SINTEF

A26860 - Åpen

Rapport

Modenhetskartlegging av programvaresikkerhet i offentlige virksomheter

Forfatter(e)
Martin Gilje Jaatun
Inger Anne Tøndel
Daniela Soares Cruzes

BSIMM logo brukt med tillatelse. Se http://bsimm.com

SINTEF IKT
Systemutvikling og sikkerhet
2015-04-10

SINTEF IKT

Postadresse:
Postboks 4760 Sluppen
7465 Trondheim

Sentralbord: 73593000
Telefaks: 73594302

postmottak.IKT@sintef.no
www.sintef.no
Foretaksregister:
NO 948 007 029 MVA

Rapport

Modenhetskartlegging av programvaresikkerhet i offentlige virksomheter

EMNEORD:
Informasjonssikkerhet;
Programvaresikkerhet;
BSIMM;
Offentlige virksomheter

VERSJON
1.1

DATO
2015-04-10

FORFATTER(E)
Martin Gilje Jaatun
Inger Anne Tøndel
Daniela Soares Cruzes

OPPDRAGSGIVER(E)
Direktoratet for forvaltning og IKT (Difi)

OPPDRAGSGIVERS REF.
14/00907

PROSJEKTNR
102009573

ANTALL SIDER OG VEDLEGG:
48, inkl. 4 vedlegg

SAMMENDRAG
Difi ønsker å få en kartlegging av modenhet knyttet til informasjonssikkerhet i utvikling og anskaffelser av IKT-løsninger i offentlig sektor. Denne rapporten beskriver resultatene fra en spørreundersøkelse knyttet til i hvilken grad aktiviteter for å sikre programvare-sikkerhet er tatt i bruk som en del av programvareutviklingsprosessene i 20 offentlige virksomheter.

Funnene tyder på at offentlige virksomheter i Norge er gode på å etterleve lover og retningslinjer når de utvikler egen kode, men at det er et stort forbedringspotensial i forbindelse med måling av effekten av innførte sikkerhetstiltak og opplæring av utviklere innen sikker programvareutvikling.

UTARBEIDET AV
Martin Gilje Jaatun

SIGNATUR

KONTROLLERT AV
Karin Bernsmed
FOR

SIGNATUR

GODKJENT AV
Eldfrid Øvstedal

SIGNATUR

RAPPORTNR	ISBN	GRADERING	GRADERING DENNE SIDE
A26860	978-82-14-05895-6	Åpen	Åpen

Historikk

VERSJON	DATO	VERSJONSBESKRIVELSE
1.0	2015-03-27	Første versjon
1.1	2015-04-07	Språkretting og oppklaringer

SINTEF

Innholdsfortegnelse

Utvidet sammendrag

Det offentlige gjennomfører en rekke digitaliseringsprosjekter, og utfører dermed mange aktiviteter knyttet til utvikling og anskaffelse av nye digitale løsninger. For at disse løsningene skal kunne håndtere de digitale sikkerhetstruslene, er det viktig at det jobbes systematisk med sikkerhet i utviklingsprosessen.

Denne rapporten presenterer resultatet av en undersøkelse av modenhetsnivået når det gjelder programvaresikkerhet hos 20 norske offentlige virksomheter. Undersøkelsen er basert på rammeverket til *Building Security In Maturity Model* (BSIMM). BSIMM er en studie av en rekke virksomheter og deres programvaresikkerhetsaktiviteter, og rammeverket beskriver aktiviteter som gjennomføres hos en større andel av de virksomhetene som er med i BSIMM-studien. Mange av disse virksomhetene er langt fremme på programvaresikkerhet, og BSIMM-rammeverket beskriver dermed aktiviteter som anses som nyttige i praksis for å bedre sikkerheten i programvare.

Modenhetskartleggingen av offentlige virksomheter har blitt gjennomført som en spørreundersøkelse. Virksomhetene har angitt hvilke av de 112 aktivitetene beskrevet i BSIMM-studien som de har tatt i bruk som en del av sin egen programvareutviklingsprosess. Svarene er fulgt opp i etterkant gjennom intervjuer med den enkelte virksomhet, for å utdype informasjonsgrunnlaget.

Modenhetsstudien konkluderer med at norske offentlige virksomheter fremstår som bedre enn det offisielle BSIMM gjennomsnittet innenfor området "Etterlevelse av lover, regler og retningslinjer" (*Policy & Compliance*), noe som er forståelig ut fra at offentlige virksomheter er vant til å måtte forholde seg til krav fra myndighetene.

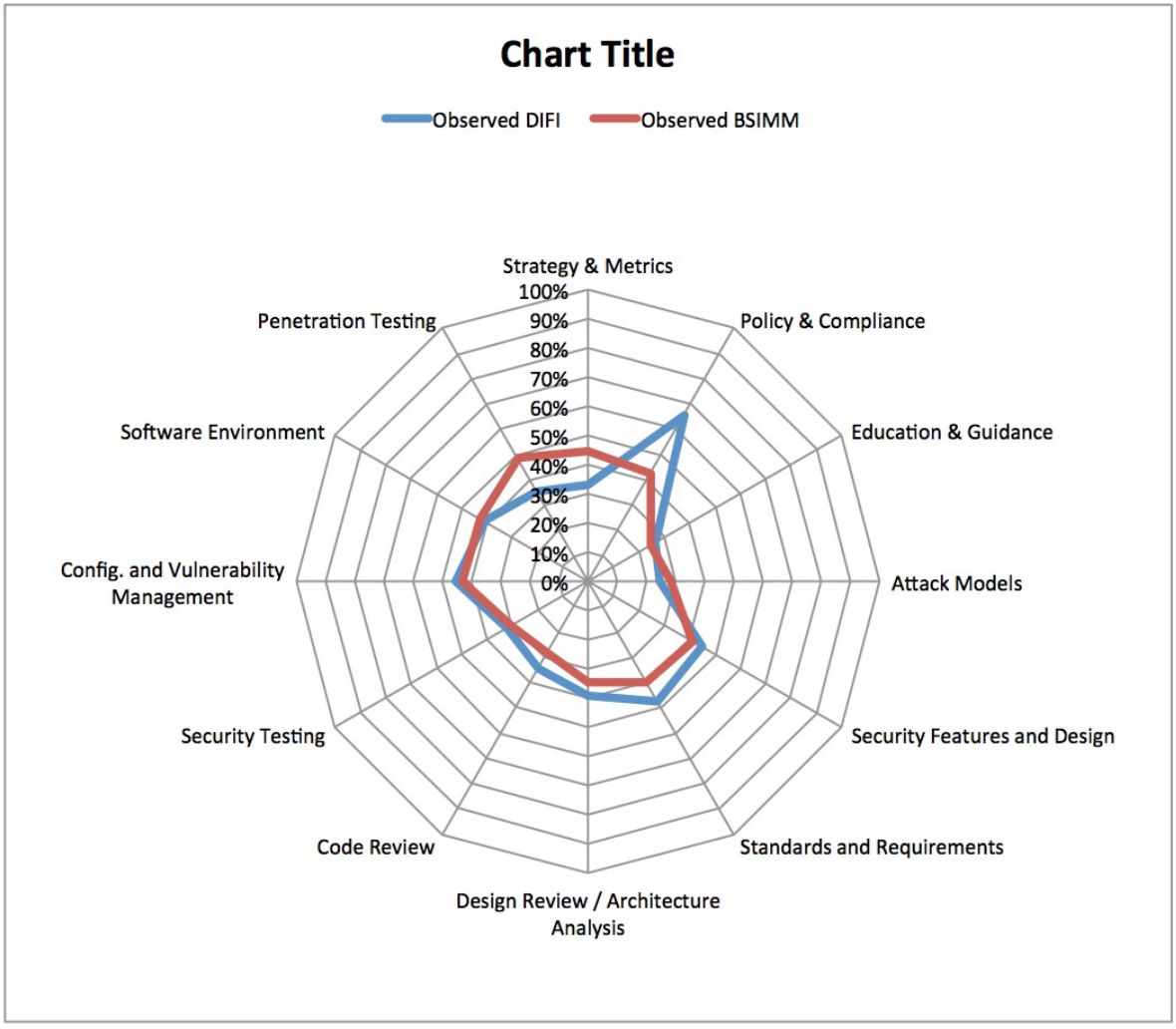
Virksomhetene er imidlertid vesentlig dårligere innenfor områdene "Strategi og måling" (*Strategy & Metrics*), "Angrepsmodeller" (*Attack Models*), og penetreringstesting (*Penetration Testing*). Selv om tallene for "Opplæring og øvelser" (*Education & Guidance* eller *Training*) ser marginalt bedre ut enn i BSIMM-studien, er hovedinntrykket at virksomhetene har for dårlige rutiner for opplæring innen programvaresikkerhet, spesielt ved innleie av utviklere. Virksomhetene er ikke i stand til å måle effekten av programvaresikkerhetstiltak som iverksettes, og de jobber ikke systematisk med å identifisere og forstå angrep som er spesielt relevante for løsningene de utvikler.

Resultatene tyder på at til tross for at virksomhetene gjør en del aktiviteter innenfor programvaresikkerhet, så er det få som har en helhetlig strategi på området. Mange er veldig avhengige av interessen og kompetansen til den enkelte utvikler. Det er også mye som tyder på at det fortsatt eksisterer et skille mellom "sikkerhetsfolk" og "utviklere" i virksomhetene, og at det fortsatt er for mange utviklere som anser at sikkerhet er noe som kun angår sikkerhetsadministratorer.

Det er viktig å påpeke at det er stor variasjon mellom virksomhetene. Den beste virksomheten har implementert 87 av de 112 aktivitetene, og den minst modne har kun implementert 9. Gjennomsnittet av de 20 virksomhetene er omtrent 44 av 112. Den gjennomsnittlige fordelingen per praksis er illustrert i Figur 1.

Studien er gjennomført som en spørreskjemaundersøkelse med oppfølgingsintervjuer for å oppklare misforståelser og uklarheter. Den kan følgelig karakteriseres som en assistert selvevaluering, der vi ikke har hatt mulighet til å innhente evidens for at virksomhetene faktisk gjør det de sier.

Areas	Observed DIFI	Observed BSIMM
Strategy & Metrics	33%	45%
Policy & Compliance	66%	43%
Education & Guidance	27%	24%
Attack Models	25%	29%
Security Features and Design	45%	41%
Standards and Requirements	48%	40%
Design Review / Architecture Analysis	39%	35%
Code Review	35%	28%
Security Testing	32%	30%
Config. and Vulnerability Management	46%	43%
Software Environment	41%	42%
Penetration Testing	36%	49%



Figur 1: Gjennomsnittlig prosentdel av aktiviteter i virksomhetene

1 Innledning

Samfunnet er i dag avhengig av informasjons- og kommunikasjonsteknologi (IKT). IKT er en grunnleggende infrastruktur for samhandling. Dette gjør at sikkerheten knyttet til IKT-systemene er viktig i et samfunns-perspektiv [1]. Tradisjonelt har IKT-sikkerhet primært dreid seg om å implementere sikkerhetsmekanismer på system- eller nettverksnivå. I senere tid har det imidlertid blitt klart at det er vel så viktig å sørge for at alle mekanismene i programvaren er sikre, dvs. utvikle programvaren fra bunnen av slik at den er sikker mot angrep [2]. Dette er det vi kaller *programvaresikkerhet*.

I denne rapporten presenteres resultater fra en undersøkelse av modenheten med hensyn til programvare-sikkerhet i 20 norske offentlige virksomheter, etter malen fra BSIMM-rammeverket [3]. BSIMM gir den enkelte virksomhet mulighet til å sammenligne egne programvareinitiativer med data fra andre norske virksomheter, og til en viss grad også med virksomheter som inngår i den offisielle internasjonale BSIMM-studien. Kartleggingen gir et bilde av modenhetsnivået til de undersøkte virksomhetene, men vil også gjøre den enkelte virksomhet i stand til å identifisere forbedringsområder hvor de har et lavere modenhetsnivå sammenlignet med andre norske virksomheter.

BSIMM måler hvilke aktiviteter som inngår i en virksomhets samlede livssyklus for sikker utvikling av programvare (Secure Software Development Lifecycle – SSDL). Sentralt i BSIMM er konseptet "programvaresikkerhetsgruppe" (Software Security Group – SSG), dvs. den eller de som har ansvaret for å følge opp programvaresikkerheten i en virksomhet. Den kan være så liten som en enkelt person, og det trenger ikke være en formell rolle, og trenger heller ikke være en fulltidsfunksjon. I tillegg brukes begrepet "Satellitten" (*The satellite*) om en mer eller mindre veldefinert gruppe utviklere som ikke er en del av SSG, men som likevel har en spesiell interesse for og kunnskap om programvaresikkerhet, og som dermed kan funger som SSGs forlengede arm i mange sammenhenger.

Rapporten er strukturert på følgende måte: Kapittel 2 gir en beskrivelse av metoden som er benyttet. Kapittel 3 presenterer hovedresultater fra spørreundersøkelsen, og resultatene diskuteres i kapittel 4. Kapittel 5 har en avsluttende oppsummering og skisserer videre arbeid.

2 Metode for datainnsamling og rangering av resultater

I dette arbeidet har vi benyttet teknikken spørreskjema med individuelle oppfølgingsintervjuer [8]. Spørreskjemaet er basert på programvaresikkerhetsrammeverket til Building Security In Maturity Model (BSIMM) slik det er dokumentert i BSIMM-V-rapporten [1]. Rammeverket beskrives overordnet i avsnitt 2.1.

Det finnes to velkjente modenhetsmodeller for programvaresikkerhet; BSIMM og OpenSAMM [4]. Begge har et felles opphav, og har mange likhetstrekk. Vi valgte å basere oss på BSIMM kontra OpenSAMM av to årsaker: forskergruppen var på forhånd mest kjent med BSIMM, og samtidig er BSIMM en mer rendyrket deskriptiv metodikk som i utgangspunktet er designet for å måle, mens OpenSAMM har et sterkere fokus på normative betraktninger, dvs. å definere hva som er "den riktige måten å gjøre det på". En annen fordel med BSIMM er at det her finnes resultater fra et større antall amerikanske og internasjonale virksomheter som det går an å sammenligne med. Utgangspunktet for den første BSIMM-undersøkelsen i 2008 var å studere hvilke programvaresikkerhetsaktiviteter ni utvalgte virksomheter gjorde. De ni virksomhetene var presumtivt langt framme på programvaresikkerhet, og aktivitetene som ble observert her dannet grunnlaget for rammeverket i Tabell 1. Cigital [5] besøkte fysisk hver virksomhet, og de første undersøkelsene ble gjort av Gary McGraw og Sammy Migues personlig over en hel dag.

Det er viktig å presisere at ressursbegrensingene for dette prosjektet impliserer at metoden som er benyttet her må karakteriseres som "assistert selvevaluering", og følgelig ikke har samme grad av grundighet som BSIMM-evalueringene utført av Cigital. Det betyr rent konkret at vi må regne med at det har vært lettere for virksomhetene å få godkjent en aktivitet i vår studie, enn det ville vært hvis Cigital hadde gjort undersøkelsen i henhold til sin vanlige praksis. Dette medfører at selv om våre resultater gir sterke indikasjoner på modenhetsnivået hos de evaluerte virksomhetene, kan ingen av virksomhetene hevde å derved ha etablert sitt "BSIMM Score", og man kan ikke uten videre sammenligne seg direkte med resultatene i de offisielle BSIMM-rapportene.

2.1 BSIMM programvaresikkerhetsrammeverk

Hensikten til BSIMM er å kvantifisere programvaresikkerhetsaktiviteter som utføres i virkelige utviklingsprosjekter i virkelige virksomheter. Ettersom disse prosjektene og virksomhetene bruker forskjellige metodikker og forskjellig terminologi, trenger vi et rammeverk som lar oss beskrive alle initiativene på en enhetlig måte [3]. BSIMM rammeverket består av tolv praksiser organisert i fire domener; Styring og ledelse (*Governance*), Etterretning *(Intelligence*), SSDL Trykkpunkter (*SSDL Touchpoints*) og Utrulling (*Deployment*) (se Tabell 1). Hver praksis har et antall aktiviteter fordelt på 3 nivåer, der nivå 1 er lavest modenhet og nivå 3 er høyest. Dette beskrives nærmere i kapittel 3; f.eks. er SM1.4 en aktivitet på nivå 1, SM 2.5 en aktivitet på nivå 2, og SM 3.2 en aktivitet på nivå 3 (se avsnitt 3.1).

I domenet "Styring og ledelse" omfatter praksisen "Strategi og måling" planlegging, tildeling av roller og ansvar, identifisering av programvaresikkerhetsmål, bestemme budsjetter, og identifisere målinger og "gates". Praksisen "Etterlevelse av lover, regler og retningslinjer" er fokusert på å identifisere kontrollmekanismer for overholdelse av bransjespesifikke regelverk, utvikle kontraktsmekanismer som Service Level Agreements (SLA) for å styre risiko relatert til hyllevare programvare, etablering av organisasjonens programvaresikkerhetsretningslinjer, og revisjon mot disse retningslinjene. Opplæring har alltid spilt en viktig rolle i programvaresikkerhet fordi systemutviklere og arkitekter ofte har svært lite sikkerhetskunnskap i utgangspunktet.

