RASK – Ramverk för IT-säkerhetskravställning

CAROLINE BILDSTEN, DANIEL EIDENSKOG, JONAS HERMELIN, JACOB LÖFVENBERG

FOI-R--4641--SE
ISSN 1650-1942 Oktober 2018

FOI-R--4641--SE

Caroline Bildsten, Daniel Eidenskog, Jonas Hermelin, Jacob Löfvenberg

RASK – Ramverk för IT-säkerhetskravställning

Bild/cover: Robert Zunikoff, Unsplash.com

Titel	RASK – Ramverk för IT-säkerhetskravställning
Title	RASK – Framwork for IT security requirements
Rapportnummer	FOI-R--4641--SE
Månad	Oktober
Utgivningsår	2018
Antal sidor	56
ISSN	ISSN-1650-1942
Uppdragsgivare	Försvarsmakten
Projektnummer	E72235
Godkänd av	Christian Jönsson
Ansvarig avdelning	Ledningssystem
Forskningsområde	Informationssäkerhet och kommunikation
FoT-område	
Exportkontroll	Innehållet är granskat och omfattar ingen information som är underställd exportkontrollagstiftningen.

Sammanfattning

IT-system utgör viktiga komponenter inom Försvarsmakten, ofta på nivån att IT-systemen är centrala för att verksamheterna ska fungera. I och med att hotbilden mot Försvarsmakten är omfattande innebär detta att IT-säkerhetskraven för systemen ofta är mycket höga. I dagsläget ställs de generella säkerhetskraven på IT-system genom regelverket *Krav på IT-säkerhetsförmågor hos IT-system v3.1 (KSF 3.1)*. Som ett steg i utvecklingen mot ett mer välanpassat regelverk har FOI fått i uppdrag av Försvarsmakten att ta fram ett regelverksförslag. Målen med arbetet var att ta fram ett förslag som reducerar risken för överdriven kravställning, ökar förutsägbarheten vid bedömning av kravuppfyllnad, ökar förståelsen för kraven samt förbättrar möjligheterna till fortlöpande uppföljning.

Arbetet utfördes genom en explorativ process som tog sin utgångspunkt i de effektmål som Försvarsmakten satt upp, kombinerat med information som sammanställts från tidigare arbeten som visar på förbättringspotential relativt det befintliga regelverket.

Resultatet av arbetet är *Ramverk för säkerhetskravställning (RASK)*, en metodik som består av såväl metoder som en generisk modell för kravställning. Kraven i RASK-metodiken delas in i *styrkenivåer*, baserat på hur kraftiga hot som lösningarna förväntas motstå. Styrkenivåerna bestäms sedan individuellt för varje krav utifrån ett antal *inparametrar* som beskriver systemets miljö och kontext. Kraven i RASK-metodiken skrivs på ett tydligt sätt med öppna, målorienterade formuleringar för att ge större förståelse för kraven samtidigt som sättet att skriva undviker att kraven förespråkar specifika tekniska lösningar. I projektet skapades även en demonstrator för att påvisa metodikens funktioner och exemplifiera hur krav kan skrivas enligt RASK-metodiken.

Nyckelord

RASK, KSF, IT-säkerhetskrav, metodik

Abstract

IT systems are vital to many of the operations in the Swedish Armed Forces. Since the threat level against the Armed Forces is high, the resulting IT security requirements are extensive. Today's regulatory framework for IT security requirements is *Krav på IT-säkerhetsförmågor hos IT-system v3.1 (KSF 3.1)*. As a step towards a more adaptable framework, the Swedish Armed Forces commissioned FOI to produce a suggestion for a new regulatory framework. The goals for the suggested framework were to reduce the risk of overly strict requirements, increase the predictability during requirement fulfillment assessment, increase the understanding of the requirements, and improve the possibility for continuous follow-ups.

The work was conducted through an explorative process, where the starting point was the impact goals given by the Armed Forces combined with information collected from previous works showing areas where the current framework can be improved.

The result is *Ramverk för säkerhetskravställning (RASK)*, *Framework for security requirement selection* in English. RASK is a methodology that consists of several metods and a generic model for handling the requirements. The requirements are subdivided into *strength levels*, based on the level of threats that the solutions are expected to withstand. The strength levels are determined individually for all requirements based on a number of *input parameters* that describe the system's environment and context. The requirements used in the RASK methodology are clearly phrased and stated as target-oriented to increase the understanding of the requirements and their motives, while avoiding requirements that prescribe specific technical solutions. A demonstrator was developed during the project to show the functionality of the methodology and exemplify how requirements can be written.

Keywords

RASK, KSF, IT security requirements, methodology

Innehåll

1 Inledning **7**
1.1 RASK 7
1.2 Effektmål 8
1.3 Uppgiftsbeskrivning 8

2 Bakgrund **11**
2.1 KSF 3.1 11
2.2 Tidigare arbete 12
2.3 Terminologi 15

3 Uppgiftsanalys **17**
3.1 Reducering av överkrav 17
3.2 Öka förutsägbarheten 18
3.3 Ökad kravförståelse 18
3.4 Analys av kravuppfyllnad och restrisk 19

4 Studiens genomförande **21**
4.1 Arbetsflöde 21
4.2 Avgränsningar 21
4.3 Identifiering av parametrar 22
4.4 Identifiering och formulering av krav och styrkenivåer 22

5 RASK-metodiken **23**
5.1 RASK-metodikens uppbyggnad 24
5.2 Metod för kravställning 33
5.3 Metod för analys av kravuppfyllnad 33
5.4 Metod för hantering av ändrade förutsättningar 34
5.5 Metod för periodisk uppföljning 35
5.6 Exempel på underlag för restriskanalys 36

6 Krav och kravhantering **41**

6.1 Kravarbetet 41
6.2 Kravprocessen 42

7 Demonstrator för metodik och verktygsstöd 45

8 Diskussion 47
8.1 Effektmålen 48
8.2 Processintegration 50
8.3 Från metodik till regelverk 51
8.4 Ett helikopterperspektiv 52

Litteratur 53

1 Inledning

FOI har fått i uppdrag av Försvarsmakten att utarbeta ett konceptregelverk som kan tillgodose det stödbehov som finns kring kravställning och uppföljning av IT-säkerhetskrav i utvecklingsprocessen. Ett nytt regelverk för gemensamma säkerhetskrav till Försvarsmaktens IT-system bör underlätta både kravställning och uppföljning, samtidigt som det resulterar i adekvata krav för de system som omfattas. Dessutom behöver regelverket vara anpassat till andra processer och metoder som har gränsytor eller beroenden mot regelverket.

Dagens regelverk, *Krav på IT-säkerhetsförmågor hos IT-system v3.1 (KSF 3.1)* (Försvarsmakten 2014), är att betrakta som en metod snarare än en process. KSF 3.1 utgör i princip en enkel metod där två parametrar fångar systemets förutsättningar så att metoden kan välja ut en uppsättning säkerhetskrav för systemet. Metoden används av kringliggande processer för att bestämma generella säkerhetskrav på IT-system som ska utvecklas. KSF 3.1 används vanligtvis en enda gång under systemutvecklingen och innehåller inte metoder för exempelvis uppföljning och kravtolkning.

1.1 RASK

Regelverksförslaget som presenteras i denna rapport går under namnet *Ramverk för säkerhetskravställning (RASK)*. RASK utgör en *metodik*, där ett antal *metoder* finns definierade för olika arbetsflöden, exempelvis kravställning och uppföljning. Utöver metoderna ingår även en *modell* som beskriver kraven och hur de bestäms utifrån systemets förutsättningar.

Modellen i RASK bygger på att förutsättningarna beskrivs genom ett antal *inparametrar* som sedan nyttjas för att beräkna individuella *styrkenivåer* för respektive krav. Styrkenivåerna innebär att kraven kan graderas utifrån hur stor skyddsförmåga lösningen förväntas uppnå för att uppfylla kravet. Kraven utformas med fokus på skyddsförmågan och formuleras med öppna, målorienterade texter. Genom denna typ av formuleringar undviker kraven att förespråka specifika tekniska lösningar och lämnar en större frihet till systemutvecklaren att uppfylla kraven på lämpligaste sätt utifrån systemet.

Genom modellens utformning är det möjligt att utforma olika metoder för kravställning, hantering av ändrade förutsättningar samt uppföljning av säkerhetsläget över tid. Sammantaget ger dessa metoder en koherent metodik, där RASK kan ge kontinuitet i kravställning över förändringar i såväl systemet som omvärldsläget.

Läsare som endast intresserar sig för RASK-metodikens utformning kan hoppa direkt till kapitel 5, men missar då beskrivningar som behandlar bakgrunden till arbetet och hur det genomförts.

1.2 Effektmål

Utgångspunkten för studien har varit en uppsättning önskade effektmål som Försvarsmakten presenterade vid de inledande diskussionerna. De önskade effektmålen var att ta fram ett regelverk som

- stödjer verksamheten i att göra korrekta designval med hänsyn till verksamheten och de hot verksamheten vill kunna motstå
- ger verksamheterna instinktiv förståelse för riskerna
- stödjer ett kontinuerligt informationssäkerhetsarbete
- ger verksamheten realistiska förväntningar på systemet
- hanterar både sekretess, riktighet och tillgänglighet
- möjliggör jämförelser mellan system.

Effektmålen sedda som helhet är både breda och djupa. Breda i det att de inte enbart berör en avgränsad del av det KSF 3.1 behandlar och djupa i det att de inte på ett vederhäftigt sätt går att uppfylla med bara detaljförändringar i KSF 3.1.

1.3 Uppgiftsbeskrivning

Utgående huvudsakligen från effektmålen i avsnittet ovan utarbetade FOI och Försvarsmakten tillsammans en uppgiftsbeskrivning som tolkade effektmålen på ett sätt som tydligare kunde utgöra inriktning för studiens arbete. Uppgiftsbeskrivningen tar upp följande arbetsuppgifter:

1. Beskriva och analysera uppgiften.
2. Analysera omgivande regelverk för att ge grundläggande förståelse för kontexten som regelverket verkar inom.
3. Ta fram förslag på modell för bestämning av IT-säkerhetskrav. Modellen ska tas fram efter följande principer:
    - Identifiera ingångsvärden som går att extrahera ur en befintlig hotmodell.
    - Ta fram avbildning mellan ingångsvärden och kravnivåer.
    - Bestäm indelning i kravnivåer och kravområden[1].
4. Ta fram specifika krav inom utvalda områden, såväl tekniska krav som assuranskrav.

Den första punkten inledde studien och redovisades tillsammans med uppgiftsbeskrivningen i *Framtagning av nya säkerhetskrav för IT-system – Problembeskrivning* (Eidenskog 2017). Den andra punkten lyftes över i en annan studie och redovisades genom rapporten *Rutiner och regelverk för att godkänna IT-system inom Försvarsmakten* (Gud-

[1] Konceptet med kravområden förkastades under arbetets gång, se avsnitt 5.1.2.

mundson Hunstad 2017). Punkt tre och fyra utgör huvudarbetet som redovisas i denna rapport.

2 Bakgrund

IT-system utgör viktiga komponenter i Försvarsmaktens verksamheter – ofta på nivån att IT-systemen är centrala för att verksamheterna ska fungera. I många fall är kommersiellt tillgängliga IT-system inte lämpliga för att fylla Försvarsmaktens behov, antingen för att de inte passar verksamheten eller för att de inte uppfyller de höga säkerhetskraven. Således finns ett stort behov av att ta fram IT-system som anpassats till Försvarsmaktens speciella förutsättningar.

Det finns ett antal olika processer och metoder som ska följas när nya IT-system tas fram till Försvarsmakten. Huvudprocessen är *Försvarsmaktens IT-process* (Försvarsmakten 2013) som täcker hela kedjan från behovsfångst i verksamheten till avveckling av systemet. I IT-processen ingår att samla in de krav som ska ställas på såväl systemet som utvecklingen av systemet, där det befintliga regelverket *Krav på IT-säkerhetsförmågor hos IT-system. (KSF)* ingår som en delkomponent för generell säkerhetskravställning. Då det i regel finns hot som är specifika för respektive IT-system och därmed inte hanteras av KSF måste kraven kompletteras utifrån en systemspecifik hotbedömning.

I processen ingår även att genomföra en granskning av systemet som ska leda till ett godkännande för användning, så kallad *ackreditering*. Granskningen genomförs delvis av MUST men kan även göras av andra parter med lämplig kompetens.

Under systemets livstid finns behov av att genomföra kontinuerliga uppföljningar och förbättringar, bland annat för att hantera upptäckta säkerhetsbrister och för att omhänderta förändringar i hotbild.

2.1 KSF 3.1

Krav på IT-säkerhetsförmågor hos IT-system v3.1. (KSF 3.1) (Försvarsmakten 2014) är det gällande regelverket för IT-säkerhetskravställning när detta skrivs. KSF 3.1 utgår från två ingångsvärden för att fastställa den kravnivå som ett IT-system måste uppfylla. Ingångsvärdena är *konsekvensnivå* och *exponering*, vilka sedan vägs samman enligt en specifik matris för att bestämma kravnivån för systemet till antingen *Grund*, *Utökad* eller *Hög*. Kravnivån används sedan för att peka ut de specifika säkerhetskrav som IT-systemet måste uppfylla.

Konsekvensnivån beskriver i övergripande formuleringar den skadeverkan, det *men*, som kan uppstå vid en oönskad händelse och graderas på skalan 1–5. I konsekvensnivån räknas sekretessförlust in och har en direkt koppling till informationssäkerhetsklasserna, Hemlig/Restricted på nivå 2 och så vidare till Hemlig/Top Secret på nivå 5.

