Ansvarig enhet

SPL

Handläggare

Dan Olofsson

Beslutande

Harmonisering mellan FMV ISD-process och FM KSF 3.1

Kravreducering och arkitektur

Innehållsförteckning

INLEDNING

1.1 Syfte

Denna delrapport syftar till att stötta designansvariga i samband med framtagning av nya IT-tjänster/-produkter eller uppgradering av befintliga IT-tjänster/-produkter.

Rapporten fokuserar på steget från STAU2 till TS (Technical Specifikation, sv: Teknisk specifikation) samt SoW (Statement of Work, sv: Verksamhetsåtagande (VÅ)) och ska ses som ett stöd i framtagningen av en lämplig systemarkitektur (med avseende på informationssäkerhet).

En stor del av rapporten upptas av frågan kring kravreducering, och med detta avses sänkning av kravnivå enligt MUST KSF 3.1 i syfte att få en kostnadseffektiv lösning utan att ge avkall på de grundläggande kraven.

Det finns i princip två sätt att göra detta på; reducera konsekvensen av informationsförlust alternativt reducera exponeringen av systemet. Båda dessa faktorer samverkar i valet av en sund och tillräckligt säker systemarkitektur.

1.2 Bakgrund

När nya IT-system ska tas fram inom Försvarsmakten ska framtagandet göras i enlighet med IT-processen. En del i detta arbete är ta fram den säkerhetsmålsättning som används som underlag när Militära underrättelse- och säkerhetstjänsten (MUST) uttalar sig om huruvida det planerade IT-systemet bedöms få en tillräcklig nivå avseende informationssäkerhet eller ej.

De ingångsvärden som finns att tillgå vid skapandet av en välbalanserad och begriplig säkerhetsmålsättning är säkerhets- och verksamhetsanalys. FM Säkerhetsmålsättningen ger vidare ingångsvärde vid applicering av MUST Krav på säkerhetsfunktioner (KSF). I KSF specificeras de miniminivåer av IT-säkerhetskrav som Försvarsmaktens system förväntas uppfylla.

I ISD-processens STAU1 fastställs den gällande kravbilden (kravfångst) utifrån beställarens och koordinatorns kravbild.

STAU2 syftar till att allokera säkerhetsfunktioner på ett sådant sätt att kravbilden uppfylls på ett adekvat och kostnadseffektivt sätt. I STAU2 fastställs också den arkitektur som ska gälla för det specifika systemet.

1.3 Sammanfattning

I MUST KSF 3.1 finns en korrelation mellan informationssäkerhetskrav och -arkitektur, som inte funnits i tidigare versioner av KSF. Korrelationen mellan krav och arkitektur består i att kravmängden till stor del bestäms av arkitekturen då ett systems kravnivå är en direkt följd av systemets bedömda exponering och konsekvens.

Kravreducering
Genom att modulera med de två faktorerna exponering och konsekvens är det möjligt att reducera kravnivån.

Reducering av exponering kan ske på ett antal olika sätt, exempelvis genom att förändra arkitektur eller på annat sätt förändra den tekniska lösningen. Exponeringen kan också reduceras genom att kravställa

driftmiljön eller begränsa systemets användning istället för tekniken. Då förändras inte den faktiska systemlösningen men kraven på omgivningen och/eller användarna sänker exponeringen.

För att reducera konsekvensnivån, och därmed få möjlighet att sänka kravnivån för hela systemet, krävs en genomgång av den information som finns i systemet i syfte att minimera mängden känslig information i systemet med utgångspunkt i verksamhetens krav och behov. Om den känsliga informationen i systemet inte är absolut nödvändig för verksamheten måste denna informations tillgänglighet i systemet ställas mot den ökade kostnad det innebär att konstruera, dokumentera och vidmakthålla systemet utifrån högre ställda krav. Efter denna analys kan den känsliga informationen sedan antingen plockas ut ur systemet eller flyttas till en eller flera separata zoner/säkerhetsdomäner.

