Hess, T., 2013; Open Source Initiative, 2007). I dagsläget täcks de flesta områden av proprietär mjukvaruförsäljning även av OSS-utvecklade alternativ, till exempel operativsystem, databashanterare och verktyg för mjukvaruutveckling (Androutsellis-Theotokis et al. 2011; Back & Silic, 2017). OSS skiljer sig från proprietär mjukvara främst genom *community*, *teknisk utveckling* (säkerhetshål och buggar), *licensiering* samt *support* och *dokumentation* (Androutsellis-Theotokis et al. 2011; Back & Silic, 2017). Det finns både fördelar och nackdelar med alla aspekter som skiljer OSS från proprietär mjukvara, beroende på vem som blir tillfrågad (Brown & Giera, 2004). Det finns även en klassisk diskussion mellan de som förespråkar proprietär mjukvara och de som förespråkar OSS gällande vilken typ av mjukvaruutveckling som bär med sig flest tekniska säkerhetsrisker (Booch & Brown, 2002; Brown & Giera, 2004; Payne, 2002).

2.1.1 Support

Eftersom OSS utvecklas utan monetära incitament blir det ofta svårt att kunna erbjuda konkret support inom den tidsram användarna efterfrågar (Buxmann et al. 2013; Conradi, Cruzes, Hauge, Skarpenes, Velle, 2010). Vissa organisationer behöver stöd från utvecklarna eller återförsäljaren dygnet runt, vilket gör att dessa organisationer väljer bort OSS-lösningar (Brown & Giera, 2004; Conradi et al. 2010). Enligt Brown & Giera (2004) är detta den största anledningen till varför företag väljer att inte investera i OSS-lösningar. Utöver detta är många organisationer rädda för risken att det kanske inte längre går att få kontakt med de som är experter inom mjukvaran eller de som har utvecklat den då de har lämnat projektet, vilket innebär att man kan behöva utöka de interna resurserna för att kunna utnyttja dessa lösningar på bästa möjliga vis (Brown & Giera, 2004; Conradi et al. 2010).

Detta har lett till att en ny typ av företag har uppkommit där affärsidén går ut på att distribuera paket med OSS och ta betalt för specialiserad support (Androutsellis-Theotokis et al. 2011; Buxmann et al. 2013). En affärsmodell som enligt Buxmann et al (2013) har varit vanligt förekommande inom kapitaliseringen av OSS är *follow-the-free*, en affärsmodell som i dagsläget oftast går under benämningen *freemium* (Au, Choi & Liu, 2014). Detta är en affärsmodell som går ut på att erbjuda en grundläggande produkt utan kostnad i förhoppningen att användaren blir låst till produkten, för att sedan sälja utökade tjänster eller funktioner tillhörande grundutbudet (Au et al. 2014; Buxmann et al. 2013). Under support

nämns även upplärning av hur mjukvaran fungerar, vidareutbildning i effektivitet och automatisk versionsuppdatering som fler exempel på tjänster som ofta säljs av de företag som använder sig av freemium-modellen (Androutsellis-Theotokis et al. 2011; Buxmann et al. 2013).

Inom support talar man även om stöd från chefer och ledningsgrupper vid införande och användande av OSS (Back & Silic, 2017; Conradi et al. 2010). Om dessa nyckelgrupper inte förstår varför man använder OSS, vad det bär med sig för interna förändringar samt vilka möjligheter och risker som följer med OSS, kommer organisationen få problem (Back & Silic, 2017; Conradi et al. 2010). Det är också viktigt att vara uppmärksam på att OSS arbetar i enlighet med samt stödjer de strategier och mål som organisationen har (Conradi et al. 2010).

Under support kategoriseras även *community*, vilket handlar om hur uppdelningen av ansvar inom utvecklingen av OSS ter sig (Moroiu, Weiss & Zhao, 2006). De som utvecklar OSS gör det på sin fritid, ofta på grund av ett gemensamt intresse, en vision eller ett mål som delas mellan alla utvecklare inom projektet (Androutsellis-Theotokis et al. 2011; Moroiu et al. 2006).

När det kommer till OSS agerar projekten inte inom samma simplicitet som i proprietära projekt, då hierarki inom OS-projekt inte alltid följer samma struktur, vilket leder till att det kan uppstå skiljaktighet mellan till exempel projektledaren och projektets medlemmar (Androutsellis-Theotokis et al. 2011). Om skiljaktigheterna blir för stora skapas klyftor som i vissa fall kan leda till att ett projekt bryts upp i två olika projekt, vilket är ett fenomen som kallas *forking* (Androutsellis-Theotokis et al. 2011; Conradi et al. 2010). Det innebär att två olika grupper utgår från koden de hade vid tidpunkten de valde att dela på sig och sedan fortsätter utvecklingen i den riktning som de själva vill, vilket är farligt då det kan leda till att en version av mjukvaran helt och hållet överges och de som använder sig av mjukvaran kan vara ovetandes om situationen samt riskerna det innebär (Androutsellis-Theotokis et al. 2011; Back & Silic, 2017).

2.1.2 Licenser

OSS får fritt användas, modifieras och distribueras, förutsatt att vissa restriktioner följs gällande upphovsrätten som ett OSS innefattar (Androutsellis-Theotokis et al. 2011; Open Source Initiative, 2007). Dessa rättigheter och restriktioner uttrycks i form av en licens, vilket kan ses som ett kontrakt mellan licensgivaren (mjukvarans ägare) och licenstagaren (användare av mjukvaran) (Androutsellis-Theotokis et al. 2011; Buxmann et al. 2013). Licensgivaren kan vara en enskild person, en grupp på flera personer eller en organisation och är den som äger upphovsrätten till mjukvaran (Androutsellis-Theotokis et al. 2011; Buxmann et al. 2013). Vilken typ av licens som appliceras för varje OSS är beroende på vad syftet med mjukvaran är och bestäms av den som äger upphovsrätten (Androutsellis-Theotokis et al. 2011).

Upphovsrättsskyddad mjukvara gör det olagligt att få tillgång till källkoden, så länge ägaren inte väljer att distribuera källkoden (Androutsellis-Theotokis et al. 2011). Eftersom programkod enkelt kan kopieras och återskapas finns det många organisationer som är måna om att ha strikt upphovsrätt gällande deras mjukvara (Androutsellis-Theotokis et al. 2011). På andra sidan spektrumet finns det enligt Androutsellis-Theotokis et al. (2011) de som förespråkar att källkod ska finnas tillgängligt för att bidra till en ökad kunskap för alla

utvecklare samt samhället i stort. Utöver tvisten gällande kunskapsbidrag finns det även problem med global lagstiftning kring upphovsrätt på grund av hur lagarna skiljer sig mellan länder, vilket också kan påverka användningsmöjligheterna av både OSS och proprietär mjukvara (Androutsellis-Theotokis et al. 2011). OSS-licenser misstas ofta för allmängods då de besitter många liknande egenskaper, men den stora skillnaden är att allmängods inte har någon licens medan OSS har en licens som ger användare mer frihet än upphovsrättsligt material vanligtvis gör (Androutsellis-Theotokis et al. 2011).

Det finns ett litet antal licenser som är vanligt förekommande inom OSS. De är alla väldigt lika och skiljer sig bara på någon punkt, se figur 2.1 (Androutsellis-Theotokis et al. 2011; Buxmann et al. 2013).

Type of license	GPL	LGPL	MPL	BSD license
Can be integrated into proprietary software and redistributed without an OS software license	No	Yes	Yes	Yes
Modifications to OS licensed source code can remain proprietary on distribution	No	No	No	Yes

Figur 2.1. Skillnader mellan några av de vanligaste licenserna (Buxmann et al., 2013, s. 193)

GNU Public License (GPL) skapades av Richard Stallmann på 80-talet och har sedan dess varit ett vanligt alternativ för OSS. Licensen innebär att om den licensierade mjukvaran eller koden används som en del i en mjukvara, måste även den mjukvaran anamma GPL (Androutsellis-Theotokis et al. 2011; Buxmann et al. 2013). Detta förhindrar att proprietära mjukvaror tar koden och lägger in den i sin egna mjukvara för att sedan sälja detta till konsumenter.

Ur GPL härstammar Lesser GPL (LGPL), eller Library GPL som den ofta kallas (Androutsellis-Theotokis et al. 2011; Buxmann et al. 2013). LGPL är mildare i restriktioner jämfört med GPL, då licensen främst togs fram för de bibliotek som implementeras i större projekt. Skillnaden är att LGPL kan användas i proprietära system utan att hela källkoden måste gå under GPL (Androutsellis-Theotokis et al. 2011; Buxmann et al. 2013).

Mozilla Public License (MPL) är lik LGPL på det sättet att koden får användas och sedan säljas i proprietär mjukvara (Buxmann et al. 2013). Skillnaden ligger enligt Buxmann et al. (2013) i att MPL måste användas som den är, det vill säga inga modifieringar får göras på källkoden.

Det finns vissa bekymmer och möjliga risker när det kommer till de olika licenserna som används inom olika OSS-projekt (Androutsellis-Theotokis et al. 2011; Booch & Brown, 2002; Brown & Giera, 2004; Conradi et al. 2010). Enligt Androutsellis-Theotokis et al (2011) finns det en generell regel som säger att ett OSS som agerar under flera olika licenser blir minst lika restriktivt som upphovsrättsskyddad mjukvara. Osäkerheterna som uppstår när licenser börjar blandas och tiden som krävs för att förstå vilken licens som ett OSS faktiskt innehar gör att en del organisationer inte vågar anamma OSS, då riskerna anses för stora (Androutsellis-Theotokis et al. 2011; Booch & Brown, 2002).

Om utvecklare inte är tillräckligt noggranna eller har några riktlinjer att följa då de väljer eller använder sig av OSS finns det stor risk att bryta många olika lagar gällande licenser (Back & Silic, 2017; Brown & Giera, 2004; Conradi et al. 2010). Ett problem som kan uppstå då man använder sig av OSS är risken att det i sig innehåller bitar eller hela segment av deriverad OSS, vilket i sig kan innehålla spår från ett annat OSS och så vidare (Back & Silic, 2017; Conradi et al. 2010).

2.1.3 Säkerhetsrisker

Det råder konstant diskussion mellan de som förespråkar proprietär mjukvara och de som förespråkar OSS och det blir aldrig någon klargörelse kring vilken typ av mjukvara som är säkrast (Back & Silic, 2017; Booch & Brown, 2002; Brown & Giera, 2004; Schryen, 2011).

De mest förekommande ämnena som brukar diskuteras är kontroll av koden samt kontroll över utvecklare (Booch & Brown, 2002; Payne, 2002; Schryen, 2011). De som säger att OSS är mindre säker än proprietär mjukvara argumenterar att när flera olika utvecklare med dold agenda sitter och skriver koden till ett program finns det risk att någon lägger in en bakdörr eller implementerar övervakning i programmet som sedan kan utnyttjas (Back & Silic, 2017; Booch & Brown, 2002; Payne, 2002; Schryen, 2011). Motargument mot detta är att det är mycket svårt att implementera farlig kod då all kod granskas av andra utvecklare innan den implementeras i en version som skickas ut till användare (Booch & Brown, 2002; Payne, 2002). Vidare argumenterar de som är för OSS att proprietär mjukvara kan vara farligare då man som användare inte vet vad som händer bakom kulisserna när det gäller mjukvaran man arbetar med samt ger information till, eller hur många som har sett över säkerheten (Booch & Brown, 2002; Payne, 2002).

Länge har det inte funnits någon konkret data över vilken typ som är säkrast, då termen säkerhet i sig är svår att definiera beroende på vad för mjukvara som jämförs. Schryen (2011) gjorde en empirisk jämförelse baserat på ett antal olika kriterier för säkerhet, främst gällande intrång, antal buggar hittade jämfört med antal buggar åtgärdade samt hur allvarliga dessa buggar var. Resultatet artikeln presenterar är att antalet buggar och säkerhetsbrister inom OSS är jämlikt med proprietär mjukvara och att det inte går att dra all mjukvara över en kam (Schryen, 2011). Istället är det många kringliggande aspekter och situationer inom både utveckling och användning av mjukvara som gör att varje fall agerar och hanteras annorlunda (Schryen, 2011).

2.2 IT Risk Management

Risk är ett övergripande och vardagligt återkommande ord vilket ofta kan förknippas med osäkerhet. Enligt Kenett & Raanan (2010) kan skillnaden mellan risk och osäkerhet definieras som att osäkerhet är att inte besitta kunskapen eller tillräckligt med data för att kunna bedöma huruvida ett fenomen kommer inträffa eller ej. Risk skiljer sig från osäkerhet genom att man både kan utföra bedömningar och mätningar över sannolikheten att ett fenomen kommer inträffa (Kenett & Raanan 2010).

Enligt ISO (2009, s. 2) definieras *risk management* som "[the] coordinated activities to direct and control an organization with regard to risk". *Risk management* utgörs av *risk management*-processen:

"[The risk management process is the] systematic application of management policies, procedures and practices to the activities of communicating, consulting, establishing the context, and identifying, analyzing, evaluating, treating, monitoring and reviewing risk." (ISO, 2009, s.2)

NIST (2012) understryker att *risk management*-processen är en löpande och iterativ process eftersom nya risker, hot och sårbarheter uppstår varje dag. Enligt ISO (2013) är *risk management*-processen en del av ett *information security management system*.

ITRM är en komponent av ett större *risk management*-system och en del av *risk management*-processen (ISACA, 2009). ISACA (2009, s. 7) definierar ITRM som: "the business risk associated with the use, ownership, operation, involvement, influence and adoption of IT within an enterprise or organization". ITRM omfattar inte bara IT-risker som kan ha en negativ inverkan på organisationen, utan också förbisedda IT-möjligheter som kan förbättra effektiviteten i verksamhetsprocesser, eller möjliggöra nya affärsinitiativ (ISACA, 2009). Enligt NIST (2012) berör ITRM många olika roller i en organisation, som till exempel företagsledningen, informationsägare, processägare, IT-inköpare, och informationssäkerhetsansvariga.

Det finns olika metoder och standarder som hanterar ITRM (ENISA, 2006). ENISA (2006) menar dock också att det saknas ett gemensamt språk och kunskap inom ITRM samt undersökningar om ITRM:s metoder och verktyg. En av de få undersökningar som finns är från Ernst & Young (2013) där de vanligast förekommande ITRM-standarderna listas:

1. ISO 27005 (2008)
2. ISACA Risk IT Framework (2009)
3. OCTAVE (Caralli, Stevens, Young, Wilson, 2007)
4. NIST (2012)

Det finns många olika varianter på ITRM-processen men innehållet är mycket snarlikt mellan de olika standarderna (Kouns & Minoli, 2010). I figur 2.2 visas de tre ITRM-huvudaktiviteterna riskstyrning (*risk governance*), riskbedömning (*risk evaluation*), och riskhantering (*risk response*) (ISACA, 2008; Katsikas; 2013). Nedan följer en genomgång av dessa huvudaktiviteter.

Figur 2.2: ISACA IT Risk Framework (ISACA, 2009, s. 15)

2.2.1 Riskstyrning (Risk Governance)

Målet med riskstyrning är att kommunicera och förankra ITRM inom organisationen (ISACA, 2009). Enligt ISO (2008) måste syftet med ITRM klargöras vilket till exempel kan vara att se till att legala krav följs eller att få bättre underlag för strategiska beslut i organisationen. Roller och ansvar för ITRM-processen ska definieras och de ansvariga ska hjälpa ledningen förstå vad ITRM är och vad IT-risker kan ha för konsekvenser på verksamheten och dess mål (ISACA, 2009; ISO, 2008; NIST, 2012). På så sätt beaktas IT-risker av ledningen innan beslut tas (ISACA, 2009). ITRM-ansvariga ska också ta reda på vilken grad av IT-relaterad risk som är acceptabel (riskacceptans) i organisationen (Caralli et al. 2007; ISACA, 2009; ISO, 2008).

ITRM bör integreras med ett existerande *enterprise risk management*-system för att få en holistisk syn på *risk management*-processen (Caralli et al. 2007; ISACA, 2009).

2.2.2 Riskbedömning (Risk Evaluation)

För att kunna bedöma och analysera IT-risker måste relevant data samlas in. Denna data innehåller information om vilka tillgångar man vill skydda, vilka sårbarheter som finns i tillgångarna, vilka hot som finns mot de identifierade tillgångarna, och vilka konsekvenser en riskincident kan få (ISACA, 2009; ISO 2008; NIST 2012). ISACA (2009) påpekar även att förbisedda IT-möjligheter ska räknas in som risk eftersom effektiviteten i verksamhetsprocesser kan bli lidande. Innan datan samlas in ska en modell upprättas som beskriver hur insamlingen ska gå till och hur datan ska kategoriseras (ISACA, 2009). Organisationen bör också identifiera vilka existerande riskhanteringsåtgärder som finns implementerade för att undvika onödigt arbete och onödiga kostnader (ISACA, 2009; ISO 2008; NIST 2012).

Efter att risker identifierats ska de bedömas och analyseras (ISACA, 2009; ISO 2008; NIST 2012). Detta ska inte bara göras ur ett tekniskt perspektiv, utan också ur ett verksamhetsperspektiv så att anställda med stort fokus på IT förstår hur IT-relaterade misslyckanden kan påverka organisationens verksamhetsmål och ledningen förstår hur IT-relaterade misslyckanden kan påverka IT-tjänster och processer (ISACA, 2009).

För att bedöma och analysera risker görs antingen en kvalitativ eller en kvantitativ bedömning, eller båda (ISACA 2008; ISO, 2008). I en kvalitativ bedömning används en skala (låg, medium, hög) för att ge uttryck för konsekvenser och sannolikheten för att dessa konsekvenser kommer att uppstå (ISO, 2008). I en kvantitativ bedömning används en skala med numeriska värden (1, 2, 3 och så vidare) för både konsekvenser och sannolikhet, med användning av data från olika källor (ISO, 2008). Oftast görs en kvalitativ bedömning först för att få en generell indikation av risknivån och för att avslöja de största riskerna (ISO, 2008). Sedan kan det vara nödvändigt att göra mer specifik eller kvantitativ analys av de stora riskerna eftersom det vanligtvis är mindre komplext och billigare att utföra kvalitativ än kvantitativ analys (ISO, 2008).

Konsekvens- och sannolikhetsvärdena räknas ut med hjälp av en formel för att mäta IT-risker (Caballero, 2009). Det finns många olika formler för att mäta IT-risker men en vedertagen sådan är:

Risk = Asset * Threat * Vulnerability (Caballero, 2009; Carr, Rainer, Snyder, 1991)

För att mäta risker måste organisationen först identifiera sårbarheter (vulnerabilities) i tillgångar (assets) som måste skyddas mot hot (threats) (Caballero, 2009). Många har kritiserat formulan för att vara för generell och ej matematiskt gångbar vilket resulterat i många olika revideringar (ISACA, 2009; Katsikas, 2013; Kouns & Minoli, 2010).

2.2.3 Riskhantering (Risk Treatment)

Syftet med riskhantering är att utveckla riktlinjer och metoder som kan bemöta risker på ett kostnadseffektivt vis samtidigt som det faller i linje med operativa mål (ISACA, 2009). Det finns fyra alternativ för att hantera risker (ISACA, 2009; ISO, 2008). Antingen försöker man

undvika risken, reducera risken, överföra risken, eller acceptera risken (ISO, 2008). Om den identifierade risken bedöms ha för hög risknivå bör den undvikas, antingen genom att helt undvika aktiviteterna där risken kan uppstå, eller genom att ändra villkoren för aktiviteterna så att risken undviks (ISACA, 2009; ISO, 2008). I många fall kan risken reduceras antingen genom att implementera riskhanteringsåtgärder eller genom att överföra eller dela risken genom outsourcing, till exempel (ISACA, 2009; ISO, 2008). Ibland kan organisationen även välja att acceptera risken och då också acceptera förlusten som risken kan medföra (ISACA, 2009; ISO 2008). Detta är inte samma sak som att vara ignorant, utan ett aktivt beslut, grundat i en rigorös analys (ISACA, 2009). För en organisation med högre riskacceptans är det mer naturligt att acceptera risker (NIST, 2012).

Direkt när det blir klart att en risk kommer inträffa, eller redan har inträffat, måste de personer och avdelningar som påverkas kontaktas och arbetet för att minimera konsekvenserna påbörjas så snart som möjligt (ISACA, 2009; ISO, 2008). Då risker kontinuerligt inträffar måste lärdomarna som dras från dessa kommuniceras mellan alla parter inblandade för att reducera chansen att det inträffar igen (ISACA, 2009).

2.3 Open Source & IT Risk Management

Det finns mycket litteratur och forskning kring möjligheterna och de positiva sidorna av OSS, medan riskerna och de negativa sidorna inte är lika väl undersökta och förstådda (Back & Silic, 2015; Conradi, Cruzes, Hauge, Skarpenes & Velle, 2010)

Ett av de vanligaste misstagen när organisationer implementerar OSS är bristfällig applicering av *risk management* (Franch et al., 2013; Franch & Susi, 2016). En holistisk *risk management*-process bidrar till att både tekniska och verksamhetsstrategiska OSS-relaterade risker kan identifieras, bedömas och hanteras (Back & Silic, 2015; Franch & Susi, 2016).

2.3.1 Riskstyrning

Enligt Belle, Brink, Roos & Weller (2006) är en nyckelfaktor vid användningen av OSS att ha stöd från ledningen. Ledningen ska tillhandahålla resurser, policies och en tydlig plan för analys (Belle et al. 2006; Fitzgerald, 2009). Enligt Fitzgerald (2009) är det också vanligt att det inte finns några tydliga roller som ansvarar för hur OSS-risker ska hanteras och att organisationer inte har några tydliga riktlinjer för hur OSS ska anskaffas.

Woods & Guliani (2005) menar att organisationer bör ha en specifik riskstyrning (risk governance) vid användning av OSS eftersom OSS har specifika risker som proprietär mjukvara inte har. Mycket handlar om att ta reda på ifall organisationen har den tekniska och operativa kompetens som behövs (Woods & Guliani, 2005).

2.3.2 Riskbedömning

Det finns stora mängder öppen data (källkod och *community*-relaterade artefakter) om olika OSS-projekt som kan användas för att stödja, förenkla och förbättra risk management-processen (Bai, Kenett, Yahav, 2014; Franch, Kenett, Susi, Galanis, Glott, Mancinelli, 2015). Datan bör användas till att identifiera både tekniska och strategiska risker (Bai et al. 2014;

Franch & Susi, 2016). Det finns olika metoder och verktyg (Franch & Susi, 2016) för att samla och analysera datan som till exempel RISCOSS. Dessa verktyg kan identifiera licensbrott och sårbarheter i koden (Mansfield-Devine, 2016).