Etterretningsdomenet er ment å skape organisasjonsomfattende ressurser, som er delt inn i tre praksiser. Angrepsmodeller fanger opp informasjon som brukes til å tenke som en angriper: trusselmodellering, utvikling og raffinement av abuse cases, dataklassifisering og teknologispesifikke angrepsmønstre. Praksisen sikkerhetsegenskaper og design er ment å skape brukbare sikkerhetsmønstre for vesentlige kontroll-

mekanismer (for å oppfylle standardene definert i neste praksis), og bygge mellomvare rammeverk for disse kontrollmekanismene, og skape og publisere annen proaktiv sikkerhetsveiledning. Praksisen Standarder og krav innebærer å innhente eksplisitte sikkerhetskrav fra organisasjonen, bestemme hvilke hyllevare produkter å anbefale, etablere standarder for viktige sikkerhetsmekanismer (som autentisering, inputvalidering, osv.), lage sikkerhetsstandarder for teknologier i bruk, og etablere en intern godkjennings-komité for standarder.

Tabell 1: BSIMM programvaresikkerhetsrammeverk

Programvaresikkerhetsrammeverk (*Software Security Framework*)			
Ledelse og styring (*Governance*)	**Etterretning** (*Intelligence*)	**SSDL Trykkpunkter** (*SSDL Touchpoints*)	**Utrulling** (*Deployment*)
Strategi og måling (*Strategy and Metrics*)	Angrepsmodeller (*Attack Models*)	Arkitekturanalyse (*Architecture Analysis*)	Penetreringstesting (*Penetration Testing*)
Etterlevelse av lover, regler og retningslinjer (*Compliance and Policy*)	Sikkerhetsfunksjonalitet og design (*Security Features and Design*)	Kodegjennomgang (*Code Review*)	Programvaremiljø (*Software Environment*)
Opplæring og øvelser (*Training*)	Standarder og krav (*Standards and Requirements*)	Sikkerhetstesting (*Security Testing*)	Konfigurasjonsstyring og sårbarhetsstyring (*Configuration Management and Vulnerability Management*)

Domenet SSDL Trykkpunkter omfatter god programvaresikkerhetspraksis som er integrert i virksomhetens programvareutviklingslivssyklus (SDLC). De to viktigste programvaresikkerhetspraksisene er Arkitektur-analyse og Kodegjennomgang. Arkitekturanalyse omfatter å avbilde programvarearkitektur i konsise diagrammer, sammenholde med lister med risikoer og trusler, velge en prosess for gjennomgang (som f.eks. STRIDE eller Arkitektonisk Risikoanalyse), og etablere en plan for vurdering og utbedring av eventuelle mangler. Praksisen Kodegjennomgang omfatter bruk av kodegjennomgangsverktøy, utvikling av skredder-sydde regler, tilpassede profiler for verktøybruk av forskjellige roller (f.eks., utviklere kontra revisorer), manuell analyse, og sporing/måling av resultater. Praksisen Sikkerhetstesting omfatter testing før utgivelse, inkludert integrering av sikkerhet i standard kvalitetssikringsprosesser. Praksisen inkluderer bruk av "black box" sikkerhetsverktøy (bl.a. fuzzere) som en inngangstest i QA, risikodrevet "white box" testing, anvendelse av angrepsmodellen, og kodedekningsanalyse. Sikkerhetstesting fokuserer på sårbarheter i konstruksjonen.

Som kontrast har vi Utrullingsdomenet, hvor praksisen Penetreringstesting involverer mer standard utenfra-og-inn testing av samme type som gjøres av sikkerhetsspesialister. Penetreringstesting fokuserer på sårbar-heter i den endelige konfigurasjonen, og gir direkte innspill til feilhåndtering og korrigering. Praksisen Programvaremiljø befatter seg med lapping og oppdatering av applikasjoner, versjonskontroll, sporing og korrigering av defekter, og hendelseshåndtering.

2.2 Deltakerne

Spørreundersøkelsen er besvart av representanter fra 20 norske offentlige virksomheter. For enkelte virksomheter har flere personer vært involvert i besvarelsen. Følgende roller har vært involvert (rollenavn noe homogenisert for å bedre anonymitet):

- IT-direktør
- Avdelingsdirektør utvikling
- IT-sjef drift
- Seksjonssjef IT
- Seksjonsleder utvikling
- Gruppeleder
- IT-leder
- Utviklingsleder
- Løsningsarkitekt
- Sjefsarkitekt
- Systemutvikler
- IKT-rådgiver
- Sikkerhetsleder
- Informasjonssikkerhetsleder
- Sikkerhetssjef
- Sikkerhetsarkitekt
- Informasjonssikkerhetskonsulent
- Sikkerhetsanalytiker
- Sikkerhetsrådgiver
- Senioringeniør
- Seniorrådgiver

2.3 Spørreundersøkelse

Selve spørreundersøkelsen (spørreskjemaet er gjengitt i Vedlegg B) ble distribuert til 32 offentlige virksomheter som ble antatt å drive med intern programvareutvikling. Vi fikk 20 positive svar (62,5%), 4 avslo å svare av forskjellige årsaker, og 8 svarte ikke på henvendelsen i det hele tatt.

Spørreskjemaet bestod av to deler. I den første delen ble respondentene bedt om å gi generell bakgrunnsinformasjon knyttet til stilling, antall utviklere i virksomheten, andel av innleide utviklere og hvorvidt virksomheten kontraherte utvikling av nøkkelferdige løsninger fra eksterne firma. Den andre delen av spørreskjemaet ble brukt til å spørre konkret om hvilke av de 112 BSIMM-aktivitetene virksomhetene utfører i sin daglige drift.

Vi valgte å beholde de engelske beskrivelsene av BSIMM-aktivitetene i spørreskjemaet, ettersom det ikke finnes noen offisiell oversettelse av BSIMM til norsk. Imidlertid har vi uavhengig av denne undersøkelsen startet en serie med norske beskrivelser av BSIMM-aktiviteter på vår blogg [6], slik at vi ser for oss at senere oppfølgingsstudier med fordel kan lage et norsk spørreskjema

2.4 Oppfølgingsintervjuer

Vi avtalte oppfølgingsintervjuer på telefon eller videokonferanseløsningen GoToMeeting [7]. Sistnevnte løsning ga mulighet til å dele visning av spørreskjemaet med respondenten under intervjuet. Hovedhensikten med oppfølgingsintervjuene var å avklare misforståelser og eventuelle spørsmål rundt enkeltaktiviteter som ikke var forstått under utfylling. Intervjuene førte i de fleste tilfellene til at enkelte svar ble endret; alle "vet ikke" ble endret til enten ja eller nei, men enkelte "ja" ble også endret til "nei", og omvendt. Vi har tatt med relevante enkeltsitater for å illustrere resultatene.

Selve intervjuguiden finnes i Vedlegg C.

2.5 Dataanalyse og måling av modenhetsnivå

Den viktigste funksjonen til BSIMM er å fungere som en tommestokk for å avgjøre hvor virksomheten står sammenlignet med andre virksomheter [3]. Spørreskjemaet forteller oss hvilke aktiviteter virksomheten har på plass[1], og basert på hvor godt de dekker opp de forskjellige praksisene kan vi bestemme modenhetsnivået til hver enkelt virksomhet. BSIMM hevder at et radardiagram etter høyvannsmetoden (basert på tre nivåer per praksis) er tilstrekkelig for å gi et grovt, overordnet bilde på modenheten. Vi har valgt å i tillegg utvikle to komplementære modenhetsmål som kan gi et mer balansert modenhetsbilde. De tre modenhetsmålene vi bruker er følgelig:

1) **Konservativ modenhet** (*Conservative Maturity*; Skala 0-3): Her vil en virksomhet kun få godkjent et modenhetsnivå dersom alle aktivitetene i nivået er oppfylt ("Yes"), pluss at alle aktiviteter på lavere nivå også er oppfylt. Dersom virksomheten utfører noen (men ikke alle) aktiviteter på et nivå, indikeres dette med en "+", dvs. hvis man har 3 av 5 aktiviteter på første nivå, blir resultatet 0+; har man alle aktiviteter på nivå 1 og 2 av 4 aktiviteter på nivå 2, blir resultatet 1+, osv. I forbindelse med utregning av gjennomsnittsverdien telles en + som 0,5. Ettersom få virksomheter gjør alle aktivitetene på nivå 1, havner de fleste i kategorien 0+.

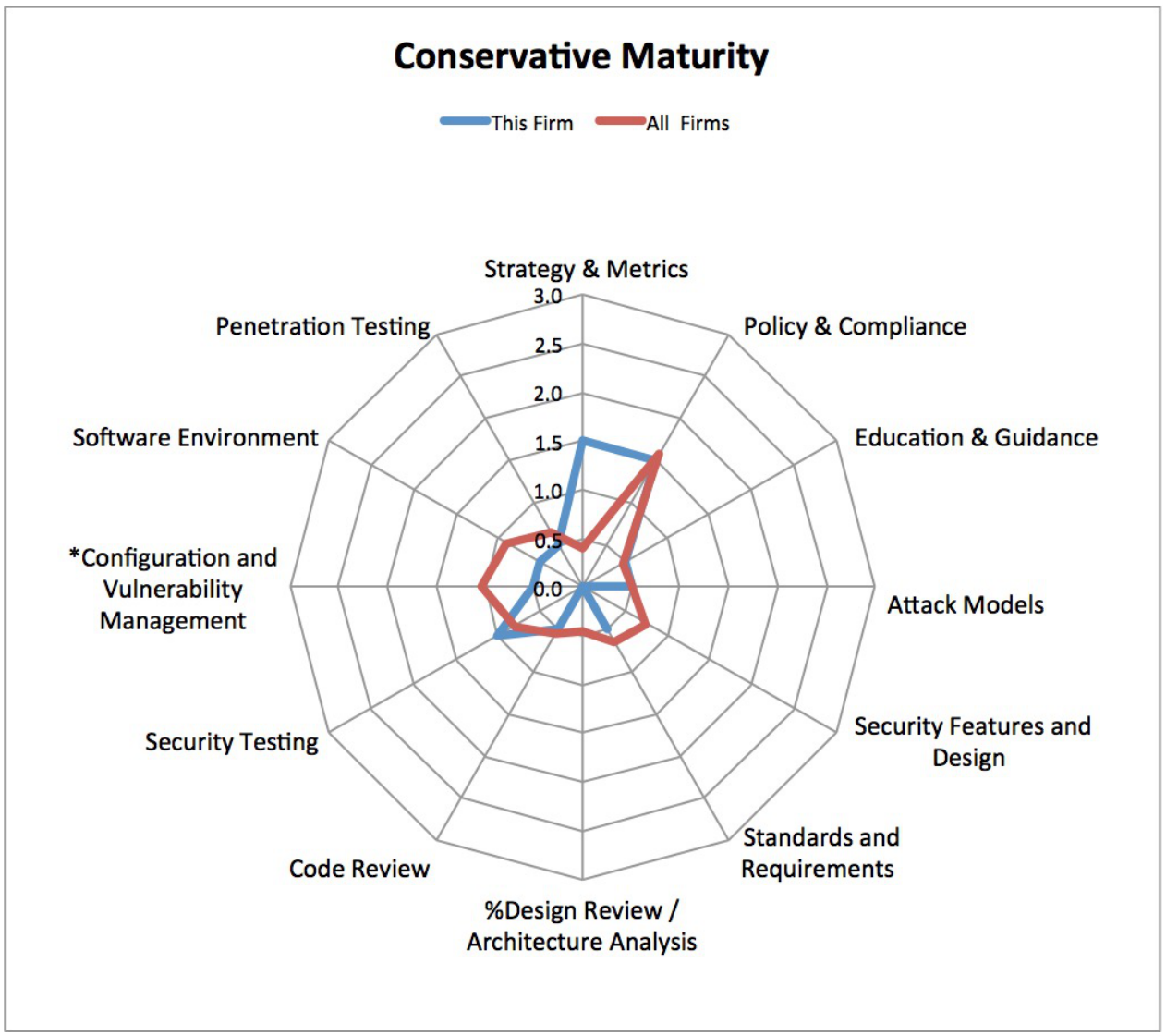
2) **Vektet modenhet** (*Weighted Maturity*; Skala 0-6): Denne verdien gir i større grad uttelling for aktiviteter på høyt nivå, også hvis laverenivå-aktiviteter ikke er fullstendig gjennomført. Verdien regnes ut etter følgende formel:

$$\frac{aktiviteter\ på\ nivå\ 1}{tot\ antall\ aktiviteter\ på\ nivå\ 1} \times 1 + \frac{aktiviteter\ på\ nivå\ 2}{tot\ antall\ aktiviteter\ på\ nivå\ 2} \times 2 + \frac{aktiviteter\ på\ nivå\ 3}{tot\ antall\ aktiviteter\ på\ nivå\ 3} \times 3$$

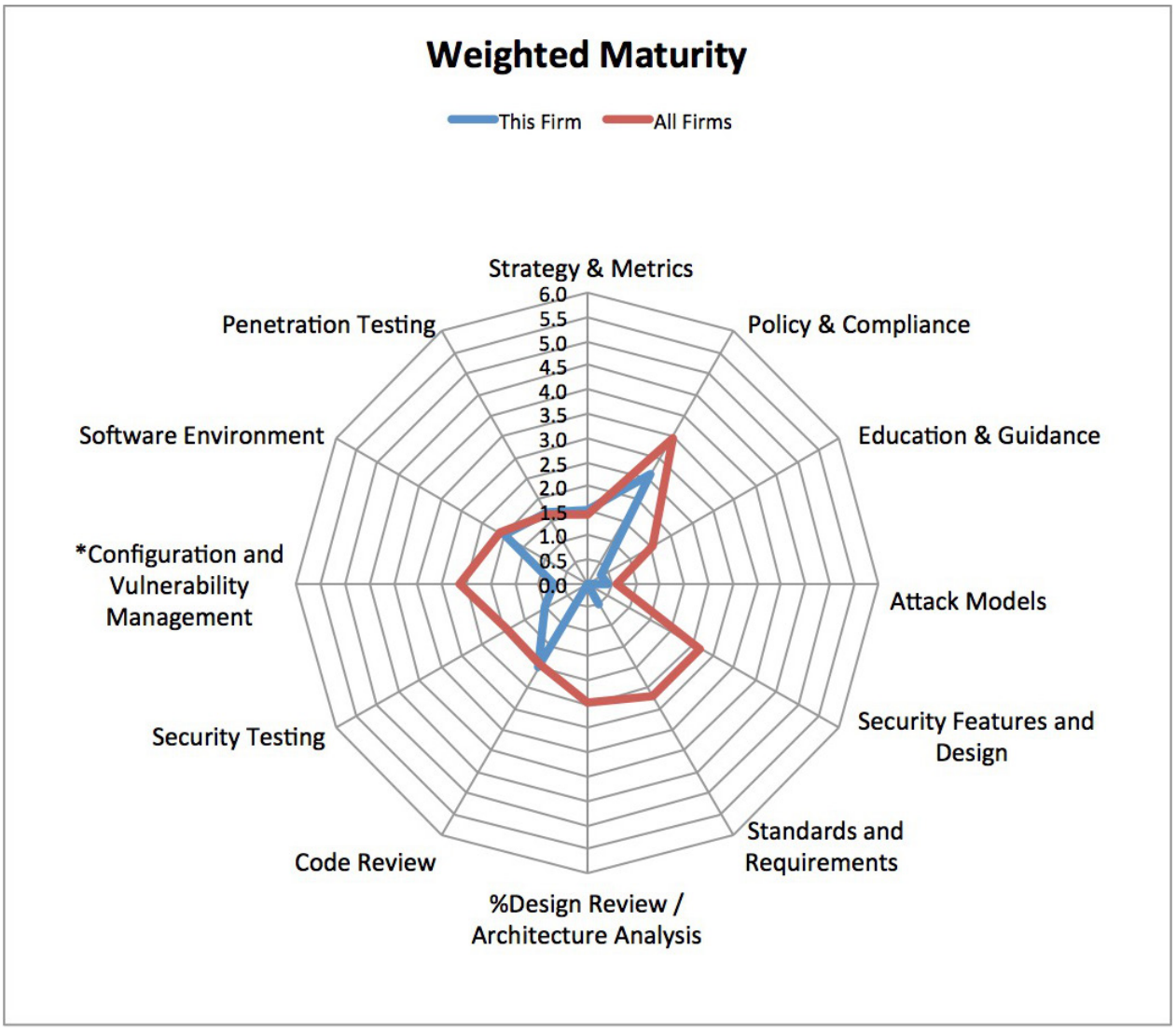
3) **Høyvannsmodenhet** (*High Watermark Maturity*; Skala 0-3): Denne beregnes på samme måte som i BSIMM [1]; hvis virksomheten har minst en aktivitet på nivå 3 får den modenhetsnivå 3. Høyvannsmodenheten sier derfor kun noe om hvilket nivå den høyest rangerte aktiviteten de utfører er på. Dette medfører at mange får nivå 2 eller 3 i noen praksiser. Høyvannsmodenheten er allikevel relevant å ha med, fordi det er denne som kan sammenlignes med den globale BSIMM-studien.

For denne studien gir det mest mening å sammenligne de to første modenhetstallene (konservativ og vektet) fra en gitt virksomhet med gjennomsnittsverdiene fra alle de norske virksomhetene for å se hvordan det står til, slik vi har gjort i radardiagrammene i Figur 2 og Figur 3.

[1] Som tidligere nevnt er det virksomheten selv som har avgjort hvorvidt de tilfredsstiller kriteriene for en gitt aktivitet; i noen tilfeller har vi kunnet justere dette basert på oppfølgingsintervjuene, men vi har ikke hatt anledning til noen systematisk evidensinnsamling slik Cigital gjør. Det er derfor grunn til å tro at det vil være lettere for en virksomhet å få "godkjent" av oss, enn hvis de hadde blitt undersøkt av Cigital.