I de fall då aktuella uppgifter inte hamnar i någon informationssäkerhetsklass väljs nivå 1.

Exponeringen tar hänsyn till personer och andra IT-system som kan komma i kontakt med IT-systemet och bedöms på skalan 1–4. Nivå 1, som är den lägsta exponeringen, motsvarar en mycket kontrollerad miljö med enbart personer som är behöriga till all information i systemet och där det inte finns några kopplingar mot andra IT-system. Den högsta exponeringsnivån motsvarar en miljö där ej säkerhetsprövade personer kan förekomma eller där det finns kopplingar till andra IT-system som inte är kontrollerade eller som i sig har hög exponeringsnivå.

En kritik mot det nuvarande regelverket är att konsekvensnivån och exponeringen utgör ett för smalt bedömningsunderlag som gör att kravställningen allt för ofta måste utgå från den högsta hotbilden vilket resulterar i höga krav. Faktorer som påverkar hotbilden (såväl uppåt som nedåt) och därmed påverkar kravbilden kan inte tas i beaktande inom det nuvarande regelverket. Exempel på faktorer som kan ge lägre hotbild är om existensen av systemet endast är känd av ett fåtal personer eller om de hot som riktas mot systemet inte kommer från en lika kvalificerad hotagent. Exempel på en faktor som kan ge högre hotbild är storskalighet, där många personer använder systemet på många platser för att hantera stora informationsmängder.

2.2 Tidigare arbete

Under de senaste åren har FOI genomfört flera studier som behandlar frågor kring utveckling av IT-system inom Försvarsmakten. I de studierna har KSF och säkerhetsgranskning återkommande tagits upp som problematiska i utvecklingsprojekten, speciellt avseende att de sägs orsaka ökad kostnad och längre utvecklingstid. En del i detta är den osäkerhet som upplevs i samband med säkerhetsgranskning av systemen, där den efterfrågade nivån på systemens säkerhetsfunktioner inte går att utläsa ur säkerhetskraven vilket upplevs leda till oförutsägbara bedömningar. Att dessa åsikter är återkommande i studierna kan tolkas som att kritiken är utbredd.

Studierna pekar även på att dagens situation bland annat kan leda till sämre säkerhet där verksamheterna håller kvar vid gamla system som inte är uppdaterade till dagens säkerhetsbehov.

Följande citat sammanfattar läget som det beskrivs i en studie.

> Det finns en upplevelse av att den analys som görs av MUST och de krav som ställs för att uppfylla regelverket är personbaserat och därmed kan variera. Uppfattningen hos en respondent är också att MUST enbart ser till statisk säkerhet, utan att hantera förändringar i förutsättningar och miljö. Detta bidrar till att Försvarsmakten har kvar gamla osäkra system, eftersom ingen vågar föreslå att ersätta dessa med nyare och mer säkra sy-

> stem, för att då riskera att MUST ska säga att säkerhetsmålsättningarna inte är uppfyllda. Respondenter menar att auktorisationen måste uppmuntra och stödja att uppdateringar görs för att öka säkerheten och att nya säkrare system ersätter äldre. Det anses vara ett grundfel att auktoriserade system stannar i den form de har vid auktorisation. Det påstås finnas ett beslut inom Försvarsmakten att system enbart behöver ackrediteras en gång. (Hermelin m. fl. 2017, s. 38)

I studierna har intervjuer gjorts med personer som på olika sätt arbetar med utveckling, granskning eller förvaltning av IT-system åt Försvarsmakten. Respondenternas svar visar att säkerhetsarbetet uppfattas sikta på absolut säkerhet, där avvägningar mellan verksamhetens behov, systemens genomförbarhet och säkerhetsaspekter inte är möjliga. De visar även ett starkt fokus på sekretess, där andra aspekter såsom tillgänglighet fått stå tillbaka. Här följer några citat från tidigare arbeten som visar på detta.

> Säkerhetsarbetet genomsyras av uppfattningen att absolut säkerhet, med starkt fokus på sekretess, driver IT-säkerhetsarbetet medan tillgänglighet och verksamhetsnytta får stå tillbaka. (Karlzén, Eidenskog och Löfvenberg 2017, s. 26)

> Det påtalas hos ett flertal respondenter vikten av att låta verksamhetens behov vara styrande och att verksamhetsnyttan i högre grad måste vägas mot eventuella risker. (Hermelin m. fl. 2017, s. 39)

> Processen för ackreditering är för krånglig och har krav på säkerhet som är orimligt höga. (Hermelin m. fl. 2017, s. 37)

> Om fokus på absolut säkerhet ska minska [. . .] behövs ett kontinuerligt och aktivt säkerhetsarbete under driftsfasen. (Karlzén, Eidenskog och Löfvenberg 2017, s. 28)

> Flera respondenter indikerade att det ofta är svårt att göra avvägningen mellan tillgänglighet och sekretess. På många ställen i Försvarsmakten är idag sekretessnivån gränssättande men i stridssituationer ökar behovet avseende tillgänglighet (Gudmundson Hunstad m. fl. 2012, s. 15)

Studierna har även pekat på att KSF och granskningsprocessen ger för lite stöd till utvecklingsprojekten, exempelvis vad gäller designval och avvägning mellan teknisk och administrativ säkerhet. Här följer några citat avseende detta.

> Därför rekommenderas att en katalog med säkerhetskomponenter relateras till [. . .] KSF3 och görs tillgänglig (Bengtsson, Sommestad och Holm 2014, s. 58)

> När tekniska skydd blir allt bättre fokuserar angripare i ökad utsträckning på sårbarheter i användares tillgång till IT-system. (Svenmarck 2017, s. 29)

> Det måste ske en mer välgrundad balansering mellan säkerhet och tillgänglighet. Nyttan med system måste få en högre prioritering i förhållande till säkerhet. (Hermelin m. fl. 2017, s. 31)

Det är inte bara studierna som pekar på att det finns utvecklingspotential. Även i de befintliga regelverken och styrande dokumenten inom Försvarsmakten finns tecken som indikerar ”behov och potential för vidare utveckling där riktighets- och tillgänglighetsaspekterna på ett tydligare och bättre sätt bidrar.” (Gudmundson Hunstad 2016, s. 25)

En viktig reflektion om kritiken mot KSF 3.1 är att den mycket väl kan ha fått en viss mängd oförtjänt kritik då den sticker ut bland processerna och metoderna som en konkret och tydlig avstämningspunkt. Genom att KSF 3.1 är såpass konkret blir det svårt att kringgå den samtidigt som det är relativt lätt att det blir diskussioner om dess inverkan på projekten.

Ytterligare en aspekt som är värd att reflektera över är omfattningen av den påverkan som KSF 3.1 ger på IT-systemen och deras utveckling jämfört med övriga processer och metoder som behöver följas och hanteras under IT-systemens livscykel. IT-säkerhet är ett relativt abstrakt område, där effekterna av olika åtgärder i många fall är svåra att överblicka och därmed ofta svåra att bedöma värdet av. Andra områden, som exempelvis personsäkerhet och miljö, är i många fall lättare att förhålla sig till och åtgärder har tydligare effekt relativt de mål som eftersträvas. FOI-rapporten *Försvarsmaktens IT-styrning: Nulägesanalys* (Hermelin m. fl. 2017) visar med tydlighet att det finns andra områden än IT-säkerhet som negativt påverkar Försvarsmaktens förmåga att ta fram och underhålla IT-system, exempelvis att processer kringgås och att designansvaret är otydligt. Ett annat exempel är att olika processer skiljer sig åt i hur roller, begrepp och metoder definieras även när de i grund och botten handlar om samma sak, vilket gör att processerna blir onödigt svåra att samordna inom samma projekt.

Effektmålen som satts för studiens arbete (se avsnitt 1.2) har stor överensstämmelse med de problem som identifierats i tidigare studier enligt ovan. Genom att förbättra regelverket i den riktning som effektmålen strävar torde regelverket bättre stödja arbetet kring utveckling och vidmakthållande, samtidigt som en större tydlighet och transparens kan nås när regelverket appliceras.

2.3 Terminologi

Följande termer används i rapporten.

öppet målkrav	Krav som skrivits så att fokus ligger på målet med kravet, det vill säga vad lösningen på kravet ska uppnå.
ackreditering	Beslutssteg som innebär säkerhetsgodkännande av systemet.
deaktiverat krav	Krav som inte är applicerbara på systemet givet inparametrarnas värden.
demonstrator	Ett enkelt verktyg för RASK-metodiken som illustrerar funktionen och möjligheterna som ges av verktygsstöd.
generell hotanalys	En hotanalys som tar hänsyn till hotbilden mot en (större) mängd av IT-system. Generella hotanalyser används för att ta fram specifika modeller.
generisk modell	Den generiska, oifyllda RASK-modellen som ska fyllas med inparametrar, sammanvägningsfunktioner och kravtexter för att skapa en specifik modell.
inparameter	Indata som fångar IT-systemets miljö och kontext på ett sätt som stipuleras av den specifika modellen.
KSF	*Krav på IT-säkerhetsförmågor hos IT-system*, Försvarsmaktens regelverk för att fastställa IT-säkerhetskrav. Termen avser regelverket oavsett version.)
KSF 3.1	KSF version 3.1. I skrivande stund gällande version av Försvarsmaktens regelverk för att fastställa IT-säkerhetskrav
metod	Sätt att utföra visst arbete.
metodik	Uppsättning metoder inom visst område.
modell	Schematisk avbildning eller strukturerad beskrivning av verkliga eller abstrakta företeelser.
parameterdelta	En differens i inparametervärden som resulterar i en specifik differens mellan kravställning och kravuppfyllnad.
permanent krav	Krav som alltid är aktiverade på en fast styrkenivå, oavsett inparametrarnas värden.

problemorienterat krav	Krav som skrivits så att fokus ligger på de specifika problemställningar som lösningen på kravet ska omhänderta.
process	Verksamhet som består av ett antal relaterade händelser som tjänar visst ändamål eller leder till visst resultat.
RASK	*Ramverk för säkerhetskravställning*, det konceptregelverk som beskrivs i denna rapport.
restriskanalys	Analys som genomförs för att bedöma den kvarstående risken efter att skyddsåtgärder vidtagits.
sammanvägningsfunktion	Matematisk funktion för att beräkna kravens styrkenivåer baserat på inparametrarnas värden.
specifik modell	En ifylld RASK-modell med inparametrar, sammanvägningsfunktioner och kravtexter.
styrkenivå	Representation av IT-systemets förväntade motståndskraft (skyddsförmåga) mot de hot som kravet hanterar.

Definitionerna av *process*, *metod*, *metodik* och *modell* är hämtade från Terminologicentrum (2002).

När detta skrivs håller MUST på att ta fram en uppdaterad version av KSF. Detta är ett arbete som är oberoende av det som presenteras i denna rapport, vilket således innebär att denna rapport inte behandlar den kommande versionen av KSF. För att undvika begreppsförvirring använder denna rapport namnen enligt följande:

- *KSF* – Avser KSF i version 3.1 eller tidigare.
- *KSF 3.1* – Avser KSF i version 3.1.
- *Ramverk för säkerhetskravställning (RASK)* – Avser förslaget till metodik för framtida regelverk som beskrivs i denna rapport.

3 Uppgiftsanalys

Detta kapitel ger en beskrivning och kort analys av de områden som identifierats för bearbetning inom ramen för studien. Kapitlet utgör bakgrund och motivering till RASK-metodiken som beskrivs i kapitel 5.

I föregående kapitel beskrivs ett antal dokumenterade problem och utvecklingsmöjligheter med det nuvarande regelverket. Det har inte genomförts någon fullständig analys av vilka områden som är viktigare eller hur de kan adresseras på bästa sätt. Då uppdraget har inneburit utveckling av ett alternativ till den befintliga KSF 3.1, har studiens föreslagna förändringar hållit sig inom den övergripande ramen för det som KSF 3.1 kravställer. Det innebär att andra åtgärder skulle kunna ge lika bra eller till och med bättre effekt, men detta har inte studerats.

3.1 Reducering av överkrav

I såväl uppdragsbeskrivningen som de tidigare beskrivna problemen och utvecklingsmöjligheterna finns tankar som handlar om att dagens KSF, och kanske utvecklingsprocessen som helhet, ställer för höga säkerhetskrav så att det blir onödigt svårt att få IT-system ackrediterade. Några frågor utifrån detta är om det verkligen är för svårt att få IT-system ackrediterade, om det i så fall beror på för höga säkerhetskrav och om det då är KSF som utgör grunden till problemet?

Frågorna ovan saknar i dagsläget tydliga svar. Författarna upplever dock att KSF får ta emot en del oförtjänt kritik, troligen beroende på att KSF är ett i sammanhanget jämförelsevis konkret dokument. Detta resulterar i att det ofta är när utvecklingsprojekten kommer till KSF-kraven som IT-säkerhetsarbetet hamnar i fokus. Det är då lätt att uppfatta det som att KSF är hindret, när det egentligen är annat som inte fungerat i utvecklingsprojektet. Faran är att problemen uppmärksammats först när det inte är möjligt att uppfylla de konkreta kraven från KSF.

Det är värt att notera att dagens regelverk i princip måste bestämma den gemensamma kravnivån utifrån det krav där behovet ligger på högst nivå baserat de två parametrarnas värden. Om den gemensamma kravnivån sätts lägre kommer vissa aspekter att underkravställas. En bättre anpassning av kravnivåerna till aktuella säkerhetsbehov kommer att minska problematiken med såväl över- som underkravställning.