Enligt Woods & Guliani (2005) är det viktigt att titta på mognadsgraden (*open source maturity*) av ett OSS-projekt för att kunna bedöma risker, innan mjukvaran implementeras. *The Open Source Maturity Model* behandlar aspekter av OSS-projekt som ger en holistisk och graderad vy av OSS-risker (Woods & Guliani, 2005; Petrinja, Nambakam, Sillitti, 2009). Dessa aspekter är bland andra:

- Projektets livslängd (age)
- Senaste utgåvan (release)
- Popularitet
- Dokumentation & Support
- Kod- och arkitekturkvalitet
- Licenser

(Woods & Guliani, 2005)

Aspekterna graderas sedan för att kvantifiera mognadsgraden av en OSS-komponent (Woods & Guliani, 2005). Aner & Cid (2010) påpekar att kod- och arkitekturkvalitet är viktigt att granska, men också att det är tids- och kostnadskrävande, samt att det kräver expertis.

2.3.3 Riskhantering

Riskhantering inom OSS handlar om att reducera och hantera riskerna som identifierats och analyserats/bedömts (Mestre, Rodrigues & Soares, 2011). Det är inte alltid det går att eliminera eller ens reducera en risk eftersom det ligger utanför organisationens domän (Conradi et al., 2010). Dock är de två tidigare aktiviteterna (riskidentifiering och riskbedömning) en sorts riskhantering i sig eftersom det bidrar till medvetenhet och utvärdering av risker (ISACA, 2009).

Några generella framgångsfaktorer som kan reducera OSS-relaterade risker är följande:

- Anställdas attityd, medvetenhet, och kompetens (Bonaccorsi, Giannangeli & Rossi, 2006; Exton, Fitzgerald & Glynn, 2005; Gomez & Moreno, 2012)
- Tillgång till support (Fitzgerald, 2009)
- Andra framgångsexempel (Ågerfalk, Deverell, Fitzgerald & Morgan, 2005)
- Ingen inlåsning (Goode, 2005)
- OSS-policy (Woods & Guliani, 2005)

En positiv attityd från de anställda gentemot OSS kan främja en organisations val att använda OSS (Bonaccorsi et al. 2006). Om de anställda också besitter kompetens och erfarenhet av OSS kommer övergången förenklas (Exton et al. 2005; Gomes & Moreno, 2012). Vidare är det viktigt att ha tillgång till support eftersom OSS ibland kan vara mindre användarvänligt än proprietära alternativ (Fitzgerald, 2009). Det är dessutom en fördel om ett OSS-projekt är populärt och har implementerats framgångsrikt av andra organisationer (Ågerfalk et al. 2005). Till sist är det vanligast att organisationer som inte är låsta till en mjukvaruleverantör har större framgång med användandet av OSS än de som är låsta (Goode, 2004).

Ytterligare en faktor som kan reducera OSS-relaterade risker är att ta fram en policy som beskriver hur OSS ska användas i organisationen (Woods & Guliani, 2005). Denna kan till exempel innehålla en lista över vilka licenser som får användas för att undvika att *intellectual property* (IP) läcker ut (Woods & Guliani, 2005).

I en undersökning gjord på Telenor av Conradi et al. (2010) identifierades följande steg som mest vitala av Telenors anställda vid användning av OSS:

- Öka de anställdas kunskap om OSS (antingen genom att anställa eller utbilda)
- Öka de anställdas medvetenhet och inställning om pågående OSS-relaterade implementeringar, initiativ och projekt
- Förankra ledningens engagemang och stöd för OSS
- Ha tillgång till support

2.4 Litteratursammanfattning

Kategori	Sammanfattning/Innehåll	Litteratur
Säkerhet	• Kontroll över kod • Kontroll över utvecklare	• Back & Silic (2017) • Booch & Brown (2002) • Brown & Giera (2004) • Payne (2002) • Schryen (2011)
Support	• Osäkerhet gällande tillgänglighet • Stöd från ledningen • Forking • Utbildning	• Androutsellis-Theotokis et al (2011) • Back & Silic (2017) • Brown & Giera (2004) • Buxmann et al (2013) • Conradi et al (2010)

Licenser	• Typer av licenser
	  ○ GPL
	  ○ LGPL
	  ○ MPL
	  ○ Apache
	• Risk för licensbrott
	• Deriverade licenser
		• Androutsellis-Theotokis et al. (2011)
		• Back & Silic (2017)
		• Brown & Giera (2004)
		• Buxmann et al. (2013)
		• Conradi et al. (2010)

Riskstyrning	• Stöd från ledningen • Definiera roller och ansvar	• Belle et al. (2006) • Caralli et al. (2007) • Fitzgerald (2009) • ISACA (2009) • ISO (2008) • NIST (2012) • Woods & Guliani (2005)

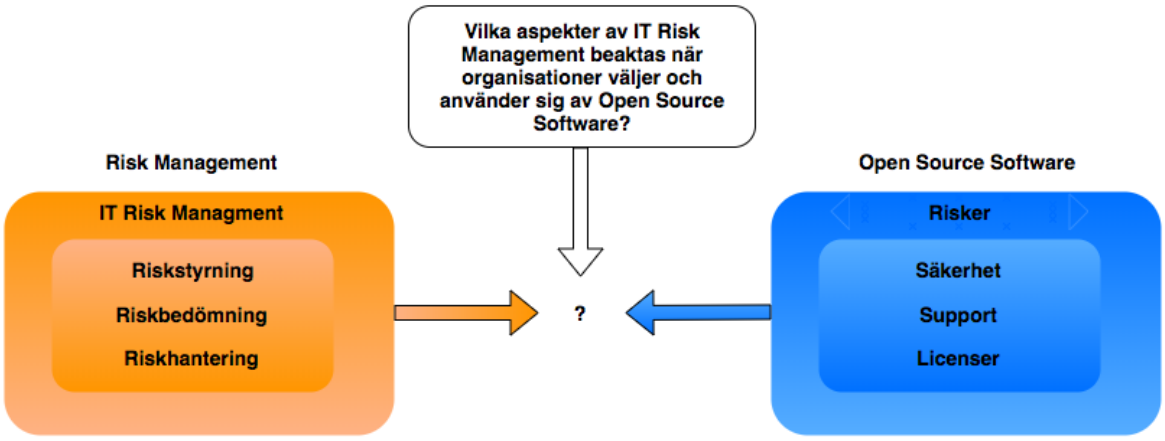
Riskbedömning	• Identifiera data om:
	  ○ Tillgångar
	  ○ Hot
	  ○ Sårbarheter
	  ○ Konsekvenser
	• Mät IT-risk (R = T * V * A)
	• Analysera datan
	  ○ Kvantitativt
	  ○ Kvalitativt
	• OSS-riskverktyg
	• Open Source Maturity Model
		• Bai et al. (2014)
		• Carr et al. (1991)
		• Franch et. al (2015)
		• Franch & Susi (2016)
		• ISACA (2009)
		• ISO (2008)
		• NIST (2012)
		• Petrinja et. al (2009)
		• Woods & Guliani (2005)

Riskhantering	• Hantera risk genom att:
	  ○ Undvika risk
	  ○ Reducera risk
	  ○ Överföra risk
	  ○ Acceptera risk
	• Faktorer som kan reducera OSS-risker:
	  ○ Anställdas attityd, medvetenhet, och kompetens
	  ○ Tillgång till support
	  ○ Andra framgångsexempel
	  ○ Ingen inlåsning
	  ○ OSS-policy
		• Bonaccorsi et. al (2006)
		• Exton et. al (2005)
		• Fitzgerald (2009)
		• Goode (2005)
		• ISACA (2009)
		• ISO (2008)
		• NIST (2012)
		• Woods & Guliani (2005)
		• Ågerfalk et. al (2005)

Tabell 2.1: Litteratursammanfattning

2.5 Undersökningsmodell

Undersökningsmodellen i figur 2.3 baseras på tidigare forskning och litteratur som sammanställts i detta kapitel (2). Vidare ligger undersökningsmodellen till grund för datainsamlingen och för att besvara forskningsfrågan. De huvudsakliga områdena vi identifierat inom ITRM är riskstyrning, riskbedömning, och riskhantering. Som tabell 2.1 visar finns det många olika aspekter inom huvudområdena och det är dessa aspekter vi syftar på i vår forskningsfråga. Vad gäller OSS har vi identifierat säkerhet, support och licenser som vanligt förekommande risker och problem enligt litteraturen. Dessa risker tillsammans med ITRM-aspekterna ligger till grund för våra intervjufrågor och därmed datainsamlingen.



Figur 2.3: Undersökningsmodell

3 Metod

För att kunna ge svar på frågeställningen krävs en god inblick i hur både val av mjukvara går till samt hur riskhantering ter sig inom en organisation. Då kriterierna inte uppfylls av många anställda inom en organisation, om ens någon, förstod vi i ett tidigt skede att vi var ute efter kvalitativ data och ej kvantitativ. Vi visste även att ämnets bredd skulle innebära att en intervjumall kunde bli svår att följa, trots detta valde vi att använda oss av en mall då detta enligt Backman (2016) är ett bra instrument för att kunna redovisa replikation och evaluering.

3.1 Datainsamling

Backman (2016) beskriver kvantitativ metodik som användningen av mätningar, matematik och kvantifiering som utmynnar i numeriska observationer och resultat. Kvalitativ metodik använder sig inte av siffror utan verbala formuleringar (Backman, 2016). Därav föll den kvalitativa metodiken mer i linje med det arbete vi valde att genomföra för att införskaffa data.

3.1.1 Intervjuobjekt

Vi fick kontakt med alla intervjuobjekt genom att maila direkt till de organisationer vi ville intervjua, oftast genom en generell info-mailadress som gick till deras kundtjänst eller service desk. I mailet uttryckte vi att vi ville bli vidarebefordrade till någon de tror skulle kunna hjälpa oss. De intervjuobjekt vi ämnade få kontakt med var de som innehar roller som vi förstod har hög förståelse för eller arbetar med både övergripande systemarkitektur och internt säkerhetsarbete (Appendix 8.2). Vi ville ha kontakt med dessa roller då vi antog att de kunde ha insyn i hur både ITRM och OSS fungerar och används inom deras organisationer. Utöver detta var kravet även att organisationen i fråga antingen använder sig av OSS eller aktivt har valt att inte använda det. Vi tänkte att en organisation som medvetet valt bort OSS-alternativ förmodligen har genomfört någon slags riskanalys och skulle således kunna ge svar på frågeställningen. Vi valde främst att fokusera på organisationer inom IT-branschen, helst de som producerar IT-tjänster eller mjukvara, då vi antog att dessa organisationer skulle ha högst chans att kunna erbjuda intervjuobjekt.

Vi erbjöd alla intervjuobjekt möjligheten att vara anonyma. Vi förstod att detta var viktigt då uppsatsens ämne till viss del hanterar väsentliga delar av en organisations säkerhet. För att garantera en ömsesidig bild av anonymitet skickade vi alltid vår pseudonyma beskrivning av organisationen samt transkriberingen av intervjun till intervjuobjektet för godkännande innan vi använde data från intervjun.

Totalt intervjuade vi fem företag med spridda arbetsområden och vitt differerande tjänster och produkter inom IT-branschen, vilka syns i tabell 3.1. En organisation ville vara anonym och där har vi ersatt organisationens namn med “Organisation X” och intervjupersonen med “Person X”.

Namn	Organisation	Position	Intervjutyp	Appendix
Magnus Persson	LDC	IT-arkitekt	Möte	8.3.1
Carl-Erik Mols	Sony Mobile	Head of Open Source	Möte	8.3.2
Philip Vendil	CGI	Certificate Service Development Team Lead	Videomöte	8.3.3
Håkan Erlandsson	Prevas	Business Unit Manager	Möte	8.3.4
Person X	Organisation X	Head of Vendor Management and Governance	Videomöte	8.3.5

Tabell 3.1: Intervjuobjekt

3.1.2 Procedur

Vi skapade frågorna (Appendix 8.1) i enlighet med undersökningsmodellens huvudområden och dess aspekter. Enligt Jacobsen (2002) medför öppna frågor att en intervju bibehåller hög intern giltighet. Detta medförde att vi försökte konstruera frågorna så öppna som möjligt. Utöver det var syftet inte att påvisa att teorin appliceras i praktiken, utan att undersöka om något av det appliceras över huvud taget. Eftersom vi valde att inte genomföra en kvantitativ undersökning var vi flexibla gällande huruvida vi ställde en fråga exakt som den stod skriven i intervjumallen eller ej. Till exempel, om vi ställt en fråga och i svaret även fått svar på en fråga som skulle ställas senare i intervjun, valde vi i vissa fall att inte ställa den senare frågan för att minimera onödig upprepning. Trots detta kunde vi ibland upprepa en fråga vi indirekt fått svar på i syftet att undvika misstolkning av svar. Vi förtydligade alltid för intervjuobjekten i början av intervjun att det ibland kunde kännas som att vi ställde frågor den redan svarat på och att det då var fritt fram att referera till ett tidigare svar.

Intervjuerna genomfördes fysiskt i intervjuobjektets lokaler om de befann sig i Skåneregionen, annars genomfördes de via Googles kommunikationsverktyg Hangouts. Intervjuerna började initialt med att vi frågade intervjuobjektet om godkännande att spela in intervjun. En av författarna agerade intervjuledare under alla intervjuer där den ställde alla frågor och byggde vidare på diskussioner medan den andre antecknade. Efter att varje intervju var färdig frågade alltid intervjuledaren personen som antecknade om någon fråga hade glömts bort. Anteckningarna behövdes egentligen inte eftersom vi spelade in intervjuerna, men då vi hade hört om tidigare studenter som har haft problem med ljudfiler valde vi att minimera riskerna.

Syftet med frågornas upplägg var att först ge intervjuobjektet en bild om vilka vi var, vår bakgrund samt varför vi skriver uppsatsen. Sedan ville vi få intervjuobjektet att börja prata lite och bli varm i kläderna genom att svara på enklare frågor om organisationen och vilket arbete intervjuobjektet genomför. Vi ville även att intervjuobjektet skulle presentera sin uppfattning av de två huvudsakliga begreppen som uppsatsen undersöker, ITRM och OSS. Om det skiljde

sig från hur vi har definierat det i det teoretiska kapitlet, förklarade vi vår definition för att försäkra oss om att vi pratade om dessa ämnen på samma nivå.

3.2 Transkribering & Analys

Det bestämdes vid ett tidigt skede att transkriberingen endast skulle innehålla det som var av relevans för intervjun och ämnet. Frågor eller utsvävningar som inte på något sätt berörde frågeställningen eller intervjufrågorna som är nedskrivna i intervjumallen (Appendix 8.2) togs därför inte med i transkriberingen. Tillfällig stamning, utfyllnadsord som "ehh" och lång betänketid är några fenomen som vi valde att inte ta med i transkriberingen då vi anser att det inte påverkar svaren samt underlättar läsbarheten. Om en ljudinspelning skulle gå förlorad var tanken att intervjun skulle återges så gott som möjligt, men inte ordagrant, baserat på de anteckningarna som fördes under intervjun.

Enligt Jacobsen (2002) kan man via telefonintervju inte alltid märka vissa fysiska signaler som intervjuobjektet visar, till exempel om den känner sig hotad, obekväm eller uttråkad. Detta skulle enligt Jacobsen (2002) kunna påverka resultatet i en kvalitativ ansats. Vi anser att det är skillnad på videointervju och telefonintervju då man faktiskt ser personen i fråga och dess fysiska signaler. Vi valde också att hålla intervjuerna korta samt med fokus på företaget som informanten representerade och inte särskilt mycket på intervjuobjektet i sig. Detta resulterade i att vi inte skiljde på den data vi samlade in via videomöte kontra fysisk intervju.

Målet med analysen är att ge den insamlade datan ändamålsenlig och tolkningsbar form för att kunna relateras till den ursprungliga frågeställningen (Backman, 2016). Detta kan enligt Backman (2016) underlättas genom en viss grov strukturering eller kategorisering av den insamlade datan, innan datainsamlingen påbörjas. Utmaningen ligger i att komma bort från enkla beskrivningar och fånga en helhetsbild med de essentiella orsakerna till dess uttryck (Backman, 2016). Detta ämnar vi att uppnå genom att kategorisera de resultat vi presenterar i enlighet med de kategorier vi har identifierat i vår undersökningsmodell.

3.3 Validitet & Reliabilitet

Enligt Jacobsen (2002) finns det två olika typer av validitet: intern och extern. Den interna validiteten handlar om resultatets giltighet (Jacobsen, 2002). Ett sätt att kontrollera den interna validiteten är att genomföra en uppgiftslämnarvalidering, vilket är då transkriberingen av en genomförd intervju skickas till intervjuobjektet i efterhand för att undersöka i vilken grad de känner igen sig i de svaren som de angivit vid intervjun (Jacobsen, 2002). Detta är en metod vi använde oss av för att försäkra oss om att vår data var korrekt. Framförallt tyckte vi detta var viktigt för de intervjuobjekten som valde att vara anonyma då vi inte ville riskera att publicera något som de ansåg skulle kunna vara identifierande uppgifter för deras organisation eller intervjuobjektet som person.

Den externa validiteten handlar om i vilken grad resultatet av en undersökning kan generaliseras (Jacobsen, 2002). Detta är svårt när det kommer till en kvalitativ ansats då målet inte är att generalisera i större sammanhang (Jacobsen, 2002). Enligt Jacobsen (2002) är målet att utveckla en mer generell teori utifrån den data som erhålls från ett mindre urval av

intervjuobjekt. Detta faller i linje med den väg vi valde att följa gällande den externa giltigheten då ämnet är av specifik natur och inte något som enkelt kan generaliseras till en större mängd.

Reliabiliteten kretsar kring ifall det är något i själva undersökningen som har påverkat resultatet av denna (Jacobsen, 2002). En av dessa effekter som Jacobsen (2002) skriver om är kontexteffekten, den effekt som påverkar hur intervjuobjektet kan bete sig beroende på hur bekväm den är i sin kontext, det vill säga hur naturlig omgivningen är. En naturlig omgivning kan enligt Jacobsen (2002) vara i intervjuobjektets hem eller på dess arbetsplats där den befinner sig dagligen och inte känner sig obekväm eller hotad. Vi valde att utföra intervjuerna på intervjuobjektets arbetsplats eller via videomöte när den befann sig på sin arbetsplats för att minimera externa påverkningar via omgivning. Utöver detta bad vi intervjuobjektet själv bestämma en tid då det passade, samt att intervjun skulle ta cirka 30 minuter (Appendix 8.2) för att ge en bild av hur stor del av arbetsdagen intervjun i sig skulle kräva. Detta faller i linje med det Jacobsen (2002) nämner gällande planerad undersökning. Om man vill ha planerade och genomtänkta synpunkter lämpar det sig bättre med en planerad undersökning, medan en oplanerad undersökning ger större chans till spontana åsikter och känslor (Jacobsen, 2002).

3.4 Etik

Enligt Jacobsen (2002) finns det tre huvudsakliga grundkrav som en undersökning måste uppfylla för att vara etisk försvarbar: informerat samtycke, rätt till privatliv och korrekt återgivning.

Informerat samtycke innefattas av flera mindre krav som bör försöka uppfyllas så gott som möjligt (Jacobsen, 2002). Dels måste de som undersöks fritt få välja om de ska delta i undersökningen eller ej, vilket kan tyckas verka löjligt, men det är viktigt för den insamlade datan att det inte finns någon bakomliggande orsak eller påtryck från en extern aktör att personen ska delta i undersökningen (Jacobsen, 2002). Dessutom måste de som undersöks enligt Jacobsen (2002) få full tillgång till information kring syftet med undersökningen samt information om eventuella fördelar och nackdelar med sitt deltagande.

Rätt till privatliv syftar till att beakta information och uppgifter rörande privatlivet hos den som blir undersökt (Jacobsen, 2002). Det är därför av stor vikt att eftersträva diskretion vilket uppnås genom att erbjuda deltagarna garanterad anonymitet och konfidentialitet, för att undvika att informationen kan bli kopplad till en privatperson (Jacobsen, 2002).

Det sista kravet är enligt Jacobsen (2002) krav på riktigt presentation av data. Detta innebär att den intervjuade personen ska bli korrekt återgiven och i rätt sammanhang (Jacobsen, 2002). För att se till att detta sköts på ett godtycklig vis kan uppgiftslämnaren alltid kräva fullständig återgivning (Jacobsen, 2002).

Vi har försökt uppnå alla dessa krav genom följande tillvägagångssätt: Vi lät organisationerna få full information gällande uppsatsens syfte och möjliga frågor vid första kontakt, vi lät även organisationerna själva hänvisa till de personer som de trodde skulle kunna hjälpa oss, det vill säga de som besatt den lämpliga kompetensen. Vi frågade alla intervjuobjekt huruvida de ville vara anonyma eller ej samt ifall de godkände en ljudinspelning av intervjun. Till sist skickade

vi fullständig transkribering för godkännande samt korrigering till varje intervjuobjekt för att försäkra oss om att båda parter var eniga i att datan som presenteras är korrekt.

4 Resultat

Nedan presenteras resultatet av den empiriska undersökningen. Intervjuerna är sammanställda och kategoriserade enligt huvudområdena i undersökningsmodellen.

4.1 Open Source Software

Vår definition av begreppet Open Source Software (OSS) är enligt intervjumallen (Appendix 8.2) källkod som är fullt tillgänglig för andra att modifiera och distribuera utan begränsningar. Under alla intervjuer definierade intervjuobjekten begreppet snarlikt. Philip Vendil beskrev det följande:

> "Det är väl att källkoden är tillgänglig, man kan ladda ned, man kan bidra, utbyta idéer, ja, som utvecklare är det ofta att man kan fixa problem själv som man upptäcker [...]" (Philip Vendil, 2018).

4.1.1 Säkerhet

De svaren som presenteras nedan dök upp i samband med att vi frågade om intervjuobjekten har stött på några säkerhetsproblem med OSS. Både CGI och Sony Mobile sade att de proaktivt arbetar för att undvika dessa risker. CGI förklarade att på grund av den utveckling som har skett inom säkerhet har det blivit lättare att undvika deriverade sårbarheter:

> "Där har det ju blivit mycket bättre på senare tid tycker jag, med att få säkerhetsvarningar i bibliotek. Tidigare visste man inte alls, då var det väldigt svårt att hålla reda på för att, det är ju liksom beroenden som av OS-bibliotek så det kan vara hundratals bibliotek så det är svårt att veta om den där längst ner på listan har fått en sårbarhet" (Philip Vendil, 2018).

CGI försvarar sig mot detta genom att genomföra kontroller av all kod de väljer att implementera.

Sony Mobile menade att OSS är säkrare än proprietär mjukvara och att den åsikten delas av många andra som använder sig av OSS:

> "Sen finns det ju säkerhetsrisker i Open Source i sig och de är ju då, enligt vad jag predikar, mindre än de som finns i proprietär mjukvara. Det är ingen som vågar säga emot det här längre, det är ingen tvekan, Open Source är trots allt bättre. Till och med vår egen security-avdelning säger det nu för tiden [...]" (Carl-Erik Mols, 2018).

Men trots det så innebär *säkrare* inte helt och hållet *säkert* enligt Sony Mobile och att man hela tiden måste vara uppmärksam på vad man tar in för komponenter. Även fast de har en god bild av säkerheten inom OSS så granskar de fortfarande all kod de tar in.