Figur 2: Sammenligning mellom eksempelvirksomhet og alle norske virksomheter (konservativ modenhet)



Figur 3: Sammenligning mellom eksempelvirksomhet og alle norske virksomheter (vektet modenhet)

Vi har laget et individuelt diagram for hver virksomhet som deltok i studien. I tillegg har vi laget en BSIMM-V "karakterbok" for hver virksomhet som illustrert i Figur 4 for en oppdiktet virksomhet. I dette skjemaet har vi også en del tilleggsinformasjon; i svar-kolonnen ("Answer") vil en celle som er merket rød indikere en aktivitet som de fleste virksomhetene i studien gjør, men som denne virksomheten ikke gjør. Dette kan være en indikasjon på at virksomheten burde vurdere om denne aktiviteten burde tas i bruk[2]. På samme måte indikerer en grønn celle at virksomheten utfører en aktivitet som de fleste andre virksomheter i studien ikke gjør; noe som kan tolkes som at virksomheten er "best i klassen" på dette området. I kolonnen "Levels" indikerer vi kortfattet i hvor stor grad aktivitetene på hvert av de tre nivåene gjøres for denne virksomhet, ● betyr at alle aktivitetene tilfredsstilles, ◑ betyr at noen av aktivitetene er tilfredsstilt, mens ○ betyr at ingen av aktivitetene på dette nivået utføres i virksomheten. Eksempelet i Figur 4 viser bare de tre første praksisene, og forkortede aktivitetsbeskrivelser (den fulle aktivitetsteksten finnes i kapittel 3, og i spørreskjemaet gjengitt i Vedlegg B).

[2] Samtidig er det viktig å påpeke at det ikke nødvendigvis gir mening for en virksomhet å gjøre alle aktiviteter; individuelle forhold kan medføre at en populær aktivitet ikke er relevant for en spesifikk virksomhet.

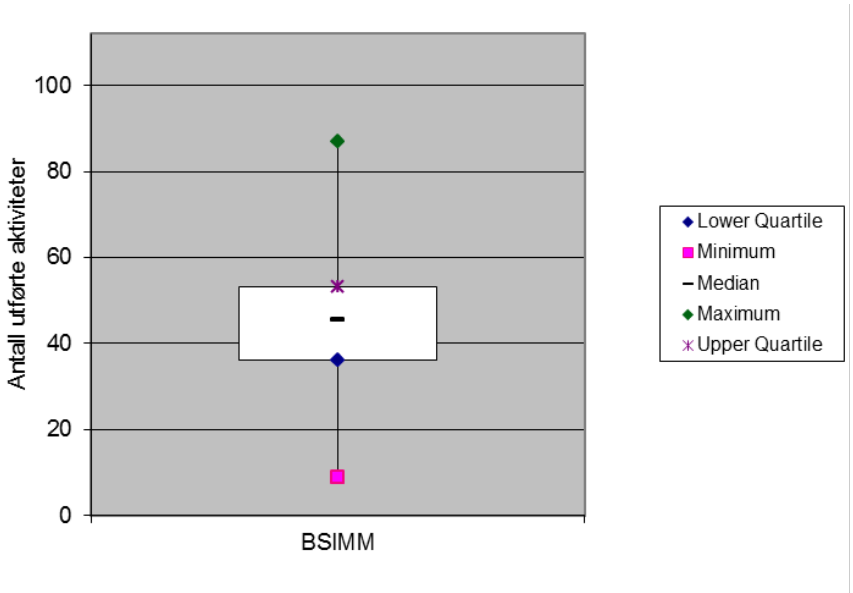
Assessment Worksheet								
Business Functions	**Security Practices**	**BSIMM**	**Activities**	**Answer**	**Levels**	**Weighted Score (0-6)**	**Conservative Maturity (0-3)**	**High Watermark (0-3)**
Governance	**Strategy & Metrics**	SM 1.1	We publish our	Yes	Level 1 ●	2,0	1+	2
		SM 1.2	We have a secure ...	Yes	Level 2 ◑			
		SM 1.3	We educate our ...	Yes	Level 3 ○			
		SM 1.4	We have *identified* ...	Yes				
		SM 2.2	We *enforce* the ...	Yes				
		SM 1.6	We have a process ...	Yes	**Percentage of Practices 63 %**			
		SM 2.1	The software	No				
		SM 2.3	In addition to the	Yes				
		SM 2.5	We have identified....	No				
		SM 3.1	The SSG has ...	No				
		SM 3.2	The SSG advertises ...	No				
	Policy & Compliance	CP 1.1	The SSG has an ...	Yes	Level 1 ●	2,6	1+	2
		CP 1.3	We have a ...	Yes	Level 2 ◑			
		CP 1.2	The SSG is ...	Yes	Level 3 ○			
		CP 2.1	We have identified	Yes				
		CP 2.2	All identified risks	No				
		CP 2.3	We can demo....	Yes	**Percentage of Practices 63 %**			
		CP 2.4	We make sure	Yes				
		CP 2.5	We promote	Yes				
		CP 3.1	We have all the ...	No				
		CP 3.2	When managing ...	No				
		CP 3.3	Information from ...	No				
	Education & Guidance	T 1.1	We have a security ...	No	Level 1 ○	0,6	0+	3
		T 1.5	We offer role	No	Level 2 ○			
		T 1.6	The security ...	No	Level 3 ◑			
		T 1.7	We deliver	No				
		T 2.5	We encourage ...	No				
		T 2.6	We provide...	No	**Percentage of Practices 8 %**			
		T 2.7	We use the ...	No				
		T 3.1	We have a reward ...	No				
		T 3.2	We provide...	No				
		T 3.3	We host external ...	No				
		T 3.4	We require an ...	No				
		T 3.5	The SSG has ...	Yes				

Figur 4: Eksempel på enkeltresultat (oppdiktet virksomhet)

3 Virksomhetenes modenhetsnivå

De virksomhetene som deltok i studien varierer med hensyn til hvor mye utvikling de gjør selv, og hvor mye de benytter seg av eksterne konsulenter. Noen få baserer seg utelukkende på eksterne konsulenter, mens mange opererer med en miks av interne og eksterne utviklere. Noen kjøper i stor grad løsningene fra system-leverandører, og gjør justeringer internt, mens andre utvikler størstedelen av løsningene selv.

Virksomheten med lavest modenhetsnivå erklærte at den gjorde 9 aktiviteter av 112, mens virksomheten med høyest modenhetsnivå gjorde 87 aktiviteter. Basert på boxplot-diagrammet i Figur 5 ser vi at mesteparten av virksomhetene kommer halvveis opp på skalaen; de gjør i gjennomsnitt 39% av aktivitetene.



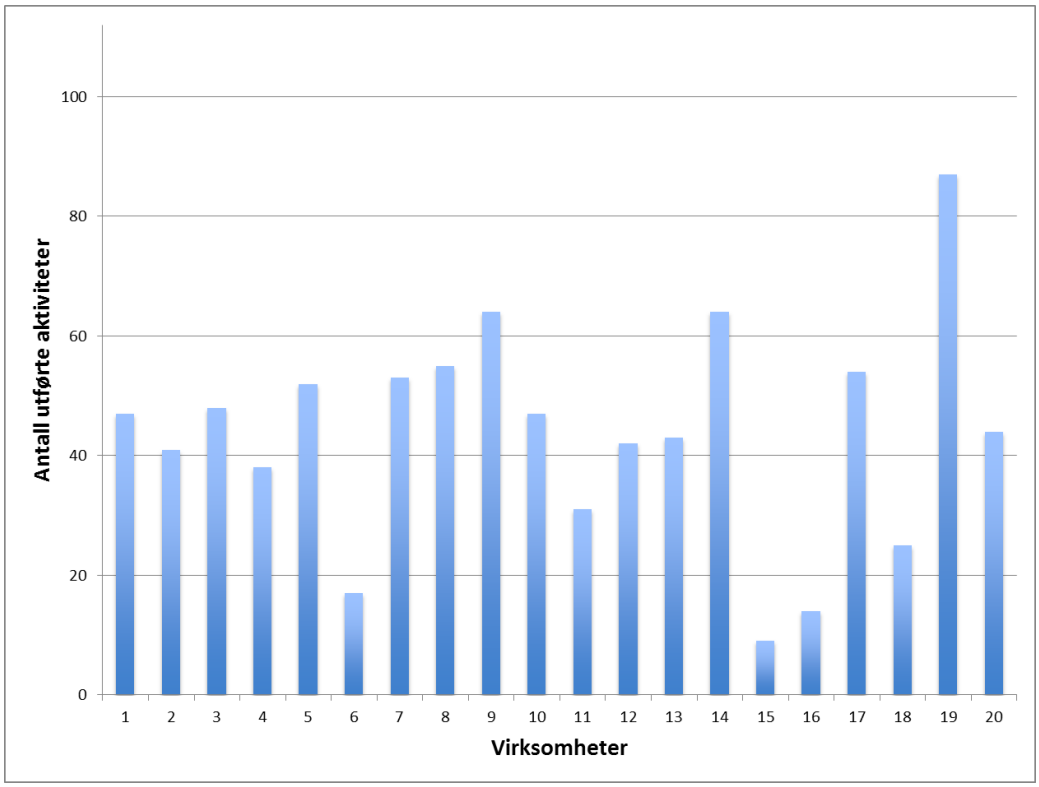
Figur 5: Boxplot-diagram for fordeling av totalt antall aktiviteter

De hyppigst utførte aktivitetene (dvs. alle aktiviteter som utføres av minst 80 % av de studerte virksomhetene) listes opp i Tabell 2.

Tabell 2: De hyppigst utførte aktivitetene blant alle norske virksomheter

ID	Aktivitetstekst	%
SE 1.2	We use accepted good practice mechanisms for host/network security.	90%
CMVM 2.1	We are able to make quick changes in the software when under attack.	85%
CMVM 2.2	We track software defects found during operations until they are closed.	85%
CP 1.1	The software security group has an overview of the regulations that our software has to comply with.	85%
CP 2.1	We have identified all the personally identifiable information stored by each of our systems and data repositories.	85%
CP 1.2	The software security group is responsible for identifying all legislation related to personally identifiable information (for example personopplysningsloven).	80%
AM 1.5	The software security group keeps up to date by learning about new types of attacks / vulnerabilities.	80%
SFD 1.2	Security is a regular part of our organization's software architecture discussion.	80%
SR 2.3	We use a limited number of standard technology stacks.	80%

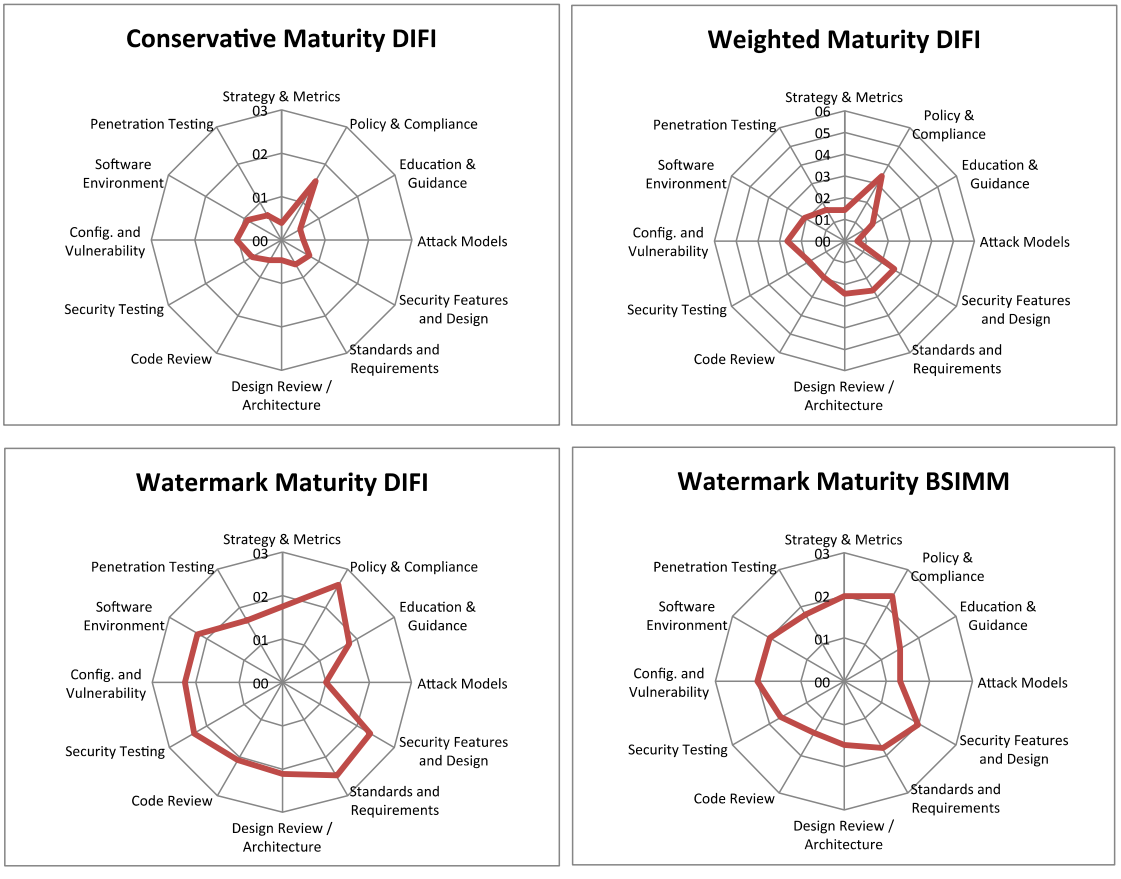
Den totale oversikten over hvor mange aktiviteter hver virksomhet gjør vises i Figur 6; dette antallet utgjør den rå poengsummen til hver virksomhet.



Figur 6: Totalt antall aktiviteter per virksomhet

De overordnede resultatene med gjennomsnittet av de tre modenhetsverdiene (konservativ, vektet og høyvann) beregnet per område presenteres i Figur 7, både tekstlig og i form av radardiagram. Til sammenligning presenteres også den gjennomsnittlige høyvannsmodenheten for virksomhetene i BSIMM-studien. Mer detaljer for hver praksis kan finnes i Figur 9, Figur 10 og Figur 11 i Vedlegg D.

Areas	Conservative Maturity DIFI	Weighted Maturity DIFI	Watermark Maturity DIFI	Watermark Maturity BSIMM
Strategy & Metrics	0,4	1,4	1,8	2,0
Policy & Compliance	1,6	3,5	2,6	2,3
Education & Guidance	0,5	1,5	1,8	1,5
Attack Models	0,5	0,6	1,0	1,3
Security Features and Design	0,8	2,7	2,4	2,0
Standards and Requirements	0,7	2,7	2,5	1,8
Design Review / Architecture Analysis	0,5	2,5	2,1	1,5
Code Review	0,6	1,9	2,1	1,4
Security Testing	0,8	1,9	2,4	1,7
Config. and Vulnerability Management	1,0	2,6	2,3	2,0
Software Environment	0,9	2,1	2,3	2,0
Penetration Testing	0,7	1,7	1,7	1,8



Figur 7: Overordnede resultater

I de neste underavsnittene skal vi går nærmere inn på resultatene i hver praksis. Vi innleder hvert underavsnitt med en overordnet beskrivelse av praksisen hentet fra BSIMM-rapporten [3].

3.1 Strategi og måling

Hovedmålet for praksisen Strategi og måling (*Strategy and Metrics*) er transparens av forventninger og ansvarlighet for resultater. Toppledelsen må klarlegge organisasjonsmessige forventninger for *Secure Software Development Lifecycle* (SSDL) slik at alle forstår viktigheten av initiativet. I tillegg må toppledelsen angi spesifikke mål for alle SSDL interessenter, og sørge for at spesifikke individer kan stilles til ansvar for at disse målene nås.

SM Nivå 1: **Få en felles forståelse av retning og strategi.** Ledere må sørge for at alle som er involvert i å lage, rulle ut, drifte og vedlikeholde programvare forstår de skrevne organisasjonsmessige programvaresikkerhetsmålene. Ledere må også forsikre seg om at organisasjonen som helhet forstår strategien for å oppnå disse målene. En felles strategisk forståelse er avgjørende for virkningsfull og effektiv gjennomføring av programmet.

SM Nivå 2: **Tilpass oppførsel med strategi, og sjekk etterlevelse.** Ledere må eksplisitt identifisere individer som skal stå ansvarlige for programvaresikkerhet risikohåndtering. Disse individene er igjen ansvarlige for vellykket gjennomføring av SSDL aktiviteter. SSDL-ledere må sørge for rask identifikasjon og modifikasjon av enhver SSDL oppførsel som resulterer i uakseptabel risiko. For å redusere uakseptabel risiko, må ledere identifisere og oppmuntre vekst av en programvaresikkerhets-satelitt.

SM Nivå 3: **Praktisér risikobasert porteføljeadministrasjon.** Applikasjonseier og programvaresikkerhetsgruppen (SSG) må orientere ledelsen om risikoen assosiert med hver applikasjon i porteføljen. SSG må reklamere for sine aktiviteter eksternt for å bygge støtte for sin tilnærming, og muliggjøre sikkerhet i økosystemet.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 3.

Tabell 3: Aktiviteter i Strategy & Metrics

Strategy & Metrics	SM 1.1	We publish our process for addressing software security; containing goals, roles, responsibilities and activities.	35%
	SM 1.2	We have a secure software evangelist role to promote software security internally.	50%
	SM 1.3	We educate our executives about the consequences of inadequate software security.	55%
	SM 1.4	We have *identified* gate locations in our secure software development process where we make go/no go decisions with respect to software security.	60%
	SM 2.2	We *enforce* the identified gate locations in our secure software development process where we make go/no go decisions with respect to software security, and track exceptions.	35%
	SM 1.6	We have a process of accepting security risk and documenting accountability. In this process we assign a responsible manager for signing off on the state of all software prior to release.	35%
	SM 2.1	The software security group publishes data internally on the state of software security within the organization.	10%
	SM 2.3	In addition to the software security group, we have also identified members of the development teams that have a special interest in software security, and have a process for involving them in the software security work.	55%
	SM 2.5	We have identified metrics that measure software security initiative progress and success.	0%
	SM 3.1	The software security group has a centralized tracking application to chart the progress of all software.	15%
	SM 3.2	The software security group advertises the software security initiative outside the organization (for example by writing articles, holding talks in conferences, etc).	15%

Strategi og måling er et av områdene med dårligst modenhet. Aktiviteten som gjøres av flest virksomheter er SM1.4 (*We have identified gate locations in our secure software development process where we make go/no go decisions with respect to software security*).