Studiens inriktning för att förbättra dagens situation har utgått från en hypotes om att användandet av KSF resulterar i en överkravställning genom att vissa krav tidigt aktiveras på onödigt hög nivå trots att de faktiska säkerhetsbehoven är lägre. Detta är en rimlig tanke då KSF dels utgår från enbart två invärden vid fastställandet av kravnivå, dels använder samma kravnivå för alla krav på hela systemet. Det är svårt att tän-

ka sig att denna systemgemensamma nivå skulle vara rätt för alla systemets säkerhetsfunktioner. Om regelverket ger möjlighet att, i så många avseenden som möjligt, ställa krav på rätt nivå minskar risken för överkravställning. Grunden för att uppnå bättre anpassning av kravnivån är dels en noggrannare analys av systemet och dess kontext, dels ett mer noggrant val av vilka krav som ska aktiveras.

3.2 Öka förutsägbarheten

Ett problem som berörts i avsnitt 2.2 är oförutsägbarheten där systemutvecklarna har svårt att veta om de säkerhetslösningar som implementeras är tillräckligt bra för att systemet ska kunna ackrediteras. De krav som ges i KSF är ofta allmänt hållna och beskriver *vad* som ska finnas men inte *hur mycket* eller *hur starkt*. Ett krav som är aktivt för alla kravnivåer i KSF kan i praktiken betyda helt olika saker i termer av vilken sorts lösning som är acceptabel, beroende på vilken kravnivå systemet hamnat på. För detta finns ingen vägledning i KSF utan det är upp till systemutvecklaren att tolka KSF och förhandla med FMV eller MUST om vilken lösning systemet ska innehålla. Då antalet krav är stort och det på grund av hög arbetsbelastning kan vara svårt att få tillgång till rätt personer för att diskutera och förhandla så är denna otydlighet kostsam, både ur ekonomiskt och kalendertidsmässigt perspektiv.

Studiens inriktning för att förbättra dagens situation har varit att formulera varje krav i olika styrkenivåer som beskriver *hur mycket* eller *hur starkt* skyddet förväntas vara på denna styrkenivå. Formuleringen utgör således ett förtydligande av kravet och gör det lättare för systemutvecklaren att välja en lösning som är i paritet med systemets skyddsbehov.

3.3 Ökad kravförståelse

Kraven i KSF är kortfattade och ger endast begränsat stöd i att förstå vad kraven egentligen betyder och vad de syftar till. Varje kravgrupp inleds med en kort förklarande text, i form av ett kortfattat resonemang eller en motivering. Därefter kommer kravkomponenterna som utgör själva kravformuleringen. Det finns dock ingen förklaring eller motivering till varje enskild kravkomponent, vilket gör att det kan vara svårt att tolka dessa på ett korrekt sätt för att nå en effektiv lösning.

Studiens inriktning för att förbättra kravförståelsen har varit att komplettera varje kravtext med en förklaring samt en beskrivning av den kontext i vilken kravet ska tolkas. En förbättrad förståelse för kraven har potiential att ge flera positiva effekter. Ju bättre systemutvecklaren förstår kraven, desto bättre är möjligheterna att utforma säkerhetsmekanismer som på ett korrekt och effektivt sätt uppfyller systemets säkerhetsbehov. En god förståelse öppnar också för ett större mod att erbjuda nya, potentiellt bättre eller billigare, lösningar där systemutvecklaren kan känna trygghet i att de är tillräckligt bra för att bli godkända trots att de är oprövade inom Försvarsmakten.

När kraven ska förklaras och sättas i en kontext så uppmuntras den som formulerar kravdokumenten att noggrant klargöra vad som är den önskade effekten av kraven även för sig själv, vilket i sig kan medföra den positiva effekten att kraven skrivs så de blir tydligare och lättare att tolka.

3.4 Analys av kravuppfyllnad och restrisk

Ett överskådligt och smidigt arbetsflöde för uppföljning och analys av eventuell restrisk som kvarstår efter att åtgärder vidtagits är centralt för att stödja kontinuerlig säkerhetsbedömning under systemets livstid. Ett sådant arbetsflöde kan även stödja återanvändning av såväl arkitekturer som implementationer av system och systemkomponenter för nya uppgifter och miljöer. Vid tidig användning under livscykeln går det även att analysera effekterna som kraven ger i systemet för att kunna uppskatta hur komplexa lösningar som krävs.

Att analysera kravuppfyllnad eller restrisk kan vara värdefullt antingen för att undersöka ett enskilt system eller för att jämföra olika system. Det kan också vara värdefullt att kunna variera systemets förutsättningar så att kravställningen kan jämföras för ett system i olika kontexter. Genom jämförelsen går det att identifiera eventuella variationer som har mycket lägre kravnivåer men ändå är tillräckligt anpassade till systemkontexten för att vara acceptabla. Även då förutsättningarna förändras för systemet är det intressant att se vilken påverkan detta har på kravställningen och hur detta påverkar den faktiska kravuppfyllnaden för systemet under de nya förutsättningarna.

Studiens inriktning är att bygga metodiken med god möjlighet till uppföljning och analys, genom att ha detta som en viktig utgångspunkt när metodik och modeller utformas. Genom att metodiken ger ett bra stöd för uppföljning erhålls ett bättre utgångsläge för att genomföra kontinuerligt underhåll och löpande säkerhetshöjning av systemet.

4 Studiens genomförande

Utgångspunkten för studien var en förfrågan om att ta fram ett koncept för säkerhetskravställning. Studien avsågs ta fram en konkret, demonstrerbar metodik för hur säkerhetskravställning kan genomföras på ett sätt som hanterar de identifierade problemen och som söker uppfylla effektmålen som efterfrågades av Försvarsmakten.

Eftersom studien innebar ett arbete med att ta fram en konkret metodik så har arbetet genomförts enligt en tydligt framskridande plan, där arbete av vetenskaplig karaktär för att samla in en mer omfattande bakgrundsinformation inte har genomförts. Studien har istället förlitat sig på tidigare arbete av FOI för att få relevant bakgrundsinformation till utvecklingen av RASK-metodiken.

4.1 Arbetsflöde

Framtagningen av RASK-metodiken har genomförts i följande huvudsteg:

1. Samla bakgrundsinformation för att försöka förstå möjliga orsaker till de upplevda problemen med KSF 3.1.
2. Genomför intern workshop med diskussioner om tänkbara metoder och modeller utifrån bakgrundsinformationen.
3. Ta fram en första ansats till modell och genomför preliminära tester av metodikens lämplighet.
4. Genomför presentation för uppdragsgivaren med återkoppling på metodikens upplägg.
5. Ta fram en uppdaterad ansats till metodik samt populera modellen med innehåll.
6. Utveckla en demonstrator för att illustrera hur verktygsstöd kan underlätta användning av metodiken. I demonstratorn ingår att ta fram en enkel modell.
7. Redovisa resultat genom denna rapport och en presentation för uppdragsgivaren.

4.2 Avgränsningar

Eftersom ett regelverk för informationssäkerhet måste hantera många olika säkerhetsaspekter så valde vi att endast bestycka en begränsad del av konceptregelverket med krav. Vi valde att inrikta oss på tekniska krav för skydd mot skadlig kod samt assuranskrav kring testning och testtäckning. Övriga kravområden har inte beaktats.

Det ligger utanför studien att ta fram heltäckande krav, då detta inte ryms inom studiens ramar. Studien fokuserar istället på att ta fram hur modellen ska utformas och exemplifiera med krav inom olika områden genom modellen som används i demon-

stratorn. Den modellen utgår från befintliga krav i KSF 3.1 där fokus lagts på några få kravområden.

4.3 Identifiering av parametrar

För att identifiera parametrar som påverkar om ett hot är giltigt och således om krav och lösningar ska tillämpas så utfördes en mycket enkel hotanalys. Hotanalysen utfördes med tre olika typsystem i åtanke för att täcka behoven hos olika typer av system. Typsystemen utgjordes av (1) ett stort landsomfattande kontorssystem, (2) ett medelstort lokalt kontorssystem samt (3) ett mindre omfattande fordonsmonterat system. Hotanalysen är inte på något sätt fullständig utan utfördes som stöd för att identifiera en delmängd parametrar och lösningar till demonstratorns modell. Hotanalysens villkor för när hotet aktiverades och med vilken styrka hotet riktades mot de tänkta systemen gav de inparametrar och inparametervärden som användes i demonstratorn.

När konceptregelverket lyfts till att omfatta krav från alla relevanta områden, så måste hotanalysen vara utförlig och genomföras rigoröst eftersom hotanalysen utgör grunden till de krav som ställs i regelverket. Genom ett strukturerat arbete kan dessutom spårbarhet erhållas, från de konkreta hoten till kraven med tydligt syfte och mål, vilket underlättar framtida underhåll och uppdateringar av hotanalys och krav. Spårbarheten kan även underlätta framtagandet av beskrivningar och motiveringar till kraven.

4.4 Identifiering och formulering av krav och styrkenivåer

Kraven i demonstratorn baseras på krav i KSF 3.1 som har strukturerats om och formulerats utifrån RASK-metodikens förslag på kravformuleringar. I samband med att kraven formulerades så bestämdes även de matematiska funktioner som beräknar styrkenivåerna från inparametrarna.

I ett skarpt regelverk måste kraven och styrkenivåerna bestämmas utifrån hotanalysen.

5 RASK-metodiken

Ramverk för säkerhetskravställning (RASK) är en metodik som används för att fastställa och följa upp IT-säkerhetskrav. De krav som ges av metodiken är anpassade till förutsättningarna för respektive system. RASK utgörs av en samling metoder och modeller, vars användning är tänkt att integreras i kringliggande processer.

Huvudmålen i RASK-metodiken är att

1. reducera överkravställning genom att anpassa kraven till systemets förutsättningar
2. öka förutsägbarheten i utveckling och ackreditering genom att öka kravens tydlighet
3. öka förståelsen för kraven genom att ge mer bakgrund och motivering
4. möjliggöra effektiv analys och uppföljning av kravuppfyllnad och resterande riskbild över tid och under ändrade förutsättningar.

En mer detaljerad beskrivning av målen återfinns i kapitel 3.

Målen uppfylls genom att RASK-metodiken använder systemets förutsättningar, inklusive dess miljö, som indata för att bestämma vilka krav som ska ställas på systemet och vilken motståndskraft som lösningarna ska tillföra systemet. Detta uppnås genom en parametriserad modell som aktiverar krav på anpassade styrkenivåer utifrån inparametrar som beskriver systemets förutsättningar.

Kraven formuleras så att det tydligare framgår vad som ska uppnås, varför det ska uppnås och vilken styrka (motståndskraft) som lösningen på respektive krav förväntas uppnå. Vid behov indelas kraven i *styrkenivåer*, där kraftigare hotbild (större angreppsstyrka eller större men) motsvaras av förväntan på högre motståndskraft. Styrkenivåerna ger möjlighet till en mer finkornig uppföljning där kravuppfyllnaden inte enbart uttrycks som *uppfyllt* eller *icke uppfyllt*, utan i stället mäts på den skala som styrkenivåerna ger.

Modellen i RASK tas fram från en generell hotanalys som utförs av exempelvis MUST. Den generella hotanalysen utgår från en allmän hotbild för system av olika karaktär. Den allmänna hotbilden ger upphov till en uppsättning skyddsbehov, som sedan utgör grunden till kraven i modellen och hur dessa aktiveras.

Tanken är att användningen av RASK ska ske i befintliga processer, varför RASK inte tillför någon egen process. Integrationen innebär att RASK används inom de IT-processer som används inom Försvarsmakten och FMV. Genom utökat stöd för regelbundna uppföljningar kommer RASK att användas under flera livscykelfaser för systemen, från

Figur 1: Modellen som används för att bestämma säkerhetskraven utifrån ett antal inparametrar är grundstenen i RASK-metodiken.

bedömning i tidiga faser via faktisk kravbestämning till regelbundna uppföljningar under systemets livstid.

5.1 RASK-metodikens uppbyggnad

RASK-metodikens centralpunkt är den modell som används för att bestämma säkerhetskraven utifrån ett antal inparametrar, se figur 1. Systemets förutsättningar beskrivs genom inparametrarna till modellen, som i sin tur bestämmer vilka krav som aktiveras på vilken nivå under dessa förutsättningar. Internt bygger modellen på att *styrkenivåer* beräknas individuellt för varje krav utifrån inparametrarnas värden. Styrkenivåerna används sedan för att välja ut de specifika kravtexter som gäller för systemet.

Kring modellen byggs RASK-metodiken upp genom ett antal metoder för olika arbetsuppgifter, exempelvis kravställning och uppföljning. Metoderna används sedan av kringliggande processer där så är lämpligt, exempelvis vid tidiga genomförbarhetsstudier, kravställning och periodiska uppföljningar såväl under som efter systemets utveckling.

Mekanismen med inparametrar och nivåer påminner om den som används i KSF 3.1, där parametrarna *konsekvensnivå* och *exponering* vägs samman för att bestämma en gemensam *kravnivå* för alla säkerhetskrav. I RASK används fler inparametrar för att noggrannare beskriva systemets kontext samtidigt som styrkenivån beräknas individuellt för varje krav. Detta ger möjlighet att få en mer anpassad kravställning utifrån det aktuella systemets förutsättningar, vilket ökar sannolikheten för att erhålla rätt nivå på kraven. Om nivån bestäms gemensamt för alla krav eller för delmängder av krav finns en fara att systemet överkravställs och/eller underkravställs (se avsnitt 3.1).