Organisation X har en intern process som utför kontroll av koden de använder. Organisation X belyste också att det är viktigt att kontinuerligt kontrollera dessa komponenter:

> "[...] för att dels kunna säga att jo men den här [komponenten] är i sitt nuvarande stadie okej men också för att få till en kontinuerlig bevakning över OS-komponenter. De kan ju i sin natur förändras utan att vi gör någonting egentligen, och där vet vi nog inte riktigt hur vi ska hantera det." (Person X, 2018).

Prevas nämnde inga konkreta säkerhetshål eller risker men uttryckte en stark oro för mer traditionella företag som ska förnya sina produkter eller maskiner med hjälp av Internet of Things. Inom Internet of Things används det enligt Prevas mycket OSS och när detta sätts i samband med verksamheter som tidigare inte jobbat med denna typ av teknologi finns det stora risker att OSS kan utnyttjas för att skada verksamheten.

LDC svarade att de enbart använder sig av större och mer beprövade typer av OSS och lägger därav ingen tid eller kraft på att undra om mjukvaran är säker, de litar dom på att den är.

4.1.2 Support

Angående support var majoriteten positiv till den support de anser att man får kring OSS. CGI sade att supporten hos proprietär mjukvara kan vara krångligt då man ofta måste skapa en support issue och sedan få vänta tills någon på företaget med rätt kompetens kan erbjuda hjälp. När det gällde supporten hos OSS tyckte CGI istället att det oftast fungerar bättre än proprietär mjukvara:

> " [...] aktiva projekt har ju nästan [alltid] något så här slack-rum eller IRC-chatt man kan koppla upp sig på så får man oftast väldigt mycket bättre support än en kommersiell supportkanal, faktiskt." (Philip Vendil, 2018).

Även Prevas höll med om god support baserat på tidigare erfarenheter med OSS där communityn är stor:

> " [...] väljer man en hyfsat vanlig [OSS] är det rätt stora communities där man kan gå in och ofta får man väldigt bra svar, rätt snabbt. Så supporten där är nog rätt så bra faktiskt." (Håkan Erlandsson, 2018).

Till skillnad från Prevas och CGI påpekade LDC att support inom OSS ibland kan vara bristfällig om de behöver en förändring då det inte finns några krav eller incitament för utvecklarna att lösa de problem LDC stöter på vid användningen av mjukvaran. Men LDC var inte bara negativ i sin syn på supporten kring OSS utan påpekade även att det finns bra communitys som har stort fokus på användarna och till och med erbjuder träning och utbildning inom mjukvaran:

> "Min kollega Peter som jobbar mest med implementationen tekniskt ska nu gå på kurs i Elastic Stack i Schweiz, vilket visar på att communityn är stor och det finns stora och bra möjligheter att utbilda sig." (Magnus Persson, 2018).

Sony Mobile nämnde att supporten kan skifta, men att de oftast får bra respons när de väl kontaktar utvecklarna. Eftersom Sony Mobile anser sig själva vara stora när det kommer till användning av OSS brukar utvecklare bli positivt överraskade när någon från Sony Mobile ber dom att ändra någonting vilket gör att de ofta får bra hjälp.

Organisation X sade att dom för närvarande fortfarande använder sig av många stora proprietära lösningar men att de sakta men säkert börjar använda mer och mer OSS. Därför visste Person X inte särskilt mycket gällande support när det kom till deras användning.

4.1.3 Licenser

Vid frågor kring licenser svarade en stark majoritet att de arbetar för att undvika licensbrott så gott som möjligt. CGI beskrev att de har en central policy som konstaterar exakt vilka licenser man får eller inte får använda och sedan är det upp till utvecklarna att se till att följa policyn då de väljer en OSS-komponent. Till exempel får de inte använda sig av GPL då det finns risk att CGIs kunder skulle behöva släppa sin mjukvara som OSS.

Prevas förklarade hur de hade gjort på tidigare projekt där kunder behövde hjälp med att genomsöka och kategorisera alla licenser de använde sig av:

> "[...] vad vi gjorde var egentligen en gap-analys på vilka OS-paket hade man och hur hade man behandlat deras licenser. [...] Det var nog så att det blev en liten sån aha-upplevelse för kunden att, okej var det så mycket att göra, det hade vi inte riktigt räknat med." (Håkan Erlandsson, 2018).

Organisation X sade att de inom koncernen har en avdelning som varje intern verksamhet kan använda som en tjänst för att kontrollera att inga licensbrott sker:

> "Ja, vi har inom koncernen en central hantering av Software Asset Management och licenser [...] då köper vi kontrollen av det som tjänst så att säga." (Person X, 2018).

Gällande licenser så svarade LDC till skillnad från de andra att de inte aktivt undersöker vilken licens det är på grund av hur dom som statlig myndighet använder sig av Open Source. Sedan förklarade LDC att de inte tror att någon som använder sig av OSS undersöker licenser över huvud taget, baserat på hur dom själva använder sig av det:

> "Jag betvivlar att någon som använder sig av OSS bryr sig, jag tror inte heller att vi gör våld på någon licens då de enbart används internt, vi vidareutvecklar det inte. Vi kan använda öppna bibliotek och kod men arbetar inte inom kommersiell natur." (Magnus Persson, 2018).

Sony Mobile sade att de använder sig av sin juridiska avdelning för att se till att alla komponenter de tar in och använder sig av har en licens som de kan och får använda sig av:

> "Men då får man ju be legal att kolla på kriterierna, håller de [licenserna] för de [kriterierna] så kan dom [legal] komma med en verdict att ja, vi anser att det här är OSS, och då är det lugnt." (Carl-Erik Mols, 2018).

Sony Mobile nämnde även att det är viktigt med strukturerade processer och att deras Head of Open Source måste se över alla val av nya OS-komponenter som görs inom verksamheten:

"Vårt bästa försvar mot sådana här copyright breach of license är att se till att vi har processer på plats, och det har vi. [...] När vi väljer att ta in Open Source görs en Open Source-evaluation som skickas till mig. Sen får legal godkänna licensen [...]" (Carl-Erik Mols, 2018).

4.1.4 Resultattabell för Open Source Software

	Säkerhet	Support	Licenser
LDC	Ser inga större problem. Använder endast OSS från stora och etablerade communities.	Support kan ibland vara bristfällig eftersom det inte finns några incitament för utvecklare att lösa specifika problem som LDC har. Nämnde också att det finns bra communityn som har stort fokus på användare och som erbjuder träning.	Undersöker inte licenser men är ganska säker på att de inte bryter mot någon licens eftersom de inte vidareutvecklar kod.
Sony Mobile	Tycker OSS är säkrare än proprietärt men menar att det ändå finns säkerhetsrisker	Supporten kan skifta. Oftast får de bra respons från communityn eftersom Sony har bra rykte i OSS-communityt.	Legal-avdelningen sköter licenser
CGI	Tycker att det blivit bättre på senare tid. Ser biblioteksberoenden som ett stort problem.	Tycker att supporten hos OSS-communityn kan vara bättre än proprietär.	Följer CGI:s licenspolicy
Prevas	Uttryckte oro inom IoT.	Tycker att supporten är bra i stora communities.	Gör GAP-analyser för att genomsöka och kategorisera licenser
Organisation X	Nämnde inga specifika problem.	Använder mycket proprietära lösningar och visste därför inte särskilt mycket gällande OSS-support	Software Asset Management-avdelningen sköter licenser

Tabell 4.1: Resultattabell för Open Source Software

4.2 IT Risk Management

Överlag var de flesta intervjupersoner mer involverade och kunniga inom OSS och utveckling än ITRM. CGI, LDC och Sony Mobile nämnde att deras organisationer arbetar efter ITRM-ramverk, mer specifikt ISO 27000-serien (LDC & CGI) och NIST (Sony Mobile). Dock var detta något som hanterades ur ett bredare verksamhetsperspektiv och därför inte märktes av speciellt mycket i det dagliga arbetet hos CGI och LDC. På frågan om organisationen använder något ITRM-ramverk svarade CGI:

> "Nej, alltså jag vet ju att CGI globalt jobbar mot ISO men det är inget jag har varit i kontakt med under utveckling eller sådär." (Philip Vendil, 2018)

Sony Mobile svarade att ITRM hanteras av Software Security-avdelningen men att risktänket även finns inom OSS:

> "Ja, nu sitter jag ju omgiven av mina kollegor som tillhör software security [...]. Lika mycket systematik som de har inom security har jag inom Open Source." (Carl-Eric Mols, 2018)

Person X var mer involverad i ITRM-processen och förklarade att den genomsyrar allt de gör:

> "[...] vi har en funktion i mitt team som kallas precis IT Risk Management [...]. Vi jobbar både proaktiv och reaktivt med riskhantering och omsätter det mesta, alla förändringar vi gör går igenom våran riskprocess och vi jobbar oftast, i alla fall när det kommer jobb, riskbaserat." (Person X, 2018)

4.2.1 Riskstyrning

Vid frågan gällande hur väl informerade ledningen är om OSS-risker var svaren blandade. Sony Mobile, CGI och Organisation X svarade alla att ledningen hade god förståelse för riskerna och möjligheterna kring användandet av OSS. Både Sony Mobile och CGI nämnde att ledningen varit med och tagit fram en OSS-policy som beskriver hur OSS ska användas. Organisation X arbetar med riskstyrning på ett bredare verksamhetsplan som innefattar alla risker:

> "Väl medvetna. Risk Managers stämmer av med CIOn varannan vecka vad progressen är på alla åtgärder för dom risker vi har identifierat. Det är en ganska levande organism, våran riskkarta, som vi ständigt jobbar med." (Person X, 2018).

Prevas svarade att de har sett det mesta när de är ute och besöker kunder och att mycket av det handlar om bakgrunden hos ledningen. Om en majoritet av ledningen har teknisk bakgrund eller god förståelse för IT går arbetet med risker kring OSS ofta bra, annars finns det en risk för friktion och att riskerna inte riktigt når fram.

LDC svarade att de ibland kan ha svårt att förklara riskerna och få ledningen medvetna om varför de måste åtgärdas proaktivt. Problemet ligger enligt LDC i att få ledningen att förstå varför man ska ta besult som kostar pengar nu för att möjligtvis få en vinst i framtiden. Speciellt blir det svårt då de inte har några säkerhetskrav på verksamheten:

"I vår organisation finns inget krav att man ska ha en viss säkerhet, förutom att vi ska försöka använda ISO27005." (Magnus Persson, 2018).

4.2.2 Riskbedömning

Samtliga intervjupersoner svarade att de genomför någon typ av riskbedömning av OSS-komponenterna de använder men tillvägagångssätten varierar. Både Sony Mobile, CGI, och Prevas använder automatiserade verktyg som upptäcker risker och problem med OSS-komponenter:

> "[...] men där har det ju börjat komma lite sårbarhetsanalyser vet jag. På GitHub vet jag till exempel att man automatiskt kan slå på att den varnar om man har en beroende som har nån rapporterad sårbarhet. Det går också i själva byggmiljön att ha en plugin som kollar upp sårbarheter i dom beroenden man har" (Philip Vendil, 2018).

Pluginet som Vendil nämner används för att kolla upp sårbarheter i OSS-sårbarhetsdatabaser.

Sony Mobile använder liknande verktyg men mer av en automatiserad helhetslösning:

> **"**Vi har krav och processer för att ta in Open Source och granska koden. Där använder vi oss mycket av National Vulnerability Database. [...] Vi använder oss av ett verktyg som heter Black Duck Protex. Det är så inbyggt i verktygskedjan att det sker per automatik. När man laddar upp någonting till vår main branch så kommer det så småningom scannas av och sen scannas det av varje dygn flera gånger osv. Så det är helt automatiserat." (Carl-Eric Mols, 2018).

För Prevas beror riskbedömningen på deras kunders krav men, likt Sony Mobile, har de också använt Black Duck:

> "Det försöker vi göra i den mån vi får lov av kunden, eller får tid eller där vi kan övertyga kunden om att ja men det är nog rätt vettigt att göra det. [...] Men vi försöker ta reda på för kunden, är ni medvetna, har ni gjort någon riskanalys, är ni medvetna om vad som händer, har ni vägt in dom parametrarna i valet? [...] bland annat finns det verktyg för det, till exempel ett företag som heter Black Duck [...] och det har jag varit med om att vi har använt." (Håkan Erlandsson, 2018).

LDC och Organisation X använder inga verktyg men har manuella processer på plats för att godkänna OSS och identifiera risker. LDC gör en enkel riskbedömning genom att kolla på OSS-specifika faktorer som de anser viktiga:

> "Vi kollar på communityn, produkten, livslängd, patchfrekvens för att försöka avgöra om det är värt det eller inte." (Magnus Persson, 2018).

Organisation X har en godkännandeprocess för OSS-komponenter som görs av vissa IT-arkitekter och säkerhetsfunktionen:

> "Det kräver ändå en extra sväng av godkännande och utlåtande från dom som vill, ja, dom inom IT-organisationen framförallt vår Enterprise-arkitekt och dess nätverk av IT-arkitekter. Men också av våran säkerhetsfunktion, för att dels kunna säga att jo men den här är i sitt nuvarande stadie okej men också för att få till en kontinuerlig bevakning över OS-komponenter." (Person X, 2018)

På frågan om organisationerna använder någon formel (T * V * A = R) för att beräkna IT-risker hade samtliga intervjupersoner hört talas om den men kunskapen var inte speciellt djup. LDC tyckte att det var svårt att beräkna risker på det sättet men att formeln hade ett teoretiskt värde. Sony Mobile, CGI, och Organisation X hade alla sin egna versioner av formeln.

4.2.3 Riskhantering

Då riskhanteringen är specifik beroende på vilken risk det gäller ser svaren ganska olika ut inom detta område. LDC mitigerar OSS-risker genom att välja OSS-komponenter som är uppbackade av ett aktivt och stort community, samt genom att undvika OSS i sina huvudplattformar:

> "Vi väljer inga små konstiga OSS då vi måste kunna garantera drift på ett eller annat sätt [...] Problemet ligger att vi måste ha leverantörer att skylla på/peka på om något går snett. Gällande till exempel huvudplattformarna måste vi därför ha en proprietär leverantör för att kunna peka på vid diverse problem." (Magnus Persson, 2018).

CGI har en kontinuerlig riskhanteringsprocess där de identifierar och analyserar risker för att sedan åtgärda dem:

> "Och sen [efter riskbedömning] har vi regelbundna möten där vi går igenom alla punkter som man skulle ha åtgärdat, där man ser status, så att det jobbas ju med regelbundet." (Philip Vendil, 2018).

Organisation X säger följande om riskhantering på ett generellt plan:

> "Vi jobbar både proaktiv och reaktivt med riskhantering och omsätter det mesta, alla förändringar vi gör går igenom våran riskprocess och vi jobbar oftast, i alla fall när det kommer jobb, riskbaserat." (Person X, 2018).

Vidare förklarar Person X att efter riskbedömning kan riskerna hanteras på olika sätt:

> "Utkomsten av den [riskbedömningen] kan ju antingen bli att ja, vi lever med risken eller vi kan göra någonting för att mitigera risken eller vi får se till att mitigera den på ett annat håll eller att vi inte vill leva med risken, det vill säga att vi säger nej till denna förändring. Det leder då till att man inte får fortsätta använda komponenten eller att man inte tar in den från första början." (Person X, 2018).

Sony Mobile:s riskhanteringsprocess involverar den juridiska avdelningen och Intellectual Property-avdelningen:

> "När vi väljer att ta in Open Source görs en Open Source-evaluation som skickas till mig. Sen får legal godkänna licensen och IP-avdelningen säger att vi ser inga IP-problem kring det hela, de gör en enkel IP-search." (Carl-Eric Mols, 2018).

Vidare påpekar Mols att kulturen och medvetenheten hos de anställda är det bästa försvaret mot risker:

> "Det är kanske det bästa försvaret mot risker, det är att få upp ett mindset, och det har vi fått upp här. Folk här är ganska skickliga i Open Source, de vet om licensriskerna och andra risker. De är medvetna om processerna och vet vart man ska gå ifall man undrar något." (Carl-Eric Mols, 2018).

4.2.4 Resultattabell för IT Risk Management

	Riskstyrning	Riskbedömning	Riskhantering
LDC	Kan vara svårt att förklara riskerna och få ledningen medvetna om varför de måste åtgärdas proaktivt.	Kollar på communityn, produkten, livslängd, och patchfrekvens.	Väljer OSS-komponenter som är uppbackade av aktiva och stora communities. Undviker OSS i huvudplattformar.
Sony Mobile	Väl medveten ledning. Förstår både risker och möjligheter men så har inte alltid varit fallet.	Använder Black Duck Protex och National Vulnerability Database för att identifiera OSS-risker.	Har en godkännandeprocess som involverar Head of Open Source, Legal-avdelningen, och IP-avdelningen. Har OSS-policy. Hög mognadsgrad i OSS.
CGI	Väl medveten ledning. Förstår både risker och möjligheter.	Använder GitHubs sårbarhetsanalyser och plugin som checkar mot Common Vulnerabilities and Exposures-databaser	Har en kontinuerlig riskhanteringsprocess där de identifierar och analyserar risker för att sedan åtgärda dem. Har OSS-policy.
Prevas	Menar att mycket handlar om bakgrunden hos kundens ledning.	Har använt Black Duck:s mjukvara för att identifiera OSS-risker. Beror på kundens krav. Försöker ta reda på om kunden är medvetna om riskerna. Om inte, försöker de övertyga kunden att riskerna måste hanteras.	Ser att många företag inte har någon riskkultur, speciellt inom IoT. Riskhanteringen prioriteras ner.
Organisation X	Väl medveten ledning. Avstämning med CIO varannan vecka.	Har en godkännandeprocess för OSS-komponenter som görs av vissa IT-arkitekter och säkerhetsfunktionen.	Inget svar om OSS-specifik riskhantering. Jobbar med riskhantering på ett generellt plan som innefattar alla risker i organisationen.

Tabell 4.2: Resultattabell för IT Risk Management

5 Diskussion

I kapitel fem analyseras och diskuteras datan från den empiriska undersökningen ytterligare och ställs mot tidigare forskning och litteratur som presenterats i kapitel två. Diskussionen är, likt tidigare, kategoriserad efter undersökningsmodellens huvudområden.

5.1 Open Source Software

Datainsamlingen gav bilden att intervjuobjekten till stor del var väldigt positiva till OSS och att användningen uteslutande hade givit mer fördelar än nackdelar.

5.1.1 Säkerhet

Gällande säkerhetsfrågor var majoriteten av reaktionerna positiva till säkerhetsarbetet som utförs inom utvecklandet av OSS. Både Sony Mobile, CGI, och Prevas pratade om möjligheten att använda verktyg som automatiskt söker igenom all kod och jämför den mot en databas av OSS för att hitta om de använder sig av OSS med kända sårbarheter. Dessa verktyg strävar mot att motarbeta ett av de största säkerhetshoten som finns kring användningen av OSS vilket är kontroll av koden (Booch & Brown, 2002; Payne, 2002; Schryen, 2011). LDC använde sig av argumentet att de enbart tar in stora och väletablerade OSS och att de förlitar sig på mjukvarans höga antal användare och således antal utvecklare som ett facit på att det är säkert, vilket enligt både Booch & Brown (2002) och Payne (2002) är ett av de starkaste argumenten för varför OSS inte innehar större säkerhetsrisker än proprietär mjukvara. Utöver de problem som tas upp i litteraturen såg CGI deriverade sårbarheter som ett av de farligaste hoten gällande säkerhet. Back & Silic (2017) och Conradi et al (2010) tar visserligen upp deriverade licensbrott men nämner aldrig deriverade sårbarheter som ett problem som OSS kan besitta.

5.1.2 Support

Support var den kategori inom OSS som fick flest och starkast positiva reaktioner från intervjuobjekten. Majoriteten svarade att om man använder sig av ett OSS som är välanvänt och som har ett stort community, är det lika bra om inte bättre än proprietär support. CGI sade att de brukar få sämre hjälp från proprietära återförsäljare än OSS, då det tar lång tid innan man får prata med en anställd som har rätt kompetens. Detta är raka motsatsen mot det Brown & Giera (2004) samt Conradi et al (2010) säger, där de påpekar att organisationer är rädda för att inte kunna få kontakt med experter inom mjukvaran när det gäller OSS.

LDC pratade om att det ibland kan vara svårt att få specifik support då de som utvecklar en OSS kanske inte har samma mål med mjukvaran som du som användare har, vilket faller i linje med det Buxmann et al (2013) och Conradi et al (2010) påstår. Detta medför problemet att om en organisation behöver hjälp med något eller ber utvecklarna ändra en funktion finns det inget incitament för utvecklarna att lyda förfrågan. Av alla intervjuer var detta den enda kritiken som riktades mot support hos OSS. LDC sa utöver detta att de ändå får väldigt bra support i form av träning och utbildning från Elastic Stack vilket är ett OSS som följer

freemium-modellen (Androutsellis-Theotokis et al. 2011; Au et al. 2014; Buxmann et al. 2013).

5.1.3 Licenser

Vid datainsamlingen gällande licenser visade de flesta organisationer på ett väletablerat och moget arbete som behandlade kontroll av licenser centralt inom organisationen. Organisation X, Sony Mobile och CGI hade policys och riktlinjer skapade av organisationens juridiska avdelningar som avgjorde om ett OSS fick användas eller ej. Inom både Sony Mobile och Organisation X var alla OSS tvungna att skickas till denna juridiska avdelning för bedömning innan det kunde användas. CGI svarade att de har en tydlig policy över vilka licenser man får eller inte får använda sig av vid utveckling av mjukvara och sedan är det upp till utvecklarna själva att följa denna policy. Det mer centraliserade licensarbetet som dessa organisationer utför går hand i hand med det som enligt litteraturen sägs vara en av de viktigaste förebyggande åtgärderna för att undvika licensbrott (Back & Silic, 2017; Brown & Giera, 2004; Conradi et al. 2010).

LDC var inte säkra att de följde alla licenser de använde sig av, men förklarade detta med att de inte vidareutvecklar någon kod och således besitter en låg risk att bryta mot licenser. På grund av hur universitets licensmodeller fungerar sade LDC att det snarare är större risk att de bryter mot proprietära licenser. Trots att det enligt litteraturen är en kritisk risk att inte undersöka och kontrollera sin licenser (Back & Silic, 2017; Brown & Giera, 2004; Conradi et al. 2010) har LDC arbetat problemfritt med OSS.

5.2 IT Risk Management

Resultatet av datainsamlingen visar att en majoritet av aspekterna inom ITRM beaktas av organisationerna vid användning av OSS. Samtliga organisationer redogjorde för hur de genomför riskbedömning och riskhantering av OSS-relaterade risker. Detta stämmer inte överens med att det finns en bristfällig applicering av ITRM inom OSS-adoption enligt Franch et al. (2013) och Franch & Susi (2016). Däremot saknades kunskap om ITRM-processen såsom den beskrivs i litteraturen. Fyra av de fem intervjuade organisationerna nämnde att de följde ett ITRM-ramverk men kunde inte utveckla hur de användes i praktiken. Detta visar på att det finns en riskmedvetenhet men också en frånkoppling av ITRM-ramverkens praktiska applicering vid användandet av OSS. Att de inte kommer i kontakt med just ITRM-ramverken även om deras organisation implementerat dem faller i linje med ENISA:s (2006) konstaterande att det saknas kunskap om ITRM inom organisationer som implementerar det.