> *"Risikovurderinger gjøres knyttet til prosjekter, men ikke når det gjelder sikkerhet – de gjelder andre ting. Har gjort risikovurdering knyttet til sikkerhet overordnet for hele virksomheten."*
>
> (Sitat fra intervjuene)

Ingen av respondentene gjør SM2.5 (*We have identified metrics that measure software security initiative progress and success*).

Flere av respondentene kommenterte i intervjuet at de opplevde å ha hatt bedre kontroll på "*gate locations*" tidligere, men at dette nå var vanskeligere på grunn av bruk av smidig utviklingsmetodikk. Det er også interessant (men ikke overraskende) å merke seg at det er flere virksomheter som sier de har identifisert *gate locations* enn det er virksomheter som sier de faktisk bruker dem konsekvent. Det kan tolkes dithen at mange virksomheter har gode intensjoner mht. definisjon av *gate locations*, men at de ikke klarer å gjennomføre bruken av dem, enten på grunn av utviklingsmetodikken eller andre årsaker.

> *"Risikovurdering skjer på høyere nivå. Det er informasjonssikkerhets-seksjonen som gjør risikoanalyser, men disse er ofte på et høyt nivå og ikke så nyttige i utviklingen."*
>
> (Sitat fra intervjuene)

> *"Regimet fungerte bra på fossefalls-prosjekter, men det er vanskeligere med smidig metodikk."*

3.2 Etterlevelse av lover, regler og retningslinjer

Hovedmålet for praksisen "Etterlevelse av lover, regler og retningslinjer" (*Compliance and Policy*) er å gi veiledning til alle interessenter og sikre at SSDL-aktiviteter kan revideres. Ledelsesgodkjente veiledninger må være tilgjengelige for alle SSDL-interessenter, inkludert leverandører, for å kunne nå sikkerhetsmål og sikre etterlevelse av relevante lover og regler. Alle SSDL-aktiviteter må generere artefakter som er tilstrekkelig for å gjennomføre en revisjon i henhold til veiledningene.

CP Nivå 1: **Dokumentér og sy sammen juridiske og kontraktsmessige forutsetninger.** SSG må samarbeide med relevante grupper for å identifisere hvilke krav fra lover og regler det er nødvendig å oppfylle, sy disse sammen til en helhet, og sørge for at denne kunnskapen blir tilgjengelig for SSDL interessenter (stakeholders).

CP Nivå 2: **Tilpass intern praksis til lovpålagte krav og retningslinjer, støttet av ledelsen.** Ledelsen må åpent støtte SSG og initiativet for sikker programvareutvikling, inkludert behovet for å etterleve lover og regler. Ledere for risikostyring må eksplisitt ta ansvar for programvare-risiko. SSG og applikasjonseiere må sørge for at tjenesteavtaler (SLA) omfatter sikkerhetsegenskaper ved leverandørenes programvareleveranser.

CP Nivå 3: **Organisasjonsmessige trusler, angrep, defekter og operasjonelle forhold driver utvikling av retningslinjer og krav til leverandører.** Ledelsen må sørge for at retningslinjene for programvaresikkerhet oppdateres jevnlig basert på faktiske data, og må demonstrere virksomhetens løpende etterlevelse av lover og regler. SSG, applikasjonseiere og juridisk avdeling må sørge for at leverandører leverer programvare som tilfredsstiller relevante deler av virksomhetens retningslinjer.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 4.

Tabell 4: Aktiviteter i Etterlevelse av lover, regler og retningslinjer

Policy & Compliance	CP 1.1	The software security group has an overview of the regulations that our software has to comply with.	85%
	CP 1.3	We have a software security policy to meet regulatory needs and customer demands.	75%
	CP 1.2	The software security group is responsible for identifying all legislation related to personally identifiable information (for example personopplysningsloven).	80%
	CP 2.1	We have identified all the personally identifiable information stored by each of our systems and data repositories.	80%
	CP 2.2	All identified risks have to be mitigated or accepted by a responsible manager.	75%
	CP 2.3	We can demonstrate compliance with regulations that we have to comply with.	70%
	CP 2.4	We make sure that all vendor contracts are compatible with our software security policy.	65%
	CP 2.5	We promote executive awareness of compliance and privacy obligations.	75%
	CP 3.1	We have all the documentation necessary for demonstrating the organization's compliance with regulations we have to comply with (for ex. written policy, lists of controls, artifacts from software development).	35%
	CP 3.2	When managing our third party vendors, we impose our software security policies on them.	65%
	CP 3.3	Information from the secure software development process is routinely fed back into the policy creation process.	20%

Etterlevelse av lover og regler får best modenhetsverdi på alle skalaene; en stor prosentandel av svarene er positive. Alle aktivitetene på nivå 1 gjøres av minst 75% av virksomhetene, og mange virksomheter har også mange aktiviteter på nivå 2. Aktivitet PC 2.1 (*We have identified all the personally identifiable information stored by each of our systems and data repositories*) gjøres av 80% av virksomhetene.

> *"Vi har mange jurister som jobber hos oss, og vi som organisasjon har mye instrukser og policyer som gjør at vi dekker dette med compliance. Men er usikker på i hvor stor grad dette har konsekvenser for kodingen".*
>
> (Sitat fra intervjuene)

Den minst vanlige aktiviteten gjøres av kun 20% av virksomhetene: CP3.3 (*Information from the secure software development process is routinely fed back into the policy creation process*). Dette kan ha sammenheng med at flere respondenter har uttrykt at deres retningslinjer ikke sier noe konkret om sikker programvareutvikling.

3.3 Opplæring og øvelser

Hovedmålene for praksisen "Opplæring og øvelser" (*Training*) er å lage en kunnskapsrik arbeidsstokk og rette opp feil i prosesser. Arbeidsstokken må ha rolle-basert kunnskap som spesifikt inkluderer evnene som kreves for å utføre deres SSDL-aktiviteter på en hensiktsmessig måte. Opplæringen må omfatte spesifikk informasjon om rotårsaker til feil som oppdages i prosessaktiviteter og resultater.

T Nivå 1: **Gjør tilpasset, rollebasert opplæring tilgjengelig ved behov.** SSG må bygge opp interessen for programvaresikkerhet i hele organisasjonen, og tilby rollespesifikt opplæringsmateriale, inkludert datamaskinbasert opplæring, som bygger inn erfaringer fra faktiske interne hendelser.

T Nivå 2: **Opprett programvaresikkerhetssatellitten.** SSG må bygge opp og forbedre en satellitt gjennom sosiale aktiviteter, inkludert opplæring og relaterte arrangementer. SSG og ledere må sørge for at nyansatte eksponeres for organisasjonens sikkerhetskultur som en del av velkomstprogrammet.

T Nivå 3: **Belønn ferdigheter og skap en karrierevei.** Bygg moral. Ledelsen og SSG må sørge for at alle ansatte får tilstrekkelig anerkjennelse for å fullføre opplæringsløpet. Ledere, applikasjonseiere og SSG må tilby opplæring til leverandører og innleide utviklere som en måte å spre sikkerhetskulturen. Ledere og SSG må fortsette å bygge opp under satellittens moment ved å markedsføre sikkerhetskulturen eksternt. SSG må være tilgjengelig, i hvert fall i perioder, for de som ønsker programvaresikkerhetsbistand og veiledning. Ledelsen må sørge for at alle ansatte mottar slik opplæring minst på en årlig basis.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 5.

Tabell 5: Aktiviteter i Opplæring og øvelser

Education & Guidance	T 1.1	We have a security awareness training program.	50 %
	T 1.5	We offer role-specific security courses (for example on specific tools, technology stacks, bug parade).	15 %
	T 1.6	The security awareness training content/material is tailored to our history of security incidents.	35 %
	T 1.7	We deliver on-demand individual security training.	45 %
	T 2.5	We encourage security learning outside of the software security group by offering specific training and events.	50 %
	T 2.6	We provide security training for new employees to enhance the security culture.	60 %
	T 2.7	We use the security training to identify individuals that have a particular interest in security.	15 %
	T 3.1	We have a reward system for encouraging learning about security.	10 %
	T 3.2	We provide security training for vendors and/or outsourced workers.	20 %
	T 3.3	We host external software security events.	10 %
	T 3.4	We require an annual software security refresher course.	0 %
	T 3.5	The software security group has defined office hours for helping the rest of the organization.	10 %

> *"Alle som begynner hos oss må gjennom obligatorisk innføring i sikkerhet, samt underskrive sikkerhetsinstruks. Men er ikke noe om programvaresikkerhet her. Siden utviklerne er innleide er det ingen av de som må gjennom dette opplegget."*
> (Sitat fra intervjuene)

Det er overraskende lave tall for opplæring, f.eks. er det bare 15% av virksomhetene som gjør aktivitet T1.5 (*Deliver role-specific advanced curriculum (tools, technology stacks, bug parade)*).

Den hyppigste aktiviteten innen opplæring er T2.6 (*We provide security training for new employees to enhance the security culture*), som gjøres av 60% av virksomhetene. Det er grunn til å tro de fleste som svarer positivt på dette tar med generell sikkerhetsopplæring av nyansatte, og at det er vesentlig færre (om noen) som tilbyr opplæring spesifikt innen programvaresikkerhet, slik også svarene fra T1.5 tyder på. En respondent spurte oss om vi tilbød kurs innen dette.

Ingen virksomheter gjør aktivitet T3.4 (*We require an annual software security refresher course*); dette befester inntrykket av at opplæringen som tilbys er innen generell sikkerhetsbevissthet snarere enn programvaresikkerhet. Flere respondenter uttalte at det typisk er de sikkerhetsansvarlige som sendes på slike kurs, ikke utviklerne.

> *"Vil ikke kalle det et program. Har ikke kjempegod struktur, men er mer stuntbasert."*
> (Sitat fra intervjuene)

3.4 Angrepsmodeller

Hovedmålet for praksisen "Angrepsmodeller" (*Attack Models*) er å etablere tilpasset kunnskap om angrep som er relevant for virksomheten. Tilpasset kunnskap må styre beslutninger både om kode og kontrollmekanismer.

AM Nivå 1: **Lag kunnskapsbase med angrep og dataverdier.** SSG må identifisere potensielle angripere og dokumentere både angrepene som representerer den største bekymringen for virksomheten, og eventuelle angrep som allerede har funnet sted. SSG må kommunisere angrepsinformasjonen til alle interessenter. Forretningsdelen av virksomheten må utforme et graderingssystem for data som SSG kan bruke for å katalogisere og prioritere applikasjoner.

AM Nivå 2: **Spre informasjon om angripere og relevante angrep.** SSG må samle informasjon om angrep og utvide sin kunnskap til å omfatte både høynivå angrepsmønstre og lavnivå "*abuse cases*". Angrepsmønstre må inkludere teknologispesifikk informasjon som er relevant for virksomheten.

AM Nivå 3: **Forske på nye angrepsmønster og mottiltak.** SSG må forske på angrep på virksomhetens programvare for å ligge på forskudd i forhold til angrepsaktivitet. SSG må tilby kunnskap og automatisering til revisorer og testere for å sikre at deres aktiviteter gjenspeiler faktiske angrep som utføres mot virksomhetens programvare, i tillegg til potensielle angrep.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 6.

Tabell 6: Aktiviteter i Angrepsmodeller

Attack Models	AM 1.1	We build and maintain a top N possible attacks list.	25 %
	AM 1.2	We have a data classification scheme and an inventory of attacks so we can prioritize applications by the data handled by them.	25 %
	AM 1.3	We maintain a list of likely attacker profiles.	25 %
	AM 1.4	We collect and publish attack stories.	20 %
	AM 1.5	The software security group keeps up to date by learning about new types of attacks / vulnerabilities.	80 %
	AM 1.6	We have an internal forum to discuss attacks.	55 %
	AM 2.1	We link abuse cases to each attacker profile.	0 %
	AM 2.2	We have a list of technology-specific abuse cases.	5 %
	AM 3.1	We have an engineering team that develops new attack methods.	5 %
	AM 3.2	We have automated the attack methods developed by our engineers.	5 %

"Har et forum for å diskutere angrep i IKT-drift, men er usikker på om ting kommer videre derfra til utvikling og forvaltning."

(Sitat fra intervjuene)

Angrepsmodeller er generelt en praksis med lav modenhet. Virksomhetene gjør i gjennomsnitt bare 25% av aktivitetene på nivå 1, og kun AM1.5 (*The software security group keeps up to date by learning about new types of attacks / vulnerabilities*) gjøres av 80% av virksomhetene. En faktor som kan bidra til dette relativt høye tallet er at mange av virksomhetene har en person som er definert som sikkerhetsansvarlig (inkludert utvikling), og denne personen er gjerne den som deltar på sikkerhetskurs og seminarer. 55% sier de gjør AM 1.6 (*Build an internal forum to discuss attacks*), men her tyder intervjuene på at mange har en veldig bred definisjon på hva et slikt forum kan være. Flere virksomheter refererer til OWASP Top Ten [9] i forbindelse med AM1.1, men det bør bemerkes at dette ikke er like bra som å lage en egen, organisasjonsspesifikk angrepsliste.

"Vet ikke hva utviklere følger med på, men mange følger med på software-komponenter de bruker. Får noen ganger krav fra utviklere om å få patchet komponenter de bruker."

(Sitat fra intervjuene)

Ingen virksomheter gjør aktiviteten AM2.1 (*We link abuse cases to each attacker profile*), noe som kan skyldes at de færreste respondentene hadde et bevisst forhold til abuse cases, som er en grafisk eller tekstlig fremstilling av hvordan en angriper kan går fram for å angripe et system eller tjeneste.

3.5 Sikkerhetsfunksjonalitet og design

Hovedmålet for praksisen "Sikkerhetsfunksjonalitet og design" (*Security Features and Design*) er etablering av tilpasset kunnskap om sikkerhetsegenskaper, rammeverk og mønster. Den tilpassede kunnskapen må drive arkitektur- og komponent-beslutninger.

SFD Nivå 1: **Publisér sikkerhetsegenskaper og arkitektur.** SSG må tilby arkitekter og utviklere veiledning på sikkerhetsegenskaper og sikkerhetsmekanismer, og bidra direkte til arkitekturgrupper.

SFD Nivå 2: **Bygg og identifisér sikkerhetsløsninger.** SSG må tilby sikker-ved-design (*secure by design*) rammeverk, og må være tilgjengelig for og kapabel til å løse designproblemer for andre.

SFD Nivå 3 **Aktivt gjenbruk av godkjente sikkerhetsegenskaper og sikkerhetsmekanismer og sikker-ved-design rammeverk.** SSG må tilby flere modne design-mønstre (*design patterns*) tatt fra eksisterende programvare og teknologistakker (*technology stacks*). Ledere må sørge for at det er formell konsensus i hele virksomheten når det gjelder sikre designvalg. Ledere må også kreve at definerte sikkerhetsmekanismer og rammeverk brukes til enhver tid der dette er mulig.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 7.

Tabell 7: Aktiviteter i Sikkerhetsfunksjonalitet og design

Security Features and Design	SFD 1.1	Our software security group builds and publishes a library of security features.	15 %
	SFD 1.2	Security is a regular part of our organization's software architecture discussion.	80 %
	SFD 2.1	The software security group facilitates the use of secure-by-design middleware frameworks/common libraries.	55 %
	SFD 2.2	The software security group is directly involved in the design of security solutions.	70 %
	SFD 3.1	We have a review board to approve and maintain secure design patterns.	20 %
	SFD 3.2	We require the use of approved security features and frameworks.	60 %
	SFD 3.3	We find and publish mature design patterns from the organization.	15 %

Det er overraskende at bare 15% av virksomhetene sier at de gjør SFD1.1 (*Our software security group builds and publishes a library of security features*), mens 80% påstår at de oppfyller SFD 1.2 (*Security is a regular part of our organization's software architecture discussion*). En forklaring på dette avviket kan være at mange virksomheter benytter seg av standard sikkerhetsfunksjonalitet fra ID-porten, men dette er ikke noe de utvikler og vedlikeholder selv. Det er imidlertid viktig å minne om at sikkerhetsfunksjonalitet omfatter mye mer enn bare autentisering, og at vi kunne forventet at de fleste virksomhetene har behov for sikkerhetsfunksjonalitet utover det som ID-porten tilbyr.

> *"I flere prosjekter er det sikkerhetskrav med fra starten. Der har vi blitt bedre. IT-sikkerhetsleder kan da være med og stille krav. Det varierer fra prosjekt til prosjekt om sikkerhet tas med. Det er mer vanlig at sikkerhet er med om det er nyutvikling enn om det er videreutvikling."*
>
> (Sitat fra intervjuene)

3.6 Standarder og krav

Hovedmålet for praksisen "Standarder og krav" (*Standards and Requirements*) er å lage retningslinjer for alle interessenter. Ledere og SSG må dokumentere programvaresikkerhetsvalg, og formidle dette materialet til alle som er involvert i SSDL, inkludert eksterne aktører.

SR Nivå 1: **Gjør sikkerhetsstandarder og krav lett tilgjengelige.** SSG må gjøre grunnleggende kunnskap tilgjengelig, i det minste inkludert sikkerhetsstandarder, sikre kodestandarder og krav for etterlevelse av relevante lover og regler. Ledere må sørge for at programvaresikkerhetsinformasjon holdes oppdatert og gjøres tilgjengelig for alle.