Värt att notera är att konsekvensbedömningen görs av olika aktörer beroende på om systemet omfattar hemlig/utrikesklassad information eller enbart ej sekretessklassad information. Bedömningen av konsekvens görs redan vid informationens inplacering i informationssäkerhetsklass då det gäller ett system med H/R-information eller högre. Konsekvensbedömningen i ett öppet system görs inte förrän vid K/E-analysen.

Designprinciper

I arbetet med arkitektur finns ett antal designprinciper att förhålla sig till, baserade på *best practice* och beprövade arbetssätt inom mjukvaruarkitektur. Följande principer beskrivs i detta dokument, med fokus på ett informationssäkerhetsperspektiv:

- **Minimalism** - Säkerhetsrelevanta funktioner ska inte göra något mer än nödvändigt
- **Least privilege** - Användare och funktioner ska endast ha nödvändiga rättigheter
- **Redundancy** - Funktioner ska ha kapacitet för att hantera hög belastning och systemfel (inklusive fysiska fel)
- **Defence in depth** - Multipla funktioner ska samverka på olika nivåer för att upprätthålla säkerheten
- **Self-protection** - Funktioner ska inte ha onödiga beroende till andra funktioner/domäner
- **Controlled data flow** - Informationsutbyte ska följa väl definierade mönster, som är under central kontroll
- **Balanced strength** - Det ska finnas en balans i styrkan hos säkerhetsfunktionerna i hela systemet
- **Hardening** – Alla funktioner som inte stödjer systemets primära syfte ska vara avstängda/avinstallerade

Utöver dessa principer finns det andra riktlinjer, såsom:

- **Economy of mechanisms** - Säkerhetslösningar måste hållas enkla, okomplicerade och begränsade till sin omfattning
- **Open design** - Säkerhetslösningar måste vara öppna i design och implementation och med så få ”hemligheter” som möjligt. Dessutom ska säkerhetslösningar inte förlita sig på skydd från endast kända hot.
- **Ease of use** - Säkerhetslösningar måste vara enkla (användarvänliga) och tillräckligt enkla för att undvika riskabla workarounds. Transparenta säkerhetslösningar bör eftersträvas.

Arkitekturarbete

Projektgruppen har även tittat på skillnaderna mellan traditionella projekt och agil utveckling, då ett iterativt arbete med krav och arkitektur i ISD-processen kommer att krävas. Grundtankarna bakom agil utveckling bygger på ett iterativt arbetssätt med ett mycket nära samarbete mellan utvecklaren och beställaren eller mottagaren. Ett agilt arbetssätt innebär också att mängden förarbete innan utvecklingen startar minimeras. Agila projekt fokuserar på att involvera verksamheten under hela projektet, inte bara i början och slutet. Förutom en studie av agil utveckling berörs också inkrementella och iterativa metoder.

Kravreducering genomförs i de initiala stegen i ISD- processen. Det första steget är inom beställarens område då denna ska kravställa nödvändiga förmågor hos systemet. Det andra kravreduceringssteget sker hos utföraren koordinatorn som kravställer och upphandlar. För att kunna avgöra vilka förändringar som är möjliga att göra behöver FMV efter genomgång av krav och behov från FM återkoppla till verksamheten. Detta måste ske för att få svar på vilka förändringar kopplat till exponering och konsekvens som är möjliga att genomföra och med bibehållen funktionalitet. Alternativt måste verksamheten godkänna avkall på funktionalitet för att exempelvis sänka kostnader kopplat till utveckling, granskning och test. Det krävs således ett iterativt arbete mellan dessa båda aktörer.