RASK-metodiken innehåller metoder för olika arbetsflöden relaterade till kravställningen. Förutom bestämning av krav från inparametrar finns även metoder som stödjer uppföljning av kravuppfyllnad, hantering av ändrade förutsättningar (inklusive ändrad användning) och periodisk uppföljning av systemens säkerhetsnivå.

5.1.1 Inparametrar – systemets förutsättningar

Hot mot IT-system som hanterar någon form av tillgångar kan motverkas på många olika sätt. Vissa aspekter av hoten hanteras genom miljön, exempelvis med bevakning och låsta utrymmen, medan andra måste hanteras av systemet i sig. Genom att ta hän-

Tabell 2: Exempel på en inparameter med giltiga värden.

Inparameter	Högsta informationssäkerhetsklass som hanteras i systemet
Värden	1. Öppen
	2. Sekretessklassificerad
	3. Hemlig/Restricted
	4. Hemlig/Confidential
	5. Hemlig/Secret
	6. Hemlig/Top Secret

syn till de yttre förutsättningarna vid kravställningen undviks att systemet tvingas hantera hot som redan omhändertagits av miljön. Sådant som kan vara givet på förhand är exempelvis informationens skyddsvärde, den fysiska miljö som systemet ska verka i, vilka personer som får befinna sig i närheten av systemet och vilka andra IT-system som det ska kommunicera med.

När säkerhetskraven ställs utifrån systemets förutsättningar så har det betydelse för den skyddsförmåga som byggs in i systemet. Om miljön ändrar sig kan det innebära att systemet inte längre klarar av att hantera de hot som det utsätts för, då dess skyddsförmåga är anpassad till den ursprungliga miljön. Ändringar i förutsättningarna kan även innebära att systemet blir överkvalificerat i förhållande till hotbilden, om exempelvis det yttre hotet minskar eller om ett starkare yttre skydd tillkommer kring systemet. Förändrade förutsättningar betyder således att en ny analys av systemets skyddsbehov behöver göras.

I RASK beskrivs systemets yttre förutsättningar genom ett antal *inparametrar* som utgör indata till RASK-modellen. Inparametrarna kan anta förutbestämda värden som utgör ordinalskalor, vilket innebär att värdena är ordnade från lättare till svårare att hantera ur säkerhetsperspektiv. Ett exempel på en inparameter återfinns i tabell 2, där den högsta tillåtna informationssäkerhetsklassen i systemet kan väljas på skalan från *Öppen* till *Hemlig/Top Secret*.

Inparametrarna beskriver olika aspekter av systemet och dess kontext. Inparametrar kan som i exemplet ovan beröra informationen som ska hanteras, men de kan även beskriva miljön som systemet verkar i, den förväntade övergripande strukturen hos systemet eller hur systemet avses användas i verksamheten. Inparametrarna ska företrädesvis vara konkreta och lätta att förstå för den som använder RASK.

5.1.2 Krav med styrkenivåer

En utmaning med kravställningen i regelverk som KSF 3.1 är att den kravställer IT-system på generisk nivå och därför behöver vara giltig för alla typer av system inom Försvarsmakten. Detta innebär en ytterligare aspekt vid kravformuleringen, då det in-

te finns specifika verksamhetsbehov eller system att beakta. Dessa generiska krav kan också riskera att bli svårtolkade då de ställs utan relation till en systemkontext. Säkerhetskraven i RASK-modellen möter dessa utmaningar med utgångspunkt i två kravmetoder: *öppna målkrav* (eng. open-target requirements) (Lauesen 2002; Lauesen 2006) och *problemorienterade krav* (eng. problem requirements eller problem-oriented requirements) (Lauesen och Kuhail 2012).

Öppna målkrav syftade ursprungligen till att lösa problem vid kravställning av system som ska integreras i en befintlig miljö och som anskaffas via offentlig upphandling. Att hitta en rimlig nivå på kraven och att hitta formuleringar som inte exkluderar relevanta lösningar är extra utmanande när anskaffningsprocessen inte är iterativ, där det råder stora osäkerheter kring vad som är en lämplig lösning eller när det är osäkert hur uppfyllnad ska mätas. Lauesen (2006) menar att en risk med skall-krav vid kravställning av befintliga system eller komponenter är att skall-krav kan vara rimliga för vissa lösningar men orimliga för andra, samtidigt som syftet med kravet är uppfyllt i båda fallen. Skall-krav riskerar därmed att exkludera alternativa lösningar som i vissa fall är bättre eller billigare. Detta kan liknas med den generiska kravställning som sker via KSF. Öppna målkrav syftar till att lösa dessa problem genom formulering av mer flexibla krav, som även tillförs en beskrivning av vad beställaren förväntar sig av systemet. Till skillnad från traditionella skall-krav behöver inte leverantörerna uppfylla samtliga krav. Öppna målkrav ger möjlighet att även acceptera alternativa lösningar så länge de uppfyller förväntningarna som finns, vilket lägger fokus på resultaten snarare än på kraven.

Problemorienterade krav tar upp specifika problemställningar avseende användningen av systemet och påminner delvis om så kallade användningsfall. Istället för den sekventiella beskrivningen i ett traditionellt användningsfall så formuleras problemorienterade krav icke-sekventiellt genom de arbetsuppgifter som finns i samband med användningen. Varje arbetsuppgift har ett eller flera problem knutna till sig som behöver beaktas. Dessa problem kan således betraktas som krav för systemet.

RASK-modellen nyttjar aspekter från både öppna målkrav och problemorienterade krav för att skapa ett flexibelt och generiskt sätt att formulera säkerhetskrav. I varje krav ingår, förutom en grundläggande kravtext, även en beskrivning av förväntningar på lösningen och syftet med kravet. Denna kombination strävar efter att göra kraven mer förståeliga för systemutvecklarna och till att skapa ett utrymme för designavvägningar, samtidigt som det finns en tydlig inriktning som referenspunkt. Följande utgör ett exempel på hur krav kan formuleras med tillhörande förväntning:

Krav:	Vid detektering av potentiellt skadlig kod ska behörig administratör kunna notifieras.
Förväntning:	Det förväntas att administratörer notifieras regelbundet om skadlig kod har detekterats.

Genom att formulera olika förväntningar på hur kravet ska realiseras kan även samma krav användas för olika styrkor på de skydd som kravet innebär. För system som bedöms ha större skyddsbehov justeras förväntningen därefter, vilket exemplifieras nedan:

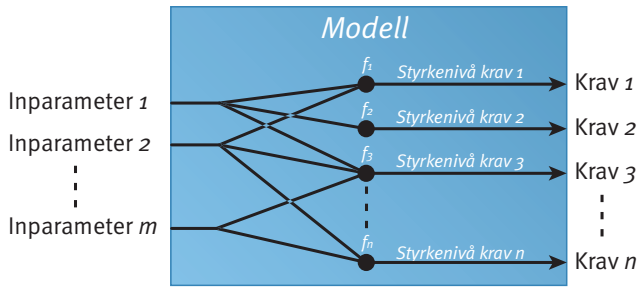
Krav:	Vid detektering av potentiellt skadlig kod ska behörig administratör kunna notifieras.
Förväntning:	Det förväntas att administratörer omedelbart informeras vid detektion av skadlig kod. Vid enanvändarsystem förväntas det finnas en administrativ rutin.

RASK-modellen använder sig av krav med tre styrkenivåer[2], där dessa motsvarar olika förväntningar på lösningen. Högre styrkenivå innebär att en mer motståndskraftig eller skyddande lösning förväntas i systemet.

Genom att kraven inte är absoluta i sin formulering utan bygger på de beskrivna förväntningarna öppnas det upp för en flexibilitet där leverantörer kan erbjuda likvärdiga eller bättre lösningar så länge dessa uppfyller förväntningarna. Sättet att skriva kraven ger även mer information till leverantörerna om vilken nivå som förväntas, vilket minskar risken för en alltför ambitiös och därmed kostnadsdrivande implementation.

För att stödja leverantören med att erbjuda alternativa lösningar som är i linje med förväntningarna så formuleras även syften kopplade till varje krav och vid behov till varje styrkenivå. Exempel på syfte på två olika nivåer ges nedan:

[2] Utöver de tre styrkenivåerna finns även deaktiverade krav samt permanenta krav (som alltid är aktiva och saknar ytterligare styrkeindelning). Se avsnitt 5.1.3 för mer information.



Figur 2: RASK-modellen bestämmer styrkenivåer från inparametrar genom sammanvägningsfunktionerna f_1–f_n.

Krav:	Vid detektering av potentiellt skadlig kod ska behörig administratör kunna notifieras.
Styrkenivå 1	
Förväntning:	Det förväntas att administratörer notifieras regelbundet om skadlig kod har detekterats.
Syfte:	Syftet är att uppmärksamma administratörerna att detektion skett för att möjliggöra analys och utredning.
Styrkenivå 2	
Förväntning:	Det förväntas att administratörer omedelbart informeras vid detektion av skadlig kod. Vid enanvändarsystem förväntas det finnas en administrativ rutin.
Syfte:	Syftet är att uppmärksamma administratörerna att detektion av skadlig kod skett, för att initiera ett administrativt agerande för att förhindra fortsatt skada eller spridning.

Detta sätt att formulera krav begränsar sig inte till funktionella (ofta tekniska) krav utan är även tillämpligt på andra kravtyper, såsom assuranskrav gällande dokumentation, utvecklingsmetodik, produktionssäkerhet, administrativa åtgärder och uppföljning.

5.1.3 Bestämning av styrkenivåer

Till varje krav finns en *sammanvägningsfunktion* som används för att beräkna styrkenivån baserat på aktuella inparametervärden. Figur 2 visar en schematiskt överblick över RASK-modellen, där sammanvägningsfunktionerna benämns f_1–f_n. Varje sammanvägningsfunktion kan använda en eller flera inparametrar som indata och ger styrkenivån för det associerade kravet som resultat.

Tabell 3: Giltiga styrkenivåer för krav i RASK-modellen.

Styrkenivå	Beskrivning
3	Högt krav
2	Medelkrav
1	Lågt krav
0	Deaktiverat krav (inget krav för aktuella parametervärden)
*	Permanent krav (alltid aktiverat och saknar styrkeindelning)

Tabell 3 visar de olika värden som styrkenivåerna kan anta i RASK-modellen, vilket även är de möjliga utvärden som sammanvägningsfunktionerna kan anta. Vissa krav faller bort under vissa förutsättningar, exempelvis kanske det inte behövs något inbyggt skydd mot fysiska angrepp om systemet består av en enda dator som alltid är under bevakning. Sådana förutsättningar leder till att kravet får styrkenivå 0, vilket innebär att kravet deaktiveras.

Det finns även krav som alltid är aktiverade med en specifik lydelse oavsett inparametrarnas värden och som därför inte delas in i olika styrkenivåer. Dessa utgör så kallade *permanenta krav* och indikeras med en asterisk (*). Asterisken kan liknas vid ett jokertecken (eng. wildcard) som matchar alla styrkenivåer, vilket i princip innebär att sammanvägningsfunktionens resultat ignoreras.

När arbetet med krav och sammanvägningsfunktioner påbörjades fanns ansatsen att kraven skulle grupperas i *kravområden* med gemensam styrkenivå, baserad på gemensamma grundbehov. Exempel på tänkbara kravområden är *skydd mot skadlig kod* och *fysiskt skydd*. Efter att ett antal olika krav och sammanvägningsfunktioner tagits fram så förkastades dessa kravområden då kraven inom de naturliga grupperingarna ofta inte passade ihop när det gällde sammanvägningsfunktionerna. Ett experiment gjordes för att se hur en matematiskt optimerad gruppering i kravområden skulle kunna se ut och då framkom att grupperna helt saknade pedagogiskt värde för användaren av metodiken, varpå själva idén med kravområden förkastades.

I arbetet med RASK-demonstratorn har sammanvägningsfunktionerna huvudsakligen bestått av kombinationer av de matematiska max- och min-funktionerna. Höga värden på vissa inparametrar lyfter skyddsbehovet till högre styrkenivåer medan låga värden på andra inparametrar i stället begränsar skyddsbehovet till lägre styrkenivåer. Ett exempel där nivån lyfts är högre tillgänglighetsbehov som leder till högre skyddsbehov i det generella fallet. På motsvarande sätt tenderar svaga hotagenter att ge lägre skyddsbehov för systemet.

5.1.4 Specifika modeller

RASK-metodiken utgör ett ramverk för kravställning av IT-säkerhetskrav, vilket innebär att metodiken i sig inte innehåller all den information som krävs för ett komplett regelverk. I RASK utgör *modellen* bäraren av den saknade informationen, det vill säga inparametrar, sammanvägningsfunktioner och krav. RASK beskriver således en *generisk modell* (se figur 2) som fylls med information för att skapa en *specifik modell*.

Den information som behöver tillföras för att erhålla en specifik modell är

- samtliga inparametrar med listor över giltiga värden
- sammanvägningsfunktioner för beräkning av styrkenivåer till alla krav
- alla krav med tillhörande kravtexter för respektive styrkenivå.

Två specifika modeller behöver inte ha något annat gemensamt än att de utgår ifrån den generiska RASK-modellen, vilket innebär att specifika modeller kan vara helt fristående från varandra. Detta kan vara fördelaktigt om olika specifika modeller tas fram för olika tillämpningsområden, exempelvis en modell för reguljära IT-system och en annan för säkerhetskomponenter. Även om RASK ger möjlighet till fristående modeller bör möjligheten nyttjas återhållsamt, då varje modell innebär arbete med underhåll samtidigt som möjligheten att jämföra system minskar om dessa använder olika modeller.