SR Nivå 2: **Kommuniser formelt godkjente standarder internt og til leverandører.** Ledere må sørge for at det er en formell prosess som brukes til å lage standarder som er spesifikk til teknologistakker. Ledere, SSG og produkteiere må sørge for at SLAer er forsterket med kontraktsspråk som er godkjent av egne jurister. SSG må sørge for at all åpen kildekode er identifisert i virksomhetens kode.

SR Nivå 3: **Krev risikostyringsavgjørelser for bruk av åpen kildekode.** Ledere og SSG må demonstrere at åpen kildekode som brukes i virksomheten er gjenstand for de samme risikostyringsprosessene som kode som lages internt, og sørge for at alle relevante standarder kommuniseres til tredjepartsleverandører.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 8.

Tabell 8: Aktiviteter i Standarder og krav

Standards and Requirements	SR 1.1	The software security group create standards that explain the accepted way to carry out specific security centric operations.	40 %
	SR 1.2	We have a portal where all security related documents are easily accessible.	55 %
	SR 1.3	The software security group assists the software development team in translating compliance constraints (for instance from legislation) into application specific security requirements.	55 %
	SR 1.4	We use secure coding standards in our software development.	60 %
	SR 2.2	We have a standards review board to formalize the process used to develop security standards.	15 %
	SR 2.3	We use a limited number of standard technology stacks.	80 %
	SR 2.5	We have a template SLA text for use in contracts with vendors and providers, to help prevent compliance and privacy problems.	50 %
	SR 3.2	We have procedures to communicate and promote our security standards to vendors.	40 %
	SR 2.4	We have a list of all open source components used in our software.	40 %
	SR 3.1	We manage the risks related to using open source components.	40 %

Rundt halvparten av virksomhetene viser tegn til generelt høy grad av modenhet innenfor denne praksisen. Det er en mulig sammenheng med lover og regler, ettersom offentlige virksomheter gjerne er vant til å forholde seg til krav fra myndighetene. 80% sier at de tilfredsstiller SR 2.3 (*We use a limited number of standard technology stacks*); i praksis betyr dette oftest at virksomheten har standardisert på Microsoft programvareprodukter.

> *"Vi har standardisert på Microsoft platform og .net."*
>
> (Sitat fra intervjuene)

3.7 Arkitekturanalyse

Hovedmålet for praksisen "Arkitekturanalyse" (*Architecture Analysis*) er kvalitetskontroll. De som utfører arkitekturanalyse må sørge for at strukturelle sikkerhetsfeil oppdages og korrigeres. Programvarearkitekter må sørge for at standarder følges, og at godkjente sikkerhetsløsninger gjenbrukes.

AA Nivå 1 **Utfør risikodrevne AA gjennomganger, ledet av SSG.** Virksomheten må gjøre en lettvekts risikoklassifiseringsskjema for programvare tilgjengelig. SSG må begynne å lede arkitekturanalyseaktiviteter, spesielt i forbindelse med høyrisikoapplikasjoner, som en måte å bygge opp kompetanse internt og demonstrere verdien på designnivå.

AA Nivå 2 **Spre bruk av dokumentert AA-prosess.** SSG må legge til rette for bruk av arkitekturanalyse gjennom hele virksomheten ved å gjøre seg selv tilgjengelig som en ressurs og mentor. SSG må definere en arkitekturanalyseprosess basert på et felles arkitekturbeskrivelsesspråk og standard angrepsmodeller.

AA Nivå 3 **Bygg opp kunnskap om revidering og forbedring av sikkerhetsfeil i arkitekturgruppen.** Programvarearkitekter må lede analyseaktiviteter på tvers av virksomheten, og må bruke analyseresultater for å lage og oppdatere arkitekturmønstre som er sikre.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 9.

Tabell 9: Aktiviteter i Arkitekturanalyse

Design Review / Architecture Analysis	AA 1.1	We perform security feature review.	40 %
	AA 1.2	We perform design review for high-risk applications.	60 %
	AA 1.3	We have a software security group that leads review efforts.	20 %
	AA 1.4	We use a risk questionnaire to rank applications in terms of the risk they are exposed to.	25 %
	AA 2.1	We have a defined process to do architecture analysis.	45 %
	AA 2.2	We have a standardized format for describing architecture that also covers data flow.	35 %
	AA 2.3	The software security group is available to support architecture analysis when needed.	50 %
	AA 3.1	The software architects lead design review efforts to detect and correct security flaws.	45 %
	AA 3.2	Failures identified during architecture analysis are used to update the standard architecture patterns.	35 %

Generelt virker det som om det er lav modenhet innen designgjennomgang og arkitekturanalyse. Den hyppigste aktiviteten (60%) var AA1.2 (*We perform design review for high-risk applications*). Flere respondenter virket usikre på hva det innebærer å gjøre en designgjennomgang eller en arkitekturanalyse, og flere kommenterte at det var mindre grad av rent arkitekturarbeid etter at de hadde begynt med smidig utvikling.

"Arkitektur involverer ofte sikkerhetsarkitekter når de lager arkitekturen, men de kan i virksomheten bli flinkere til å sjekke at sikkerhetsarkitekter er involvert. Nå er det prosjektet som bestiller ressurser, f.eks. en sikkerhetsarkitekt. Det er vanlig at sikkerhetsarkitekter er med når det er åpenbart sikkerhets-ting, men dette kan falle gjennom om fokus er på funksjonaliteten."

(Sitat fra intervjuene)

3.8 Kodegjennomgang

Hovedmålet for praksisen "Kodegjennomgang" (Code Review) er kvalitetskontroll. De som utfører kodegjennomgang må sørge for at sikkerhetsfeil oppdages og korrigeres. SSG må sørge for at standarder blir fulgt og at godkjente sikkerhetsmekanismer gjenbrukes.

CR Nivå 1: **Bruk manuell og automatisert kodegjennomgang med sentralisert rapportering.** SSG må gjøre seg tilgjengelig ovenfor andre for å øke bevisstheten om og etterspørsel etter kodegjennomgang. SSG må utføre kodegjennomgang på høyrisiko-applikasjoner når den kan bli involvert i prosessen, og må bruke kunnskapen som tilegnes til å informere virksomheten om typen feil som oppdages. Ledelsen må gjøre kodegjennomgang obligatorisk for alle programvareprosjekter. SSG må tvinge gjennom bruk av sentraliserte rapporteringsverktøy for å samle inn kunnskap om tilbakevendende feil, og kanalisere denne informasjonen inn i strategi og opplæring.

CR Nivå 2: **Følg opp standarder via kodegjennomgangsprosessen.** SSG må lede utvikler-oppførsel ved å følge opp kodestandarder vha. automatiserte verktøy og verktøy-mentorer. SSG må kombinere automatiserte vurderingsteknikker med skreddersydde regler for å effektivt finne problemer.

CR Nivå 3: **Bygg en automatisert kodegjennomgangsfabrikk med skreddersydde regler.** SSG må bygge opp en evne til å finne og luke ut spesifikke feil i hele kodebasen.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 10.

Tabell 10: Aktiviteter i Kodegjennomgang

Code Review	CR 1.1	We create a list with top N software security defects list.	20 %
	CR 1.2	The software security group does ad-hoc code reviews.	20 %
	CR 1.4	We use automated tools (such as static analysis) along with manual review to detect software security defects.	50 %
	CR 1.5	We make code review mandatory for all projects before release.	45 %
	CR 1.6	The software security defects found during code review are tracked in a centralized repository.	45 %
	CR 2.2	We enforce coding standards to improve software security.	60 %
	CR 2.5	We have mentors for code review tools for making most efficient use of the tools.	35 %
	CR 2.6	We use automated tools with tailored rules to improve efficiency and reduce false positives.	40 %
	CR 3.2	We combine assessment results so that multiple analysis techniques feed into one reporting and remediation process.	15 %
	CR 3.3	When a software defect is found we have tools to search for that defect also in the whole codebase.	30 %
	CR 3.4	We perform automated code review on all code to detect malicious code.	20 %

Kodegjennomgang var også en praksis med gjennomgående lav modenhet. Det er mange som nevner i intervjuene at utviklerne sjekker hverandres kode. Dette er jo en god ting, men trekkes noe ned av at inntrykket er at det er liten kunnskap om programvare-sikkerhet hos utviklerne. Det er derfor grunn til å tro at det som gjøres av kodegjennomgang bare i liten grad kan sies å være spesifikt på sikkerhet.

> *"Det dukker av og til opp feil, og da blir dette tatt opp med utviklerne, men vet ikke hva utviklerne gjør med det."*
> (Sitat fra intervjuene)

Den største andelen positive svar var 60% for CR2.2 (*We enforce coding standards to improve software security*). Det er imidlertid ingenting som tyder på at kodestandardene er spesifikt innrettet på programvaresikkerhet, og det er grunn til å anta at sikkerhetsgevinsten mer kommer som en sideeffekt av å gjøre ting på en enhetlig måte. Vi kan derfor gå ut fra at det er flere gevinster å hente ut på å sørge for at

kodestandarder også omfatter programvaresikkerhet.

3.9 Sikkerhetstesting

Hovedmålet for praksisen "Sikkerhetstesting" (*Security Testing*) er kvalitetskontroll utført i løpet av utviklingssyklusen. De som utfører sikkerhetstesting må sørge for at sikkerhetsfeil oppdages og korrigeres. SSG må sørge for at standarder blir fulgt og at godkjente sikkerhetsmekanismer gjenbrukes.

ST Nivå 1: **Forbedre QA utover det funksjonelle perspektivet.** QA må omfatte funksjonell kant og grenseverdi-vilkår testing i testregimet. Testsamlinger må også omfatte funksjonell sikkerhetstesting.

ST Nivå 2: **Integrer angriperens perspektiv i test planer.** QA må integrere "black-box" sikkerhetstestverktøy i prosessen. QA må bygge testsamlinger for funksjonelle sikkerhetsegenskaper. SSG må dele sin sikkerhetskunnskap og testresultater med QA.

ST Nivå 3: **Lever risiko-basert sikkerhetstesting.** QA må inkludere sikkerhetstesting i automatiserte regresjonssamlinger. SSG må sørge for denne sikkerhetstestingen, og dybden må veiledes av kunnskap om kodebasen og den assosierte risikoen, i tillegg til aggressiv testing som simulerer angriperens perspektiv.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 11.

Tabell 11: Aktiviteter i Sikkerhetstesting

Security Testing	ST 1.1	We perform adversarial tests with edge and boundary values.	45 %
	ST 1.3	We create our tests based on existing security requirements and security features.	60 %
	ST 2.1	We integrate black box security tools into the testing process (including protocol fuzzing).	10 %
	ST 2.4	We share security test results with QA.	55 %
	ST 3.1	We include security tests in QA automation.	20 %
	ST 3.2	We perform fuzz testing customized to application APIs.	15 %
	ST 3.3	We base the security tests on the security risks analysis.	55 %
	ST 3.4	We use code coverage tools to ensure that security tests cover all parts of the code.	10 %
	ST 3.5	We write tests cases based on abuse cases provided by the software security group.	20 %

Også for sikkerhetstesting observerer vi et generelt lavt modenhetsnivå. Flest virksomheter (60%) svarte ja på ST1.3 (*We create our tests based on existing security requirements and security features*), mens bare 10% gjør ST2.1 (*Integrate black box security tools into the QA process*). Flere respondenter indikerte at testing og QA er noe som utviklerne selv gjør, men at fokuset er på funksjonell testing, ikke sikkerhetstesting. Det er derfor nærliggende å tenke at funksjonelle sikkerhetskrav oppfattes som en kurant ting å teste, mens ren sikkerhetstesting oppleves som mer perifert.

> *"Vi har ikke egne, spesifikke tester for sikkerhet. [...] Kvalitetssikringstestere utfører ikke sikkerhetstester."*
>
> (Sitat fra intervjuene)

3.10 Konfigurasjonsstyring og sårbarhetsstyring

Hovedmålet med praksisen Konfigurasjonsstyring og sårbarhetsstyring (*Configuration Management and Vulnerability Management*) er endringsledelse. SSG og applikasjonseiere må holde styr på autoriserte endringer, og avdekke og/eller forhindre uautoriserte endringer. Applikasjonseiere må sørge for at virksomhetens retningslinjer blir fulgt.

CMVM Nivå 1 **Få det som observeres i driften til å drive utviklingen.** SSG må støtte opp om hendelseshåndtering. SSG må bruke data fra driften til å foreslå endringer i SSDL og utvikleroppførsel.

CMVM Nivå 2 **Sørg for at krisehåndtering er tilgjengelig når applikasjoner er under angrep.** Ledere og SSG må understøtte krisehåndtering ved pågående angrep på applikasjoner. Ledere og SSG må opprette og vedlikeholde en oversikt over all kode. SSG bruker data fra driften til å styre evolusjon av SSDL og utvikleroppførsel.

CMVM Nivå 3 **Lag en tett tilbakekoblingssløyfe mellom drift og utvikling.** SSG må sørge for at SSDL både adresserer kodefeil som oppdages i driften, og inkluderer forbedringer som eliminerer assosierte rotårsaker.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 12.

Tabell 12: Aktiviteter i Konfigurasjonsstyring og sårbarhetsstyring

Configuration and Vulnerability Management	CMVM 1.1	The software security group has procedures for incident response, in collaboration with the incident response team (if it exists).	65 %
	CMVM 2.1	We are able to make quick changes in the software when under attack.	85 %
	CMVM 3.3	We perform drills to ensure that incident response capabilities minimize the impact of an attack.	25 %
	CMVM 1.2	We identify software defects found in operations (for ex. by intrusion detection systems) and feed back to development.	55 %
	CMVM 2.2	We track software defects found during operations until they are closed.	85 %
	CMVM 2.3	We maintain a matrix of all installed applications in order to identify all places that need to be updated when a piece of code needs to be changed.	45 %
	CMVM 3.1	When a software defect is found in a piece of code during operations we have a process to search for that defect also in the whole codebase.	25 %
	CMVM 3.2	We do software security process improvement based on the analysis of cause of software defects found in operations.	25 %
	CMVM 3.4	We have a system for paying rewards to individuals who report security flaws in our software.	0 %

Mer enn halvparten av virksomhetene gjør aktivitetene på nivå 1 i denne praksisen, som omfatter CMVM1.1 (*The software security group has procedures for incident response, in collaboration with the incident response team (if it exists)*) og CMVM 1.2 (*We identify software defects found in operations (for ex. by intrusion detection systems) and feed back to development*). 85% av virksomhetene gjør to aktiviteter på nivå 2; CMVM 2.1 (*Have emergency codebase response*) og CMVM 2.2 (*Track software bugs found during ops through the fix process*). Sistnevnte er åpenbart et tolkningsspørsmål, ettersom de fleste hevdet å ha et bra system for å spore feil, men samtidig ikke hadde konfigurert systemet slik at sikkerhetsfeil kunne spores spesielt. Basert på intervjuene kan det virke som om virksomhetene er overoptimistiske når det gjelder CMVM 1.1, ettersom de typisk refererer til at de har meget kompetente sikkerhetsadministratorer som håndterer hendelser på en bra måte. Det er imidlertid ikke like klart hvor tett koblingen mellom disse sikkerhetsadministratorene og utviklerne er.

> *"Om en feil skulle oppdages trekker vi inn de som kjenner produksjons-systemet. Siden de har bygget det selv vet de hvor komponenten er i bruk. Dette er kunnskap som ligger i hodene til folk."*
>
> (Sitat fra intervjuene)

Ingen virksomheter gjør CMVM 3.4 (*Operate a bug bounty program*), men dette er ganske naturlig ettersom virksomhetene primært utvikler programvare til eget bruk.

3.11 Programvaremiljø

Hovedmålet for praksisen "Programvaremiljø" (*Software Environment*) er endringsledelse. De som er ansvarlige for programvaremiljøet må sørge for at de er i stand til å gjøre autoriserte endringer, og avdekke uautoriserte endringer. Ledere må sørge for at virksomhetens retningslinjer blir fulgt.

SE Nivå 1: **Sørg for at programvaremiljøet støtter programvaresikkerhet.** Driftsgruppen sørger for de nødvendige sikkerhetsmekanismene for datamaskiner og nettverk fungerer, og overvåker programvare proaktivt, inkludert input til applikasjoner.

SE Nivå 2: **Bruk publiserte installasjonsveiledninger og kodesignering.** SSG må sørge for at programvareutviklingsprosesser beskytter integriteten til koden. SSG må sørge for at veiledninger for installasjon og vedlikehold av applikasjoner er utarbeidet for bruk av driftsgruppen.

SE Nivå 3: **Beskytt klient-side kode og overvåk oppførselen til programvare aktivt.** SSG må sørge for at all kode som forlater virksomheten er beskyttet. Driftsgruppen må overvåke oppførselen til programvaren.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 13.

Tabell 13: Aktiviteter i Programvaremiljø

Software Environment	SE 1.1	We monitor the input to software we run in order to spot attacks on our software.	40 %
	SE 1.2	We use accepted good practice mechanisms for host/network security.	90 %
	SE 2.2	The software security group creates and publishes installation guides to ensure that our software is configured securely.	45 %
	SE 2.4	We create digital signatures for all binaries that we deliver.	10 %
	SE 3.2	We use code protection such as obfuscation to make reverse engineering harder.	0 %
	SE 3.3	We monitor the behavior of our software looking for misbehavior and signs of attacks.	60 %

Virksomhetene har god tiltro til eget sikkerhetsnivå når det gjelder nettverk og datamaskiner; 90% bekrefter SE 1.2 (*We use accepted good practice mechanisms for host/network security*). Dette tallet virker kanskje høyt sammenlignet med andre aktiviteter, men det er viktig å huske på at nettverkssikkerhet generelt er mye modnere enn programvaresikkerhet. Videre er det slik at nettverkssikkerhet og systemsikkerhet ivaretas av systemansvarlige, og i mye mindre grad av den enkelte utvikler. Tallene passer også godt med den offisielle BSIMM-studien [1].