Slutsatser

- Om det finns möjlighet reducera att sänka kravnivån för ett system från hög till utökad eller ännu hellre från hög till grund så finns stora ekonomiska fördelar att uppnå.
- En lägre kravnivå kan också medföra att konstruktion, test och granskning av systemet tar mindre tid i anspråk vilket gör att verksamheten snabbare kan få tillgång till systemet.
- Det kan finnas tillfällen då sänkning av kravnivån inte är önskvärd, exempelvis då systemet är tänkt att kommunicera med andra system med en viss sekretessnivå eller då säkerhets- och/eller verksamhetsanalys kravställer en högra skyddsnivå än vad KSF anger.
- Vid en eventuell reducering av kravnivå är det viktigt att beakta den färdiga systemlösningen. Risk finns att IT-systemet inte längre kan användas på rätt sätt om funktionalitet eller information plockats bort.
- För att ISD-processen ska kunna förhålla sig till och på ett effektivt sätt kunna arbeta med KSF 3.1 kommer det att behövas iterationer inom arkitekturarbetet.

2 BASFAKTA

2.1 Dokumenthistorik

Version	Datum	Beskrivning	Handläggare
1.0	2015-10-22	Första utgåvan	SPL/SP Metodstöd, Dan Olofsson

2.2 Använt underlag

ID	Dokument	Beteckning
[1]	Beslut angående etablering av ISD-processen	FMV skr 2014-02-25, 14FMV47-10:1
[2]	KSF, Krav på IT-säkerhatsförmågor hos IT-system, v3.1	HKV skr 2014-06-13, FM2014-5302:1
[3]	IT-säkerhetskrav i Försvarsmakten KSF3 och tillkommande säkerhetskrav	FOI skr FOI-R-4000-SE December 2014
[4]	Sammanställning FM MUST KSF 3.1 krav	FMV skr 2015-10-22, 15FMV9326-2:1

2.3 Begrepp och definitioner

Begrepp	Betydelse
COTS	Commercial off-the-shelf
E	Exponering, enligt KSF definition, anges i nivå E1-E4
GFE	Government furnished equipment
GOTS	Government off-the-shelf
IDS	Intrusion detection system
IPS	Intrusion prevention system
IRS	Interface requirement specification
ISD	Informationssäkerhetsdeklaration
ISU	Informationssäkerhetsunderlag
ITSS	Informationssäkerhetsspecifikation, del av KSF 3.1
K	Konsekvens, enligt KSF definition, anges i nivåer K1-K5
KSF	Krav på säkerhetsfunktioner
MOTS	Military off-the-shelf
SiF	System i fokus
SoW	Statement of Work
STAU	Säkerhetstekniskt ackrediteringsunderlag, anges i nivåer STAU1-STAU4
TS	Technical specification

3 MUST KSF 3.1 OCH ISD-PROCESSEN

I MUST KSF 3.1 [2] finns en korrelation mellan informationssäkerhetskrav och -arkitektur, som inte funnits i tidigare versioner av KSF. Då detta bedöms vara en av de större metodmässiga förändringarna och utmaningarna i arbetet med KSF 3.1 har projektgruppen valt att ägna fas 2 av projektet åt att klargöra hur ett sådant krav- och arkitekturarbete bör bedrivas på ett effektivt och säkerhetsmedvetet sätt.

Korrelationen mellan krav och arkitektur består i att kravmängden till stor del bestäms av arkitekturen då ett systems kravnivå är en direkt följd av systemets bedömda exponering och konsekvens. Ju lägre exponering och konsekvens fastställs vara desto lägre blir kravnivån. Förhållandet mellan exponering, konsekvens och kravnivå åskådliggörs i tabellen nedan. Kravnivåerna är enligt KSF 3.1 graderade Grund(G), Utökad (U) samt Hög (H).

Konsekvensnivå	Exponeringsnivå			
	E1	E2	E3	E4
5	H	H	H	H
4	U	H	H	H
3	U	U	U	H
2	G	U	U	U
1	G	G	G	G

Tabell 1 Kravnivåer

Det borde alltså vara möjligt att reducera den mängd krav som ställs på ett system genom att sänka exponeringsnivån och/eller konsekvensnivån. En av grundfrågorna som ställts under arbetet med denna rapport har således varit hur man sänker exponering och konsekvens för ett system och hur detta påverkar den faktiska kravmängden för systemet.