I och med att RASK-metodiken kan hantera flera specifika modeller blir det möjligt att periodiskt uppdatera dessa för att följa omvärldsläget. Periodisk uppdatering av specifika modeller ger möjlighet att genomföra en rationell uppföljning av läget i systemen över tid, för att på så sätt underlätta att systemen hålls på den önskade säkerhetsnivån under sin livstid. Att ta fram en specifik modell är således inte att betrakta som ett engångssteg, då den lämpligtvis behöver hållas uppdaterad över tid.

En specifik modell med samma tillämpningsområde som KSF 3.1 skulle utgöra huvudmodellen för IT-system inom Försvarsmakten. Utöver denna modell är det även möjligt att ta fram specifika modeller för andra tillämpningsområden, exempelvis för kravställning av säkerhetskomponenter och kryptosystem.

Varje specifik modell tar sin utgångspunkt i en *generell hotanalys*, som i sin tur ger det underlag som behövs för att ta fram inparametrar, matematiska funktioner och krav som ingår i modellen. Det finns inget givet sätt att genomföra generella hotanalyser, men det är viktigt att metoden tar hänsyn till den diversitet som kan förekomma bland systemen där modellen ska användas. Exempelvis kan hotanalysen utgå från en uppsättning representativa typsystem i olika typmiljöer.

Den generella hotanalysen bör utgöra ett strukturerat och väldokumenterat arbete, speciellt med tanke på att uppdateringar av modellen då kan genomföras utifrån en ”delta-hotanalys” där tidigare arbete kan återanvändas för att minska arbetsmängden.

Metoden för att genomföra hotanalysen och därmed för att ta fram de specifika modellerna ligger utanför det arbete som beskrivs i denna rapport.

5.1.5 Underlag för restriskanalys

Vissa metoder i RASK producerar underlag som kan användas vid en restriskanalys, där systemets kvarstående risker bedöms utifrån en samlad bild över alla systemets aspekter. I bedömningen utgör underlagen från RASK-metodiken endast en del, varför RASK-metodiken inte tar upp hur en sådan bedömning ska göras. Underlagen för restriskanalys kan användas vid bedömning av kvarstående risk oavsett när i systemets livscykel bedömningen sker.

RASK-metodiken baserar underlagen på hur väl systemet uppfyller de krav som ingår i den specifika modell som används. Krav som uppfyllts till kravställd styrkenivå antas hantera samtliga risker som de ska motverka och tas inte upp i underlagen. Underlagen tar således endast upp de ställda krav som systemet inte uppfyller fullt ut.

Underlaget som produceras är tänkt att ge stöd vid bedömningen av vilka miljöfaktorer som inverkar på de krav som inte uppfyllts. Genom att jämföra aktuella förutsättningar, representerade genom inparametrarna och deras värden, med aktuell kravuppfyllnad går det att få en bild över vilka områden som är problematiska och vilka åtgärder – i systemet eller miljön – som kan lösa problemen.

När underlaget för restriskanalys ska produceras appliceras RASK-modellens sammanvägningsfunktioner baklänges, varpå resultatet av beräkningarna ger en mängd av *parameterdeltan*. Ett parameterdelta utgör differensen mellan angivna inparametervärden och en uppsättning inparametervärden som skulle ge krav på de styrkenivåer som faktiskt uppfyllts. Då RASK-modellen generellt sett inte är entydig när den appliceras baklänges[3] så ger metoderna i regel en mängd parameterdeltan som vart och ett motsvarar skillnaden mellan kravställda och uppfyllda styrkenivåer för de krav som inte uppfyllts. Genom att analysera samtliga parameterdeltan tas olika scenarion fram för att kunna bedöma restrisken.

5.1.6 Verktygsstöd

Modellen som används i RASK är betydligt mer komplex än exempelvis modellen i KSF 3.1. Detta innebär att RASK-metodiken sannolikt kräver verktygsstöd för att ge full effekt och för att åstadkomma effektivt och rationellt arbetsflöde. Utan verktygsstöd är vissa delar av RASK-metodiken mer eller mindre ogenomförbara i praktiken;

[3] Matematiskt uttryckt är sammanvägningsfunktionerna för att beräkna styrkenivåer *surjektiva* men inte *injektiva*, vilket innebär att de inte är entydigt inverterbara.

speciellt kan metoderna för periodisk uppföljning av systemen bli för arbetskrävande.

Nackdelen med verktygsstöd, jämfört med en manuell metod, är att verktyget måste utvecklas och att det behöver underhållas mer aktivt än motsvarande manuella metod. Det är även viktigt att verktyget utformas med användarna i fokus, så att det kan bli det avsedda stödet i verksamheten snarare än ytterligare något som uppfattas som en belastning. Som för de allra flesta programvaror bör utvecklingen således ske i nära samarbete med användarna.

RASK-metodiken lämpar sig väl för verktygsstöd, då relationerna mellan parametrar och krav relativt enkelt kan beskrivas och implementeras genom de specifika modellerna samtidigt som metoderna bitvis är krävande att utföra manuellt. Verktygsstödet kan därmed underlätta arbetsflödet avsevärt.

I verktyget bör samtliga metoder i RASK-metodiken implementeras för att nå full effekt. Med en komplett metodik implementerad ger verktyget god möjlighet att både beskriva nuläge och genomföra utforskande användning där olika förutsättningar för systemet kan testas i varierande framtidsscenarion.

Verktygsstöd kan dessutom ge stöd med spårbarhet vid ändringar och underlätta jämförelser i samband med förändringar i förutsättningarna och vid uppföljningar. Verktygsstödet underlättar även uppdateringar av kravbilden då en ny, generell hotbedömning har resulterat i en uppdaterad specifik modell.

Utöver metoderna och modellen i RASK-metodiken finns det ytterligare funktionalitet som är lämplig att implementera i stödverktyget. Här följer några exempel på sådan funktionalitet:

- Versionshantering av användardata, exempelvis i form av inparametervärden, kravuppfyllnad och metadata. Versionshantering underlättar spårbarhet och ger större möjlighet att genomföra utforskande användning av RASK-metodiken. Det underlättar även den periodiska uppföljningen när historiken för systemet finns tillgänglig.
- Mallar i form av typsystem som kan användas för tidig uppskattning av den förväntade kravbilden för ett kommande system.
- Möjlighet att dokumentera godtagna avvikelser från kraven, exempelvis efter förhandlingar mellan systemutvecklarna och MUST. För uppföljningsarbetet är det viktigt att alla krav som ställs genom metodiken kan hanteras i verktyget, inklusive eventuella krav som omförhandlats eller tagits bort för det specifika systemet.
- Möjlighet att dokumentera implementerad lösning på kraven med lämplig detaljgrad för att underlätta den periodiska uppföljningen vid förändringar i såväl kravställning som implementation.

De delmoment där användandet av verktygsstöd rekommenderas tas upp i metodbeskrivningarna i följande avsnitt.

5.2 Metod för kravställning

Metoden som används för kravställning utgår från att systemutvecklaren väljer inparametervärden som sedan används för att bestämma säkerhetskraven. RASK-metoden för kravställning inkluderar följande steg:

1. *Bestämma inparametervärden* – Utifrån systemets förutsättningar ansätter systemutvecklaren lämpliga värden på samtliga inparametrar. Parametervärden för typsystem kan användas för att ge en utgångspunkt vid inmatningen.
2. *Beräka styrkenivåer* – Från inparametrarna beräknas styrkenivåer för samtliga krav med hjälp av sammanvägningsfunktionerna.
3. *Peka ut säkerhetskrav* – Styrkenivåerna används för att peka ut de säkerhetskrav som gäller för systemet.

Steg 2–3 är lämpliga att implementera med verktygsstöd.

Då kraven som ges av metoden är skrivna utifrån en generell systembild så är det möjligt att vissa krav inte är applicerbara på alla specifika system. Därför måste kraven gås igenom för att bedöma om dessa är relevanta för det specifika systemet. Krav som inte är relevanta bör förhandlas bort, exempelvis i samråd med MUST.

5.3 Metod för analys av kravuppfyllnad

RASK innehåller en metod för att ta fram underlag till en restriskanalys utifrån faktisk kravuppfyllnad. Genom att ange den styrkenivå som systemet uppnått för respektive krav kan metoden påvisa vilka faktorer i miljön och systemets förutsättningar som inte omhändertas helt på grund av att ställda säkerhetskrav inte har uppfyllts.

Underlaget som metoden producerar är tänkt att ge förståelse för de säkerhetsaspekter och miljöfaktorer som systemet inte hanterar fullt ut, för att på så sätt ge en bättre möjlighet att bedöma om dessa faktorer innebär en hanterlig risknivå för systemet eller om andra åtgärder måste vidtas. Metoden ger ett underlag med tydligare koppling mellan icke-uppfyllda krav och systemets förutsättningar i form av inparametrarna och deras värden.

Permanenta krav är alltid aktiverade med en specifik lydelse, vilket innebär att de inte går att koppla tillbaka till inparametervärden. Permanenta krav som inte uppfyllts bedöms utifrån de motiveringar och syften som angivits för kraven.

Analys av kravuppfyllnad genomförs i följande steg:

1. *Ange kravuppfyllnad* – Systemutvecklaren anger uppfyllnad för respektive krav. För krav som är indelade i styrkenivåer anges den högsta styrkenivå som uppfyllts, oavsett om denna är lägre, lika eller högre än kravställd nivå. Permanenta krav (som saknar styrkenivåindelning) anges som uppfyllda eller ej uppfyllda.
2. *Beräkna parameterdeltan* – Utifrån de maximala inparametervärden som är acceptabla vid den sammantagna kravuppfyllnaden beräknas parameterdeltan.
   a) Hitta alla krav som är indelade i styrkenivåer och inte är uppfyllda.
   b) Beräkna de högsta alternativa inparametervärden som resulterar i styrkenivåer som är lika med eller lägre än de uppfyllda styrkenivåerna.
   c) Förenkla mängden av parameterdeltan. De parameterdeltan som beräknats i de föregående stegen kan överlappa eller utgöra delmängder av varandra. Dessa sammanförs och förenklas för att ge ett minimalt antal olika uppsättningar av inparametervärden till nästa steg.
3. *Sammanställ underlag för restriskanalys* – Med utgångspunkt i de parameterdeltan som beräknats i föregående steg formuleras ett skriftligt underlag för restriskbedömning, baserat på skillnader mellan ursprungliga och alternativa inparametervärden kopplade till faktisk kravuppfyllnad. Notera att underlaget kan bli omfattande i de fall där många olika uppsättningar kvarstår efter föregående steg.

Steg 2–3 är lämpliga att implementera med verktygsstöd, då dessa steg är arbetskrävande att utföra manuellt.

Det är tänkbart att metoden kan ge många alternativ med olika inparametervärden som kan resultera i ohanterligt mycket information i sammanställningen. Detta problem hanteras lämpligen genom heuristiska metoder i steg 2c, exempelvis genom att begränsa avståndet mellan ursprungliga och alternativa inparametervärden.

5.4 Metod för hantering av ändrade förutsättningar

Säkerhetskraven kan ändras om förutsättningarna för systemet ändras, exempelvis om information med annan klassning ska hanteras i systemet eller om det ska placeras i en ny miljö. Metoden används för att identifiera de krav som ändras och därför behöver hanteras av systemet under de nya förutsättningarna.

Vid ändrade förutsättningar används följande metod:

1. *Ange ursprungliga inparametervärden* – Systemutvecklaren anger de inparametervärden som motsvarar systemets ursprungliga förutsättningar.
2. *Ange nya inparametervärden* – Systemutvecklaren anger de inparametervärden som motsvarar systemets nya förutsättningar.
3. *Frivilligt: Ange systemets kravuppfyllnad* – Systemutvecklaren anger kravuppfyllnad för ursprungligt system på det sätt som beskrivs i avsnitt 5.3, punkt 1. Denna punkt hoppas över om det inte finns någon kravuppfyllnad att förhålla

sig till, exempelvis om förutsättningarna ändras under pågående utvecklingsprojekt.

4. *Beräkna respektive kravställning* – Kravställningen beräknas för såväl ursprungliga som nya förutsättningar.
5. *Sammanställ underlag för jämförelse av kravställning* – Jämför kravställningen mellan de ursprungliga och nya inparametervärdena. Formulera ett skriftligt underlag som tar upp samtliga krav där kravställningen skiljer sig åt, med lydelse för både nytt och ursprungligt krav. Om ursprunglig kravuppfyllnad har angetts jämförs denna med den nya kravställningen varpå de nya kraven markeras som redan uppfyllda eller ännu ej uppfyllda.

Steg 4–5 är lämpliga att implementera med verktygsstöd.

5.5 Metod för periodisk uppföljning

Periodiska uppföljningar är ett sätt att kontrollera att systemets säkerhetsnivå ligger kvar på en acceptabel nivå över tid. Så länge den specifika modell som användes för kravställningen fortfarande är applicerbar går uppföljningen att göra på samma sätt som för en analys av kravuppfyllnad, se avsnitt 5.3. Om en ny generell hotbedömning har genomförts och resulterat i en ny specifik modell är det lämpligt att göra uppföljning mot denna så att nyidentifierade hot inte missas i uppföljningen. Eftersom inparametrar, sammanvägningsfunktioner och krav kan ändras mellan specifika modeller så kräver en sådan uppföljning att inparametervärden och kravuppfyllnad för systemet uppdateras till den nya modellen.