5.2.1 Riskstyrning

Den aspekten av riskstyrning som berörts i den här uppsatsen är hur väl ledningen är medvetna och involverade i OSS-relaterade risker. Både Sony Mobile, CGI, och Organisation X hade OSS-policies framtagna av ledningen vilket är en nyckelfaktor vid användningen av OSS enligt Belle et al. (2006). Dock påpekade Sony Mobile att ledningen inte alltid ställt sig positiv till och engagerat sig i OSS och LDC menade att detta existerar än idag. Skillnaden mellan ledningen hos Sony Mobile och LDC är att Sony Mobile:s ledning först var skeptiska

till användningen av OSS men sedan förstod fördelarna, medan LDC fortfarande hade problem att förmedla riskerna och få ledningen medvetna om varför de måste åtgärdas proaktivt. Vi tycker oss se en parallell mellan LDC:s ledning och det Prevas sade om att bakgrunden hos ledningen spelar roll. En ledning med en god förståelse för IT förstår både risker och möjligheter i OSS medan en ledning utan teknisk bakgrund har svårt att se vinsten i att investera IT-riskförebyggande. Citatet från Magnus Persson på LDC är ganska talande: "Det är alltid svårt att sälja in till ledningen, hade vi inte haft en lag som säger att man måste ha trafikförsäkring hade många skippat det." (Magnus Persson, 2018).

5.2.2 Riskstyrning

Det går inte att hitta någon gemensam eller standardiserad process för riskbedömningen då samtliga företag använder olika tillvägagångssätt. Dock kan vi konstatera att samtliga på olika sätt identifierar och bedömer OSS-relaterade risker. Något som tre av de fem företagen nämnde var att de använde eller har använt OSS-riskverktyg. Detta kan stödja, förenkla, och förbättra ITRM-processen enligt Bai et al. (2014) och Franch et al. (2015). LDC, som inte använde något OSS-riskverktyg, utförde istället en riskbedömning som var mycket snarlik *The Open Source Maturity Model* (Woods & Guliani, 2005). Dock saknar de en process för att hantera licensrelaterade risker vilket de försvarar med att de inte vidaredistribuerar någon kod, allt används internt.

Riskformeln (T * V * A = R) för att beräkna IT-risker var inte något som intervjupersonerna arbetade med i någon större utsträckning. Majoriteten hade sina egna versioner av formeln men det var inget som märktes av när OSS-risker beräknades. LDC kritiserade formeln och resten hade sina egna versioner av den. Detta faller i linje med att formeln fått kritik för att vara för generell vilket resulterat i många olika revideringar (ISACA, 2009; Katsikas, 2013; Kouns & Minoli, 2010).

5.2.3 Riskhantering

Precis som med riskbedömningen ser resultatet olika ut även inom riskhanteringen. Sony Mobile och CGI har en OSS-policy som specificerar arbetet med OSS. Detta är enligt Woods & Guliani (2005) en faktor som kan reducera OSS-relaterade risker. Sony Mobile nämnde att de ansåg att deras anställda som jobbar med OSS är kompetenta, medvetna om riskerna och vet vart man ska vända sig om man behöver hjälp. Även detta nämns i litteraturen som en viktig faktor för att reducera risker (Bonaccorsi et al. 2006; Exton et al. 2005; Gomez & Moreno, 2012).

LDC mitigerar OSS-risker genom att endast välja OSS-komponenter med många användare och ett stort och aktivt community. Detta kan härledas till Ågerfalk et al:s (2005) konstaterande om att en populär OSS som implementerats av andra organisationer är en framgångsfaktor. Vidare undviker LDC OSS i deras huvudplattformar eftersom de måste ha en leverantör som de kan peka på vid diverse problem. Här ser vi ett tydligt exempel på när man väljer att undvika risk (ISACA, 2009; ISO, 2008) då den bedöms ha för hög risknivå. Även Organisation X var medvetna när det kom till olika sätt att hantera OSS-risker. De utgick ifrån att man antingen undviker, reducerar, överför, eller accepterar risk vilket stöds i litteraturen av ISO (2008) och ISACA (2009).

6 Slutsats

Vårt huvudsakliga syfte med den här uppsatsen var att undersöka om teorin kring ITRM korrelerar med organisationers hantering av OSS-relaterade risker. Därav var forskningsfrågan följande: *Vilka aspekter av IT Risk Management beaktas när organisationer använder sig av Open Source Software?*

I vår undersökning har vi kunnat konstatera att en majoritet av aspekterna inom ITRM beaktas när de intervjuade organisationerna använder OSS. Däremot saknas kunskap om ITRM-processen såsom den beskrivs i litteraturen. Denna paradox tror vi kan bero på att ITRM hanteras ur ett bredare verksamhetsperspektiv och inte specifikt för att hantera OSS-risker. Att vi ändå ser att ITRM:s aspekter beaktas visar på att riskmedvetenhet och kunskap finns men kommer nödvändigtvis inte från ITRM-ramverken.

Inom riskstyrning beaktade tre av fem organisationer att ledningen måste vara införstådd i såväl möjligheter som risker som kan medfölja vid användandet av OSS. Här fick vi ofta svaret att det har blivit bättre med tiden vilket vi tolkar som en ökande mognad av OSS inom organisationer.

Vidare kan vi konstatera att alla organisationer genomför riskbedömning men att tillvägagångssätten är unika för varje organisation. I denna riskbedömning identifieras och bedöms OSS-risker och en majoritet av organisationerna använder riskverktyg. Att använda ett automatiserat riskverktyg ser vi, och litteraturen, som ett mycket nödvändigt steg i att förebygga licens- och säkerhetsrelaterade OSS-risker eftersom manuella processer omöjligt kan fånga upp alla dessa risker. En människa kan helt enkelt inte analysera tusentals, eller miljontals, rader kod för att hitta säkerhetshål eller leta upp alla licenser för alla OSS-komponenter som används och kontrollera att inget licensbrott sker.

Utöver användningen av OSS-riskverktyg fanns även manuella processer på plats hos majoriteten av organisationerna för att komplettera analys av OSS-risker som automatiserade verktyg inte kan fånga upp. Här belystes aspekter som community, support, och risk för övergivna projekt.

Riskhantering beaktades unikt inom varje organisation och tog form genom bland annat tydliga policys för hantering av OSS eller genom att enbart använda välkända OSS och således förlita sig på att andra organisationer redan genomfört riskbedömning på mjukvaran. Vi tycker att den sistnämnda aspekten av riskhantering är godkänd då dessa communities och OSS är mycket etablerade och av god kvalitet.

Den aspekt som lyste med sin frånvaro i datainsamlingen var användningen av riskformeln (T * V * A = R). Kontentan var att den antingen var för generell eller för teoretisk vilket vi, och delvis litteraturen, håller med om. Revideringar av formeln har gjorts, både inom litteraturen och av de intervjuade organisationerna. Detta tolkar vi som ett bevis på att det är svårt att beräkna och estimera risker.

De problem och risker med OSS som introducerades i kapitel ett och som utvecklades i kapitel två var överlag inte något som de intervjuade organisationerna kände igen sig i. Detta tolkar vi som att riskhanteringen och riskmedvetenheten är fungerande samt en ökande mognad av OSS inom organisationerna.. Vi är medvetna om att det i kapitel ett presenterades

rapporter och exempel på att riskerna och problemen med OSS existerar och blir till verklighet. Att vår slutsats delvis motsäger detta tror vi beror på att de organisationer som blivit utsatta är mer attraktiva måltavlor än de organisationer vi intervjuat i denna uppsats.

När vi reflekterar över uppsatsen är vi nöjda med den data vi samlat in och tycker att vi har fått ett godtyckligt svar på vår forskningsfråga. Trots att studien varit av kvalitativ ansats där intervjumallen innehöll många öppna frågor har den insamlade datan i hög grad korrelerat med litteraturen som ligger till grund för ämnesområdet. Detta tolkar vi som att både den litteratur vi har analyserat samt de organisationer vi haft kontakt med har varit av mycket god relevans för att uppnå uppsatsens syfte.

7 Referenser

Androutsellis-Theotokis, S., Spinellis, D., Kechagia, M., Gousios, G. (2011). Open Source Software: A Survey from 10,000 Feet. Foundations and Trends in Technology, Information and Operations Management: Vol. 4: No. 3–4, s. 187-347.

Aner, Y., & Cid, C. (2010). Open-source security assessment. Royal Holloway Series.

Au, Y., Choi, H., Liu, C. (2014). Effects of Freemium Strategy in the Mobile App Market: An Empirical Study of Google Play, *Journal of Management Information Systems*, 2014, Vol. 31, s. 326-354. Tillgänglig via LUBsearch: http://ludwig.lub.lu.se/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=102702526&site=eds-live&scope=site [Hämtad 17 April 2018]

Back, A., Silic, M. (2017) Open Source Software Adoption: Lessons from Linux in Munich, *IT Professional (IT Pro)*, 1, s. 42.Tillgänglig via LUBsearch: http://ludwig.lub.lu.se/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.7839852&site=eds-live&scope=site [Hämtad 25 April 2018]

Backman, J. (2016) Rapporter och Uppsatser. Lund: Studentlitteratur

Bai, X., Yahav, I., Kenett, R. S., (2014). Risk Based Testing of Open Source Software (OSS). 2014 IEEE 38th International Computer Software and Applications Conference Workshops. doi:10.1109/compsacw.2014.107

Belle, J., Brink, D., Roos, L., Weller, J. (2006), Critical Success Factors for Migrating to OSS-on-the-Desktop: Common Themes across Three South African Case Studies. In Damiani, E., Fitzgerald, B., Scacchi, W., Scotto, M., Sued, G. (Eds.) Open Source Systems. USA: Springer.

Black Duck Software (2017) Open Source Security & Risk Analysis, Tillgänglig via: https://www.blackducksoftware.com/download/open-source-security-risk-analysis-2017 [Hämtad 22 Mars]

Bonaccorsi, A., Giannangeli, S., & Rossi, C. (2006). Entry Strategies Under Competing Standards: Hybrid Business Models in the Open Source Software Industry. Management Science, 52(7), 1085-1098. doi:10.1287/mnsc.1060.0547

Booch, G., Brown, A. (2002). Reusing Open-Source Software and Practices: The Impact of Open-Source on Commercial Vendors, *ICSR-7 2002 Proceedings*, Tillgänglig via: https://link.springer.com/content/pdf/10.1007%2F3-540-46020-9.pdf [Hämtad 11 April 2018]

Brodkin, J. (2010) Microsoft: 'We love open source', *NetworkWorld*, 23 Augusti, Tillgänglig via: https://www.networkworld.com/article/2216878/windows/microsoft---we-love-open-source-.html [Hämtad 28 Mars 2018]

Brown, A. & Giera, J. (2004) The Costs And Risks Of Open Source: Debunking The Myths, Forrester Inc.

Buxmann, P., Diefenbach, H., Hess, T. (2013). The Software Industry Economic Principles, Strategies, Perspectives. Tyskland: Springer

Caballero, A. (2009). Information Security Essentials for IT Managers: Protecting Mission-Critical Systems. In J. Vacca (Ed.) Computer and Information Security Handbook. USA: Morgan Kaufmann Publishers.

Caralli, R. A., Stevens, J. F., Young, L. R., & Wilson, W. R. (2007). Introducing OCTAVE Allegro: Improving the Information Security Risk Assessment Process. Tillgänglig via: https://resources.sei.cmu.edu/asset_files/TechnicalReport/2007_005_001_14885.pdf [Hämtad 3 april]

Carr, H. H., Rainer, R. K., & Snyder, C. A. (1991). Risk Analysis for Information Technology. Journal of Management Information Systems, 8(1), 129-147. doi:10.1080/07421222.1991.11517914

Carver, B. (2005). Share and Share Alike: Understanding and Enforcing Open Source and Free Software Licenses, *Berkeley Technology Law Journal*, vol 1, s. 443-481

CompTIA (2016). Cyberstates 2016, Tillgänglig via: https://www.comptia.org/advocacy/resources/cyberstates-2016?tracking=cyberstates [Hämtad 28 Mars 2018]

Conradi, R., Cruzes, D., Hauge, Ø., Skarpenes, T., Velle, K., (2010) Risks and Risk Mitigation in Open Source Software Adoption: Bridging the Gap between Literature and Practice. P. Ågerfalk et al. (Eds.) *Open Source Software: New Horizons*: IFIP AICT 319, s. 105–118.

Deloitte (2016) Global risk management survey, 10th edition, Tillgänglig via: https://www2.deloitte.com/insights/us/en/topics/risk-management/global-risk-management-survey.html [Hämtad 28 Mars 2018]

ENISA (European Network and Information Security Agency). (2006). Risk Management: Implementation principles and Inventories for Risk Management/Risk Assessment methods and tools. Tillgänglig via: https://www.enisa.europa.eu/publications/risk-management-principles-and-inventories-for-risk-management-risk-assessment-methods-and-tools [Hämtad 1 april]

Ernst & Young (2014) Addressing the evolving challenges of IT risk: IT Risk Management Survey. Tillgänglig via: http://www.ey.com/Publication/vwLUAssets/EY-it-risk-management-survey-2014/$FILE/EY-it-risk-management-survey-2014.pdf [Hämtad 26 mars 2018]

Exton, C., Fitzgerald, B. & Glynn, E. (n.d.). Commercial adoption of open source software: An empirical study. 2005 International Symposium on Empirical Software Engineering, 2005. doi:10.1109/isese.2005.1541831

Feller, J., Fitzgerald, B., Scacchi, W., Sillitti, A. (2007). Open Source Development, Adoption and Innovation. Irland: Springer

Fitzgerald, B. (2009). Open Source Software Adoption. Software Applications, 1675-1698. doi:10.4018/978-1-60566-060-8.ch099

Franch, X., Susi, A., Annosi, M., Ayala, C., Glott, R., Gross, D., Kenett, R., Mancinelli, F., Ramsamy, P., Thomas, C., Ameller, D., Bannier, S., Bergida, N., Blumenfield, Y., Bouzereau, O., Costal, D., Domingues, M., Haaland, K., Lopez, L., Morandini, M., Siena, A. (2013). Managing Risk in Open Source Software Adoption. Proceedings of the 8th International Joint Conference on Software Technologies.

Franch, X., & Susi, A. (2016). Risk Assessment in Open Source Systems. 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C), 896-897. Tillgänglig via LUBsearch: http://ludwig.lub.lu.se/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.7883433&site=eds-live&scope=site [Hämtad 3 Maj 2018]

Franch, X., Kenett, R. S., Susi, A., Galanis, N., Glott, R., & Mancinelli, F. (2015). Community Data for OSS Adoption Risk Management. The Art and Science of Analyzing Software Data, 377-409.

Free Software Foundation (2008). Free Software Foundation Files Suit Against Cisco For GPL Violations, Tillgänglig via: https://www.fsf.org/news/2008-12-cisco-suit [Hämtad 2 April 2018]

Gates, B. (1976). “An open letter to hobbyists,” Homebrew Computer Club Newsletter, vol. 2, no. 1, p. 2, January.

GitHub (2017). Open Source Survey, Tillgänglig via: http://opensourcesurvey.org/2017 [Hämtad 2 april 2018]
GitLab (2016). 2016 Global Developer Report, Tillgänglig via: https://page.gitlab.com/2016-developer-survey_2016-developer-survey.html [Hämtad 30 Mars 2018]
Gomes, J. C. & Moreno, V. D. (2012). Benefits And Success Factors Of Open-Source Web Services Development Platforms For Small Software Houses. Journal of Information Systems and Technology Management, 9(3). doi:10.4301/s1807-17752012000300008
Goode, S. (2005). Something for nothing: Management rejection of open source software in Australia's top firms. Information & Management, 42(5), 669-681. doi:10.1016/j.im.2004.01.011
Greene, T. (2001) Ballmer: "Linux is a cancer", *The Register*, 2 Juni, tillgänglig via: https://www.theregister.co.uk/2001/06/02/ballmer_linux_is_a_cancer/ [Hämtad 30 Mars 2018]
Hubbard, D. (2009) The Failure of Risk Management. USA: John Wiley & Sons Inc
IDATE (2016). DigiWorld Yearbook 2016, Tillgänglig via: https://en.idate.org/product/digiworld-yearbook-2016/ [Hämtad 28 Mars 2018]
ISACA. (2009) Risk IT Framework. Tillgänglig via: http://www.isaca.org/Knowledge-Center/Risk-IT-IT-Risk-Management/Documents/Risk-IT-Brochure.pdf [Hämtad 12 mars 2018]
ISO (International Organization for Standardization). (2009) ISO Guide 73. tillgänglig via: http://vai.org.vn/docs/ISO/ISO_GUIDE_731.PDF [Hämtad 12 april 2018]
ISO (International Organization for Standardization). (2013) ISO 27001. tillgänglig via: https://www.iso.org/isoiec-27001-information-security.html [Hämtad 2 april 2018]
ISO (International Organization for Standardization). (2008). tillgänglig via: https://www.iso.org/standard/56742.html [Hämtad 27 mars]
Jacobsen, D. (2002) Vad, hur och varför? Om metodval i företagsekonomi och andra samhällsvetenskapliga ämnen. Lund: Studentlitteratur.
Kadura & Schryen (2009) Open Source vs. Closed Source Software: Towards Measuring Security, *Proceeding SAC '09 Proceedings of the 2009 ACM symposium on Applied Computing*, pp. 2016-2023, tillgänglig via: https://dl.acm.org/citation.cfm?id=1529731 [Hämtad 28 Mars 2018]
Katsikas, S. K., (2013). Risk Management. In J. Vacca (Ed.) Computer and Information Security Handbook. USA: Morgan Kaufmann Publishers.
Kennet, R., & Raanan, Y. (2011). Operational Risk Management: A practical approach to intelligent data analysis. Chichester: John Wiley & Sons.
Koskelainen, A. (2018). Dataintrånget mot Equifax kan ha blivit det dyraste i företagshistorien, *ComputerSweden*, 5 Mars, tillgänglig via: https://computersweden.idg.se/2.2683/1.698804/equifax-dyrt [Hämtad 2 April 2018]
Kouns, J., & Minoli, D. (2010). Information Technology Risk Management in Enterprise Environments: A Review of Industry Practices and a Practical Guide to Risk Management Teams. Somerset: Wiley.
Linux Foundation (2018). Participating in Open Source Communities, Tillgänglig via: https://www.linuxfoundation.org/participating-open-source-communities/#2 [Hämtad 30 Mars 2018]
Mansfield-Devine, S. (2016). The secure way to use open source. Computer Fraud & Security, 2016(5), 15-20.
Mestre, P., Rodrigues, P. E. & Soares, S. (2011). Risk management, a open-source approach. 2011 IEEE EUROCON - International Conference on Computer as a Tool. doi:10.1109/eurocon.2011.5929374

Moroiu, G., Weiss, M. & Zhao, P. (2006). Evolution of Open Source Communities, *International Conference on Open Source Software*, Tillgänglig via LUBsearch, http://ludwig.lub.lu.se/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=edselc&AN=edselc.2-52.0-33749160317&site=eds-live&scope=site [Hämtad 10 April 2018]

National Vulnerability Database. (2018). CVSS Severity Distribution Over Time, Tillgänglig via: https://nvd.nist.gov/general/visualizations/vulnerability-visualizations/cvss-severity-distribution-over-time [Hämtad 11 april 2018]

NIST (National Institute of Standards and Technology). (2012). Tillgänglig via: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf [Hämtad 14 april 2018]

Open Source Initiative (2007). The Open Source Definition, Tillgänglig via: https://opensource.org/osd [Hämtad 30 Mars 2018]

Open Source Initiative (2017). Microsoft Makes it Official: Becomes Sponsor of Open Source Initiative, Tillgänglig via: https://opensource.org/node/901 [Hämtad 22 Mars 2018]

OpenLogic (2011). OpenLogic Scan Shows Open Source License Violations for iPhone and Android, Tillgänglig via: http://www.marketwired.com/press-release/openlogic-scan-shows-open-source-license-violations-for-iphone-and-android-1407577.htm [Hämtad 2 April 2018]

Payne, C. (2002) On the security of open source software, Information Systems Journal, vol.12, s. 61–78. Tillgänglig via LUBsearch: http://ludwig.lub.lu.se/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=6066186&site=eds-live&scope=site [Hämtad 13 April 2018]

Petrinja, E., Nambakam, R., & Sillitti, A. (2009). Introducing the OpenSource Maturity Model. 2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development. doi:10.1109/floss.2009.5071358

Schief, M. (2014). Business Models in the Software Industry, Darmstadt: Springer Gabler

Schryen, G. (2011). Is Open Source Security a Myth? *Communications of the ACM.* Vol. 54, Issue 5, s. 130-140. Tillgängliga via LUBsearch: http://ludwig.lub.lu.se/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=60863987&site=eds-live&scope=site [Hämtad 5 April 2018]

Snyk (2017). The State of Open Source Security, Tillgänglig via: https://snyk.io/stateofossecurity/pdf/The%20State%20of%20Open%20Source.pdf?utm_source=blog&utm_campaign=osssecurity [Hämtad 30 Mars 2018]

Stallman, R. (2002). Free Software, Free Society: Selected Essays of Richard M. Stallman, tillgänglig via: https://www.gnu.org/philosophy/fsfs/rms-essays.pdf [Hämtad 28 Mars 2018]

Svahn, C., Örstadius, K. (2017). Sparkad generaldirektör röjde sekretessbelagda uppgifter, *DN*, 6 Juli, tillgänglig via: https://www.dn.se/nyheter/sverige/sparkad-generaldirektor-rojde-sekretessbelagda-uppgifter/ [Hämtad 3 April 2018]

Techopedia (2018) IT Risk Management. Tillgänglig via: https://www.techopedia.com/definition/25836/it-risk-management [Hämtad 25 mars 2018]

W3Techs (2018-a). Usage statistics and market share of Unix for websites, Tillgänglig via: https://w3techs.com/technologies/details/os-unix/all/all [Hämtad 22 Mars 2018]

W3Techs (2018-b). Usage of content management systems for websites, Tillgänglig via: https://w3techs.com/technologies/overview/content_management/all [Hämtad 22 Mars 2018]

Wasserman, T. (2014, February 21). 7 Reasons Not to Use Open Source Software (Rubens, P., Intervjuare) tillgänglig via: https://www.cio.com/article/2378859/open-source-tools/7-reasons-not-to-use-open-source-software.html [Hämtad 2 april 2018]

Woods, D., & Guliani, G. (2005). Open source for the enterprise: Managing risks, reaping rewards. Sebastopol, CA: OReilly.