Ingen av virksomhetene gjør aktivitet SE 3.2 (*Use code protection*), men dette er helt som forventet ettersom ingen av virksomhetene selger ferdig programvare til tredjeparter. Denne aktiviteten gir kun mening for uavhengige programvareselskaper.

3.12 Penetreringstesting

Hovedmålet med praksisen "Penetreringstesting" (*Penetration Testing*) er kvalitetskontroll av programvare som er i ferd med å bli rullet ut. De som utfører penetreringstesting må sørge for at sikkerhetsdefekter oppdages og korrigeres. SSG må sørge for at standarder blir fulgt, og at godkjente sikkerhetsmekanismer gjenbrukes.

PT Nivå 1: **Oppdater kode etter penetreringstestingsresultater.** Ledere og SSG må initiere penetreringstestingsprosessen, med interne eller eksterne ressurser. Ledere og SSG må sørge for at oppdagede feil adresseres, og at alle er gjort kjent med fremdriften.

PT Nivå 2: **Planlegg jevnlig penetreringstesting av informerte penetreringstestere.** SSG må legge til rette for en penetreringstestingsaktivitet som periodisk utføres på alle applikasjoner. SSG må dele sin sikkerhetskunnskap og testresultater med alle penetreringstestere.

PT Nivå 3: **Utfør dypdykk-penetreringstesting.** Ledere må sørge for at virksomhetens penetreringstestingskunnskap holder følge med fremskrittene som angriperne gjør. SSG må dra nytte av organisasjonsmessig kunnskap for å skreddersy penetreringstestings-verktøy.

Spørsmålene som hver virksomhet besvarte er illustrert i Tabell 14.

Tabell 14: Aktiviteter i Penetreringstesting

Penetration Testing	PT 1.1	We use external penetration testers on our software.	60 %
	PT 1.2	Defects found in penetration testing are inserted in our bug tracking system and flagged as security defects.	55 %
	PT 1.3	We use penetration testing tools internally.	45 %
	PT 2.2	The penetration testers have access to all available information about our software (for example: the source code, design documents , architecture analysis results and code review results).	40 %
	PT 2.3	We periodically perform penetration tests on all our software.	10 %
	PT 3.1	We use external penetration testers to do deep-dive analysis for critical projects to complement internal competence.	35 %
	PT 3.2	The software security group has created customized penetration testing tools and scripts for our organization.	5 %

Minst halvparten av virksomhetene gjør de to første aktivitetene på nivå 1, PT1.1 (*We use external penetration testers on our software*) og PT 1.2 (*Defects found in penetration testing are inserted in our bug tracking system and flagged as security defects*), og 45% gjør den siste aktiviteten på nivå 1, PT 1.3 (*Use penetration testing tools internally*). Spesielt for PT1.1 er tallet betydelig lavere enn i den offisielle BSIMM-studien, men det er egentlig minst like interessant å se på den relativt lave andelen som gjør PT1.3. Her igjen fester inntrykket seg at penetreringstesting er noe som (de innleide) sikkerhetsfolkene holder på med; bare i liten grad kjører utviklerne penetreringstesting internt på egen kode. Dette harmonerer med resultatene fra 3.9 (Sikkerhetstesting), som viser at selv om utviklerne er vant til å teste hverandres kode, er det lite fokus på sikkerhetstesting.

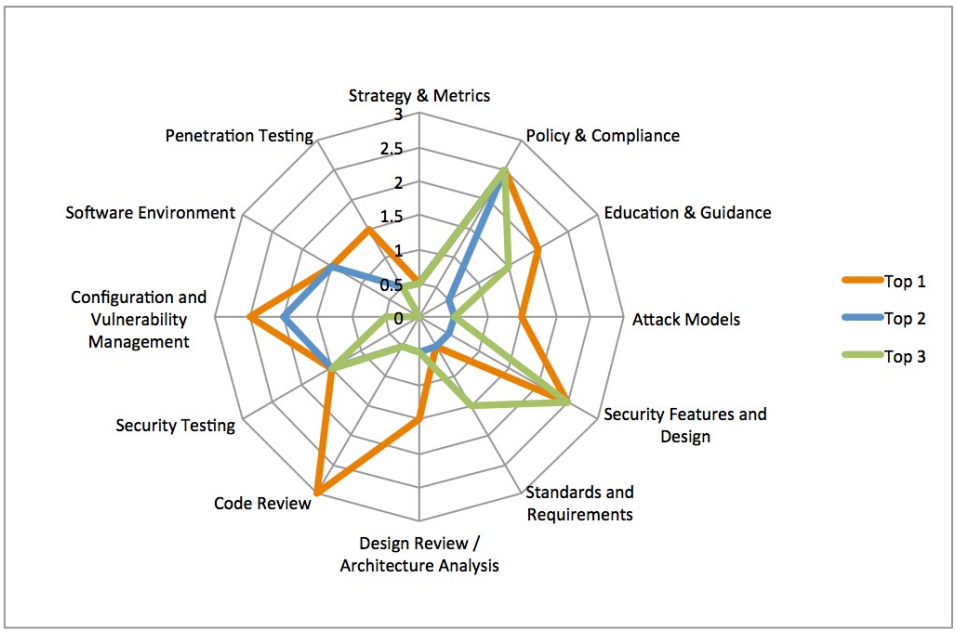
> *"Initiativer til å gjøre penetrasjons-testing kommer ikke fra utviklersiden men fra nettverkssiden. Da gjøres det ikke testing spesielt av egenutviklet kode, eller på prosjekter, men bredere."*
>
> (Sitat fra intervjuene)

4 Diskusjon

I dette kapitlet diskuteres resultatene i studien med fokus på følgende tema: ulikheter blant virksomhetene som deltok i studien, områder med høy og lav modenhet, videre aktiviteter som virksomhetene ser for seg, og gyldighet av resultatene fra studien.

4.1 Virksomhetene i studien

Det er stor variasjon i modenhet hos de ulikevirksomhetene i studien. Den beste virksomheten har implementert 87 av de 112 aktivitetene, og den minst modne har kun implementert 9. Gjennomsnittet av de 20 virksomhetene er omtrent 44 av 112. Hvis vi ser på de tre mest modne virksomhetene som illustrert i Figur 8, er det konservative modenhetsnivået konsistent med de andre virksomhetene med hensyn til at "Strategi og måling" og "Angrepsmodeller" viser ganske lav modenhet, mens det er maksimal variasjon i praksisen "Kodegjennomgang"; fra 0 til 3.



Figur 8: Konservativt modenhetsnivå for de tre mest modne virksomhetene

Noen virksomheter har liten oversikt over hva deres leverandører gjør av aktiviteter knyttet til informasjons-sikkerhet.

Virksomhetene som har deltatt i modenhetsstudien har ulike karakteristikker, både når det gjelder mengden av utvikling de deltar i, og i hvor stor grad de gjør utvikling selv. Dette påvirker hvordan de jobber med sikkerhet, og også i hvor stor grad de har oversikt over programvaresikkerhetsaktiviteter. Noen av respondentene hadde fått tilbakemeldinger fra sine leverandører på spørreskjemaet. Andre hadde liten oversikt over hva leverandørene gjorde av aktiviteter knyttet til informasjonssikkerhet. Usikkerheten gjelder både opplæringstiltak knyttet til innleide utviklere, samt hvilke aktiviteter som gjøres i utviklingen. Ut fra svarene kan vi ikke si at det er større forskjeller i modenhet grunnet i hvordan utviklingen er organisert. Det er tydelig at noen av aktørene som har deltatt i studien har relativt høy kompetanse, ressurser og erfaring knyttet til programvareutvikling generelt, mens andre har mindre erfaring knyttet til dette. Samtidig finnes det mange gode praksiser også blant de som ikke har et stort antall utviklere. De to virksomhetene som har høyest score (vektet modenhet) har 10-20 utviklere totalt (internt og innleid).

BSIMM-rammeverket er basert på ideen om at det finnes en formelt definert programvaresikkerhetsgruppe (SSG), og aktivitetene er sentrert rundt denne gruppen. De færreste av virksomhetene har en slik formelt definert gruppe. Flere virksomheter har en leder med mer eller mindre uttalt ansvar for programvare-sikkerhet, men da oftest som en del av et overordnet sikkerhetsansvar i virksomheten.

4.2 Områder og praksiser med høy grad av modenhet

Blant de undersøkte virksomhetene fant vi høyest grad av modenhet innenfor området "Retningslinjer og etterlevelse av lover og regler" (*Policy and Compliance*); mer enn 80% av de spurte svarte ja på mesteparten av aktivitetene på dette området. Resultatet er ikke overraskende, ettersom det dreier seg om offentlige virksomheter som er vant til å forholde seg til regler og krav fra myndighetene. Det virker som det er bedre etterlevelse av denne praksisen blant norske offentlige virksomheter enn gjennomsnittet fra den offisielle BSIMM-studien. Samtidig er det viktig å se de positive modenhetsverdiene på dette området i sammenheng med hvordan virksomhetene er organisert. Mange av de offentlige virksomhetene har egne jurister som håndterer etterlevelse, og har god oversikt over krav fra lover og regler. Men det er ikke dermed gitt at denne kompetansen kommer til anvendelse i utviklingsprosjektene. I mange tilfeller kan det være ganske lang veg fra den interne kompetansen knyttet til lover og regler, og utviklerne eller de innleide konsulentene som er sentrale i programvareutviklingen.

Det kan være ganske lang veg fra den interne kompetansen knyttet til lover og regler, til utviklerne eller de innleide konsulentene som er sentrale i programvareutviklingen.

Aktiviteten CP1.1 (*The software security group has an overview of the regulations that our software has to comply with*), hadde flest bekreftende virksomheter innen denne praksisen. Her er det grunn til å tro at svarene i mindre grad gjelder (en eventuell) SSG, men snarere virksomheten generelt.

De neste praksisene med høy modenhet er konstruksjon og etterretning (*construction and intelligence*), sikkerhetsfunksjonalitet og design (*security features and design*) og standarder og krav (*standards and requirements*). 80% av virksomhetene sier at de gjør SFD1.2 (S*ecurity is a regular part of our organization's software architecture discussion*), og 80% sier også at de gjør SR 2.3 (*We use a limited number of standard technology stacks*) som i de fleste tilfellene betyr at virksomheten har standardisert på Microsoft-produkter i utvikling og produksjon (Microsoft Active Directory, Microsoft Internet Information Services, etc.).

I praksisen konfigurasjonsstyring og sårbarhetsstyring (*Configuration Management and Vulnerability management)*, sier 85% av virksomhetene at de tilfredsstiller CMVM1.1 (*The software security group has procedures for incident response, in collaboration with the incident response team (if it exists)*). De hevder også at de gjør CMVM 2.2 (*We track software defects found during operations until they are closed*), men det kan virke som om mange setter likhetstegn mellom dette og deres interne feilsporingssystem som ofte ikke tar spesielt hensyn til sikkerhetsfeil. I slike tilfeller er det nødvendig at sikkerhetsfeilene prioriteres høyt nok for at de skal håndteres; hvis ikke er det ingen garanti for at de faktisk lukkes innen rimelig tid. Det er også noe uklart i hvor stor grad det er prosedyrer for samarbeid under sikkerhetshendelser, men respondentene forteller at utviklere vil kunne involveres ved behov.

Det kan virke som om virksomhetene er overoptimistiske når det gjelder CMVM 1.1, ettersom de typisk refererer til at de har meget kompetente sikkerhetsadministratorer som hånd-terer hendelser på en bra måte. Det er imidlertid ikke like klart hvor tett koblingen mellom disse sikkerhets-administratorene og utviklerne er

90% sier at de oppfyller SE1.2 (*We use accepted good practice mechanisms for host/network security*), men dette er ikke så overraskende ettersom dette strengt tatt ikke dreier seg om programvaresikkerhet. Sikkerhet virker å ha et relativt stort fokus hos de som driver med nettverk, drift og infrastruktur. Flere forteller imidlertid at det er et skille

Det er ulike kulturer hos de som jobber med infrastruktur og de som jobber med programvareutvikling.

mellom utviklere og sikkerhetsfolk. En respondent uttaler at sikkerhet nok kan oppfattes som en hindring blant utviklere, siden de som jobber med sikkerhet kanskje stenger for mye gjennom brannmurer o.l. Det er ulike kulturer hos de som jobber med infrastruktur og de som jobber med programvareutvikling.

4.3 Områder og praksiser med lav grad av modenhet

Området med lavest modenhet er Angrepsmodeller (*Attack models*), tett fulgt av Strategi og Måling (*Strategy and Metrics*).

Når det gjelder angrepsmodeller, så sier 80% at de gjør AM1.5 (*The software security group keeps up to date by learning about new types of attacks / vulnerabilities*), og 55% sier de gjør AM 1.6 (*Build an internal forum to discuss attacks*), men alle de andre aktivitetene i AM gjøres av færre enn 25% av virksomhetene.

Det er bra at utviklerne sjekker hverandres kode, men ettersom det virker som det er liten kunnskap om programvaresikkerhet hos utviklerne, er det grunn til å tro at det som gjøres av kodegjennomgang bare i liten grad kan sies å være spesifikt på sikkerhet.

Det er typisk de sikkerhetsansvarlige som sendes på sikkerhetsrelaterte kurs, ikke utviklerne

En grunn til at disse aktivitetene kun gjøres i liten grad kan være at det er aktiviteter som går veldig spesifikt på sikkerhet, i tillegg til eller utenpå det som ellers gjøres i utviklingsprosessen. Flere virksomheter har uttalt at de er bevisst på at dette er områder hvor de har et forbedringspotensial. Når det gjelder de to aktivitetene nevnt over som relativt mange gjør, så er det flere av respondentene som forteller at disse i stor grad gjøres utenfor utviklingsmiljøene. De som jobber med drift og infrastruktur får gjerne varsler eller følger med på nye angrep, og disse diskuteres etter behov. Respondentene antar da at utviklere vil varsles om ting som er relevant for dem, men det virker å være lite systematisk arbeid med å følge med på angrep i utviklingsmiljøet. Flere respondenter forteller at enkeltutviklere er flinke til å følge med også på sikkerhetsfeltet, for eksempel for å få kunnskap om problemer knyttet til komponenter de selv benytter, men dette arbeidet virker å være relativt ustrukturert og i stor grad avhengig av den enkelte utvikler.

Risikovurderinger på virksomhetsnivå oppleves ikke som relevante for programvareutviklingen

Når det gjelder arbeid med strategi, er det noen virksomheter som er i gang med et arbeid på dette. Men slik det er nå har få virksomheter en strategisk og systematisk tilnærming til programvaresikkerhet, der de tydelig plasserer ansvar, lager planer og strategier, og følger opp gjennomføringen og effekten av disse. Dette kan for eksempel vises igjen i at få virksomheter har et klart og tydelig svar på hvem hos dem som er SSG. Alle virksomheter gjør noen aktiviteter knyttet til programvaresikkerhet, og noen gjør mange, men siden dette ikke gjøres systematisk er det vanskelig å kunne si noe om effekten av det arbeidet som gjøres i virksomhetene i dag knyttet til dette. På bakgrunn av undersøkelsen kan vi ikke si noe om årsakene til dette, men noen uttalelser i intervjuene kan tyde på at det er lite oppmerksomhet om programvaresikkerhet fra ledelsen. Som eksempel ble det i ett intervju uttalt at det som gjøres av risikovurderinger på virksomhetsnivå ikke var relevant for programvareutviklingen.

4.4 Områder og praksiser hvor virksomhetene er mest interessert i forbedring

Mange av virksomhetene ga uttrykk for at de er klar over at de har et lavt modenhetsnivå innen programvare-sikkerhet, og de fleste hevdet at de jobber med forbedringer på flere områder.

Smidig utviklingsmetodikk ble nevnt som en utfordring, spesielt i forbindelse med sjekkpunkter for godkjenning av utviklingsfaser. Det er lite som tyder på at norske virksomheter (heller ikke offentlige) har tenkt å gå bort fra smidig utvikling med det første, så dette fremstår som et område hvor det er behov for bedre tiltak, mer veiledning og erfaringsutveksling, samt forskning.

Under oppfølgingsintervjuene spurte vi virksomhetene om hvilke typer verktøy de bruker i forbindelse med de forskjellige aktivitetene. Mange bruker verktøy, men mange var også interessert i å høre om vi hadde anbefalinger til hvilke verktøy som er lure å bruke. Dette ligger ikke innenfor mandatet for denne rapporten, men kunne være et område for videre studier.

Vi ble også spurt om vi spesifikt tilbyr kurs innen programvaresikkerhet. Det ble nevnt under intervjuene at det ville være interessant å lære av hva andre aktører gjør, og høre deres erfaringer med ulike programvare-sikkerhetstiltak. Noen av virksomhetene i denne studien er kommet lengre enn de andre, og har et høyere modenhetsnivå, og et mulig tiltak videre er å la disse virksomhetene få dele erfaringer med andre offentlige virksomheter. Dette er i tråd med anbefalingene om praksisfellesskap fra rapporten om behov knyttet til informasjonssikkerhet i forvaltningen [1].