Ett antal frågor har legat till grund för arbetet, exempelvis:

- Vilka motiv finns till att sänka kravnivån?
- Vad är vinsten med att sänka kravnivån från Hög till Utökad respektive från Utökad till Grund?
- Hur sänks exponeringsnivån?
- Hur sänks konsekvensnivån?
- Hur uppnås spårbarhet i kravarbetet?
- Är det alltid önskvärt att sänka kravnivån på ett system?
- Vad måste uppmärksammas vid kravreducering?
- Hur borde arbetet med kravreducering ske inom ISD-processen?

4 KRAVREDUCERING

För att kunna reducera kravmängden för ett system och således nå ekonomiska och tidsmässiga fördelar krävs att de ingående faktorer som avgör kravnivån är möjliga att påverka. Dessa faktorer är som bekant konsekvens och exponering.

4.1 Olika reduceringssteg

KSF-kraven är grupperade i tre nivåer; Grund, Utökad och Hög. Generellt ökar mängden krav vid kravställning från Grund och uppåt. Utöver fler krav, kan vissa krav komma att skärpas och det finns också hela klasser som tillkommer vid nivåökning (främst inom assuranskrav).

4.2 Vinsterna med kravreducering

Vilken kravnivå som gäller för en IT-tjänst, ett IT-system eller en IT-produkt bestäms av förhållandet mellan Exponering (E) och Konsekvens (K), enligt **Tabell 1**.

Genom att modulera med de två faktorerna E och K, kan det vara möjligt att reducera kravnivån. Syfte och/eller orsaker avseende detta kan t ex vara:

- Användande av COTS/MOTS/GOTS, som kan vara svåra eller dyra att påverka
- Kostnadsreducering avseende säkerhetsfunktion (-er)
- Minskad komplexitet i säkerhetslösningen
- Användning av arvssystem

Vid en eventuell reducering av kravnivå är det viktigt att beakta den färdiga systemlösningen. Risker finns att IT-systemet inte kan användas på rätt sätt då det inte uppfyller den tänkta användningen. För att hantera denna risk kan en dialog med *Beställare* och *Nyttjare* behöva föras, så att *Utföraren* anskaffar/implementerar en korrekt systemdesign/systemarkitektur. Det kan också finnas vinster med att skapa en systemlösning bygger på tekniska lösningar vilka är kända av användarna och driftorganisationen. Om det är möjligt kan ingångsvärden avseende systemarkitekturen inhämtas från användare/driftorganisation. Detta görs då lämpligen i i verksamhetsanalysfasen.

4.3 Sammanfattning av skillnader i kravnivå

Referens [4] innehåller en översiktlig beskrivning av vilka krav som ställs på en viss funktion/förmåga för de olika kravnivåerna avseende de funktionella säkerhetskraven och assuranskraven.

Samtliga säkerhetsfunktioner beskrivs med utgångspunkt från Grund-nivå.

Observera att assuranskraven i kapitel 4.3.2 endast tar upp krav som berör leverantörer, dvs krav på dokumentation (D-komponenter) och dess innehåll (C-komponenter). Det tillkommer yterliggare krav avseende evaluering (E-komponenter) enligt MUST KSF. Dessa evalueringskrav berör dock inte leverantörerna, och är således inte förtecknade i kapitel 4.3.2.

4.3.1 Funktionella krav

På den utökade nivån finns 124 krav, 41 fler än på Grund-nivån (som innehåller 83 krav). Två grundläggande krav (SFID_DAT.8 och SFIS_INT.1) försvinner eftersom de ersätts av mer krävande formuleringar men i övrigt är alla grundläggande krav kvar.