Metoden för periodisk uppföljning används för att göra en förnyad bedömning av systemets IT-säkerhetsbehov och jämföra dessa mot den faktiska kravuppfyllnaden. Genom att versionshantera både systemets data (inparametrar och kravuppfyllnad) och specifika modeller är det möjligt att bedöma befintliga system utifrån uppdaterade modeller med minimal arbetsinsats.

Den periodiska uppföljningen utgörs av följande steg:

1. *Välj specifika modeller att jämföra* – Systemutvecklaren väljer de specifika RASK-modeller som ska användas för uppföljningen. Typiskt används den modell som användes vid senaste godkännandet plus den senaste specifika modell som finns tillgänglig.
2. *Ange ursprungliga inparametervärden* – Systemutvecklaren anger de inparametervärden som motsvarar systemets förutsättningar vid senaste godkännande.
3. *Ange värden för ändrade inparametrar* – Om inparametrarna har ändrats mellan ursprunglig och uppdaterad specifik modell, så anger utvecklaren värden på de ändrade inparametrarna som motsvarar systemets förutsättningar.
4. *Ange systemets kravuppfyllnad* – Systemutvecklaren anger kravuppfyllnad för befintligt system på det sätt som beskrivs i avsnitt 5.3, punkt 1. Kravuppfyllnad anges mot ursprunglig kravställning.

5. *Beräkna respektive kravställning* – Kravställningen beräknas för både ursprunglig och uppdaterad specifik modell.
6. *Beräkna parameterdeltan* – Utifrån de maximala inparametervärden som är acceptabla enligt den uppdaterade specifika modellen och den angivna kravuppfyllnaden beräknas parameterdeltan. Detta görs på det sätt som beskrivs i avsnitt 5.3, punkt 2.
7. *Sammanställ underlag för restriskanalys* – Med utgångspunkt i de parameterdeltan som beräknats i föregående steg formuleras ett skriftligt underlag för restriskbedömning, baserat på den uppdaterade specifika modellen där skillnader mellan ursprungliga och nya inparametervärden kopplas till faktisk kravuppfyllnad. Notera att underlaget kan bli omfattande i de fall där många olika uppsättningar kvarstår efter föregående steg.
8. *Sammanställ underlag för jämförelse av kravställning* – Jämför kravställningen mellan ursprunglig och uppdaterad specifik modell. Formulera ett skriftligt underlag som tar upp samtliga krav där kravställningen skiljer sig åt, med lydelse för både nytt och ursprungligt krav. Kravuppfyllningen jämförs med den nya kravställningen varpå de nya kraven markeras som redan uppfyllda eller ännu ej uppfyllda.

Steg 5–8 är lämpliga att implementera med verktygsstöd, då dessa steg är arbetskrävande att utföra manuellt.

5.6 Exempel på underlag för restriskanalys

Detta avsnitt visar exempel på hur underlag för restriskbedömning tas fram och hur de kan användas. Exemplen bygger på ett mycket begränsat utsnitt ur en modell som endast visar hur styrkenivån bestäms för ett krav, *eget fysiskt skydd*. En komplett modell där kravet ingår består av många krav med tillhörande sammanvägningsfunktioner och relevanta inparametrar. Utsnittet är gjort av överskådlighetsskäl för att resonemangen som förs ska vara lätta att följa.

Sammanvägningsfunktionen för kravet på eget fysiskt skydd baseras på två inparametrar i form av det högsta skyddsvärdet hos information som systemet hanterar samt den fysiska miljö som systemet befinner sig i vid användning. Dessa inparametrar med giltiga värden ges av tabell 4. Sammanvägningsfunktionen från inparametrar till styrkenivå ges av tabell 5.

Exempelsystemets förutsättningar är att det hanterar *sekretessklassificerad* som högsta skyddsvärde hos informationen och att det befinner sig i en *bevakad* miljö. När dessa inparametervärden vägs samman enligt tabell 4 erhålls styrkenivå 2 för kravet *eget fysiskt skydd*.

Tabell 4: Utsnitt av inparametrar.

Inparameter	**Värden**
Skyddsvärde	Hemligt (H)
	Sekretessklassificerad (SK)
	Öppen (Ö)
Fysisk miljö	Oskyddad
	Bevakad
	Skyddad

Tabell 5: Styrkenivå för *eget fysiskt skydd*.

		Fysisk miljö		
		Skyddad	Bevakad	Oskyddad
Skyddsvärde	Ö	1	1	2
	SK	1	2	2
	H	2	3	3

5.6.1 Exempel: Ej uppnådd styrkenivå

Antag att det färdiga systemet endast uppnår styrkenivå 1 för *eget fysiskt skydd* istället för den kravställda styrkenivån 2. Då kan metoden för analys av kravuppfyllnad användas för att ta reda på vad som styr nivån för det ouppfyllda kravet, för att på så sätt få en bättre bild av vilka förutsättningar som inte hanteras av systemet.

Eftersom systemet når styrkenivå 1 för *eget fysiskt skydd* ges två parameterdeltan (det vill säga möjliga inparameterförändringar) för att matcha den uppfyllda nivån:

1. Inparameter *skyddsvärde* ändras till *öppen*.
2. Inparameter *fysisk miljö* ändras till *skyddad*.

Situationen kan hanteras på fyra olika sätt, där två av sätten ges av de möjliga inparameterförändringarna. De två andra sätten är generella.

1. Minska det högsta skyddsvärde som systemet hanterar till *öppen*.
2. Ändra användningsmiljön så den är att betrakta som en *skyddad* miljö.
3. Besluta om ett undantag från icke-uppfyllda krav om det är motiverat utifrån den verksamhet som systemet ska stödja och den risk som ges av de krav som inte uppfyllts.
4. Besluta att systemet inte får användas, givet de rådande förutsättningarna.

5.6.2 Exempel: Ändrade förutsättningar för systemet

Antag att det färdiga systemet får förändrade förutsättningar, exempelvis genom en förändrad driftmiljö eller om systemet ska användas i ett annat sammanhang. Då kan metoden för hantering av ändrade förutsättningar ge underlag för att bedöma den nya situationen.

I detta exempel antas att systemet ska användas för att hantera det högre skyddsvärdet Hemligt vilket ger att styrkenivå 3 ska uppfyllas för *eget fysisk skydd*. Om systemet uppfyllt den tidigare ställda styrkenivån för kravet kommer det vara designat för att hantera styrkenivå 2. Då skyddsvärdet är givet från de förändrade förutsättningarna, ges endast ett parameterdelta i detta fall:

1. Inparameter *fysisk miljö* ändras till *skyddad*.

Denna differens kan hanteras på följande sätt:

1. Ändra användningsmiljön så den är att betrakta som en *skyddad* miljö.
2. Besluta om ett undantag från icke-uppfyllda krav om det är motiverat utifrån den verksamhet som systemet ska stödja och den risk som ges av de krav som inte uppfyllts.
3. Besluta att systemet inte får användas under de nya förutsättningarna.

5.6.3 Exempel: Uppdaterad specifik modell

Antag att omvärldsläget förändrats då nya typer av sårbarheter och angreppsmetoder blivit kända. Detta medför att en uppdaterad specifik modell tas fram. I det generella fallet kan detta ge ändringar i vilka inparametrar som används, hur styrkenivåerna beräknas och kravens innehåll. I detta exempel antas att endast beräkningen av styrkenivåer påverkas vilket gör att de tidigare uppfyllda kraven inte längre är helt relevanta för systemet.

Tabell 6 visar hur den uppdaterade specifika modellen innehåller en skärpning som påverkar den styrkenivå som systemet ska uppfylla för kravet på *eget fysiskt skydd*. I tabellen framgår att styrkenivån för de specifika inparametervärden som gällde vid kravställningen har höjts till 3 från tidigare 2. Om systemet sedan tidigare når upp till styrkenivå 2 så finns följande parameterdeltan:

1. Inparameter *skyddsvärde* ändras till *öppen*.
2. Inparameter *fysisk miljö* ändras till *skyddad*.

På samma sätt som i avsnitt 5.6.1 fås då följande alternativ för att hantera situationen:

1. Minska det högsta skyddsvärde som systemet hanterar till *öppen*.
2. Ändra användningsmiljön så den är att betrakta som en *skyddad* miljö.

Tabell 6: Exempel: Skärpt styrkenivå för kravet *eget fysiskt skydd*. Ändrade värden är markerade med fetstil.

		Fysisk miljö		
		Skyddad	Bevakad	Oskyddad
Skyddsvärde	Ö	1	1	2
	SK	**2**	**3**	**3**
	H	**3**	3	3

3. Besluta om ett undantag från icke-uppfyllda krav om det är motiverat utifrån den verksamhet som systemet ska stödja och den risk som ges av de krav som inte uppfyllts.
4. Besluta att systemet inte får användas, givet den nya kravställningen.

5.6.4 Komplexitet i realistiska fall

Som tidigare påtalats bygger exemplen ovan på ett utsnitt ur en modell där endast ett krav hanteras. När underlag för restriskanalys tas fram för ett verkligt system kan potentiellt ett stort antal krav vara hanterade på såväl högre som lägre styrkenivåer än de som kravställts.

Då en inparameter kan påverka ett stort antal krav är det tänkbart att en liten förändring i systemets förutsättningar kan göra stor skillnad för kravställningen. I sådana fall är det också tänkbart att en mindre ändring kan göra stor nytta för restrisken, exempelvis kan bättre bevakning eller starkare skalskydd helt eller delvis mitigera ett flertal brister relaterade till systemets egna fysiska skydd. När kraven som inte uppfyllts fullt ut ligger inom olika områden är hanteringen mer komplex och flera olika åtgärder kan behövas för att hantera bristerna om dessa inte anses acceptabla för systemet.

Redan vid de enkla exempel som ges ovan finns det flera olika sätt att hantera den differens som uppstår när kraven inte kan uppfyllas eller när förutsättningarna ändras. Då en komplett modell innehåller flera inparameter och en större mängd krav är det mycket möjligt att parameterdifferensen blir komplex. Det är därmed viktigt att beskrivningen av differensen och vad den innebär blir lätt att förstå och förhålla sig till för den som ska ta ställning till eventuella åtgärder.

Verktygsstöd för att ta fram underlag för restriskanalys blir extra värdefullt i och med den komplexitet som kan uppstå. Verktygsstödet har möjlighet att avlasta stora delar av arbetet med att ta fram underlaget, så att fokus kan läggas på att bedöma underlaget tillsammans med de andra aspekterna såsom verksamhetsbehov och verksamhetsrisker.

Det finns ett stort batteri med tänkbara åtgärder för att mitigera en restrisk, exempelvis att ta fram en handlingsplan för hur systemet ska nå korrekt kravuppfyllnad eller

att acceptera risken under de rådande förutsättningarna. Ytterligare ett sätt är att villkora användningen till specifika omständigheter för att minska den resterande risken, något som kan vara tillräckligt för att systemet ska anses uppfylla kraven. Villkoren är då typiskt sådana att de tar hänsyn till förutsättningar som inte kan fångas av inparametrarna, exempelvis att systemet bara får användas i vissa miljöer.

6 Krav och kravhantering

I avsnitt 5.1.2 ges en introduktion till hur kraven i RASK formuleras utifrån ett semi-öppet beskrivningssätt, där målsättningarna med kraven och förväntningarna på lösningarna är centrala. Inspiration till sättet att formulera kraven är taget från *öppna målkrav* (eng. open-target requirements) och *problemorienterade krav*, två koncept som arbetats fram för att användas vid kravställning av mjukvara i samband med offentliga upphandlingar (Lauesen 2006; Lauesen och Kuhail 2012).

Öppna målkrav innebär att kraven skrivs på ett öppet sätt, där målsättningen med kravet samt relevant kontext ges av kravtexten. Detta innebär ett ökat ansvar hos systemutvecklaren (anbudsgivaren vid en upphandling) att besvara kraven utifrån den lösning som denne avser ta fram, för att påvisa att dennes lösning uppfyller förväntningarna utifrån kraven. Sättet att skriva krav innebär en större flexibilitet att nå kravets mål på olika sätt beroende på förutsättningarna för respektive systemlösning.

Kravtexterna ska i största möjliga mån undvika att föreskriva specifika lösningar eller specifik systemdesign, även om det är möjligt (och i vissa fall rekommenderat) att ge högnivåförslag på lösningar som förtydligande i kravtexten. I de fall där en specifik lösning krävs, exempelvis vid integration med ett befintligt system, kan detta anges med en sluten kravtext. Detta alternativ ska dock användas sparsamt då det inte är givet att denna lösning är fördelaktig vare sig verksamhetsmässigt eller ekonomiskt. Det kan vara bättre att öppna för att systemutvecklaren ersätter en befintlig lösning, exempelvis om denne redan har en passande, färdigintegrerad lösning.

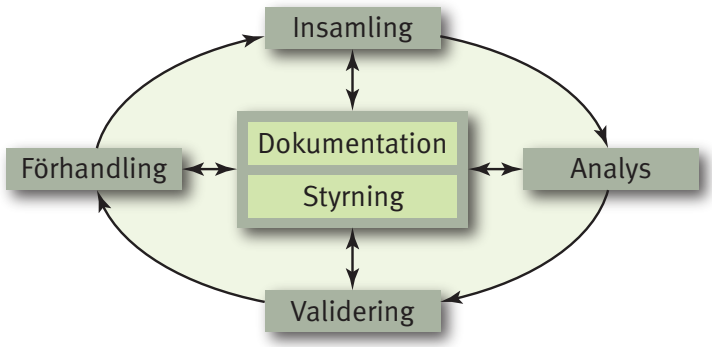
6.1 Kravarbetet

I RASK-metodiken finns det två distinkta kravarbeten – arbetet för att ta fram och hantera säkerhetskrav som ingår i en specifik RASK-modell samt arbetet för att ta fram och hantera säkerhetskrav för respektive IT-system. Medan kraven till en specifik modell tas fram genom en generisk hotanalys som inte har så många beroenden utåt, så är säkerhetskraven på IT-systemet endast en del i den totala kravmängden. För systemen består kraven typiskt av en sammansättning av olika typer av krav från många olika källor. Förutom IT-säkerhetskraven kan det exempelvis ingå verksamhetskrav, personsäkerhetskrav, miljökrav, dokumentationskrav, underhållskrav, utbildningskrav och processkrav.