4.5 Gyldighet av svarene

Som nevnt tidligere kan metoden som er brukt i denne modenhetskartleggingen karakteriseres som "assistert selvevaluering". Respondentene fra virksomhetene har, i et spørreskjema, angitt hvilke programvare-sikkerhetsaktiviteter som gjøres hos dem, og de har deltatt i et oppfølgingsintervju med formål å avklare usikkerhet knyttet til spørreskjemaet. Mange av respondentene uttalte at de syntes det var vanskelig å svare på en del av spørsmålene i spørreskjemaet. I noen tilfeller var dette fordi de ikke forstod hva aktivitetene innebar, for eksempel fordi de ikke kjente til begreper som ble brukt. I andre tilfeller manglet de kunnskap om praksis i egen organisasjon eller hos konsulenter og systemleverandører. I flere tilfeller var imidlertid usikkerheten knyttet til utfordringer med å svare et enkelt "ja" eller "nei" på om de gjennomfører aktiviteten. Disse tilfellene ble spesielt diskutert i oppfølgingsintervjuene, med formål å komme til en mest mulig lik vurdering for de ulike virksomhetene. I de fleste tilfeller medførte oppfølgingsintervjuene at det ble gjort noen endringer i de opprinnelige svarene. Samtidig er det viktig å påpeke at ulike respondenter vil ha ulik oppfatning av hva de ulike aktivitetene innebærer. Det er sannsynlig at noen av respondentene har vært opptatt av å få frem så mye som mulig av hva de gjør, og få en god score, mens andre har vært mer beskjedne på egne vegne, og ukomfortable med å si at de gjør en aktivitet som de kanskje ikke gjør i full grad. En annen usikkerhetsfaktor er at oppfølgingsintervjuene ble gjennomført av tre ulike forskere. Disse har hatt samtaler på forhånd og undervegs for å bidra til at de har en mest mulig lik oppfatning av hva som kreves knyttet til de ulike aktivitetene. Samtidig er det sannsynlig at forskerne kan ha gjort ulike vurderinger knyttet til hva som skal "godkjennes" som en aktivitet.

Siden studien er basert i stor grad på selvevaluering er det grunn til å tro at "BSIMM-scoren" er høyere enn den ville vært med en gjennomgang på linje med den som gjøres av Cigital i den opprinnelige BSIMM-studien. Samtidig øker gyldighetene av svarene på grunn av oppfølgingsintervjuene, sammenlignet med en ren spørreundersøkelse. Vi er av den oppfatning at oppfølgingsintervjuene har vært svært viktige for å øke gyldighetene av resultatene fra studien.

5 Oppsummering og videre arbeid

Denne undersøkelsen viser at offentlige virksomheter gjør en del aktiviteter som bidrar til sikkerhet i programvaren som utvikles. Samtidig er det tydelig at få jobber strategisk med programvaresikkerhet, der de har en helhetlig og systematisk tilnærming og følger opp om tiltakene virker. Mange virksomheter er veldig avhengige av interessen, kompetansen og initiativet til den enkelte utvikler når det kommer til å holde seg oppdatert på sikkerhet og ta grep i utviklingen for å sikre at sikkerhet blir ivaretatt. Dette står i kontrast til drift/nettverks-siden i virksomhetene, der sikkerhet virker å ha en klar prioritet.

De fleste virksomhetene som deltok i undersøkelsen er klare på at de er interesserte i temaet programvare-sikkerhet, men mange peker på at de har begrensede ressurser. Noen er tydelige på at de har prioritert andre områder høyere i arbeidet med å bedre sikkerheten. Men alle er positive til at det tas tak i dette temaet.

Et viktig satsningsområde fremover, for å sette offentlige virksomheter i bedre stand til å jobbe strategisk og systematisk med programvaresikkerhet, er å iverksette opplæringsaktiviteter innen dette temaet. Program-varesikkerhet er så langt i svært liten grad en del av arbeidet med å øke kompetanse og bevissthet om informasjonssikkerhet i de ulike virksomhetene.

Difi har allerede gitt uttrykk for at det er ønskelig å gjenta denne undersøkelsen om noen år, og dette er noe vi stiller oss bak. Basert på intervjuene i denne studien er det grunn til å forvente seg store forbedringer i årene som kommer.

A Referanser

[1] Inger Anne Tøndel, Nils Brede Moe, Daniela Soares Cruzes: "Behov knyttet til informasjons-sikkerhet i forvaltningen", SINTEF Rapport A25874, januar 2014

[2] Gary McGraw: "Software security," *Security & Privacy, IEEE* , vol.2, no.2, pp.80,83, Mar-Apr 2004, doi: 10.1109/MSECP.2004.1281254

[3] The Building Security In Maturity Model (BSIMM-V), https://www.bsimm.com/

[4] Software Assurance Maturity Model (SAMM): A guide to building security into software development, http://www.opensamm.org/

[5] Leaders in Software Security & Application Security – Cigital, http://www.cigital.com/

[6] SINTEF – Faggruppe informasjonssikkerhet, http://infosec.sintef.no/?s=bsimm

[7] Easy Online Meetings With HD Video Conferencing | GoToMeeting, http://www.gotomeeting.com/online/entry

[8] Colin Robson: "Real World Research", 3rd ed., John Wiley & Sons 2011

[9] OWASP Top Ten Project, https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

B Spørreskjema

DIFI MODENHETKARTLEGGING INFORMASJONSIKKERHET	
Virksomhet:	COPY HERE THE ANSWERS FROM THE COMPANY
Rolle(r) til personen(e) som har fylt ut spørreskjemaet	
Avdeling/underenhet (hvis relevant)	
Driver dere med systemutvikling internt?	
Hvor stor del av den totale programvareutviklingen for virksomheten skjer internt?	
Hvor mange utviklere er ansatt i virksomheten?	
Hvor mange utviklere er innleid?	
Setter virksomheten ut utvikling av nøkkelferdige programvareløsninger til eksterne leverandører?	
Kontaktinformasjon (epost)	
Dato for utfylling:	
Dato for oppfølgingsintervju http://doodle.com/9dauquxif494cppt	

Assessment Worksheet			
Business Functions	**Security Practices**	**Activities**	**Answer (Yes, No, Don't Know)**
Governance	Strategy & Metrics	We publish our process for addressing software security; containing goals, roles, responsibilities and activities.	
		We have a secure software evangelist role to promote software security internally.	
		We educate our executives about the consequences of inadequate software security.	
		We have *identified* gate locations in our secure software development process where we make go/no go decisions with respect to software security.	
		We *enforce* the identified gate locations in our secure software development process where we make go/no go decisions with respect to software security, and track exceptions.	
		We have a process of accepting security risk and documenting accountability. In this process we assign a responsible manager for signing off on the state of all software prior to release.	
		The software security group publishes data internally on the state of software security within the organization.	
		In addition to the software security group, we have also identified members of the development teams that have a special interest in software security, and have a process for involving them in the software security work.	
		We have identified metrics that measure software security initiative progress and success.	
		The software security group has a centralized tracking application to chart the progress of all software.	
		The software security group advertises the software security initiative outside the organization (for example by writing articles, holding talks in conferences, etc).	
	Policy & Compliance	The software security group has an overview of the regulations that our software has to comply with.	
		We have a software security policy to meet regulatory needs and customer demands.	
		The software security group is responsible for identifying all legislation related to personally identifiable information (for example personopplysningsloven).	
		We have identified all the personally identifiable information stored by each of our systems and data repositories.	
		All identified risks have to be mitigated or accepted by a responsible manager.	
		We can demonstrate compliance with regulations that we have to comply with.	
		We make sure that all vendor contracts are compatible with our software security policy.	
		We promote executive awareness of compliance and privacy obligations.	
		We have all the documentation necessary for demonstrating the organization's compliance with regulations we have to comply with (for ex. written policy, lists of controls, artifacts from software development).	
		When managing our third party vendors, we impose our software security policies on them.	
		Information from the secure software development process is routinely fed back into the policy creation process.	
	Education & Guidance	We have a security awareness training program.	
		We offer role-specific security courses (for example on specific tools, technology stacks, bug parade).	
		The security awareness training content/material is tailored to our history of security incidents.	
		We deliver on-demand individual security training.	
		We encourage security learning outside of the software security group by offering specific training and events.	
		We provide security training for new employees to enhance the security culture.	
		We use the security training to identify individuals that have a particular interest in security.	
		We have a reward system for encouraging learning about security.	
		We provide security training for vendors and/or outsourced workers.	
		We host external software security events.	
		We require an annual software security refresher course.	
		The software security group has defined office hours for helping the rest of the organization.	

Assessment Worksheet			
Business Functions	**Security Practices**	**Activities**	**Answer (Yes, No, Don't Know)**
Construction / Intelligence	Attack Models	We build and maintain a top N possible attacks list.	
		We have a data classification scheme and an inventory of attacks so we can prioritize applications by the data handled by them.	
		We maintain a list of likely attacker profiles.	
		We collect and publish attack stories.	
		The software security group keeps up to date by learning about new types of attacks / vulnerabilities.	
		We have an internal forum to discuss attacks.	
		We link abuse cases to each attacker profile.	
		We have a list of technology-specific abuse cases.	
		We have an engineering team that develops new attack methods.	
		We have automated the attack methods developed by our engineers.	
	Security Features and Design	Our software security group builds and publishes a library of security features.	
		Security is a regular part of our organization's software architecture discussion.	
		The software security group facilitates the use of secure-by-design middleware frameworks/common libraries.	
		The software security group is directly involved in the design of security solutions.	
		We have a review board to approve and maintain secure design patterns.	
		We require the use of approved security features and frameworks.	
		We find and publish mature design patterns from the organization.	
	Standards and Requirements	The software security group create standards that explain the accepted way to carry out specific security centric operations.	
		We have a portal where all security related documents are easily accessible.	
		The software security group assists the software development team in translating compliance constraints (for instance from legislation) into application specific security requirements.	
		We use secure coding standards in our software development.	
		We have a standards review board to formalize the process used to develop security standards.	
		We use a limited number of standard technology stacks.	
		We have a template SLA text for use in contracts with vendors and providers, to help prevent compliance and privacy problems.	
		We have procedures to communicate and promote our security standards to vendors.	
		We have a list of all open source components used in our software.	
		We manage the risks related to using open source components.	

Assessment Worksheet			
Business Functions	**Security Practices**	**Activities**	**Answer (Yes, No, Don't Know)**
Verification / Touchpoints	Design Review / Architecture Analysis	We perform security feature review.	
		We perform design review for high-risk applications.	
		We have a software security group that leads review efforts.	
		We use a risk questionnaire to rank applications in terms of the risk they are exposed to.	
		We have a defined process to do architecture analysis.	
		We have a standardized format for describring architecture that also covers data flow.	
		The software security group is available to support architecture analysis when needed.	
		The software architects lead design review efforts to detect and correct security flaws.	
		Failures identified during architecture analysis are used to update the standard architecture patterns.	
	Code Review	We create a list with top N software security defects list.	
		The software security group does ad-hoc code reviews.	
		We use automated tools (such as static analysis) along with manual review to detect software security defects.	
		We make code review mandatory for all projects before release.	
		The software security defects found during code review are tracked in a centralized repository.	
		We enforce coding standards to improve software security.	
		We have mentors for code review tools for making most efficient use of the tools.	
		We use automated tools with tailored rules to improve efficiency and reduce false positives.	
		We combine assessment results so that multiple analysis techniques feed into one reporting and remediation process.	
		When a software defect is found we have tools to search for that defect also in the whole codebase.	
		We perform automated code review on all code to detect malicious code.	
	Security Testing	We perform adversarial tests with edge and boundary values.	
		We create our tests based on existing security requirements and security features.	
		We integrate black box security tools into the testing process (including protocol fuzzing).	
		We share security test results with QA.	
		We include security tests in QA automation.	
		We perform fuzz testing customized to application APIs.	
		We base the security tests on the security risks analysis.	
		We use code coverage tools to ensure that security tests cover all parts of the code.	
		We write tests cases based on abuse cases provided by the software security group.	

Assessment Worksheet			
Business Functions	**Security Practices**	**Activities**	**Answer (Yes, No, Don't Know)**
Deployment	Configuration and Vulnerability Management	The software security group has procedures for incident response, in collaboration with the incident response team (if it exists).	
		We are able to make quick changes in the software when under attack.	
		We perform drills to ensure that incident response capabilities minimize the impact of an attack.	
		We identify software defects found in operations (for ex. by intrusion detection systems) and feed back to development.	
		We track software defects found during operations until they are closed.	
		We maintain a matrix of all installed applications in order to identify all places that need to be updated when a piece of code needs to be changed.	
		When a software defect is found in a piece of code during operations we have a process to search for that defect also in the whole codebase.	
		We do software security process improvement based on the analysis of cause of software defects found in operations.	
		We have a system for paying rewards to individuals who report security flaws in our software.	
	Software Environment	We monitor the input to software we run in order to spot attacks on our software.	
		We use accepted good practice mechanisms for host/network security.	
		The software security group creates and publishes installation guides to ensure that our software is configured securely.	
		We create digital signatures for all binaries that we deliver.	
		We use code protection such as obfuscation to make reverse engineering harder.	
		We monitor the behavior of our software looking for misbehavior and signs of attacks.	
	Penetration Testing	We use external penetration testers on our software.	
		Defects found in penetration testing are inserted in our bug tracking system and flagged as security defects.	
		We use penetration testing tools internally.	
		The penetration testers have access to all available information about our software (for example: the source code, design documents , architecture analysis results and code review results).	
		We periodically perform penetration tests on all our software.	
		We use external penetration testers to do deep-dive analysis for critical projects to complement internal competence.	
		The software security group has created customized penetration testing tools and scripts for our organization.	

C Intervjuguide

Intervjuguiden som presenteres her fungerte som utgangspunkt for alle oppfølgingsintervjuene, men alle delene ble ikke brukt mot alle virksomheter; der det var klart at virksomheten ikke gjorde noen aktiviteter innen et område, ble ikke dette tatt opp videre i intervjuet.

Post Interview			
	Date:		
	Note:	Please ensure that you have printed the form answered by the company and all documents provided by them. And that you follow up all the don't know questions as well as all the questions that the answers were not expected (for example, answer no to all security testing questions). The questions bellow, you have to first see if the answer for that company was yes or no and cut accordingly. Before the interview remember to ask if it is ok that we will record the interview, and that the recording is just for us to not have to type in everything and only SINTEF will have access to it.	
Business Functions	**Security Practices**	**Activities**	
Governance (20 min)		**On this set of practices what are there any questions that you had some problems to answer? Or that you were not sure what to answer?**	
	Strategy & Metrics	Could you describe your process for addressing software security? How is is implemented and spread in your organization? Do you have a software security group.	
		Do you perform security risks evaluation? How does it work? Do you have an example you can give us?	
		In addition to the software security group, do you have also identified members of the development teams that have a special interest in software security? Do you have activities for involving them in the software security work? How do you do that?	
		How metrics for software security initiative progress measurements is performed?	
	Policy & Compliance	Do you have a set of regulations that your software has to comply with? How do you demonstrate compliance with regulations that we have to comply with?	
		Do you have a security policy? How is it used?	
	Education & Guidance	How is your training program?	
		Any further comments on Governance?	

Business Functions	Security Practices		Activities
Construction / Intelligence (20 min)		**On this set of practices what are there any questions that you had some problems to answer? Or that you were not sure what to answer?**	
	Attack Models	Could you describe how you create, use and maintain your attack list?	
	Security Features and Design	Could you describe how mature is your base of design patterns, and how useful they are for the development?	
	Standards and Requirements	How do you deal with open source components? Do you do risk analysis of these components?	
		Any further comments on the construction/ intelligence?	

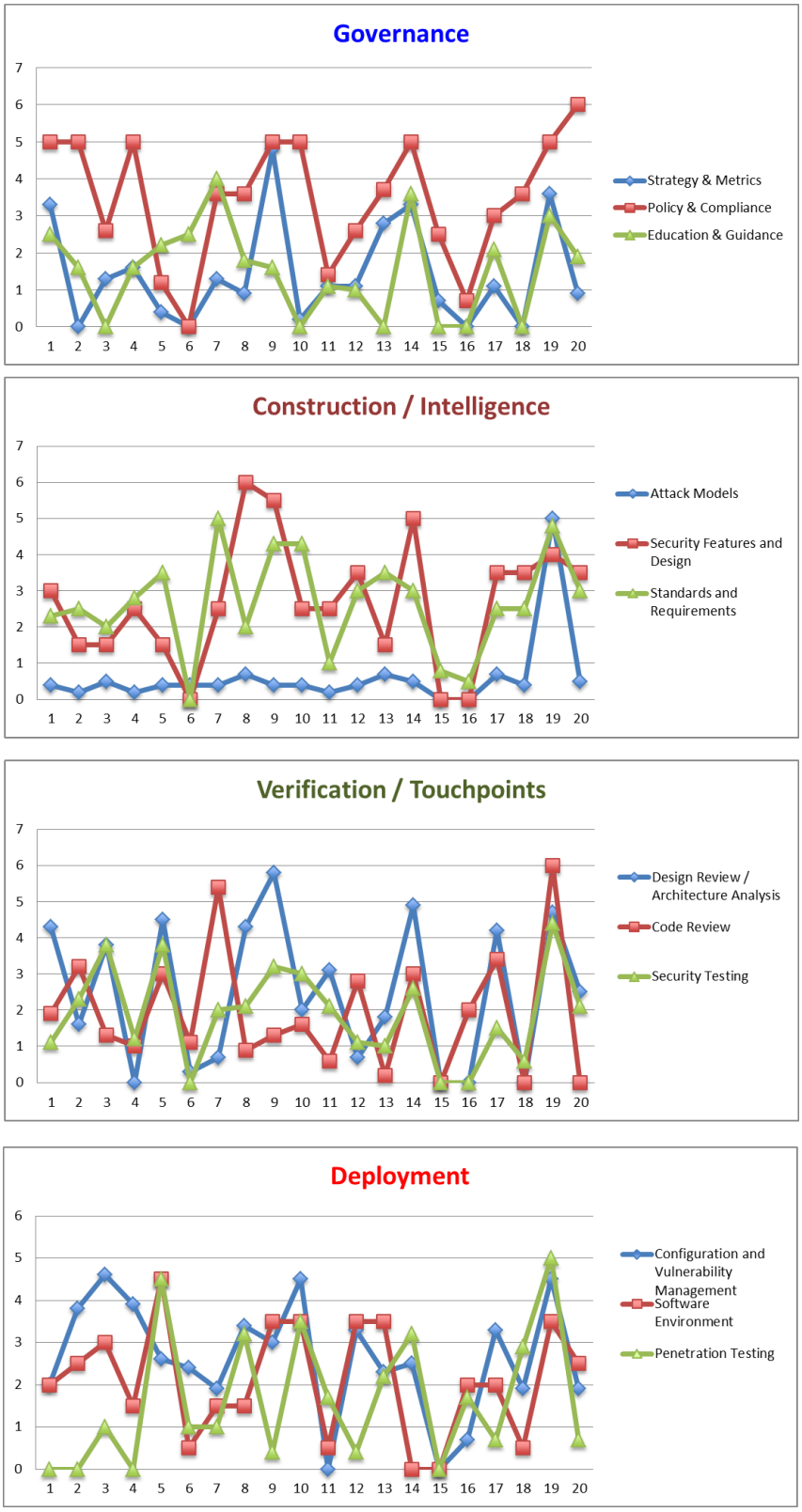
Business Functions	Security Practices		Activities
Verification / Touchpoints (20 min)		**On this set of practices what are there any questions that you had some problems to answer? Or that you were not sure what to answer?**	
	Design Review / Architecture Analysis	Could you please describe your design review process?	
		Could you please describe you architecture analysis process?	
		Do you have tools? Which tools?	
	Code Review	Could you please describe your process for code reviews? Could you describe how you create, use and maintain your defect list?	
		Do you use tools? Which tools?	
	Security Testing	How is your security testing process?	
		How is the integration between security testers and quality assurance people?	
		Which tools do you use?	
		Any further comments on verification?	