I korthet handlar de 43 nya kraven om:

- separation av roller/behörigheter (SFBK_ADM.2-3)
- autentisering och sessionshantering (SFBK_AUT.14-17)
- skyddsmärkning av objekt (SFBK_ÅTK.5-6)
- tidssynkronisering med externa säkerhetstjänster (SFGK_TID.3)
- realtidsanalys av loggar (SFID_ANA.9-13 och SFSL_ANA.10-11) och gemensam analysfunktion för loggar (SFID_DAT.9-10)
- omfattning och hantering av säkerhetsloggar och driftloggar (SFSL_OAV.1, SFSL_REG.5 och SFSL_SKY.4-7)
- bättre härdning av maskiner (SFIS_HRD.3-4)
- procedurer för att verifiera riktigheten på systemet (SFSK_RIK.3 och SFSK_UPD.4)
- starkare autentisering av intern kommunikation (SFIS_INT.2-4)
- mer kontroll och begränsning på informationsutbyte (SFOA_KBL.3, SFSK_KIN.3-4, SFIS_KIN.5-6 och SFIS_KUT.3-7)

På den högsta nivån finns det 140 krav, 16 fler än på den utökade nivån. Sex av de utökade kraven försvinner och 22 tillkommer. Inga nya typer av krav tillkommer, men kraven blir skarpare på flera områden.

Skillnaderna jämfört med utökad nivå är främst:

- striktare krav på behörighetshantering och separation av roller (SFBK_ADM.4-6)
- starkare autentisering och kontinuerlig behörighetskontroll (SFBK_AUT.18-20 och SFBK_ÅTK.7)
- mer ingående härdning av systemet (SFIS_HRD.5-7)
- ännu mer skydd och kontroll av kommunikation (SFOA_KBL.4, SFIS_INT.5, SFIS_KIN.7-8)
- automatiska kontroller av riktigheten i systemet (SFSK_EXE.7, SFSK_RIK.4, SFSK_UPD.5) istället för manuella
- krav på oavvislighet (SFSL_OAV.2-3, SFSL_REG.6)
- krav på fristående system för logganalys (SFID_ANA.14)

4.3.2 Assuranskrav

Kraven på struktur och innehåll i ITSS (SASS) förändras inte mellan olika kravnivåer (G/H/U).

Skillnaderna mellan Grund och Utökad är främst:

- Systemutvecklingsdokumentation tillkommer. Fokus på säkerhetsrelaterade komponenter. (SALC_UTV)
- Krav på spårbarhet i konfigurationsledningssystemet (SALC_KFG)
- Systemets riktighet ska kunna verifieras av mottagaren samt efter leverans (SALC_LEV)
- Ökade krav på ingående komponenters lämplighet avseende säkerhetspåverkan (SALC_LCM)
- Tillkommande krav på beskrivning av SiF gränsytor och dess säkerhetsfunktioner (SADE_GRÄ)
- Systemutvecklaren ska beskriva systemets säkerhetsarkitektur (SADE_ARK)
- Ökade krav på ingående komponenters lämplighet avseende säkerhetspåverkan (SADE_DFA)

- Systemutvecklaren skall tillhandahålla designdokument för systemet. Dokumentationen ska omfatta alla säkerhetsrelevanta komponenter. (SADE_DES)
- Tillkommande krav avseende verifiering av korrekt installation av SiF (SAOP_INS)
- Tillkommande krav avseende separation av administratörsroller samt förstärk administration av intrångsskydd (SAU_ÅTK)
- Införande av krav avseende rutiner för säkerhetskopiering och förvaring av säkerhetsloggar (SARU_INT)
- Tillkommande krav avseende CCB (SARU_KNF)
- Krav på att testtäckningsanalysen ska visa hur testfallen överensstämmer med säkerhetsfunktionella krav och att alla dessa krav omfattas av testerna samt att systemutvecklaren ska tillhandahålla testdokumentation (SATS_TTK, SATS_FUN, SATS, ANG)

Skillnaderna mellan Utökad och Hög är främst:

- Striktare krav på acceptanskriterier samt krav på integrationsdokumentation gällande alla ingående komponenter (SALC_UTV)
- Utökade krav på sårbarhet i konfigurationsledningssystemet (SALC_KFG)
- Livscykelmodellen ska motsvara ISO 9001 och påvisa komponternas (säkerhetspåverkande) lämplighet (SALC_LCM)
- Systemutvecklaren ska tillhandahålla en dataflödesanalys för samtliga komponenter (SADE_DFA). Systemutvecklaren skall tillhandahålla designdokument för systemet. Dokumentationen ska omfatta samtliga komponenter i systemet (SADE_DES)
- Utökade krav avseende separation av administratörer för behörighetskontroll (SARU_ÅTK)
- Utökade krav avseende test av säkerhetsrelevanta komponenter (SATS_TTK)

4.4 Exponeringsbedömning

Med exponeringsnivå avses bedömningen av hur utsatt systemet är för olika aktörer och vilken möjlighet dessa aktörer har att påverka systemet. Denna möjlighet kan vara såväl fysisk, d.v.s. att någon kommer åt systemets tekniska utrustning, som logisk via systemets olika gränssnitt.

Ökade risker för någon hotaktör att påverka systemet definieras som en högre exponeringsnivå, vilket i sin tur medför högre krav på systemets säkerhetsförmåga och därmed också en högre kostnad för att konstruera, dokumentera, granska och vidmakthålla systemet.

4.4.1 Exponeringsnivå

Ett systems exponering uttrycks i fyra nivåer med E1 som lägsta och E4 som högsta exponeringsnivå. Ökade möjligheter för någon aktör att påverka systemet definieras som en högre exponeringsnivå, vilket i sin tur medför högre krav på systemets säkerhetsförmåga. Kriterierna för de fyra exponeringsnivåerna framgår av Tabell 2.

Exponeringsnivå	Kriterier för exponeringsnivå		
	Tillgång till systemets fysiska och logiska gränsytor		**Informationsutbyte**
E4	Alla fall som inte uppfyller kriterierna för någon av exponeringsnivå E1-E3 nedan.		Alla fall som inte uppfyller kriterierna för någon av exponeringsnivå E1-E3 nedan.
E3	Alla personer med tillgång till någon av systemets gränsytor är säkerhetsprövade.	och	Samtliga system som systemet utbyter information med är ackrediterade till en högre konsekvensnivå. eller Samtliga system som systemet utbyter information med är ackrediterade till samma konsekvensnivå med högst exponeringsnivå E3.
E2	Alla personer med tillgång till någon av systemets gränsytor är behöriga till någon information inom den högsta konsekvensnivån som behandlas i systemet.	och	Samtliga system som systemet utbyter information med är ackrediterade till en högre konsekvensnivå eller Samtliga system som systemet utbyter information med är ackrediterade till samma konsekvensnivå med högst exponeringsnivå E2.
E1	Alla personer med tillgång till systemets gränsytor är behöriga till all information som behandlas i systemet.	och	Systemet utbyter ingen information med andra system.

Tabell 2 Exponeringsnivåer

Exponeringsnivån är den lägsta nivån för vilken det som anges för båda kriterierna, d.v.s. *Tillgång till systemets fysiska och logiska gränsytor* och *Informationsutbyte*, är uppfyllt.

Exempel på hur resonemang kring bedömning av exponering kan föras står att läsa i KSF [2].

4.4.2 Reducering av exponering

Reducering av exponering kan ske på ett antal olika sätt, exempelvis genom att förändra arkitektur eller på annat sätt förändra den tekniska lösningen. Exponeringen kan också reduceras genom att kravställa driftmiljön eller begränsa systemets användning istället för tekniken. Då förändras inte den faktiska systemlösningen men kraven på omgivningen och/eller användarna sänker exponeringen.

Det är dock viktigt att se till att den förändrade systemlösningen fortfarande uppfyller verksamhetens behov och krav. Om förändringen innebär förändringar av systemets funktionalitet måste detta stämmas av med och accepteras av verksamheten. Det kan exempelvis finnas funktionalitet som är ”nice to have” men inte ”need to have”. Om avlägsnandet av sådan funktionalitet kan ge fördelar i form av en minskad exponering så kan denna typ av förändring göras, så länge det är förankrat i verksamheten.