Ågerfalk, P. J., Deverell, A., Fitzgerald, B., & Morgan, L. (2005). Assessing the role of open source software in the European secondary software sector: a voice from industry. Conference on Open Source Software

8 Appendix

8.1 Intervjumall

Intervjuperson:
Titel:
Företag:
Antal anställda:
Kort beskrivning av företaget:
Förklaring av vår uppsats: Vår uppsats handlar om hur organisationer hanterar risker och problem vid användning och val av Open Source-mjukvara. Vi vet att Open Source har många positiva sidor som att det främjar innovation, minskar kostnader och accelererar utvecklingstiden. Vi vet också att det finns risker och problem med Open Source som till exempel säkerhetshål, buggar, licenser, dokumentation, support och övergivna projekt. Det vi vill undersöka är om organisationer som använder Open Source tänker på de här sakerna och om det finns utarbetade riktlinjer eller processer för att hantera riskerna och problemen.

Våra begreppsdefinitioner:
Open Source
- Källkod som är fullt tillgänglig för andra att modifiera och distribuera utan begränsningar

IT Risk Management
- Hanteringen av IT-relaterade risker och problem
- Inom OS: säkerhetshål, buggar, licenser, support, dokumentation

Frågor:
- Vad är din uppfattning av begreppet OS-mjukvara?
  - Om annorlunda än vår, presentera vår uppfattning
- Vad är din uppfattning om/av begreppet IT Risk Management?
  - Om annorlunda än vår, presentera vår uppfattning
- Kan ni ge några exempel på OS-mjukvaror ni använder?
- Hur går det till när ni väljer OS-mjukvara?
  - Är det fritt fram för utvecklarna att välja vad de behöver eller har ni specifika riktlinjer?
  - Gör ni någon form av riskidentifiering, riskbedömning, eller riskhantering av OS-mjukvaran?
    - Hur identifieras risker?
    - Hur går riskbedömningen till?

        - Uppdateras riskbedömningen kontinuerligt efter att OS-mjukvaran är implementerad?
    - Hur hanteras riskerna?
    - Använder ni något OSS-specifikt riskhanteringsverktyg, som RISCOSS t.ex.?

- Har ni upplevt några problem med OS-mjukvaran? (buggar, support/dokumentation, säkerhetshål)
    - Om ja, vad och hur har ni hanterat de problemen?
    - Om inte, vad hade ni gjort ifall något problem skulle uppstå?

- Hur kontrollerar ni att OS-licenserna följs korrekt?
    - Om nej, hur vet ni att ni inte riskerar legala följder?

- Använder ni något IT Risk Management-ramverk som ISO 27005 eller ISACA:s IT Risk Framework (för att nämna några)?
    - Om ja, hur appliceras det på valet och användningen av Open Source-mjukvara?
    - Använder ni någon formel för att beräkna IT-risker? (R = T * V * A)
    - Har ni anställda som enbart jobbar med detta eller är det något som alla anställda tar del av som en del av arbetet?
    - Hur väl informerade/medvetna är ledningen/”högre instanser” om riskerna med OS?

8.2 Mailmall

Hej,

Vi är två studenter som studerar Systemvetenskap vid Lunds Universitet. Just nu håller vi på att skriva vår kandidatuppsats och letar efter företag att intervjua. Uppsatsen handlar om hur organisationer hanterar risker och problem vid val och användning av Open Source-mjukvara. Om ni ställer upp på en intervju kommer ni givetvis få tillgång till uppsatsen när den är färdig. Då kommer ni enkelt kunna jämföra er med hur andra företag som vi intervjuat jobbar med Open Source-mjukvara.

Vi skulle vilja komma i kontakt med någon som har ansvar eller kunskap om IT-arkitekturen i er organisation. Gärna någon som har koll på val eller inköp av mjukvara. En intervju tar max 30 minuter och önskas anonymitet kan vi ordna det. Personer med följande eller liknande titlar och roller är av intresse:

- CTO (IT-chef)
- CIO (Informationschef)
- IT-Projektledare
- IT Solution Manager/Architect
- IT-inköpare
- Teknisk Produktägare
- IT-beslutsfattare
- Utvecklare/Programmerare (helst någon senior med stor insikt i hur IT-beslut tas i företaget)

Med vänliga hälsningar,

Fredrik Moberg

Tel: 072-252 95 05

Mail: sys15fm1@student.lu.se

Axel Hellström

Tel: 0765-547203

Mail: muv13ahe@student.lu.se

8.3 Intervjutranskriberingar

8.3.1 LDC

Denna intervju spelades in men ljudfilen förstördes. Intervjun har återgetts så gott som möjligt via de anteckningar som fördes. Intervjuobjektet har även informerats om detta samt godkänt transkriberingen.

Intervjuperson: Magnus Persson (MP)
Titel: IT-Arkitekt
Antal anställda: 100
Kort beskrivning av företaget: LDC är ett centralt organ inom Lunds Universitet som ansvarar för många underliggande IT-tjänster som t.ex. drift och säkerhet.

Intervjuledare: Axel Hellström (AH)
Notarie: Fredrik Moberg (FM)

AH: Vi kan ju berätta lite om oss och vår uppsats då, vi går sista terminen på Lunds Universitet, systemvetenskap pluggar vi, och vi skriver uppsats, kandidatuppsats. Den handlar ju om som vi skrev i mailet där, hur företag hanterar risker och problem vid användning och val av Open Source mjukvara. Och vi har ju en ganska positiv inställning till OS då, men vi vet också att det finns risker och problem med OS som till exempel säkerhetshål, buggar, licenser, support, dokumentation, såna saker och vissa är ju inte specifika för OS, det finns ju säkerhetshål i proprietär programvara också såklart, men det är dom grejerna vi kollar på där. Så det vi vill ta reda på är om företag tänker på dom här sakerna över huvud taget, om det finns några utarbetade riktlinjer eller processer för att hantera dom här OS-specifika riskerna då. Så det ena vi kollar på är open source då, det andra vi kollar på är IT risk management kallas det, som är som en-, hanteringen av IT-relaterade risker och problem kan man säga.

AH: Vad har din avdelning för ansvarsområde?

MP: Vi har ett primäransvar för driften som ligger hos varje institution och centralt är det lite som beställts. Hur IT-driften ska skötas bestäms lokalt på institutionerna och LDC sköter därför inte driften för alla institutioner. LDC är en av dom 6 nationella datacentralerna för universitet som bildades för 50 år sedan ungefär, vi är dom enda som är kvar i den ursprungliga formationen. Vi är en service-organisation som säljer tjänsterna till universitetet och fakulteten, främst när det kommer till sånt som är svårt att lägga ut på andra tex telefonsystem. Vill en fakultet sköta driften själv som på Juridicum eller LTH så får man det, annars kan man betala LDC som gör det. Detta gör t.ex. LUSEM, samhällsfakulteten och några fler. Det finns några centrala avtal gällande nätet, drift av centrala applikationer vilket ibland sköts här och ibland i molnet, dessa tjänster driftas ibland av oss och ibland av en annan part. IT säkerhet är även en central del i vår organisation.

AH: Vad är din roll och dina arbetsuppgifter?

MP: IT säkerhet och inom det är jag IT Arkitekt.

AH: Vad är din uppfattning om/av begreppet Open Source-mjukvara?

MP: Egentligen så skulle jag vilja säga att det är FOSS, men det är inte bara det. Vi kör ju två operativsystem om man kollar i våra servrar, Windows eller Redhat. Redhat är inte i sig OSS men dom säljer tjänster som hanterar OSS. Sen har vi annan OSS som vi inte betalar för då vi anser att communityn är så pass stor att vi inte behöver avtal. Till exempel gällande Bind kan vi inte få ett service-avtal men det ingår i Redhat licensen. För IT säkerhet har vi i princip gått över till att köra elastic stack på nästan allting, log stash elastic search. Idag har den så stor community och det finns så många som jobbar i det att där är vi trygga med att den lever så länge som vi behöver den, då är valet enkelt för att vi inte har pengar att köpa något proprietärt. Ska vi ha den här funktionaliteten så har vi inte råd med proprietära val, vi skulle få mycket sämre lösning om vi skulle betala för den, vilket gör valet enkelt. Vi är så fattiga att vi inte kan betala men vi kör gärna OSS. Problemet ligger att vi måste ha leverantörer att skylla på/peka på om något går snett. Gällande till exempel huvudplattformarna måste vi

därför ha en proprietär leverantör för att kunna peka på vid diverse problem.

AH: Hur gör ni om ni inte kan peka på någon?

MP: Redhat kan vi ju peka på till exempel, om vi tittar på IT-säkerhetsverktyg så kan vi inte peka på någon, det går inte. Om vi står vid valet mellan att hitta något att betala för och att få den funktionen vi vill ha och inte betala men få funktion utan säkerhet, då är det enkelt för oss. Vi kollar på communityn, produkten, livslängd, patchfrekvens för att försöka avgöra om det är värt det eller inte. Universitetet körde openBSD i 10 år förut och vågade köra universitetets brandväggar för openBSDs trackrecord är så pass bra att vi vågade, om något händer med utvecklingen så får vi leva med den sista versionen tills vi har hittat en tjänst eller mjukvara att byta ut den med.

AH: Vad är din uppfattning om/av begreppet IT Risk Management?

MP: Ju mer IT du har desto större risk utsätter du dig för, man vet hur illa saker och ting fungerar och hur lätt det är att sänka kärnverksamhet i ett öppet nät som de flesta är.

AH: Gör ni någon form av riskbedömning, riskanalys eller utvärdering?

MP: Alla system som utvecklas, nu pratar vi inte enbart OSS, utan alla system som vi köper eller utvecklar, allt från stora som ladok ned till småsystem för analys eller liknande, måste hanteras enligt PM3. PM3 är en modell för hur man utvecklar projekt och förvaltar projekt, mer om förvaltning egentligen, och förvaltning av system, systemförvaltare, systemägare. Det är samma för IT-resurs, alla har olika roller och sedan går man igenom sin mall för hur man ska utveckla den, i PM3 ingår även en riskanalys.

AH: Hur utförs riskbedömningen?

MP: Man bör göra riskanalys om man använder OSS, man måste tänka "vad händer om min infrastruktur försvinner?". Det har hänt, inte ofta men det har hänt, till exempel kan utvecklarna forka av och bli ovänner och allt delar upp sig.

AH: Hur skiljer sig valet av OSS från valet av Proprietär mjukvara?

MP: Du har mer säkerhet om du väljer en proprietär mjukvara. Om IBM tar på sig att dom säljer en produkt har dom en livshanteringscykel, dom måste underhålla i flera år trots säljstopp. Den bestäms ofta långt i förväg så att man kan planera sin systemdrift enligt denna. Mozilla och Apache är för stora för att forka till exempel men mindre OSS kan lätt forka, kolla bara på openBSD.

AH: Har ni kollat på eller använder ni er av RISCOSS?

MP: Nej, vi använder inte det. Det är ganska få som tar med den i sin riskanalys så vitt jag vet. Vad händer om en kärndel i infrastruktur förändras på ett sätt så jag inte kan använda den

längre? Vi har haft saker och ting vi har behövt förändringar på, vi har skickat förslag till OSS-utvecklarna och dom har inte förstått varför det behöver förändras. Vi kan inte få något att förändras utan bara föreslå förslag och hoppas dom implementerar det, vi kan leverera "business impact report" till dom men vi betalar inget och då kan vi inte kräva någon förändring. Hade IBM fått samma rapport hade dom lyssnat då vi troligtvis är en stor kund hos dom.

AH: Har ni upplevt några problem med OS-mjukvaran? Till exempel säkerhetshål, buggar, support/dokumentation etc?

MP: Vi väljer inga små konstiga OSS då vi måste kunna garantera drift på ett eller annat sätt. De grejer vi kör i säkerhetsgruppen drabbar egentligen bara oss, om ett av våra verktyg står still drabbas bara vi, men det är illa nog. Om vi inte kan göra vårt jobb drabbas universitetet i slutändan. Min kollega Peter som jobbar mest med implementationen tekniskt ska nu gå på kurs i elastic stack i Schweiz, vilket visar på att communityn är stor och det finns stora och bra möjligheter att utbilda sig. Dock så har elastic stack ett företag bakom sig. Vissa verktyg kan man betala för, bland annat om du vill ha X-pack, då kan man välja specifika index som andra grupper får läsa. Det behövs inte hos LDC då vi bara är tre personer, men vi kommer börja använda elastic stack för i princip alla loggar. Vi ska genomföra projekt för säker hantering av IT-drift, vi vill ha en projektplattform där vi ska jobba och detta kommer ske genom elastic stack, vilket gör att OSS kommer användas inom fler områden inom universitetet. Detta kom till för att vi såg vad vi behöver nu och vad som fanns tillgängligt och i framtiden kommer allting sakta men säkert hanteras av elastic stack. Vi har inte valt i ett led utan vi tog det som vi har sett fungerar bra och utökar det i enlighet med verksamheten och behovet.

AH: Kontrollerar ni så att OS-licenserna följs korrekt och i så fall hur?

MP: Nej, när det gäller licenssidan kan vi säga att vi kontroller att det är open source, sen är det för vår del inte aktuellt att göra en förändring och sälja vidare. Innan så var det svårt med licenser, det är mycket mindre sånt även om det finns. Jag betvivlar att någon tänker på det i dagsläget, till exempel att LDC är en myndighet och det kan räknas som en kommersiell användning av den licensen som finns. Jag betvivlar att någon som använder sig av OSS bryr sig, jag tror inte heller att vi gör våld på någon licens då de enbart används internt, vi vidareutvecklar det inte. Vi kan använda öppna bibliotek och kod men arbetar inte inom kommersiell natur.

AH: Hur vet ni att ni inte riskerar legala följder?

MP: Vi vet inte med säkerhet att alla licenser följs men beroende på typen av vår verksamhet är det inte så troligt att att vi bryter mot OSS-licenser. Det är mer troligt att vi inte följer licenser för proprietära program till punkt och pricka. Det sker nog oftast omedvetet på grund av att det kan vara lätt att missförstå hur våra licensmodeller fungerar. Vi har många Campus-licenser som tillåter användning för anställda och studenter i arbetet, studierna och i vissa fall hemma. Licensmodellerna skiljer sig mellan olika programvaror och därmed hur man får använda dem.

AH: Använder ni något IT Risk Management-ramverk som ISO 27005 eller ISACA:s IT Risk Framework (för att nämna några)?

MP: ISO 27005 använder vi, och vi försöker arbeta enligt det så gott det bara går.

AH: Skulle du säga att ni implementerar det steg för steg?

MP: Vi försöker följa det på LDC så gott det går. Utöver det är ITIL en bra plattform för att saker och ting ska göras strukturerat, vem gjorde det, hur och när är det klart etc. Bara det gör att vi fyller många krav i andra standarder som tex ISO, då är det lättare att införa ISO27000-tänket och de saker som ska göras. På frågan om vi är helt compliant? Nej, men det finns inget krav att certifiera oss, bara att följa den, och gällande den centrala biten följer vi den men kanske inte genom hela verksamheten. Vissa är väldigt duktiga men andra struntar i det helt och hållet, när jag började för 30 år sedan var allt centraliserat och skulle precis decentraliseras. Det har det gjort i 15 år men nu börjar det bli mer centraliserat igen, det är jobbigt att sköta, kostar massa pengar och de ekonomiska ramarna blir snävare och snävare. Även om universitetet har mycket pengar, kan vi inte ta det och stoppa in i daglig drift, dom ska användas till satsningar som kan göra saker bättre, det är ett litet problem för oss.

AH: Använder ni någon formel för att beräkna IT-risker? (Risk = Threat * Vulnerability * Assets)

MP: Nej vi gör inte det, jag tror att det är svårt att räkna på risker på det sättet, teoretiskt är det väl bra, och det börjar komma tillbaka det här hur räknar vi risk. IT-säkerhet är en försäkring som du måste betala för, det är en brandkår som väntar på att det ska börja brinna, dom måste vara utbildade när branden väl kommer. Så är det med all säkerhet, du betalar för att det kan börja brinna, inte när det väl brinner. Det är samma med all riskhantering, man kan inte börja med riskhantering när risken väl har skett. Om vi säger såhär, har jag ingenting att göra på mitt jobb så har jag gjort ett väldigt bra jobb då alla säkerhetsaspekter är väl omhändertagna. Om jag visar ett bra arbete kanske chefen vill ge mig mindre resurser då den tänker att det inte finns några säkerhetsrisker, och då går det i kras. Konceptet är svårt att sälja in, och det är samma med all riskhantering. Vad är risken med att välja program X istället för program Y till exempel. Men när man gör det valet är det är bara en bedömningsfråga. Man kan alltid titta bakåt och undra om man valde rätt? Nej, kanske inte, men jag hade inte rätt information, nu har jag informationen och får ta med det in i framtiden inför nya val. Man kommer inte gå genom livet utan att välja fel, men man måste göra sin hemläxa, kommer jag få ta del av så mycket information som möjligt innan jag väljer? Det gäller att prata med dom som har lång erfarenhet av detta sedan innan. Väljer jag OSS så utsätter jag mig för större risk då jag måste utföra en egen riskhantering, väljer jag proprietär mjukvara så gör någon riskhanteringen åt mig. Om dom säger till mig att det ska funka, ja då ska det funka, står det i broschyren så ska det funka på det sättet. Om jag köper en OS-mjukvara och den inte fungerar exakt som det står och jag rapporterar detta till dom, då tar dom bort den funktion från hemsidan dagen efter, och det finns inga krav på att dom inte får göra så. Jag har ju ingen rätt att klaga på gratis mjukvara.

AH: Hur väl informerade/medvetna är ledningen/”högre instanser” om OSS-risker?

MP: Ibland tas det inte på allvar, ibland är det svårt att få saker och ting igenom och att få folk att förstå vikten av varför man ska ta beslut som kostar pengar nu för att kanske få en vinst i framtiden. Jag kan aldrig garantera vinsten här och nu, man köper snarare en försäkring och då måste man kolla på vad som kommer löna sig mest. Det är alltid svårt att sälja in till ledningen, hade vi inte haft en lag som säger att man måste ha trafikförsäkring hade många skippat det. I vår organisation finns inget krav att man ska ha en viss säkerhet, förutom att vi ska försöka använda ISO27005. Större incidenter ska rapporteras in till (MSB?. När det kommer till säkerhet är det svårt att hitta en specifik nivå vi ska upprätthålla. Vi har inte bara en plattform och ett system, om du nämner ett operativsystem så kan jag lova att jag hittar det på vårt interna nät. För några veckor sen hittades IBM OS2 på en laptop som en konsult hade med sig till exempel. Vi hittar verkligen allt möjligt, vi kan inte hålla någon typ av full säkerhet och därför måste vår risk management koncentrera sig på våra kronjuveler. Vi måste kolla var är vi mest sårbara och lägga tid och ork där vi gör mest nytta, vi får inte låta DNS-systemen eller LADOK bli hackade. Vi har identifierat de riktigt dyra sakerna och vad som skulle göra mest skada på vår image om det blir hackat. www.lu.se och de centrala webbserverna är exempel på detta, om det skulle dyka upp någon text på startsidan för hela Lunds Universitet hade det inte varit bra för vår image. Det hade kommit i tidningarna direkt även om det bara var en oskyldig bild eller text i några minuter. Full ITRM kan vi inte göra men på våra centrala system och mest värdefulla tillgångar, ska vi försöka att göra det på. Problemet är att det inte bara är hårdvara och mjukvara utan även på människor som är säkerhetshot och det är den svåraste biten. Har vi tid och möjlighet att tvinga på 2FA på vissa inloggningar och människor vill vi skaffa detta. En rektor på KTH blev lurad i ett phishing-försök och de blev av med nära 500 000kr. Förövarna hade studerat detta noga genom att använda sig av oskyddade inloggningar och loggat allt som hände på hemsidan, på så sätt lyckades dom få skolan att betala ut pengar. Beroende på vilken tjänst personen besitter borde vissa konton kanske skyddas mer, till exempel förvaltningschef, rektor, chefer på olika ställen med mera.

AH: Du pratade tidigare om ITIL, skulle du kunna förklara lite mer kring det?

MP: ITIL är ju ett sätt att sköta verksamheten på, det är en standard som bygger på en ISO-standard. Målet är att få folk på rätt plats, på rätt tid, med rätt material grundat i en strategi som användes under Falklandskriget. LDC bestämde sig för längesen att följa detta och det är hur vi sköter vår verksamhet inom allt från utveckling och förvaltning till hur vi rapporterar in saker som ska förändras. Det styr även hur det kommer loggas och hur det sköts genom systemet.

8.3.2 Sony Mobile

Intervjuperson: Carl-Eric Mols (CEM)
Titel: Head of Open Source
Antal anställda: 7100
Kort beskrivning av företaget: Sony Mobile Communications är ett telekommunikationsföretag som tillverkar mobiltelefoner och tillbehör.

Intervjuledare: Axel Hellström (AH)
Notarie: Fredrik Moberg (FM)

AH: Vi kan ju berätta lite om oss och vår uppsats då, vi går sista terminen på Lunds Universitet, systemvetenskap pluggar vi, och vi skriver uppsats, kandidatuppsats. Den handlar ju om som vi skrev i mailet där, hur företag hanterar risker och problem vid användning och val av Open Source mjukvara. Och vi har ju en ganska positiv inställning till OS då, men vi vet också att det finns risker och problem med OS som till exempel säkerhetshål, buggar, licenser, support, dokumentation, såna saker och vissa är ju inte specifika för OS, det finns ju säkerhetshål i proprietär programvara också såklart, men det är dom grejerna vi kollar på där. Så det vi vill ta reda på är om företag tänker på dom här sakerna över huvud taget, om det finns några utarbetade riktlinjer eller processer för att hantera dom här OS-specifika riskerna då. Så det ena vi kollar på är open source då, det andra vi kollar på är IT risk management kallas det, som är som en-, hanteringen av IT-relaterade risker och problem kan man säga

AH: Vad är din uppfattning av begreppet OS-mjukvara?