Oavsett vilket regelverk IT-säkerhetskraven tas fram utifrån är det viktigt att inse att de endast utgör en delmängd av den totala uppsättningen krav. I alla realistiska system kommer det att finnas krav som står i konflikt med varandra vilket innebär att kompromisser nästan alltid krävs för att kunna bygga fungerande system.

En viktig aspekt när krav från olika källor sammanställs är spårbarhet – att det tydligt går att se vem som är huvudintressent i kravet, var behovet som ligger bakom kravet uppstår och vad syftet är med kravet. Behovet av spårbarhet blir extra tydligt när krav hamnar i konflikt, då det är viktigt att kunna gå tillbaka till de ursprungliga intressenterna för att förhandla om kraven. För att underlätta hanteringen av kraven (inklusive spårbarhet) är det fördelaktigt att använda ett kravhanteringsverktyg.

6.2 Kravprocessen



Figur 3: Kravprocesscykeln (Sommerville 2005)

Kravprocessen är inte en enkel linjär process där intressenterna förmedlar sina behov som sedan i ett steg sammanställs till perfekta krav. Verkligheten ligger närmare den kravprocesscykel som Sommerville (2005) beskriver och som återfinns i figur 3. Cykeln utgör en iterativ process som fortgår under hela systemets livstid, från första idé till sista uppdatering.

Figur 3 visar att insamlade krav analyseras för att förstå kravens innebörd samt eventuella överlapp och konflikter. Sedan valideras kraven för att säkerställa att kraven speglar de faktiska behov som intressenterna har. Efter detta krävs en förhandling för att nå samsyn i intressenternas bild av systemet och där kravkonflikter hanteras. Under hela processen krävs det styrning (eng. management) och dokumentation av kraven.

I kravarbetet ingår det att samla in såväl behov som regelverksbaserade ramkrav från samtliga relevanta intressenter. I kravarbetet ingår att jämka ihop dessa intressenters behov så att systemet kan nå önskad effekt för verksamheten inom givna ekonomiska och regelmässiga ramar. Följande lista tar upp ett antal av de intressenter som kan vara relevanta när det gäller IT-system:

- verksamheten
- förvaltningsorganisationen
- organisationen för kontinuerlig uppföljning (exempelvis logganalyser)

- driftsorganisationen
- säkerhetsstyrning (MUST)
- ekonomistyrning
- leverantörer.

Intressesfärer och behov hos de olika intressenterna står i många fall i konflikt med varandra. Exempelvis kan verksamheten ha ett behov där många arbetsplatser vore fördelaktigt medan detta står i konflikt med den ekonomiska ramens begränsningar. Säkerhetsbehoven avseende sekretesskydd kan ofta stå i konflikt med enkelhet i användning, något som påverkar verksamheten när personal och information inte kan röra sig fritt.

7 Demonstrator för metodik och verktygsstöd

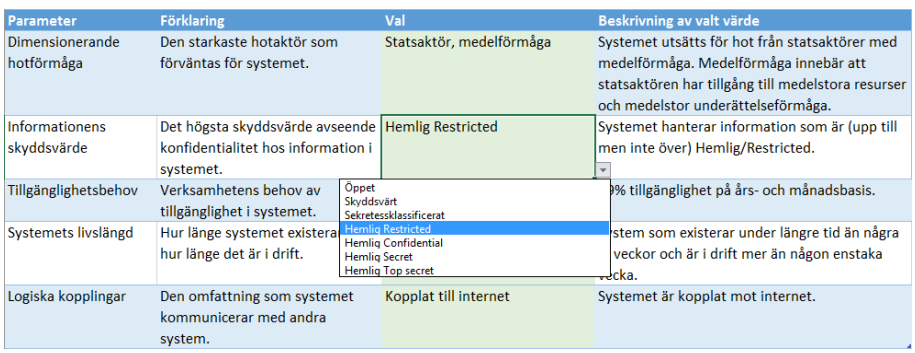
Under projektet har en demonstrator utvecklats för att testa RASK-metodiken och exemplifiera hur den fungerar samtidigt som demonstratorn visar de fördelar som kan fås med verktygsstöd. Demonstratorn har gett möjlighet att undersöka om metodiken fungerar i praktiken genom att ge en konkret miljö där idéerna kan testas. Arbetet har visat att grundidéerna i RASK fungerar väl och att användningen av metodiken kan bli smidig med verktygsstöd. Det har även varit rättframt att fylla modellen med krav och kravtexter, baserat på krav från KSF 3.1 och den mycket enkla hotanalys som genomförts i projektet.

Demonstratorn har utvecklats som en arbetsbok i Microsoft Excel 2013 och använder makron skrivna i Visual Basic for Applications (VBA).

Demonstratorn är en förenkling av det verktygsstöd som behövs för att metodiken ska få komplett stöd, där det främst är användbarhet och funktioner för uppföljning som fått stå tillbaka i demonstratorutvecklingen. Demonstratorn har stöd för inmatning av inparametrar, kravställning, inmatning av kravuppfyllnad samt enkla uppföljningsfunktioner. Ett exempel på inmatning av inparametervärden återfinns i figur 4.

Följande funktioner är implementerade i demonstratorn:

- inmatning av inparametervärden
- beräkning och presentation av kravställning



Figur 4: Inmatning av inparametervärden i demonstratorn.

Uppfylld nivå mot **Kravställd nivå**

Kravställd nivå	Uppfylld nivå 0	1	2	3
0	16	0	0	0
1	0	11	0	0
2	0	0	8	1
3	0	0	1	3

Figur 5: Exempel på förenklad uppföljning i demonstratorn.

- beräkning och presentation av kravställningar baserade på alternativa inparametervärden
- inmatning av kravuppfyllnad
- uppföljning av differens mellan kravuppfyllnad och kravställning.

En funktion som av tekniska skäl inte är implementerad i demonstratorn är produktion av underlaget för restriskbedömning. Funktionen kräver betydligt mer texthantering i verktyget, något som Excel inte lämpar sig särskilt väl för.

Uppföljningen sker på ett förenklat sätt, där skillnader mellan kravställning och kravuppfyllnad färgmarkeras i en översiktlig tabell över samtliga krav. Skillnaderna sammanställs också i tabeller som visar fördelningen av uppfyllda, underuppfyllda och överuppfyllda krav. Ett exempel på en sådan sammanställning återfinns i figur 5. I exemplet visas ett fall där de flesta kraven uppfyllts på ställd nivå, medan ett har underuppfyllts (kravställd nivå 3, uppfylld nivå 2) och ett har överuppfyllts (kravställd nivå 2, uppfylld nivå 3). Underuppfyllda krav markeras med röd bakgrundsfärg medan överuppfyllda krav markeras med grön bakgrundsfärg.

Demonstratorn är avsedd att visa hur metodiken kan användas och antyda vilka fördelar ett verktygsstöd ger. Flera funktioner behöver läggas till eller signifikant förbättras för att nå ett fullgott verktygsstöd. Exempelvis saknas stöd för att ta fram restriskunderlag och för att hantera olika versioner av de specifika modellerna.

8 Diskussion

IT-system är en viktig byggsten i Försvarsmaktens verksamheter och allt fler materielsystem blir beroende av fungerande och pålitliga IT-system. Utifrån Försvarsmaktens förutsättningar – att verka under ett militärt hot från främmande makt – blir IT-säkerheten avgörande för IT-systemens möjlighet att fungera i ett utsatt läge. Dagens regelverk för kravställning av IT-säkerhetsfunktioner, *Krav på IT-säkerhetsförmågor hos IT-system v3.1. (KSF 3.1)* (Försvarsmakten 2014), har kritiserats för att bland annat vara otydligt, inte ta tillräckligt noggrann hänsyn till systemens förutsättningar, att ha för starkt sekretessfokus och att inte ge tillräckligt utrymme för avvägningar gentemot verksamhetsbehoven.

Kritiken mot att dagens regelverk är otydligt återkommer vidare i kritik om att tolkningen av regelverket upplevs som personberoende och att det därmed finns en variation i vad som anses vara tillräckligt för att uppfylla kraven. Otydligheten upplevs därmed påverka projekten negativt, där exempelvis svårigheter att bestämma hur olika säkerhetsfrågor ska lösas leder till förseningar och att fokus hamnar på sådant som inte ger verksamhetsnytta.

Regelverket utgör dock endast en del i ett större ekosystem med processer, metoder och regelverk som används vid utveckling och förvaltning av Försvarsmaktens IT-system. KSF är relativt oberoende i förhållande till de processer som omgärdar regelverket, varför det bör vara förhållandevis lätt att arbeta in ett alternativt regelverk för kravställning av IT-säkerhetsfunktioner. Ett alternativt regelverk ger sannolikt endast liten påverkan på arbetsprocessen när det gäller den avgränsade del av utvecklingsprocessen som avser kravställning av IT-säkerhetsfunktioner. Men när blicken lyfts till hela utvecklings- och förvaltningsarbetet kan ett alternativt regelverk ge stor effekt, exempelvis genom förbättrad tydlighet i hur kraven ska tolkas och förenklad uppföljning av kraven. Med en större samsyn på kravens tolkning mellan olika aktörer (exempelvis MUST, verksamhetsrepresentanter, FMV och leverantörer) kan tidskrävande kravdiskussioner och dyra omtag undvikas samtidigt som fokus kan flyttas till de systemfunktioner som ger verksamhetsnytta.

Att förändra processer inom Försvarsmakten tar tid, bland annat därför att påbörjade projekt som regel fortsätter att följa befintliga processer och regelverk genom hela utvecklingsarbetet. Att byta processer och regelverk under ett projekts genomförande kostar pengar samtidigt som bytet kan medföra ökad risk för oklarheter och förseningar, vilket gör att bytet kan möta motstånd.

8.1 Effektmålen

Arbetet med RASK-metodiken har tagit sikte på de effektmål som efterfrågades av Försvarsmakten, se avsnitt 1.2, och strävan har varit att ge så bra uppfyllnad av effektmålen som möjligt. I praktiken finns det flera faktorer som påverkar uppfyllanden av effektmålen förutom metodiken i sig, exempelvis den specifika modell som används och det verktygsstöd som finns tillgängligt.

I följande diskussioner om effektmålen tas KSF 3.1 som utgångspunkt för jämförelser.

8.1.1 Korrekta designval och realistiska förväntningar

Genom att metodiken ger möjlighet att fånga upp en mer detaljerad bild av systemets kontext finns det utrymme för att arbeta med avvägningar mellan olika miljöfaktorer. Under förutsättning att inparametrarna i den specifika modellen är tillräckligt detaljerade går det att göra avvägningar mellan exempelvis en hanterlig säkerhetskravställning, lämplig användningsmiljö och önskad informationssäkerhetsklass. Verksamheten kan välja att justera en eller flera inparametrar för att nå en lämplig kompromiss där så behövs. Genom att kombinera denna möjlighet med mer utförliga kravtexter, där även en syftet med kravet framgår, öppnar metodiken för en ökad förståelse för säkerhetskravställningen och hur den relaterar till systemkontexten. Ökad förståelse ger bättre förutsättningar för att hitta en lämplig avvägning mellan krav från olika intressenter.

Den ökade förståelse som uppstår i och med tydligare krav med dokumenterade motiveringar ger även bättre förutsättningar för att ge realistiska förväntningar på systemen och hur systemen kan användas. När kraven tydligare pekar på vad systemet behöver uppnå i form av säkerhetslösningar blir konflikter mellan krav från olika intressenter tydligare. Detta kan ge en tidigare indikation på olika önskemål som blir problematiska i praktiken, exempelvis att i samma system hantera information som bör vara åtskild eller att använda samma system i helt olika kontexter. Genom att tidigt få indikation på vad som är möjligt och vad som är olämpligt blir det lättare att hålla förväntningarna på systemet på en realistisk nivå.

Det sätt att formulera kraven som beskrivs i avsnitt 5.1.2 är inte specifikt för RASK-metodiken, utan kan nyttjas oavsett metodik. Tydligare krav med fokus på vilka problem systemet ska lösa – snarare än på hur dessa ska lösas – ger utvecklaren frihet att bygga systemet utan att låsas in i fördefinierade lösningar som kan vara icke-optimala. Krav som formuleras så objektivt mätbart som möjligt ger mindre tveksamheter vid bedömning av om kraven är uppfyllda, vilket gör utvecklingen mer förutsägbar och leder till bättre precision i de designval som görs under utvecklingsarbetet.

En nackdel med tydligare formulerade krav är att de kräver betydligt mer arbete när den specifika modellen tas fram. Detta arbete behöver dock bara genomföras en gång

oavsett hur många system den specifika modellen används till, varför det torde vara en rimlig arbetsinsats sett till den totala vinst som tydligare krav kan innebära.

8.1.2 Instinktiv förståelse för riskerna

Att ge förståelse för risker kan vara ganska svårt, speciellt om det gäller risker som upplevs som abstrakta. Många IT-säkerhetsrisker upplevs dessutom inte bara som abstrakta utan kan även vara subtila på så sätt att en liten och till synes irrelevant detalj är det som skapar risken. Därmed är det i många fall svårt att ge en instinktiv förståelse för IT-säkerhetsrisker.