4.4.2.1 Förändrad arkitektur

Att förändra arkitekturen och på så sätt uppnå en lägre exponering av systemet kan göras på olika sätt. Ett *stand alone*-system är alltid att föredra ur exponeringssynpunkt (ur ett logiskt perspektiv). Om detta inte är möjligt bör samtliga gränsytor noga definieras och trafik filtreras för att minimera och kontrollera exponering av systemets information.

Uppsäkring och/eller begränsning av informationsutbytet mellan system kan sänka exponeringen, exempelvis genom att kapsla in känslig information eller genom att centralisera systemlösningen och på så sätt uppnå en lägre exponering av systemet som helhet. Antalet behöriga användare är också en faktor som kan vara möjlig att påverka. Ju färre användare desto lägre exponering.

Ytterligare en möjlighet är att skapa ”titthål” in i systemet där läsrättighet ges men inte skrivrättighet. Denna lösning är mest av värde där riktighet eller tillgänglighet är i fokus snarare än sekretess.

4.4.2.2 Förändrad teknisk lösning

Hur förändring av ett systems tekniska lösning i syfte att minska systemets exponering kan göras är något svårare att exemplifiera då den tekniska lösningen inte är fullt lika starkt kopplad till exponeringen som exempelvis arkitekturen. Det finns dock några generella möjligheter att fundera kring.

Att införa kryptering i ett system är ett effektivt sätt att sänka exponeringen. Det bör då genomföras en analys för att se vilka anslutningar och kommunikationsvägar som kan vara mer exponerade än andra och att sedan försöka rikta sina insatser mot dessa.

Att göra ett medvetet val av operativsystem kan också vara ett sätt att reducera exponeringen, framför allt vad gäller skadlig kod, men till viss del även riktade, illvilliga angrepp från icke behöriga individer/system.

4.4.2.3 Förändrad driftmiljö

Vad gäller ett systems driftmiljö så finns det en hel del åtgärder som kan vidtas för att sänka systemets exponering. I de fall då systemet ska driftsättas i kontorsmiljö eller motsvarande kan det exempelvis handla om tillträdesbegränsning i de lokaler där systemets hårdvara är placerad. Kanske finns det också möjlighet att drastiskt förändra driftmiljön genom att istället för att driftsätta systemet i kontorsmiljö välja ett bergrum eller annan högsäkerhetsmiljö.

Det är också viktigt att beakta den context i vilket ett system ska driftsättas. Om ackrediteringen innehåller säkerhetskrav i systemets driftmiljö, måste dessa vara uppfyllda för alla miljöer systemet driftsätts i.

Kopplat till driftmiljö finns det också möjlighet att förändra eller begränsa vilka användningsfall som tillåts/möjliggörs. Ett exempel kan vara en verksamhet som har önskemål om att systemet ska kunna

användas både på stabsledningsplats (kontrollerad kontorsmiljö) och ute i terrängfordon (betydligt mindre kontrollerad miljö). Frågan kan då ställas om systemet verkligen behöver vara möjligt att använda ute i fält. Om så inte är fallet kan användningsfallen reduceras till att endast omfatta stabsledningsplats och på så sätt har systemets exponering sänkts.

4.5 Konsekvensbedömning

I säkerhetsanalysen identifieras och prioriteras alla skyddsvärda tillgångar (personal, materiel, information, verksamhet och anläggningar) vilka kommer att bearbetas, lagras eller på annat sätt hanteras av systemet (inklusive systemet självt) och därefter görs en bedömning av vilken konsekvens som uppstår och omfattningen av denna om uppgifter rörande de identifierade tillgångarna utsätts för en oönskad händelse som påverkar sekretess, tillgänglighet, riktighet och spårbarhet. Resultatet av säkerhetsanalysen utgör ingångsvärden till KSF genom att konsekvensnivån har beskrivits och bedömts enligt en fastställd skala.