Business Functions	Security Practices		Activities
Deployment (20 min)	On this set of practices what are there any questions that you had some problems to answer? Or that you were not sure what to answer?		
	Configuration and Vulnerability Management	Could you describe the procedures for incident response related to software security? Could you describe what happens when a security software defect is found in a piece of code?	
	Software Environment	Could you describe how you monitor the inputs to your software? Could you describe how you monitor the behavior of your software? Which tools do you use? Who configures your firewalls, IDS and anti-virus? How do you manage the security of your binary code?	
	Penetration Testing	Could you describe your process of performing penetration testing?	
	Any further comments on deployment?		

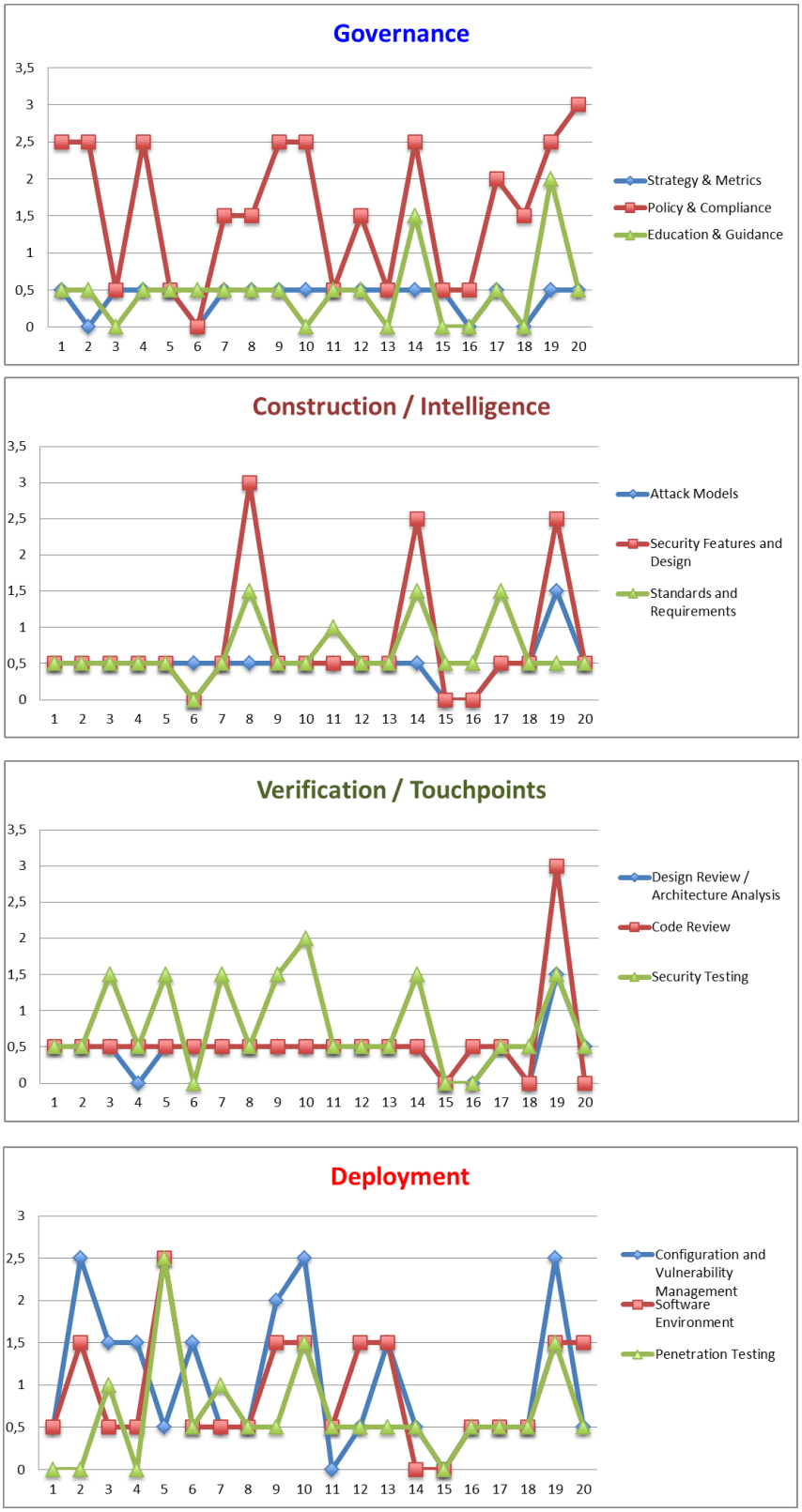
Business Functions	Security Practices	Activities
Extra Follow-up (10 min)	Do you think that the questions here covers your security activities?	
	By answering the questions here, did you already started to think about changes in your security activities?	
	Any further comments ?	
	THANK YOU!	

D Oversikt over funn

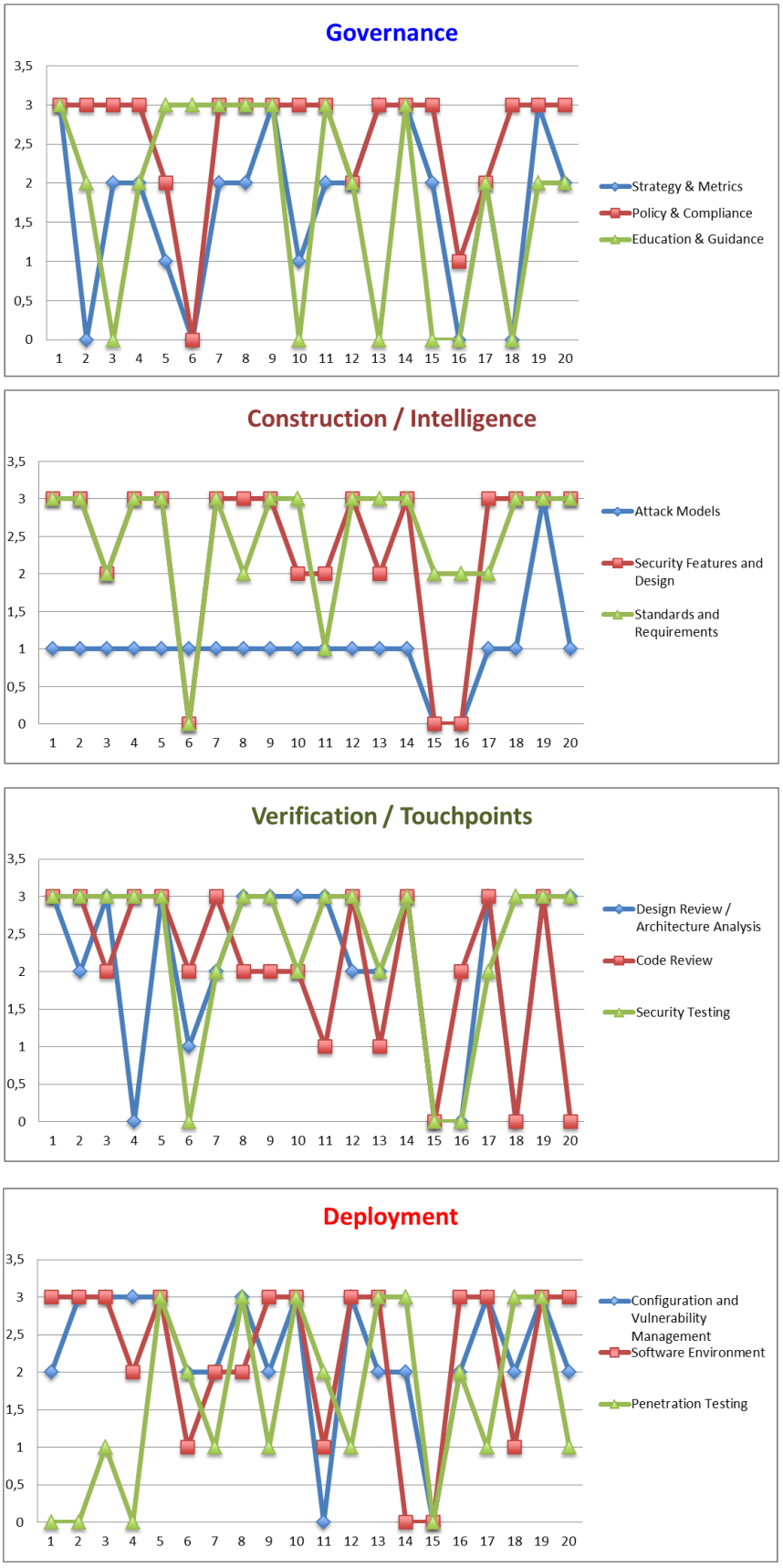
I det følgende presenterer vi totalresultatene for hvert domene (*Governance, Intelligence, SSDL Touchpoints, Deployment*) for alle de undersøkte virksomhetene (1-20). I Figur 9 presenteres totale verdier for vektet modenhetsnivå, Figur 10 gir resultatene for den konservative modenhetsverdien, mens Figur 11 gir høyvannsmerket for alle virksomhetene. Tallene oppsummeres i Tabell 15.



Figur 9: Vektet modenhet



Figur 10: Konservativ modenhet



Figur 11: Høyvannsmodenhet

Tabell 15: Totalresultater fra undersøkelsen

Business Functions	Security Practices	BSIMM	Observed DIFI	Observed BSIMM	Conservative Maturity DIFI	Weighted Maturity DIFI	Watermark Maturity DIFI	Watermark Maturity BSIMM
Governance	Strategy & Metrics	SM 1.1	35,00%	65,67%	0,4	1,4	1,8	2,0
		SM 1.2	50,00%	50,75%				
		SM 1.3	55,00%	50,75%				
		SM 1.4	60,00%	85,07%				
		SM 2.2	35,00%	46,27%				
		SM 1.6	35,00%	53,73%				
		SM 2.1	10,00%	38,81%				
		SM 2.3	55,00%	40,30%				
		SM 2.5	0,00%	29,85%				
		SM 3.1	15,00%	23,88%				
		SM 3.2	15,00%	8,96%				
	Policy & Compliance	CP 1.1	85,00%	64,18%	1,6	3,5	2,6	2,3
		CP 1.3	75,00%	67,16%				
		CP 1.2	80,00%	77,61%				
		CP 2.1	80,00%	35,82%				
		CP 2.2	75,00%	41,79%				
		CP 2.3	70,00%	43,28%				
		CP 2.4	65,00%	37,31%				
		CP 2.5	75,00%	52,24%				
		CP 3.1	35,00%	20,90%				
		CP 3.2	65,00%	16,42%				
		CP 3.3	20,00%	11,94%				
	Education & Guidance	T 1.1	50,00%	74,63%	0,5	1,5	1,8	1,5
		T 1.5	15,00%	43,28%				
		T 1.6	35,00%	34,33%				
		T 1.7	45,00%	49,25%				
		T 2.5	50,00%	13,43%				
		T 2.6	60,00%	19,40%				
		T 2.7	15,00%	13,43%				
		T 3.1	10,00%	5,97%				
		T 3.2	20,00%	5,97%				
		T 3.3	10,00%	11,94%				
		T 3.4	0,00%	13,43%				
		T 3.5	10,00%	7,46%				
Construction / Intelligence	Attack Models	AM 1.1	25,00%	31,34%	0,5	0,6	1,0	1,3
		AM 1.2	25,00%	64,18%				
		AM 1.3	25,00%	44,78%				
		AM 1.4	20,00%	17,91%				
		AM 1.5	80,00%	62,69%				
		AM 1.6	55,00%	23,88%				
		AM 2.1	0,00%	10,45%				
		AM 2.2	5,00%	16,42%				
		AM 3.1	5,00%	5,97%				
		AM 3.2	5,00%	8,96%				
	Security Features and Design	SFD 1.1	15,00%	80,60%	0,8	2,7	2,4	2,0
		SFD 1.2	80,00%	79,10%				
		SFD 2.1	55,00%	38,81%				
		SFD 2.2	70,00%	43,28%				
		SFD 3.1	20,00%	13,43%				
		SFD 3.2	60,00%	19,40%				
		SFD 3.3	15,00%	13,43%				
	Standards and Requirements	SR 1.1	40,00%	71,64%	0,7	2,7	2,5	1,8
		SR 1.2	55,00%	64,18%				
		SR 1.3	55,00%	67,16%				
		SR 1.4	60,00%	40,30%				
		SR 2.2	15,00%	34,33%				
		SR 2.3	80,00%	28,36%				
		SR 2.5	50,00%	32,84%				
		SR 3.2	40,00%	17,91%				
		SR 2.4	40,00%	28,36%				
		SR 3.1	40,00%	11,94%				

Business Functions	Security Practices	BSIMM	Observed DIFI	Observed BSIMM	Conservative Maturity DIFI	Weighted Maturity DIFI	Watermark Maturity DIFI	Watermark Maturity BSIMM
Verification / Touchpoints	Design Review / Architecture Analysis	AA 1.1	40,00%	83,58%	0,5	2,5	2,1	1,5
		AA 1.2	60,00%	52,24%				
		AA 1.3	20,00%	35,82%				
		AA 1.4	25,00%	62,69%				
		AA 2.1	45,00%	14,93%				
		AA 2.2	35,00%	11,94%				
		AA 2.3	50,00%	29,85%				
		AA 3.1	45,00%	16,42%				
		AA 3.2	35,00%	5,97%				
	Code Review	CR 1.1	20,00%	35,82%	0,6	1,9	2,1	1,4
		CR 1.2	20,00%	50,75%				
		CR 1.4	50,00%	74,63%				
		CR 1.5	45,00%	34,33%				
		CR 1.6	45,00%	37,31%				
		CR 2.2	60,00%	14,93%				
		CR 2.5	35,00%	22,39%				
		CR 2.6	40,00%	26,87%				
		CR 3.2	15,00%	5,97%				
		CR 3.3	30,00%	8,96%				
		CR 3.4	20,00%	1,49%				
	Security Testing	ST 1.1	45,00%	76,12%	0,8	1,9	2,4	1,7
		ST 1.3	60,00%	82,09%				
		ST 2.1	10,00%	40,30%				
		ST 2.4	55,00%	19,40%				
		ST 3.1	20,00%	16,42%				
		ST 3.2	15,00%	11,94%				
		ST 3.3	55,00%	8,96%				
		ST 3.4	10,00%	7,46%				
		ST 3.5	20,00%	10,45%				
Deployment	Configuration and Vulnerability Management	CMVM 1.1	65,00%	88,06%	1,0	2,6	2,3	2,0
		CMVM 2.1	85,00%	74,63%				
		CMVM 3.3	25,00%	2,99%				
		CMVM 1.2	55,00%	88,06%				
		CMVM 2.2	85,00%	65,67%				
		CMVM 2.3	45,00%	44,78%				
		CMVM 3.1	25,00%	8,96%				
		CMVM 3.2	25,00%	8,96%				
		CMVM 3.4	0,00%	2,99%				
	Software Environment	SE 1.1	40,00%	50,75%	0,9	2,1	2,3	2,0
		SE 1.2	90,00%	91,04%				
		SE 2.2	45,00%	46,27%				
		SE 2.4	10,00%	37,31%				
		SE 3.2	0,00%	14,93%				
		SE 3.3	60,00%	13,43%				
	Penetration Testing	PT 1.1	60,00%	92,54%	0,7	1,7	1,7	1,8
		PT 1.2	55,00%	76,12%				
		PT 1.3	45,00%	64,18%				
		PT 2.2	40,00%	35,82%				
		PT 2.3	10,00%	40,30%				
		PT 3.1	35,00%	19,40%				
		PT 3.2	5,00%	11,94%				

Teknologi for et bedre samfunn

www.sintef.no