Genom sin ökade tydlighet i kraven och med metoden för att ta fram underlag för restriskanalys har RASK-metodiken möjlighet att förbättra förståelsen för eventuell kvarstående riskbild. I RASK-metodiken är det den specifika modellen som innehåller information som ligger till grund för restriskunderlaget. I praktiken krävs det dessutom verktygsstöd för att ta fram restriskunderlaget, vilket innebär att modellen och verktyggstödet har stor inverkan på hur bra riskerna kan förstås.

8.1.3 Stöd för kontinuerligt säkerhetsarbete

Ett IT-system som inte uppdateras kontinuerligt utgör en stor säkerhetsrisk, då sårbarheter i gemensamma systemkomponenter och mjukvaror fortlöpande upptäcks och ny kunskap om olika potentiella säkerhetsbrister blir kända. Även ur ett funktionellt perspektiv är det viktigt att kunna uppdatera systemet, exempelvis för att rätta fel och för att lägga till nya funktioner.

RASK-metoden för restriskanalys ger möjlighet att genomföra periodiska uppföljningar, där systemet fortlöpande kan stämmas av mot säkerhetskraven. Restriskanalysen ger indikationer på eventuellt ökade risker som kommer genom förändringar i systemet. Genom att använda uppdaterade specifika modeller ger restriskanalysen även en indikation på den ändring i riskbild som kommer av förändringar i hotbild och allmänt kunskapsläge, så som det fångas i den specifika modellen.

Metoden för hantering av ändrade förutsättningar ger även den stöd för det kontinuerliga säkerhetsarbetet i de fall där systemet används i olika kontexter. Med verktygsstöd kan metoderna för restriskanalys och ändrade förutsättningar i kombination med versionshanterade specifika modeller ge en överskådlig sammanställning av systemets IT-säkerhetsläge och den associerade restrisken.

8.1.4 Jämförelser mellan system

I vissa lägen finns behov av att jämföra olika IT-system för att se vilket av dem som är lämpligast för en uppgift, exempelvis när befintliga system ska användas för att fylla ett nytt behov. Jämförelsen av IT-säkerhetsnivån kan förenklas genom att basera den-

na på vilka krav som respektive system uppfyller, för att på så sätt kunna se skillnader mellan systemen och jämföra med den kravnivå som följer av det nya behovet.

De befintliga systemen som jämförs kan ha byggts utifrån olika förutsättningar och tagits fram vid olika tidpunkt, vilket kan ge svårigheter när systemen ska mätas mot varandra. Genom RASK-metodikens stöd för versionshanterade specifika modeller och metoderna för periodisk uppföljning och hantering av ändrade förutsättningar underlättas jämförelsen där de befintliga systemen kan mätas mot varandra eller utifrån uppdaterade krav.

8.1.5 Hantera både sekretess, riktighet och tillgänglighet

Den mest konkreta säkerhetsaspekten är sekretess, där det i praktiken endast ingår ett enda mål med skyddet – att hindra obehöriga från att ta del av sekretessbelagd information. De andra aspekterna är mer komplexa, exempelvis omfattar riktighet både den initiala tillförlitligheten (ursprunglig korrekthet) hos information och att skydda den från obehörig modifiering när den befinner sig i systemet. Säkerhetsaspekten tillgänglighet är även den komplex, då innebörden egentligen är att informationen[4] ska vara tillgänglig för den som behöver informationen när den behövs. Tillgänglighet kan hanteras på många olika sätt men kan sällan garanteras fullt ut.

I KSF 3.1 är kraven på skyddsåtgärder endast kopplade till sekretess (Försvarsmakten 2014, s. 12), vilket också avspeglas i hur konsekvensnivån definieras. Konsekvensnivån är den enda parameter i KSF som beskriver informationen i systemet och dess skyddsvärde. Genom att RASK-metodiken öppnar upp för fler inparametrar finns även möjlighet att ta in delar av tillgänglighets- och riktighetsaspekterna, i den mån dessa går att generalisera. I demonstratorn har detta illustrerats genom en inparameter som fångar tillgänglighetsaspekten på en skala som inspirerats av så kallade servicenivåavtal[5].

Genom att inparametrarna ger möjlighet att fånga in en större del av systemets kontext än i KSF ges en ökad möjlighet att hantera både sekretess, riktighet och tillgänglighet i regelverk som baseras på RASK-metodiken.

8.2 Processintegration

Hur metoderna som beskrivs i denna rapport ska integreras i övriga processer är inte klarlagt. Det är tydligt att integrationen måste göras för att nå den verkan som öns-

[4] I vissa fall kan det vara funktionen som upplevs vara viktig ur tillgänglighetsaspekt, exempelvis i ett brandlarm. Dock är det ofta så att det ändå handlar om information – att mottagaren får reda på att det brinner, snarare än att funktionen ”larm” är det viktiga.

[5] Servicenivåavtal (eng. service level agreement, SLA) är avtal som ingås mellan en tjänsteleverantör och en kund där det specificeras hur stor andel av tiden som systemet måste vara tillgänligt. Ofta specificeras tillgänglighetsgraden som en procentandel av en fast tidsperiod, exempelvis en månad.

kas med ett nytt regelverk. Följande punkter beskriver några gränsytor som finns mot processer inom IT-området i Försvarsmakten:

- *Verksamhetsanalys* – Ger ingångsvärden för att bestämma verksamhetsnära inparametervärden i RASK.
- *Säkerhetsanalys* – Ger indata för att bestämma säkerhetsrelevanta inparametervärden i RASK. Använder säkerhetskraven från RASK för att komplettera med systemspecifika säkerhetskrav.
- *Kravhantering* – Tar hand av säkerhetskraven från RASK och sammanställer dessa med övriga krav på systemet. I detta ingår även analys och hantering av säkerhetskrav som står i konflikt med krav från andra områden.
- *Kravorienterad riskanalys* – Använder RASK för återkommande analyser av säkerhetsnivån. Här finns även möjligheten att undersöka hur systemet kan uppfylla säkerhetskraven utifrån de designbeslut som tas under utvecklingsarbetet.
- *Restriskbedömning* – Använder RASK för uppföljning av hur väl säkerhetskraven uppfylls och hur detta relaterar till säkerhetsläget och verksamhetens behov.
- *Återkommande säkerhets- och riskanalyser* – Använder RASK för regelbundna uppföljningar av förändringar i system och omvärld genom upprepad bedömning av restrisk.

8.3 Från metodik till regelverk

Denna rapport beskriver RASK-metodiken utifrån ett generellt perspektiv, där modellen inte har fyllts med det innehåll som behövs för att skapa ett komplett regelverk. Den demonstrator som tagits fram visar att metodiken är praktiskt användbar och att det är ett genomförbart – om än omfattande – arbete att gå från metodik till regelverk.

För att nå ett komplett regelverk återstår exempelvis att ta fram lämpliga inparametrar, kravtexter och sammanvägningsfunktioner. Dessutom krävs att regelverket integreras med de kringliggande processerna och etableras i organisationen. För att få full potential från RASK-metodiken måste även ett genomtänkt verktygsstöd tas fram.

För att ge möjlighet till kontinuerlig utvärdering och löpande förbättring av de delar som tas fram är det lämpligt att bryta ner arbetet i mindre arbetspaket. Här följer några exempel på sådana arbetspaket:

- Ta fram en guide för kravformulering som utgår från hur öppna målkrav och problemorienterade krav tas fram. Denna guide är användbar oavsett om RASK-metodiken anammas eller inte.
- Ta fram en komplett modell med inparametrar, sammanvägningsfunktioner och kravtexter. Denna kan antingen utgå från en helt ny generell hotanalys eller utgå från den hotanalys om ligger till grund för befintlig KSF.
- Utveckla en fullskalig demonstrator. Detta är ett steg på vägen mot ett fullskaligt verktygsstöd.

- Ta fram en guide till livscykelhantering av specifika modeller inklusive riktlinjer för hur förändringar i modellen ska göras så att jämförelser mellan olika versioner underlättas.
- Identifiera relevanta gränsytor mellan RASK-metodiken och andra processer/metoder för att få kunskap om var metodiken behöver integreras.

8.4 Ett helikopterperspektiv

Regelverket för IT-säkerhetskravställning – oavsett om det är KSF, RASK eller något annat – utgör endast en liten byggsten i IT-säkerhetsarbetet inom Försvarsmakten. Med RASK söker vi nå förbättringar inom de områden där regelverket har verkan, något som förhoppningsvis märks i systemutvecklingsprojekten, exempelvis genom lägre kostnader och större förutsägbarhet i bedömningarna.

Vi tror dock att kritiken som riktats mot KSF och svårigheterna att få IT-system godkända är symptom på djupare, ännu oidentifierade, svårigheter med IT-säkerhetsarbetet inom Försvarsmakten, vilket skulle kunna förklara bredden och variationen bland de problem som refereras i avsnitt 2.2. Det är därmed troligt att det finns andra förändringar än en omstrukturering av regelverket som skulle kunna ge storskaliga och avgörande effekter på Försvarsmaktens IT-utvecklingsarbete.

För att identifiera och bearbeta dessa djupare svårigheter krävs en annan typ av utredning som i ett samlat grepp genomlyser hela Försvarsmaktens arbete med IT-säkerhet, med avsikt att ta reda på vad som fungerar bra och vad som kan förbättras. Utifrån den kunskap som samlas in finns sedan möjlighet att välja ut åtgärder som kan ge bra effekt för att underlätta utveckling och underhåll av Försvarsmaktens IT-system.

Litteratur

Bengtsson, J., T. Sommestad och H. Holm (2014). *IT-säkerhetskrav i Försvarsmakten: KSF3 och tillkommande säkerhetskrav*. Tekn. rapport FOI-R--4000--SE. FOI, Totalförsvarets Forskningsinstitut.

Eidenskog, D. (2017). *Framtagning av nya säkerhetskrav för IT-system – Problembeskrivning*. Tekn. rapport FOI Memo 6131. FOI, Totalförsvarets Forskningsinstitut.

Försvarsmakten (2013). *Fastställande Handbok IT-processen*. HKV 09 100:60203.

Försvarsmakten (2014). *KSF, Krav på IT-säkerhetsförmågor hos IT-system, v3.1*.

Gudmundson Hunstad, A. (2016). *Informationssäkerhetsegenskaper: Avvägningar och prioriteringar*. Tekn. rapport FOI-R--4341--SE. FOI, Totalförsvarets Forskningsinstitut.

Gudmundson Hunstad, A. (2017). *Rutiner och regelverk för att godkänna IT-system inom Försvarsmakten*. Tekn. rapport FOI-R--4526--SE. FOI, Totalförsvarets Forskningsinstitut.

Gudmundson Hunstad, A., T. Gustafsson, H. Karlzén, F. Mörnestedt och L. Westerdahl (2012). *Objektbaserad säkerhet: Behov och möjligheter*. Tekn. rapport FOI-R--3484--SE. FOI, Totalförsvarets Forskningsinstitut.

Hermelin, J., N. Hallberg, C. Stenius, P. Nilsson, L. Westerdahl och J. Allgurén (2017). *Försvarsmaktens IT-styrning: Nulägesanalys*. Tekn. rapport FOI-R--4449--SE. FOI, Totalförsvarets Forskningsinstitut.

Karlzén, H., D. Eidenskog och J. Löfvenberg (2017). *Rutiner och regelverk för att godkänna IT-system inom Försvarsmakten*. Tekn. rapport FOI-R--4423--SE. FOI, Totalförsvarets Forskningsinstitut.

Lauesen, S. (2006). ”COTS Tenders and Integration Requirements”. I: *Requirements Engineering* 11 (2). DOI: 10.1007/s00766-005-0022-5.

Lauesen, S. (2002). *Software Requirements – Styles and Techniques*. Addison-Wesley Longman, Inc.

Lauesen, S. och M. A. Kuhail (2012). ”Task Descriptions Versus Use Cases”. I: *Requirements Engineering* 17.1, s. 3–18. DOI: 10.1007/s00766-011-0140-1.

Sommerville, I. (2005). ”Integrated Requirements Engineering: A Tutorial”. I: *IEEE Software* 22 (1). DOI: 10.1109/MS.2005.13.

Svenmarck, P. (2017). *Litteraturöversikt av samspelet mellan människa, teknik och organisation för IT-säkerhet*. Tekn. rapport FOI-R--4425--SE. FOI, Totalförsvarets Forskningsinstitut.

Terminologicentrum (2002). *Process, metod, metodik, modell*. URL: http://www.tnc.se/termfraga/process-metod-metodik-modell/ (hämtad 2018-10-04).

FOI är en huvudsakligen uppdragsfinansierad myndighet under Försvarsdepartementet. Kärnverksamheten är forskning, metod- och teknikutveckling till nytta för försvar och säkerhet. Organisationen har cirka 1000 anställda varav ungefär 800 är forskare. Detta gör organisationen till Sveriges största forskningsinstitut. FOI ger kunderna tillgång till ledande expertis inom ett stort antal tillämpningsområden såsom säkerhetspolitiska studier och analyser inom försvar och säkerhet, bedömning av olika typer av hot, system för ledning och hantering av kriser, skydd mot och hantering av farliga ämnen, IT-säkerhet och nya sensorers möjligheter.

FOI
Totalförsvarets forskningsinstitut
164 90 Stockholm

Tel: 08-55 50 30 00
Fax: 08-55 50 31 00

www.foi.se