CEM: Det finns en officiell uppfattning från Sony Mobile, som jag har varit med om att fästa vad det är. Och det är såhär att det finns en definition i en organisation som heter Open Source Initiative, och de är ju ganska välkända nu. De har ju gjort en definition på open source och den har vi då för övrigt i vårt kursmaterial, vi har ju en webkurs. Men det korta svaret är säger OSI att den [licensen] är en open source-licens och den går igenom definitionen på vad en open source-licens är, så godtar vi det som open source-licens här [på Sony Mobile]. OSI har listat 66-67 godkända licenser men de la av där för att de inte hade tid att gå igenom fler, det finns många fler licenser som inte är listade där men som ändå inte är open source. Men då får man ju be legal att kolla på kriterierna, håller de för de så kan de komma med en verdict att ja, vi anser att det här är open source, och då är det lugnt. Och det har ju skett på lite mystiska licenser genom åren. Och vi har ju testat det och sagt att ja, det här är open source, då godtar vi det. Det var det korta svaret, sen finns det ett lite längre svar. Open source är inte bara en licens och det handlar inte bara om fri och gratis kod utan nästa steg är som en lök. Första skalet är kodlicens, det är bara någonting som utvecklare och möjligtvis jurister är intresserade av, och sen går management och bestämmer massa saker som är dumma i huvudet och som inte fungerar alls. För att med open source så följer det också med en kultur, eller om man ska vara mer formell, en utvecklingsmetodik och den har visat sig extremt framgångsrik. Så det är en form av software engineering som man måste lära sig att hantera som företag för att få någonting ut av det [open source] och bli duktig inom det och lyckas med det. Så kodlicens är första skalet, sen kommer kultur och utvecklingsmetodik och sen finns det ytterligare en aspekt och det är där jag är idag och tittar närmre på. Det är helt enkelt hur gör man business av open source? Och då är det enkla svaret att det gör man inte alls. I sig är ju open source, du kan ju köpa open source. Vissa licenser tillåter det. Men oftast är det så att det är ingen idé att sälja någonting där du kan få källkoden gratis, finns ingen business i det. Sånt som är MIT och Apache-licensierat, där är det upp till dig att publicera koden och dina modifieringar eller inte. Och det är det vi har i våra Android handsets, vi publicerar bara den koden vi måste publicera, som GPL-licensierad kod till exempel. Annat struntar vi i för det behöver vi inte. Så då kan vi se att i någon mån så säljer vi mjukvaran men det gör vi ju inte heller utan vi har en vad vi kallar för en indirekt affärsmodell, och det är där jag håller på att jobba med nu. Vilka affärsmodeller har man runt open source? Det finns tre

affärsmodeller. Första har vi den klassiska extenderade modellen, sånt som Red Hat sysslar med. Det vill säga att de distribuerar open source och tar betalt för distributionen, inte för koden. Och de kan de motivera genom att de tar betalt för verktyg som inte är open source och för annat som är bra att ha.

AH: Support till exempel?

CEM: Support till exempel, inte minst. Det är den extenderade modellen. Sen har vi den indirekta modellen och det är precis vad vi och Apple och de andra gör. Vi tar inte betalt för mjukvaran. Den ingår i det som du säljer. Så egentligen kan man se det som att vi tar en premium för en hårdvara. Elektroniken för en iPhone, till exempel, är ju inte särskilt dyr, men ni får ändå betala skjortan och alldeles för mycket enligt mig haha. Men det är ju för att Apple och vi tar ju betalt för systemet där mjukvaran ingår, men då ingår det också gratis underhåll och uppdateringar och allt det där. Så du behöver ju aldrig betala för mjukvaran i sig. Och så finns det en tredje affärsmodell, den asymmetriska. Då finns det ett kungsexempel, det är vad Google gör. Google har ju släppt Android som open source, alltså källkoden är fri, vem som helst kan ta den och skriva in den i en telefon. De kan inte kalla det för Android för de har vissa saker och ting som ställer och styr det men i princip kan vem som helst göra vad de vill med det och Google tar inte betalt för det här. Men, att Android finns i mobiler är mycket medvetet gjort av Google när de insåg att mobilerna håller på att ta över från PC. Och att världen skulle börja titta på mobilskärmar istället för PC-skärmar. Det är där de har sin business, search and add-business som man kallar Googles affär är ju att du gör en sökning och dom matchar din sökning mot någonting och där har då andra servrar möjligheter att exponera dig för det som du söker på. Så söker du efter husvagnar så kommer du garanterat få massa annonser för husvagnstillverkare. Flera år efter du köpt den förbannade husvagnen så kommer de fortfarande visa nya husvagnar haha. Och det beror just på det här search and add-business. Så Google säljer annonser till annonsinnehavare och dom såg det att den här marknaden måste de vara med i, när det går över till mobil. Att driva detta var att ge ett operativsystem som är modernt, som främjar websurfing, och annat. Då kan vi flytta vår PC-business in i mobilvärlden som kommer. De ger någonting gratis men tjänar pengar på något annat. Än mer, som är föga känt, är att det finns något som heter revenue-sharing, och faktum är att vi [Sony Mobile] får intäkter på det Google tjänar på att sälja annonser. En liten skärva går till oss eller till teleoperatören, idag går det mesta till teleoperatören. Så om du har en Androidtelefon, du gör en sökning, så är det annonsören som betalar för att exponera en annons, eller det är ju Google som tar hand om det. Och då får ju annonsören betala.

AH: Vad är din uppfattning om/av begreppet IT Risk Management?

CEM: Ja, nu sitter jag ju omgiven av mina kollegor som tillhör software security haha, så att det blir ett långt svar. Jag själv jobbar inte med det men vi kan säga så här att det begreppet är ju väldigt mycket, väldigt stort. Det handlar ju på många olika nivåer. Vi kan ju ta det först som är relaterat till open source sen finns det ju annat då som inte tillhör open source så det kanske jag bara lämnar. Vilka risker finns det med open source? Det första är att management ser en risk men open source för att ens intellectual property [IP] kan läcka ut till höger och vänster. Sen har de insett att man kan inte vara utan open source och det kan man inte idag för att all modern mjukvara utvecklas idag som open source. Det är ungefär som att världen kollektivt har bestämt sig för att vi ska inte ha det som vi hade det på 90-talet med Microsoft.

Pratar du om cloud, pratar du om IoT, pratar du om allting nytt idag så är allting totalt dominerat av open source. Så det finns inget val att undkomma open source idag så även de mest hårdnackade människorna i management får ge med sig och släppa in open source men då brukar man oftast fastna i det som man kallar en fälla, mer om den sen. Så första utmaningen är att man vill få kontroll så att ens egen IP inte läcker ut. Sen finns det ju säkerhetsrisker i open source i sig och de är ju då, enligt vad jag predikar, mindre än de som finns i proprietär mjukvara. Det är ingen som vågar säga emot det här längre, det är ingen tvekan, open source är trots allt bättre. Till och med vår egen security-avdelning säger det nu för tiden haha. Det finns dock utmaningar ändå va, bara för att det är open source så är det inte säkert. Oftast är det ju så att det finns luckor och genvägar som inte är omhändertagna. Så man måste ju ändå granska open source när man tar in det. Det har vi också här på Sony Mobile. Vi har krav och processer för att ta in open source och granska koden. Där använder vi oss mycket av National Vulnerability Database som amerikanska homeland-department hanterar och NIST. När det gäller Android skickar Google ut security-bulletins som det kallas, att nu har vi upptäckt ett hål här eller ett problem där och vi tar de här åtgärderna och så vidare. Och vi har ett maskineri för att skicka ut updates till alla mobiler. Som sagt, jag sitter mitt i den avdelningen [software security] som jobbar med vad som kallas incident management där de får in säkerhetsrapporter som måste åtgärdas och så har de ett sätt att göra det på.

AH: Vet du om den processen görs enligt NIST som du nämnde tidigare? Är det software security som jobbar efter NIST-modellen?

CEM: Det vet jag inte riktigt men jag vet att det finns något annat sånt där standardiserat ramverk för hur man tar hand om software security som börjar på B-någonting **[**BSIMM**]**. Det är ett ganska stort industrilett samarbete kring security och det finns modeller hur man gör det och man kan klassificera säkerhetsbrott och liknande. Det har de följt sedan många år tillbaka. Lika mycket systematik som de har inom security har jag inom open source. Då har vi pratat om två olika risker. Det ena var att hantera att IP kan komma in eller ut och sedan har vi då säkerhetsproblem i själva koden. Det finns ju andra IT-relaterade risker som har med open source att göra. Då handlar det om att du inte hanterar din kod korrekt, om du gör en breach. Om du inte hanterar koden enligt licens, då blir det breach of agreement eller breach of license. Det första som kan hända är att du får någon som kan stämma dig för ett copyright breach eller patent breach. Du följer inte licensen eller du gör ett patentintrång och du blir stämd i domstol för detta. Den stora faran här är att till skillnad från contract law där du oftast får betala skadestånd i förhållande till skadan så är det när det gäller patent breach i förhållande till den upplevda skadan. Det finns ett väldigt perfekt exempel här i Sverige och det är Pirate Bay. För det första, dom gjorde ju inte copyright brott, de blev dömda för medhjälp till copyright brott. Då kan du bli straffad lika mycket som om du gjorde brottet själv. Om du sitter i bilen och väntar på dina bankrånarkompisar som ska komma ut med penningpåsarna är du bankrånare lika mycket som de som tog penningpåsarna. Medhjälp är lika straffvärdigt. Men som sagt, när det gäller copyright breach. Pirate Bay blev stämda, tror det landade i hovrätten på 130 miljoner. Av de 130 miljonerna förde då Antipiratbyrån följande resonemang: Dom sa att det finns 40 miljoner användare av Pirate Bay och via Pirate Bay får de tag i musiken. Dom uppskattade att ungefär 10 procent var intresserade av de 27 låtarna som de laddades ner via Pirate Bay som bevisade att via Pirate Bay kunde du ladda ner musik, 27 låtar. Hade det varit sånt här kontraktsbrott hade det varit 27 * 10 kr per låt, vilket är 270 spänn, här har ni pengarna, inga problem. Men det var inte det dom sa utan de sa att vi

uppskattar att 4 miljoner har laddat ner de här 27 låtarna, och så gånger 10, och det var så synd om oss. De förstör hela vår marknadsföringsplan och så vidare. Det sista tyckte ju hovrättan att jajaja, haha, sån skada gjorde det inte. Men det andra köpte dom och sen landade de på 130 miljoner. Det var upplevd skada. Och därför är det så här att breach of license är mycket dyrare än breach of contract. Nu finns det någonting som ska läggas till. Det ska bevisas att du gjorde det här med vett och vilja. Det berättar jag inte här inne haha, man ska hålla folk lite skraja. Men det är ju såhär att om du gör ett misstag så kan du inte bli fälld för det. Om du gör ett misstag och någon påtalar det hela och du försöker rätta till det så finns det inget case mot dig. Utan det ska vara med berott mod och vilja.

AH: Så som ett exempel. Om man skulle ta in en open source-licens, kanske läsa igenom den men missa någonting, inkorperera den och sen blir något fel, vad händer då?

CEM: Då är det bara upphovsrättsinnehavaren som kan stämma dig i domstol. Domstolen avgör om det är brott mot licensen. Det är ganska sällsynt att det kommer till domstol. Oftast är det så att någon klagar, du rättar till det och sen blev det ingenting. Men om du inte rättar direkt utan du slöar och slarvar kan de hota med domstol om inte du fixar det här. Men settlements är ganska vanligt i det här området och som sagt, svårigheten är att bevisa att det var med berott mod och vilja. Slarv är dock ingen ursäkt. Det räknas som berott mod och vilja. Vårt bästa försvar mot sådana här copyright breach of license är att se till att vi har processer på plats, och det har vi. Så vi sköter allt det där. Faktiskt så att jag ska skryta, best in class. Av ett hackercommunity som heter XDA som är de som hackar Android-mobiler, gigantiskt stort, av det hackercommunityt så var vi utnämnda till OEM-manufacturer of the year 2012 redan haha. Och ännu bättre är det så att ett team där som heter FreeXperia, ledaren där i det teamet, de var ett 20-tal som höll på att hacka, vi har ju börjat samarbeta med dom för många år sedan och de gjorde ett sådant bra jobb inom strömförbrukning. Så standardläget för en Xperia-mobil var att den håller i två dygn, till skillnad från era sunkiga mobiler så håller Xperia två dygn, hörde ni det haha. Då fick vi hjälp av det här hackerteamet. De hjälpte oss att hitta strömbovarna i våra mobiler och de gjorde sånt bra jobb att han som var ledare, han fick ett erbjudande av oss att komma hit och jobba för oss. Och han kom från Rumänien, flyttade hit och jobbar för oss sedan många år tillbaka. Så bilden av oss som schysta och vi har våra saker i ordning, vi vet vad vi gör och vi ser till att leva upp till licenserna. Det är det bästa försvaret för alla vet att du kan inte stämma någon och få de fällda om det inte är med berott mod och vilja. Visst gör vi misstag men vi ser till att rätta till dom och vi har till och med tidsgränser. Till exempel om vi skulle missa att släppa ut arkivet med alla copyleft-licensierade filer som Linux-kernel till exempel som finns i Android. Det är under GPL-licensen. Det måste släppas ut och det gör vi alltså regelbundet. Vi har alltså internt här, fem arbetsdagar från det att vi har lanserat en mobil på marknaden så måste den [GPL-licensierade koden] ligga där. Det är inte alltid vi lyckats med det men det är inte värre än det att till exempel en organisation som heter GPL-violations ringer till mig. Så pratar vi lite grann och så säger dom: Du, vi har inte sagt den där. Då säger jag: Oj, då ska vi fixa och kolla inom 48 timmar. Då är det okej, helt okej.

AH: Kan du ge några exempel på OS-mjukvaror ni använder?

CEM: Främst Android, men vi har Open Source i allting. Verktyg, till och med Corporate IT är glada i Open Source nu för tiden haha. Big data, cloud och IoT och till och med drönare. Sony Mobile i Japan äger ett litet drönarföretag och tittar man i koden för den drönaren så är det GPL v3 i alltihopa. Så det är Open Source överallt.

AH: Hur går det till när ni väljer OS-mjukvara?

CEM: Grunden för allting är någonting som heter Open Source-direktiven och man kan se det här som corporate law, de stadgades 2009 av executive management. Det är företagslagstiftningen. Jag är då chairman av det som heter Open Source Board, vi godkänner contributions, ifall någonting ska lämna bolaget till ett community. Men vi är också satta på att administrera Open Source-direktivet och i Open Source-direktivet definieras vad Open Source är och hur vi ska förhålla oss till Open Source. Där är vi väldigt tydliga och säger att beslutet tas i businessen, av någon som är affärsansvarig, och beslutet ska vila på ett teknisk utvärdering och en affärsmässig utvärdering. Men det finns ingen preferens mellan det ena och det andra [mellan Open Source, proprietärt och att bygga någonting själv]. När vi väljer att ta in Open Source görs en Open Source-evaluation som skickas till mig. Sen får legal godkänna licensen och inga problem. Och IP-avdelningen säger att vi ser inga IP-problem kring det hela, de gör en enkel IP-search. Idag är det så att 98% av det vi tar in går automatiskt in utan att det behöver godkännas eftersom det är Android och vi säger att allt som är Android är okej. Så det är bara sånt som inte är Android-relaterat som vi kör den här processen. Men många gånger är det folk som vi ta in "yet another SQL-parser" och jag är trött på dem, haha. "Får jag ta in den här" frågar dem. "Ja, dina kollegor tog in den här redan 2013, fem år sedan, ta in den, varsågod". Men det är inbyggt här, alla vet att man bör kolla med Calle först att allt går rätt till. Det är kanske det bästa försvaret mot risker, det är att få upp ett mindset, och det har vi fått upp här. Folk här är ganska skickliga i Open Source, de vet om licensriskerna och andra risker. De är medvetna om processerna och vet vart man ska gå ifall man undrar något.

AH: Gör ni någon form av riskidentifiering, riskbedömning, eller riskhantering av OS-mjukvaran?

CEM: Det vi gör med Android gör vi nu med allt annat också. För Sony Mobile är upp påväg på en resa. Om tre års tid kommer vi inte bara syssla med mobiler utan det finns mycket annat som vi har börjat jobba med, IoT inte minst.

AH: Men hur tar just riskidentifiering, riskbedömning, och riskhantering sig i uttryck när ni väljer OS-mjukvara?

CEM: Jo, det jag tänkte komma till är att de processer och strukturer vi har på plats när det kommer till Android, transplanterar vi till andra verksamheter här i Lund också. IoT-gänget jobbar under samma regler och riktlinjer. Det finns ju just nu någonting som är superhot och som alla springer omkring och är väldigt uppmärksamma på är GDPR. Vi började med det här jobbet för två år sedan. För oss är liksom, vi har allting på plats. Vi har processerna på plats,

vi har protokollen på plats. Det ingår som en tollgate i vår utvecklingsprocess. Och sen har vi då policys hur det ska se ut, vad vi gör, vad vi inte gör och så vidare. Vad vi får samla för information, vad vi inte får samla för information. Allt det där finns på plats redan. Faktum är att innan idag hade jag ett möte om hur vi ska få in andra Sony-bolag att börja jobba med det här. För det är lite grann att Sony Mobile här i Lund har börjat ta lead inom Sony när det gäller avancerade mjukvara. Vill inte säga att vi är bäst men vi tillhör definitivt toppskicket inom Sony. Sen finns det något som heter SCALARE-projektet som jag varit med och jobbat på [SCALARE är ett forskningsprojekt för att ta reda på hur organisationer kan förhålla sig till att mjukvara blir en kritisk del av verksamheten]. Där har vi kommit fram till att du till exempel ska införa en Open Source Officer, en sån som mig. Du ska ha samarbete med Legal, Software Engineering, Security, Management, och IPR [Intellectual Property Rights]. Det är min roll.

AH: Använder ni något verktyg för att identifiera, analysera och hantera OSS-risker? Det finns till exempel ett verktyg som heter RISCOSS.

CEM: Ja, men vi använder oss av ett annat verktyg som heter Black Duck Protex. Det är så inbyggt i verktygskedjan att det sker per automatik. När man laddar upp någonting till vår main branch så kommer det så småningom scannas av och sen scannas det av varje dygn flera gånger osv. Så det är helt automatiserat. Sen har vi sedan en tid tillbaka börjat kolla på den svenska lokalutmanaren, har ni hört talas om FOSSID?

AH: Nej, tror inte det. Men FOSS vet vi ju vad det står för.

CEM: Ja, precis, vi har tagit in det verktyget, vi har en licens. Protex använder vi för Android, och det har sina fördelar och nackdelar, det börjar bli ganska gammalt och Black Duck har själva aviserat att de skulle vilja stänga ner det och ersätta det med Black Duck Hub. Protex kör vi ju på många servrar eftersom Android är ganska stort. Den har då fördelen att vi kan gå ner på en djup analys, den kräver bara typ 10 - 20 rader kod så kan den identifiera vad det är för typ av open source, vilken licens och eventuella säkerhetsproblem. Men det har ju nackdelar eftersom man måste köra egna servrar och det är klyddigt att använda det gamla gränsnittet, det är inte optimerat. Så Black Duck själva vill ju att vi flyttar till Black Duck Hub. Den har dock en nackdel att den klarar inte av deep analysis. Det är därför vi tar in FOSSID.

AH: Har ni upplevt några problem med OS-mjukvaran? (buggar, support/dokumentation, säkerhetshål)

CEM: Oh ja, men som jag var inne på lite förut har vi processer för att hantera det här. Och jag ska också säga att huvuddelen av verksamheten är fortfarande Android. Google själva har ju ett gediget säkerhetsarbete då de skickar ut säkerhetsbulletiner och sedan har vi en hel avdelning som tar emot de här säkerhetsbulletinerna och jobbar efter incident management. Det där är en jätteorganisation som redan är på plats. Så när det gäller Android är allting på plats och det finns stort stöd för detta va. Men sen är det så att när det gäller Open Source

generellt, som när vi utvecklar tjänster i cloudet till exempel, då tar vi ju in helt annan Open Source. Då används ju också de här strukturerna jag pratade om, som cyber security. De är ju inte bara till för Android. Skillnaden är ju att vi inte har fördelen i att Google skickar ut säkerhetsbulletiner utan då är vi ju lite mer "on our own". Sen är det ju skillnad, det beror ju på hur mogna olika organisationer är. Tar du in Open Source från ett litet glatt gäng på fyra personer som gör någonting som är väldigt litet, nördigt men också väldigt bra för vår del och just för att lösa den grejen. Och visst, då är det lite tunnare vad det gäller säkerhet. Man slår på National Vulnerability Database men det är ju så litet så det passerar. De [National Vulnerability Database] har ju också hål va, de tittar ju mest på de stora Open Source-projekten och kanske missar de mindre projekten. Då blir det ju så att säga egenscanning, och ha disciplin på det. Vi har en form av risk assessment. Då landar det mer på kvalitet på de olika projekten. Hur pass kompetenta de är. Men eftersom vi är ganska avancerade här har de flesta teamen okej nivå på riskhanteringen, och de har ju stöd. Det finns ju interna resurser att ta stöd ifrån. Man kan ju ta parallellen med GDPR, där har vi allting på plats, alla policies, processer och så. Och Android är inga problem, det är i maskineriet. Men när det kommer till sånt som är nytt, till exempel IoT-satsningar. Då har vi ju kvalitetsgenomgångar och olika tollgates där vi kontrollerar att GPDR följs till exempel. Men det blir ju mer manuellt till skillnad från det stora "maskineriet" där det tas hand av någon avdelning.

AH: Du pratade om Black Duck-verktyget förut. Ser det till att Open Source-licenserna följs?

CEM: Ja, det ingår i Black Duck-verktyget. Så här går det till, och det är samma sak om FOSSID och de andra verktygen. Den gör något som heter code prints, som ett fingeravtryck på koden. Vad den gör är att den samlar på sig mjukvara från olika communities, GitHub, SourceForge, sen kör den någon form av hash-encoding och får ut en code-print, ett fingeravtryck. Detta fingeravtryck använder du på din egen kod och sen kör du matchning. Om fingeravtrycket för din egen kod matchar fingeravtrycket på det de har samlat in då vet du vad det är för någonting. Nästan allting. Det finns ju mismatch. Den säger att den matchar men lägger man upp koden bredvid varandra så av någon konstig anledning säger algoritmen [fingeravtrycket] att det är samma grej men det ser inte ens likt ut, det är inte samma kod. Men det hjälper dig kanske 94 - 96% av alla matchningar. Men i och med att den identifierar vad det är för Open Source har du också metadata, d.v.s. vad är det för licens? Och Black Duck har ju specialiserat sig på att ha förteckning på alla möjliga Open Source-licenser. Nu är det svårt att säga hur många de finns men Black Duck brukar påstå att det finns 1400+ licenser. Sen är många av dem varianter av varandra, till exempel Linux. Tittar du på Linux kernel, som alla vet kör GPL v2, men det är inte så många som vet att det finns sex olika versioner av Linux kernel som inte har exakt samma GPL v2 licens eftersom någon skrivit om något ord eller mening som de tyckte skulle ändras. Det finns sex olika GPL v2-varianter. Men Black Duck fångar upp dem hyfsat bra och då har du ju också en regelmaskin som du själv kan förändra i verktyget. Dels har du några självklara inbyggda, som till exempel att du inte kan kombinera GPL v2 med Apache licensen till exempel. Då kommer systemet visa röd flagga och att du måste åtgärda. Då kan det ju vara så här att det bara är testkod som inte kommer komma med i distributionen. Men då får du ändå skriva om koden eller be communitiet snällt om de kan tänka sig att licensiera det i en GPL v2-kompatibel licens som till exempel BSD. Det är en sån workaround som jag har haft roligt med bland utvecklarna. De säger "Kan vi göra det?" "Javisst" säger jag. De [communitiet] är ju upphovsrättsinnehavare. De får ju sätta vilken licens de vill. Ta kontakt med han John McKelly, skicka honom ett email och fråga om det går bra med BSD och kolla reaktionen.

Och alla gånger har det varit "OJ, Sony Mobile vill använda min kod, yes, vilken licens vill ni ha? BSD?". Då blir de lite besvikna, "Finns det ingen coolare licens? Men visst, det är inga problem!". Så vi ser till här på Sony att leva upp till licenskraven men då också köra licenser där vi kan välja att hålla delar av vår kod för oss själva medan vi kan Open Sourca andra delar. För har du någonting som är fräckt som ingen annan har och Open Sourcar det så blir du ju av med det. Så därför är till exempel Apache-licensen mer business-vänlig. Företagen kan hålla det som är differentierade för sig själva.

AH: Precis, så man kan ju skydda sin Intellectual Property även med Open Source.

CEM: Ja, och det var anledningen varför Legal [-avdelningen på Sony] köpte Open Source ganska tidigt. Först konstaterade dom att Open Source var legalt, upphovsrätt gäller. Sen förstod dom att Open Source var en nödvändighet affärsmässigt, att vi måste gå in på Open Source för företagets långsiktiga överlevnad. Med de två sakerna ställer de sig bakom Open Source. Sen har de ju sett att Open Source är till och med gynnsamt för ens business för det hjälper en att skärpa affärssynen. Vad är sådant som är differentierande, det kan vi hålla stängt och det är helt okej. Och sånt som är commodity kan vi dela med andra på ett väldigt bra sätt och sänka kostnader för att underhålla o.s.v. Så Legal är smarta, de kan ingenting om kod, men de är smarta. Vill ni veta vilken avdelning som var sist ut [att anamma Open Source]?

AH: Jättegärna

CEM: Corporate IT. De var sena på det. Inte idag, idag är de glada Open Sourcare. 2008 när vi började leka med Open Source då fick de en shock när plötsligt en 300-400 Ubuntu-datorer kopplade in sig på nätet. Då fick de ett nervöst sammanbrott och tyckte att "Sådär får ni inte göra, det är förbjudet att köra Linux här". Men vi skulle ju ta fram Android-produkter så de fick ju serva oss, det var inte så att vi skulle leva upp vad de ville. Så de fick ju ändra sitt arbetssätt och tillåta Linux-burkar. Corporate IT var ju ganska Microsoft-orienterat då.

AH: Använder ni någon formel för att beräkna IT-risker? (R = T * V * A)

CEM: Jag har hört om den och jag tror till och med att vi använde en faktor till, alltså angreppspunkter. Och detta kan Software Security [-avdelningen] allting om. Jag hänvisar till dem, det är inte mitt bort hehe.

AH: Hur väl informerade/medvetna är ledningen/”högre instanser” om riskerna med OS?

CEM: Här är det ju inga större problem idag. Vi lever symbiotiskt ihop med Google så vi studerar ju vad de gör och Google är väldigt avancerade i det. Det är risker på olika nivåer. Det börjar med en legal risk, licensrisk, och IP-risk. Det är det som management först reagerar på. Men när jag började här, 2007, så var det kontroversiellt med Open Source. Visserligen fanns det 200 000 projekt men jag har den nackdelen att jag lever kvar i att jag måste övertyga folk om att Open Source är viktigt men världen har sagt "Ja, och? Det vet vi ju" haha. Men jag

har fortfarande kvar mindsetet att jag måste frälsa ledningen. Här hos oss är det ju totalt genomslag. 2012 - 2013 gick det en kulturell chock igenom bolaget då man bara "Nu tar vi emot Open Source helhjärtat".

8.3.3 CGI

Intervjuperson: Philip Vendil (PV)
Titel: Certificate Service Development Team Lead
Antal anställda: 72 000 globalt
Kort beskrivning av företaget: CGI är ett globalt IT-service konsultbolag som styrs från Kanada.

Intervjuledare: Axel Hellström (AH)
Notarie: Fredrik Moberg (FM)

AH: Vi kan ju berätta lite om oss och vår uppsats då, vi går sista terminen på Lunds Universitet, systemvetenskap pluggar vi, och vi skriver uppsats, kandidatuppsats. Den handlar ju om som vi skrev i mailet där, hur företag hanterar risker och problem vid användning och val av Open Source mjukvara. Och vi har ju en ganska positiv inställning till OS då, men vi vet också att det finns risker och problem med OS som till exempel säkerhetshål, buggar, licenser, support, dokumentation, såna saker och vissa är ju inte specifika för OS, det finns ju säkerhetshål i proprietär programvara också såklart, men det är dom grejerna vi kollar på där. Så det vi vill ta reda på är om företag tänker på dom här sakerna över huvud taget, om det finns några utarbetade riktlinjer eller processer för att hantera dom här OS-specifika riskerna då. Så det ena vi kollar på är open source då, det andra vi kollar på är IT risk management kallas det, som är som en-, hanteringen av IT-relaterade risker och problem kan man säga

PV: Mm

AH: Så det är det vi håller på med, men vi kan väl börja med om du kan berätta lite om CGI och din roll, dina arbetsuppgifter.

PV: Ja, om CGI, är väl ett jättestort konsultbolag, kanadensiskt bolag som finns i stora delar av Europa och Nordamerika i varje fall. Och jag jobbar ju som huvudutvecklare på Security-delen och vi tar fram en certifikattjänst heter det, vi tar fram säkerhetslösningar, det finns digitala signaturer och PKIer och så, åt kunder. Jag själv har ju väldigt lång bakgrund inom OS, jag har utvecklar OS-mjukvara hela mitt yrkesliv egentligen, så jag har väldigt mycket erfarenhet av det. Just vår avdelning, vi använder nästan uteslutande OS i dom tjänster vi tar fram, för kunder.

AH: Okej, yes, vi skulle kunna börja med att du kort bara berättar din uppfattning om begreppet OS, vad det innebär för dig, så vi ligger ungefär på samma nivå.

PV: Open source för mig. Det är väl, ja, källkoden är tillgänglig, man kan ladda ned, man kan bidra, utbyta idéer, ja, som utvecklare är det ofta att man kan fixa problem själv som man upptäcker som man oftast, om det är en kommersiell produkt så är man ju ganska låst då när det säger stopp, medan vi försöker ju då hitta lösningar och använda OS för att göra liksom helhetslösningar, så att, ja-

AH: Jamen det är ungefär samma som vi har definierat det, sen då nästa område som vi undersöker är det här då som jag sa, IT risk management, har du hört om det, vad tänker du om det begreppet, vad innebär det för dig?

PV: Risk management är ju som jag tolkar, alltså vi gör ju riskanalyser i våra tjänster när vi tar fram problem och hot sådär, så kategoriserar vi det i olika kategorier och sen så gör vi då en åtgärdsplan hur vi fixar det, så att det är vad det står för för mig, det är så vi jobbar.

AH: Ja men okej, det låter bra, vi kommer nog, jag ska säga det också att jag kör frågorna en och en och ifall det kommer en fråga som du känner att du redan har svarat på kan du referera till det eller att du har redan sagt det eller att det var det jag pratade om innan liksom, så att du är med på det.

PV: Ja!

AH: Då kan vi gå in på om du skulle kunna ge några exempel på Open Source mjukvara som ni använder i era tjänster eller lösningar idag?

PV: Ja, alltså det är ofta dom stora, vi har ju mariaDB, det är ju, vi jobbar ju i Java-världen så det finns väldigt mycket open source där. Så det är ju det, spring framework heter det också, jag försöker komma på allt men vi använder nästan uteslutande OS, det är väldigt lite proprietära lösningar som vi arbetar med över huvud taget. Utan alla bibliotek, all utveckling vi gör då, bygger på OS.

AH: Okej, hur går det till när ni väljer OS mjukvara, vi förstår att det är efter behov, att ni har specifika problem som ni måste lösa, men är det fritt fram för utvecklare att välja vad dom behöver eller har ni, du var inne på det lite, ni har specifika riktlinjer då för hur det går till kanske?

PV: Ja vi har ju liksom på företaget en policy om OS som är framtaget, där är det ju specificerat vilka licenser vi får använda så att det inte blir några juridiska problem för CGI eller CGIs kunder då. Till exempel är licensformen GPL förbjuden, det får vi inte ha över huvud taget, för det skulle innebära att kanske kunders egenutvecklade mjukvara måste släppas som OS och det skulle ju inte vara populärt då kanske. Dom har ju definierat vilka licenser som vi får använda,

AH: Och det är dom lite mer öppna, tillåtande licenserna då kanske?

PV: Ja, det finns ju Apache, MIT, LGPL, får man ju använda. Och den policyn är mest för juridiskt, ofta är det ju juridiska hänseenden som bestämmer det där, just säkerhetsmässiga hänseenden får man sköta själv då. Där har det ju blivit mycket bättre på senare tid tycker jag, med att få säkerhetsvarningar i bibliotek. Tidigare visste man inte alls, då var det väldigt svårt att hålla reda på för att, det är ju liksom beroenden som av OS-bibliotek så det kan vara hundratals bibliotek så det är svårt att veta om den där längst ner på listan har fått en sårbarhet, men där har det ju börjat komma lite sårbarhetsanalyser vet jag. På GitHub vet jag till exempel att man automatiskt kan slå på att den varnar om man har en beroende som har nån rapporterad sårbarhet. Det går också i själva byggmiljön att ha en plugin som kollar upp sårbarheter i dom beroenden man har, det försöker vi ju införa i vårt byggsystem för att fånga dom aspekterna.

AH: För det är mycket Open Source projekt som är beroende av andra projekt, det är det du menar med dependencies?

PV: Ja, juste, det är små hjälpbibliotek som i sin tur är beroende av ett bibliotek. Ifall man bara drar in ett bibliotek, då kan man få femtio på köpet och då vet man inte vem som har hand om dom och så.

AH: Men då har ni, då sa du att det finns GitHub som kan kontrollera det?

PV: Ja, det är ett verktyg och sen finns det andra just i byggmiljön så finns det verktyg som slår upp mot den sårbarhetsdatabasen då-

AH: Juste, Open Source Vulnerability Database?

PV: Ja, CVE heter den väl?

AH: Den har vi kollat lite på ja. Vet du, har ni några verktyg för det som ni kör mot koden, eller i byggmiljön?

PV: Ja vi håller på att införa det nu för att kunna ha bättre översikt över vad det är som händer och då är det när man bygger så laddar den ner den här databasen och kollar alla beroenden i alla projekt.

AH: Okej, kanon! Du var inne lite förut på att ni gör riskanalys och det har vi också kollat på i det här området IT Risk Management, där har dom då listat olika steg som man ska göra i den här processen, och det är riskidentifiering, riskbedömning och riskhantering. Om vi börjar med riskidentifiering, hur gör ni för att identifiera risker med er OSS användning?

PV: Ja, vi har ju inget specifikt att mäta just för OSS men det ingår i hela tjänsten vi levererar så vi har haft, och det ska man ha regelbundet varje år, så har vi möte där man sitter i grupp och försöker komma på risker på såna här små post-its. Sen går man igenom dom tillsammans, en och en, och bedömer om det är hög sannolikhet och hur stor impact, påverkan, det skulle ha på en skala mellan ett till fyra och sen multiplicerar man dom. Sexton är då när det har högst prioritet och ett har ju minst prioritet på åtgärdslistan sen. Och sen har vi regelbundna möten där vi går igenom alla punkter som man skulle ha åtgärdat, där man ser status, så att det jobbas ju med regelbundet.

AH: Okej, då har du nog gått igenom lite riskbedömning och riskhantering där att ni följer upp detta och försöker lösa det. Vi har kollat på ett ramverk som heter RISCOSS, är det något som du har hört om? Det är både ett teoretiskt ramverk och verktyg för att hantera risker gällande Open Source, är det någonting du har-

PV: Nej, det har jag aldrig hört talas om faktiskt.

AH: Okej, har ni upplevt några tydliga problem med OSS-mjukvaran som ni har behövt hantera, typ buggar, säkerhetshål, support, dokumentation, något sånt?

PV: Ja alltså det största problemet som jag har upplevt med OSS-användning det är att man väljer projekt som inte är tillräckligt aktiva, det kan vara en person som har skrivit någonting och han har tröttnat på det och så är man beroende av det och projektet är dött. Det är en stor risk, det ser jag som det största problemet vi har haft, då får man ta över och underhålla det projektet själv då. Och det är inte så roligt, så att, det är det största problemet. Annars tycker jag OSS är, ja, vissa projekt kan vara riktigt dåligt dokumenterade också, men då har man oftast tillgång till källkoden så att man kan ha nån hum, för det är inte säkert att en kommersiell produkt är bättre dokumenterad, erfarenhetsmässigt hehe. Och då kan det ju vara väldigt krångligt att få hjälp om man ska göra nån support issue, innan man får prata med någon som faktiskt kan något på det företaget så kan det var väldigt knackigt.

AH: Oftast med lite mer aktiva communities så kanske man få hjälp på forum och liknande?

PV: Ja, riktigt aktiva projekt har ju nästan något så här slack-rum eller IRC-chatt man kan koppla upp sig på så får man oftast väldigt mycket bättre support än en kommersiell supportkanal, faktiskt. Så att just supportdelen har jag inte upplevt att det har varit så jättestora problem med.

AH: Nej, okej, du var inne på det förut, men hur kontrollerar ni att OSS licenserna följs korrekt, för det finns ju en del?

PV: Ja det är ju en utmaning, men det finns ju verktyg som scannar av det där, vi kollar på ett men det var väldigt dyrt så vi har ju gjort det lite manuellt där vi har som rutin att man kollar upp vilken licens det är innan vi kan använda det. Men det kan ju ha beroenden och sånt där, så det är ju lätt att det kan slinka in något, typ en felaktig licens. Det är en utmaning att ha koll på det.

AH: Ja, vet du om ni använder något slags IT Risk Management ramverk? Det finns t.ex. ISO 27005 eller ISACA IT Risk Framework, är det något du har hört om?

PV: Nej, alltså jag vet ju att CGI globalt jobbar mot ISO men det är inget jag har varit i kontakt med under utveckling eller sådär.

AH: Nej, vi är medvetna om det också att det är på en högre nivå, mer för management. Men okej, i det här området IT risk Management finns det en formel för att beräkna IT Risker, som är Threat * Vulnerability * Assets = Risk, är det någonting du har stött på eller använt?

PV: Nej men det liknar ju lite vår riskanalys där vi har hot * påverkan, men vi har inte med priset där, det har vi inte. Det är ju dom första två där som vi bedömer utifrån.

AH: Ja, skulle du säga att, hur väl informerade är er ledning eller högre instanser, är dom medvetna om riskerna med OSS? Dels riskerna och, är dom med på användningen som OSS, ser dom det som någonting bra eftersom ni använder det mycket?

PV: Ja, jo men det skulle jag vilja säga att dom vet om. För det var ju en policy som dom tog fram för ganska många år sedan att dom gick igenom vilka licenser man fick använda inom organisationen och sådär.

AH: Det kom ovanifrån då eller vad man ska säga?

PV: Ja, men sedan tror jag att i alla stora organisationer används väldigt mycket OSS över huvud taget, både IBM och Microsoft och dom släpper ju också väldigt mycket OSS nu för tiden. Så jag tror att det är ganska väl förankrat.

AH: Precis, ja, det var dom frågorna vi hade faktiskt, så om du känner att du vill lägga till någonting så får du jättegärna göra det, all information är bra information för oss liksom. Men annars-

PV: Nej jag vet inte om jag har något mer inom området

AH: Då känner vi oss nöjda där, vi får tacka så mycket för att du ställde upp.

8.3.4 Prevas

Intervjuperson: Håkan Erlandsson (HE)
Titel: Business Unit Manager
Antal anställda: 548
Kort beskrivning av företaget: Tekniskt IT-företag som erbjuder lösningar, konsulttjänster och produkter till kunder som utvecklar produkter med ett stort IT-innehåll.

Intervjuledare: Axel Hellström (AH)
Notarie: Fredrik Moberg (FM)

AH: Vi kan ju berätta lite om oss och vår uppsats då, vi går sista terminen på Lunds Universitet, systemvetenskap pluggar vi, och vi skriver uppsats, kandidatuppsats. Den handlar ju om som vi skrev i mailet där, hur företag hanterar risker och problem vid användning och val av Open Source mjukvara. Och vi har ju en ganska positiv inställning till OS då, men vi vet också att det finns risker och problem med OS som till exempel säkerhetshål, buggar, licenser, support, dokumentation, såna saker och vissa är ju inte specifika för OS, det finns ju säkerhetshål i proprietär programvara också såklart, men det är dom grejerna vi kollar på där. Så det vi vill ta reda på är om företag tänker på dom här sakerna över huvud taget, om det finns några utarbetade riktlinjer eller processer för att hantera dom här OS-specifika riskerna då. Så det ena vi kollar på är open source då, det andra vi kollar på är IT risk management kallas det, som är som en-, hanteringen av IT-relaterade risker och problem kan man säga

AH: Kan du kort berätta om Prevas, vad ni gör samt din roll och lite så?

HE: Ja, lite snabbt, Prevas är ett teknikkonsultföretag och vi jobbar framförallt mot företags utvecklingsavdelningar men också mot deras produktionssajter. Vi jobbar med i princip alla ingenjörsdisciplinerna från mekanik, elektronik, embedded programmering, lite högre nivå av programmering, kvalitetssäkring lite kort. Och så automation inom programmering inom IT-sidan då. Jag är ansvarig för den biten som heter PDS det vill säga vi som jobbar mot utvecklingsavdelningarna, så har jag en kollega som ansvarar för dom som jobbar mot produktionen. Vilket innebär att jag har mekanik, elektronik, embedded och applikationsmjukvara i princip, plus QA-biten, under mina vingar så att säga. Och där erbjuder våra tjänster till våra kunder, både att vi kan ta helhetsprojekt-åtaganden och utveckla en produkt åt folk, men också "vanlig konsulting" då, att vi tillsätter resurser till kundens projekt helt enkelt. Vi jobbar hos dom, sitter hos kunder. Om vi går in på biten med OS som jag misstänker är det som är mest intressant för er-

AH: Ja precis vi skulle kunna börja med att vi tar din uppfattning av begreppet av OS så att vi ligger på ungefär samma plan?

HE: Mm, jag har själv en bakgrund som civilingenjör, jobbat ungefär 10 år som utvecklare, projektledare, systemingenjör sen, ja 15 år till och med kanske, och sen lite mer åt konsultchefs-hållet. Så dom senaste åren har jag inte programmerat själv eller varit utvecklare eller konstruktör så att säga. Men jag har träffat en hel del kunder och varit inblandad i en hel

del projekt, framförallt på medicinstekniksidan. Där var det som så att OS var kanske inte så populärt på medicintekniksidan, för där finns en hel del regelverk som innebär att man kan få det ganska tungt att validera dom här stora mjukvarupaketen. Å andra sidan har man mognat lite grann i den insikten och framförallt efter att många i USA har insett att den här OS-mjukvaran är faktiskt inte så dålig kvalité eftersom det är väldigt många som har tittat på den koden. Och man har då kunnat som medtech-företag faktiskt kunnat ta in koden och sen validera dom bitarna som man använder och göra en riskbedömning på vad kan gå fel här, istället för att kanske behöva gå igenom hela koden rad för rad, vilket i praktiken är omöjligt. Och när man har gjort det så, hehe, så har vi ju en ny version som har ändrat sig. Men mina erfarenheter där gällande OS-kod är ju två bitar, för ett tag sedan började dom här bitarna komma in rätt mycket och då uppenbarade sig dom här licensavtalen som kommer med OS, det var rätt många som inte insåg vad det innebar. Där har jag varit med om rätt många projekt där vi går igenom företagens OS, vilka licensavtal har dom faktiskt tagit in i sitt paket som dom i sin tur säljer. Där är det en del företag som har råkat väldigt illa ut, där finns möjligheter att titta på det, bland annat finns det verktyg för det, till exempel ett företag som heter blackduck, som jag misstänker att ni känner till?

AH: Ja, det har vi hört talas om.

HE: De har ett verktyg som de tillhandahåller, och det har jag varit med om att vi har använt, hos en del kunder. Så där är ju en bit med open source kod som man kan titta på. Det som börjar bli den andra biten nu som ni kanske också är intresserade av här är ju just säkerhetsaspekten som har blivit väldigt uppenbar och kommer bli ännu mer uppenbar när vi pratar om Internet of Things och alla grejer blir uppkopplade vilket också innebär att det finns stora möjligheter att hacka saker och ting. Om folk inte är uppmärksamma på dom bitarna. Och där ser jag att det finns ett rätt stort gap mellan medvetenheten och vad som faktiskt behövs, både på nya saker som nya produkter som utvecklas men kanske framförallt skulle jag vilja säga på, hos gamla företag som förnyar sina produkter och affärstjänster genom att koppla upp sina gamla" mekanikprodukter. Gamla produkter som får elektronik, mjukvara i sig och nu även blir uppkopplade, där man lite grann har missat det här, man har ingen företagskultur i att göra riskbedömningar på vad som händer när du kopplar upp din maskin mot internet. För dom är ju jättevana, dom kan hela sin bransch och har jobbat hur många år som helst med den här typen av produkter men dom öppnar upp ett helt nytt fönster som jag tror många faktiskt inte har någon aning om.

AH: Nej, det börjar bli många klassiska företag, börjar ju nu bli mjukvaruföretag också.

HE: Precis och dom är rätt så fokuserade på hur ska vi nu hänga med i det här, hur ska vi kunna tjäna pengar på det och kanske misstänker jag att man, den här riskhanteringen är inte det som ligger högst upp i prioriteringsordningen. Och där tror jag, där har vi nog inte sett vad som kommer hända ännu, utan hela problemet är på väg att byggas upp nu.

AH: Precis, och det är det andra vi undersöker då, det här med Risk Management, IT Risk Management har vi kollat på vilket är ett begrepp över hur man hanterar risker. Har du hört det, har du kommit i kontakt med det?

HE: Vi tittar ju ganska mycket på det som man kallar cyber security eller vad man nu ska säga. Det är princip det jag pratade om, det är alldeles för få som kan alldeles för lite om det, eller snarare alldeles för många som kan alldeles för lite om det skulle man kunna säga. Jag menar, jag själv har varit med i techbranschen jättelänge och är van att göra riskbedömningar men samtidigt inser man ju att, när vi gör riskbedömningar på medtech-sidan till exempel, så är man väldigt duktig på att göra det men då har man helt glömt den här saken med att man kanske nu ska koppla upp sin medical device på nätet. Det har man aldrig gjort innan så att den här biten med att folk kommer att försöka hacka ditt medical device, där har man ju ingen erfarenhet av att göra riskbedömningar där. Jag tänker att vi kommer se en del otrevliga saker dyka upp där.

AH: Ja det händer ju grejer hela tiden, om vi går tillbaka till OS lite, kan du ge några exempel på OSS ni använder? Ni är ju ute mot kunder mest kanske då, men har du några exempel?

HE: Man kan väl säga Linux-distribution är ju rätt så populärt. Sen håller vi på rätt mycket med embedded, men det är kanske inte så mycket OS utan lite mindre utvecklingsmiljöer som är väldigt specialiserade mot en viss process. Det är väl frågan om man kan kalla det OS eller inte, men Linux-distribution är en stor bit givetvis.

AH: Okej, hur går det till när ni väljer OS, om ni jobbar mot en kund och väljer OS, är det upp till- är det fritt fram för dom konsulterna som jobbar med det att välja eller har ni några specifika riktlinjer för det där?

HE: Vi har inga specifika riktlinjer utan det är helt beroende på kunden. Ibland är det så att vi kommer in på projekt och så har kunden redan valt att man ska använda ditten och datten, operativsystem eller OS eller vad det nu kan vara. Då är det bara att göra det, ibland kommer vi in tidigare i projekt där vi faktiskt får göra en förstudie och ta reda på lite grann vilken typ av verktyg ska vi använda, hur ska systemet se ut och ska vi använda OS eller inte. Det har ju mycket av dom här bitarna, så det beror mycket på när var och hur i projektet.

AH: Juste, gör ni någon form av riskanalys eller riskidentifiering, riskhantering av dom här OS-mjukvarorna?

HE: Ja! Det försöker vi göra i den mån vi får lov av kunden, eller får tid eller där vi kan övertyga kunden om att ja men det är nog rätt vettigt att göra det. Eller så har kunden själva gjort det tidigare och tycker att det räcker så. Men vi försöker ta reda på för kunden, är ni medvetna, har ni gjort någon riskanalys, är ni medvetna om vad som händer, har ni vägt in dom parametrarna i valet?

AH: Har ni någon metod, något ramverk eller verktyg som du pratade om tidigare, Black Duck då, som ni använder för att analysera OS-mjukvara?

HE: Jag ska säga att jag vet faktiskt inte, jag är lite för ny inom Prevas, jag har bara varit här i lite drygt ett halvår så att, det kan jag inte säga.

AH: Nej, vet du om några konkreta exempel på när ni har upplevt problem med OS, till exempel som vi pratade om licenser, dålig support eller dokumentation, något sånt?

HE: Nej, det ska jag inte säga att jag på rak arm kan, tvärtom är det väl lite så att OS har, väljer man en hyfsat vanlig är det rätt stora communities där man kan gå in och ofta får man väldigt bra svar rätt snabbt. Så supporten där är nog rätt så bra faktiskt.

AH: Det brukar ju vara en säkerhet i sig där att om ett OS-projekt är stort och mycket använt så är det oftast bra.

HE: Precis.

AH: Det här med OS-licenser, du berättade om någon gång ni kontrollerade att OS-licenserna var korrekt. Skulle du kunna utveckla lite mer kring hur ni gjorde om du kommer ihåg?

HE: Det var faktiskt några år sedan nu men vi, vad vi gjorde var egentligen en gap-analys på vilka OS-paket hade man och hur hade man behandlat deras licenser.

AH: En gap-analys?

HE: Alltså, var är man och var borde man vara, hur mycket är det kvar att göra för att faktiskt uppfylla dom licenserna som man mer eller mindre medvetet hade plockat in i sin produkt. Det var nog så att det blev en liten sån aha-upplevelse för kunden att, okej var det så mycket att göra det hade vi inte riktigt räknat med.

AH: Nej, juste det kan ju lätt bli en licensdjungel där. Vi har ju kollat lite på dom här IT Risk Management-ramverk eller metoder, det finns ju något som heter ISO 27005 som är en sån industristandard, ISACAs IT Risk Framework, är det något du har hört om eller är det något ni jobbar med?

HE: Nej det ska jag erkänna att jag känner inte igen just dom siffrorna av ISO standarden.

AH: Okej, sen i dom här ramverken finns det några, en formel för att beräkna IT-risker, den är väldigt generell och den är Threat * Vulnerability * Asset = Risk.

HE: Precis, den funkar ungefär som dom flesta riskanalyser gör

AH: Nej exakt den är ju inte speciell för just IT, men den känner du igen?

HE: Japp

AH: Har ni använt den nån gång?

HE: Jag kan inte riktigt säga om Prevas har använt den, jag tror det, men jag har inget direkt jag kan säga.

AH: Okej, har du märkt av någonting, en del av IT Risk Management är ju också det här att man, det är viktigt att ha ledningens stöd, att ledningen är medveten om olika IT-risker som finns. Är det något du har märkt, att ledningen inte informeras tillräckligt om risker som finns med IT, eller möjligheter som finns med IT? Har du märkt av friktionen där eller att det inte riktigt förmedlas?

HE: Ja, det är nog ganska vanligt faktiskt. Och det är lite beroende på vilken typ av kund, bransch men också vad ledningen har för bakgrund, i företaget så att säga. Har dom teknik- och IT-medvetande eller inte, det gör en jättestor skillnad ofta. Vad ledningen själva har för preferenser och erfarenheter. Där har jag sett allt från extremt kompetenta ledningsgrupper i företag till, om inte totalt ignoranta eller okunniga så på gränsen till vad man säga.

AH: Okej, då var det nog bra för vår del om du inte har något mer att tillägga?

HE: Nej!

8.3.5 Organisation X

Intervjuperson: Person X
Titel: Head of Vendor Management and Governance
Företag: Organisation X
Antal anställda: 330
Kort beskrivning av företaget: Bank

Intervjuledare: Axel Hellström (AH)
Notarie: Fredrik Moberg (FM)

AH: Vi kan ju berätta lite om oss och vår uppsats då, vi går sista terminen på Lunds Universitet, systemvetenskap pluggar vi, och vi skriver uppsats, kandidatuppsats. Den handlar ju om som vi skrev i mailet där, hur företag hanterar risker och problem vid användning och val av Open Source mjukvara. Och vi har ju en ganska positiv inställning till OS då, men vi

vet också att det finns risker och problem med OS som till exempel säkerhetshål, buggar, licenser, support, dokumentation, såna saker och vissa är ju inte specifika för OS, det finns ju säkerhetshål i proprietär programvara också såklart, men det är dom grejerna vi kollar på där. Så det vi vill ta reda på är om företag tänker på dom här sakerna över huvud taget, om det finns några utarbetade riktlinjer eller processer för att hantera dom här OS-specifika riskerna då. Så det ena vi kollar på är open source då, det andra vi kollar på är IT risk management kallas det, som är som en-, hanteringen av IT-relaterade risker och problem kan man säga

AH: Men du kan väl berätta lite kort om dig själv och din roll och så?

X: Javisst! Jag önskar att jag hade gått systemvetarlinjen kan jag säga, när det begav sig. Jag läste industriell ekonomi för, ja jag gick ut för 12 år sen och konsultade lite, av en slump hamnade jag medan jag skrev mitt exjobb inom IT-branschen och där blev jag sen kvar av många händelser. Konsultade lite först och så hamnade jag för 10 år sen på Organisation X och där har jag stannat inom organisationen men har bytt jobb ganska många gånger inom koncernen. Jag började i en teknikgrupp för infrastruktur så jag är ganska väl bekant med både säkerhetsaspekter men även andra aspekter inom infrastruktur. Sen har jag gått vidare inom Governance generellt, tjänstepaketering men också inom applikationsutveckling. För tre och ett halv år sedan började jag på Organisation X, inom koncernen. Först fokuserade jag på att trimma våra utvecklingsteam (? bröts lite) internt, sen var jag pappaledig och nu är jag tillbaka i ännu en ny roll där jag ansvarar för ett team som kallas för Vendor Management and Governance, det är egentligen stabsfunktionen till IT, man skulle kunna kalla det IT-office eller någon form av management support. Vi hanterar egentligen governance-frågor runt IT, vi hanterar inköp av externa tjänster, vi tittar på regelverk och compliance, när det kommer revisorer och tittar på om vi som bank sköter oss så är det vi som koordinerar det också. Plus att vi har koncernens IT-funktion som sitter hos oss också. Det är väldigt mycket stödfunktioner till dom levererande delarna av IT så att säga, i sammanhanget kan det väl vara bra att känna till sådär kort, jag vet inte om ni har fått någon introduktion av vårt företag från annat håll?

FM: Jag har suttit och spärrat kort hos Organisation X på nätterna via ett annat företag, så jag vet lite grann men inte jättemycket!

X: Okej! Superkort då, vi är en medelstor bank, drygt 700 000 bankkunder. Vi anses fortfarande vara en nischbank även om vi har alla produkter som en stor bank har också, vi är såklart pyttesmå jämfört med dom fyra stora bankerna. Vi har ett eget försäkringsbolag som är ett dotterbolag till oss, som alla banker står vi under finansinspektionens tillsyn och det kommer jag säga många gånger säkert under denna timmen, för det styr väldigt mycket. Banker i Sverige är väldigt hårt hållna skulle jag vilja säga, hur vi beter oss, vad vi får göra och hur vi bör tänka. Inte minst när det kommer till riskhantering. Det kan tyckas tråkigt men det gäller ju att vara smart under de förutsättningar som ges. Vi har väl ungefär att hälften av vår applikationsportfölj är egenutvecklad och andra hälften är köpt, vi är delvis moderna när vi ligger mycket på windows-plattform men också ganska omoderna då vi har ganska mycket i Mainframe-världen, COBOL fortfarande.

AH: Okej

X: Genomsnittsåldern på dom utvecklarna är väl 72 snart hehe, det är ju bara dom som håller på med det

AH: Haha, ja det några få som har den kunskapen fortfarande. Men vad bra, vi skulle kunna gå vidare och fråga vad din uppfattning om begreppet OSS är, så att vi ligger ungefär på samma plan?

X: Jag är inte den främsta experten på tekniken bakom det, men självklart är det ju ett väldigt effektivt sätt att bedriva utveckling på eftersom man kan återanvända mycket komponenter. Det vill man ju allt som oftast göra. Jag tror att det blir vanligare och vanligare i våran värld att göra det, där vi kommer från dom här rigida mainframe/unix-baserade systemet som är väldigt specialanpassade till banker till att bygga mycket mer modulärt, både med OS-komponenter när vi gör i egen regi men också, vilket är lite sidospår härifrån, men ändå att vi tittar mer mot cloud-tjänster, vi tittar på köpta tjänster för att dela utvecklingskostnaden med andra mindre företag, där man inte har råd att utveckla allting i helt egen regi, så därför tycker jag absolut att OS är relevant för oss. Sen har vi ju säkerhets-aspekten såklart som vi säkert kommer komma till också.

AH: Ja juste, vi kommer lite till det sen också. Det ena vi undersöker är då OS, det andra är det som kallas IT Risk Management, vad är din uppfattning om begreppet, har du hört det innan?

X: Om IT Risk Management?

AH: Ja, precis.

X: Ja det jobbar vi ju jättemycket med, vi har en funktion i mitt team som kallas precis IT Risk Management och jag vill tro att vi är åtminstone inom koncernen är vi ganska mogna inom vår riskhantering. Vi jobbar både proaktiv och reaktivt med riskhantering och omsätter det mesta, alla förändringar vi gör går igenom våran riskprocess och vi jobbar oftast, i alla fall när det kommer jobb, riskbaserat. Inte för att vi ska ta bort alla risker utan för att vi ska kunna hantera dom på bästa sätt. Banken har en definierad riskaptit som alla våra verksamheter ska rymmas inom och alla förändringar som sker påverkar bankens riskaptit. Då är det upp till oss att använda vår del av den aptiten på bästa sätt. Utan risk finns det inga vinster, så det behövs ju, det är en naturlig del av våran verksamhet.

AH: Juste, bra! Om vi kommer tillbaka till OS då, har du några exempel på OSS ni använder just nu eller har använt?

X: Jag kan nog inte namedroppa några men jag vet att vi har, frågan dök från början upp inom vårt system för hantering av betalning och transaktioner inom kort. Där vi snabbt upptäckte, vi har ju en ganska rigid process över hur vi hanterar risker och den fyller nog upp ett ganska stort behov. Utöver den har vi en extra hantering kan man säga om när vi tar in OS-komponenter där vi, ja det finns ett visst flöde och en process, jag kan skicka över den faktiskt det är inget konstigt, den är ganska simpel. Det kräver ändå en extra sväng av godkännande och utlåtande från dom som vill, ja, dom inom IT-organisationen framförallt vår Enterprise-arkitekt och dess nätverk av IT-arkitekter. Men också av våran säkerhetsfunktion, för att dels kunna säga att jo men den här är i sitt nuvarande stadie okej men också för att få till en kontinuerlig bevakning över OS-komponenter. De kan ju i sin natur förändras utan att vi gör någonting egentligen, och där vet vi nog inte riktigt hur vi ska hantera det. Samma effekt gäller för molntjänster där man abonnerar på någonting som är föränderligt, då är det svårt att säga när man tog in den, eller det är ju ett avstamp när man tog in den men sen fortsätter den att förändras och det gäller att bevaka den här förändringen så att den inte glider iväg för mycket för vår riskaptit helt enkelt.

AH: Nej, Okej, jag ska säga det också att vi ställer frågor nu som du kanske har svarat på innan nu också, men det är för att vi har en mall som vi går efter så du är medveten om det. Om det kommer något som du känner att du har svarat på så kan du bara säga det, eller referera till det.

X: Haha, okej, ja eller så säger jag någonting helt annat!

AH: Haha, ja precis då blir det svårt. Men, du har varit inne lite på det, hur går det till när ni väljer OSS? Då förstår jag att det inte är fritt fram för utvecklarna att välja precis vad dom vill ha utan dom måste genomgå någon slags process för detta då?

X: Ja precis det är lite beroende på hur, i vilket skede som det är. Om vi säger att vi ska etablera en helt ny funktion, någon funktion som idag inte finns och där OS kan bli aktuellt, då har vi normalt sett ett projekt som etablerar det här och då sker det urvalet i projektet under överseende av projektledaren men också en blandning av lösningsarkitekter som då blir involverade beroende på vilken teknikstack det handlar om. Vi tar ganska mycket stöd när det kommer till detaljerade tekniska frågor så har vi ganska mycket stöd från koncernen där vi har en central funktion med mer tekniktunga arkitekter. Våra är mer verksamhetsvända. Sker det inom en befintlig funktion där man gör en mindre förändring och det avser skifta ut någonting mot OS eller ändra någonting, då är det det systemets tekniska ägare som gör urvalet och föreslår att "jag skulle vilja göra en grej som innebär att vi tar in den här OS-komponenten" det slutar ändå med samma process för att godkännandet ska gå igenom där arkitekterna och säkerhetsfunktionen ska godkänna komponenten och det handlar väl också om att vi också kontinuerligt följer upp det. Något som är lite svårt, det är ännu en process vi använder oss av som kallas Risk och Regelverksanalys som syftar till att alla förändringar som sker någonstans inom bankens miljöer ska gå genom en sån här analys. Där blir det jättesvårt då man tar in någonting som ständigt förändras, då vet man inte när man ska göra analysen, där vi får vi fortfarande ta ställning till när förändringen är tillräckligt stor för att faktiskt lyftas upp. Om vi kan göra en bedömning att den här påverkar vår riskexponering på något sätt, då måste vi göra en sån här Risk och Regelverksanalys. Utkomsten av den kan ju antingen bli att

ja, vi lever med risken eller vi kan göra någonting för att mitigera risken eller vi får se till att mitigera den på ett annat håll eller att vi inte vill leva med risken, det vill säga att vi säger nej till denna förändring. Det leder då till att man inte får fortsätta använda komponenten eller att man inte tar in den från första början.

AH: Då förstår jag att ni gör det här, ett godkännande då, vet du något mer ingående hur den processen går till? Vi har till exempel identifierat i det här IT Risk Management som du har varit inne på också, så finns det tre stora delar, identifiering, bedömning och hantering. Vet du om dom på något sätt följer den mallen eller den metoden eller hur det går till?

X: Ja jag skulle vilja säga godkännandet är lite vid sidan av, våran riskprocess den löper ju på allt vi gör, oavsett om det är OS eller inte. Risker flödar genom organisationen, vi gör riskutvärderingar, med jämna mellanrum och det finns många inflöden till det. Utöver den sker också ett godkännande, normalt sett är det den normala riskrutinen som säger att riskägaren är samma person som ansvarar för systemet, eller för den delen för affärsprocessen som flödar genom systemet. Då blir det ju lite så här att man ställer krav på sig själv. Medan godkännandeprocessen för OS, den är ju, den görs av samma funktioner det är arkitektur och säkerhet alldeles oavsett vilken del av banken som avser implementera den här OS-funktionen.

AH: Okej

X: Dom överlappar/kompletterar varandra lite beroende på hur man väljer att se på det.

AH: Ja, det finns en rätt generell riskprocess där.

X: Mm, sen i vanlig riskutvärdering, det har vi också en mall för som jag skulle kunna skicka över som underlag, ganska simpelt där man gör en matris med risk, konsekvens och så har vi graderingar i fem steg tror jag att det är. Där plottar man ut risken och över en viss nivå blandas flera (?) (samtalet bröts lite grann), längst upp i högra hörnet ja då är det i princip VDn som måste besluta om vi kan leva med den här risken eller inte och nere i vänstra hörnet då är den så liten att då kan en delegerad del av organisationen ansvara för risken själva.

AH: Juste, vi har ju kollat lite på liknande, vad ska man säga ramverk som är mer specifika för OS, det finns ett som heter RISCOSS och OSS Maturity Model, är detta någonting du har hört om eller någonting du vet om?

X: Nej, faktiskt inte, jag tror att våran OS-godkännande, den är nog väldigt intern. Den är nog i sin absolut simplaste form, det är i stort sett upptäck, review och godkänn.

AH: Okej, kanon. Vet du om att ni har upplevt några problem med just OS som buggar, för lite support/dokumentation, säkerhetsbrister?

X: Nej, sen ska jag säga att vårt användande är ganska begränsat och mig veterligen har vi nog inte tagit in det för någon funktion som är superduper-verksamhetskritisk. I den mån som vi har haft problem så har det inte fått så stora konsekvenser. Sen är jag ganska säker på att det kommer hända så vi utnyttjar mycket mer publika funktioner och standardiserade funktioner, så att det kanske kommer uppstå.

AH: Ja, det kommer säker med användningen. Vi var inne lite på compliance och sånt förut. Det finns ju OS-licenser också, OSS är licensierad på olika sätt, vet du hur ni kontrollerar att dom följs korrekt och att det inte bryts mot där och så?

X: Ja, vi har inom koncernen en central hantering av Software Asset Management och licenser därtill som, jag ska väl inte säga att vi slaviskt följer den men dom allra flesta inköp av licenser görs där genom. Då köper vi kontrollen av det som tjänst så att säga, både hjälper till att förhandla fram i den mån det är förhandlingsbart gällande licensvillkor men också ser till att vi, det finns lika många licensmodeller som det finns licenser näst intill så att det är en hel vetenskap att hålla koll på det. Där vi försöker i god mån nyttja den centrala funktionen, och då kan vi dessutom göra inköpen gemensamt mellan bolag i koncernen, det är inte ovanligt att någon annan har samma behov någon annanstans inom en stor koncern.

AH: Juste

X: Exakt hur dom jobbar med det vågar jag faktiskt inte, jag kan förmedla en kontakt till det men jag vet faktiskt inte hur dom, om dom har något verktyg eller vad dom håller på med.

AH: Nej precis, vi vet ju om att det berör många olika delar som kan vara svårt för en person att hålla koll på, men det är bra. Använder ni något IT Risk Management ramverk eller standarder som t.ex. ISO 27005 eller ISACAs IT Risk Framework?

X: Vi är väl inte ISO 27005-certifierade men det mesta av det vi gör är ju inspirerade av, det är ju inte svar på er fråga men vi har ett, ERC heter det, det är någon hyfsat stor och känd programvara för riskhantering som ju de facto, eftersom den är stor och hyfsat standardiserad så styr den in oss mot best practice. Men jag vet faktiskt inte om vårt ramverk är kopierat från nåt av dom eller om det är vårt egenuppfunna men jag skulle tro på det förstnämnda.

AH: Att det är kopierat?

X: Ja att det är åtminstone väldigt hårt inspirerat från något större ramverk.

AH: Precis, många av dom ramverken är ju väldigt snarlika när vi har kollat på dom och jag tänka mig att den processen som finns i dom ramverken är det man ofta kör efter i organisationer bara att man kanske inte har exakt, att man säger att man implementerar ett ramverk till punkt och pricka.

X: Nej exakt, jag vill ju tro att det är någon auditor/konsult som har implementerat det på vårt företag och då gör dom det enligt sin best practice, sen att vi kallar det något annat.

AH: Ja precis, okej, någonting som dom här ramverken har gemensamt är en formel för att beräkna IT-risker som är, eller risker överlag, som är T * V * A = R, det är en sån som vi har stött på ganska mycket nu, är det någonting du har hört om eller vet du om ni jobbar med det här?

X: Ja, vi jobbar med något annat som jag tror hör till finansbranschen som är någon förkortning som hette, IKLU hette den nog, Intern Kapital någonting. Där man (samtalet bröts lite) där inträffade risker, man tog med också verkliga inträffade incidenter som ett mått på de här riskerna. Sen var man tvungna, vi är ju ålagda att ha kapitaltäckning baserat på vilken, ja om man kommer till en hög siffra så behöver vi ju mycket pengar på vårat sparkonto för att hantera dom riskerna om dom väl slår in. Om det blir konsekvenser eller när det blir det på riktigt, så såna revisioner gör vi en gång per år för att säkerställa att vi har kapitaltäckning för dom riskerna som vi utsätter oss för utefter vår riskaptit. Jag tror och är rätt säker på att det där är något i finansbranschen känt sätt att göra det på, jag tror finansinspektionen till och med har krav på att vi gör det på ett visst sätt. Det är lite regler som bestämmer att så här ska bankerna räkna sin vad det nu än är, lånerisk, kreditrisk till exempel. Det får man inte hitta på som man vill utan det görs via en standard för att staten ska ha kontroll på bankerna.

AH: Okej, juste, något annat som dom här ramverken nämner väldigt, eller, trycker på är att ledningen eller högre instanser inom organisationen ska vara medvetna om riskerna med IT, i vårt fall med OS, hur känner du dig att det arbetet fungerar hos er, är ledningen med på det så att säga eller är det att det sköts på lägre nivå?

X: Ja men det tycker jag att vi är ganska duktiga på, våran senaste VD, eller hen som sitter nu, hen kom in i ett läge när det var ganska lösa (volymer?) och fick det i tydligt uppdrag från styrelsen att få kontroll på bland annat det här. Vi har ju en väldigt tydlig eskaleringsväg för risker, varenda avdelning på banken sitter kvartalsvis med en kommitté där man får rapportera av på sin riskkarta som den ser ut just nu och eskalera vid behov eller för den delen berätta om man har kontroll på sina risker. Eftersom min chef som är CIO på banken har den här rapporteringsvägen till sin chef så behöver ju han ha kontroll på sina detaljer ganska mycket. Vi har ju våran motsvarighet till det då inom IT-organisationen varannan vecka där våra risk managers stämmer av med CIOn vad progressen är på alla åtgärder för dom risker vi har identifierat. Det är en ganska levande organism våran riskkarta som vi ständigt jobbar med.

AH: Det finns en kommunikation där då. Då tror jag faktiskt att vi är klara där!

FM: Vi kommer ju transkribera det här så fort vi har tid och så kommer vi skicka det till dig så gott vi har kunnat sammanställa det så får du bara godkänna det innan vi använder oss av datan såklart.

X: Absolut

AH: Om inte du har något att tillägga så är vi nöjda så faktiskt.

X: Nej, inte vad jag kan komma på nu!