Konsekvens är ett mått på den skada (men) som en oönskad händelse får om den inträffar. Vad gäller systemutveckling inom ISD-processen så är exempel på dessa oönskade händelser informationsförlust (tillgänglighet), -manipulation (riktighet) och/eller röjande av information (sekretess), beroende på hur informationen i systemet är klassificerad.

4.5.1 Konsekvensnivå

Konsekvensnivån på ett system är således uteslutande kopplad till den information som lagras eller bearbetas i systemet i fråga. Konsekvensen bedöms i fem nivåer, 1-5, där 1 representerar försumbar konsekvens och 5 synnerligen allvarlig konsekvens. I KSF 3.1 finns två konsekvensbeskrivningar för varje nivå, en generell beskrivning som omfattar konsekvens vid informationsförlust av sekretessklassificerade uppgifter och en som beskriver konsekvensen vid förlust av hemliga eller utrikesklassificerade uppgifter.

Gradering		Generell konsekvensbeskrivning samt konsekvensbeskrivning vid informationsförlust av sekretessklassificerade uppgifter.	Konsekvensbeskrivning avseende informationsförlust av hemliga eller utrikesklassificerade uppgifter
5	**Synnerligen allvarlig**	Förväntade konsekvenser medför en synnerlig negativ effekt. Konsekvenserna innebär synnerligen allvarliga negativa effekter av stor omfattning, under lång tid och utgör ett direkt hot mot organisationen. Konsekvenserna är inte begränsade till enstaka förmågor eller funktioner inom organisationen.	Hemliga uppgifter vars röjande kan medföra synnerligt men för totalförsvaret eller förhållandet till en annan stat eller en mellanfolklig organisation eller i annat fall för rikets säkerhet (kvalificerat hemliga uppgifter). Hemlig handling som har åsatts beteckningen TOP SECRET eller motsvarande av en utländsk myndighet eller mellanfolklig organisation.
4	**Allvarlig**	Förväntade konsekvenser är betydande. Konsekvenserna är allvarliga, av stor omfattning eller av väsentlig art och innebär ett direkt hot, om än mot avgränsade förmågor eller funktioner inom organisationen.	Hemliga uppgifter vars röjande kan medföra betydande men för totalförsvaret eller förhållandet till en annan stat eller en mellanfolklig organisation eller i annat fall för rikets säkerhet. Hemlig handling som har åsatts beteckningen SECRET eller motsvarande av en utländsk myndighet eller mellanfolklig organisation.
3	**Kännbar**	Förväntade konsekvenser är inte obetydliga och äventyrar, vållar skada, hindrar, underlättar, innebär större avbrott samt medför påtagliga negativa effekter om än i begränsad omfattning.	Hemliga uppgifter vars röjande kan medföra ett inte obetydligt men för totalförsvaret eller förhållandet till en annan stat eller en mellanfolklig organisation eller i annat fall för rikets säkerhet. Hemlig handling som har åsatts beteckningen CONFIDENTIAL eller motsvarande av en utländsk myndighet eller mellanfolklig organisation.
2	**Lindrig**	Förväntade konsekvenser är ringa och begränsas till att påverka, försvåra, hindra, undergräva, misskreditera eller störa verksamheten i mindre omfattning.	Hemliga uppgifter vars röjande kan medföra endast ringa men för totalförsvaret eller förhållandet till en annan stat eller en mellanfolklig organisation eller i annat fall för rikets säkerhet. Hemlig handling som har åsatts beteckningen RESTRICTED eller motsvarande av en utländsk myndighet eller mellanfolklig organisation.
1	**Försumbar**	Konsekvenser för verksamheten är försumbara.	Uppgifter är inte hemliga eller utrikesklassificerade.

Tabell 3 Konsekvensnivåer och beskrivningar

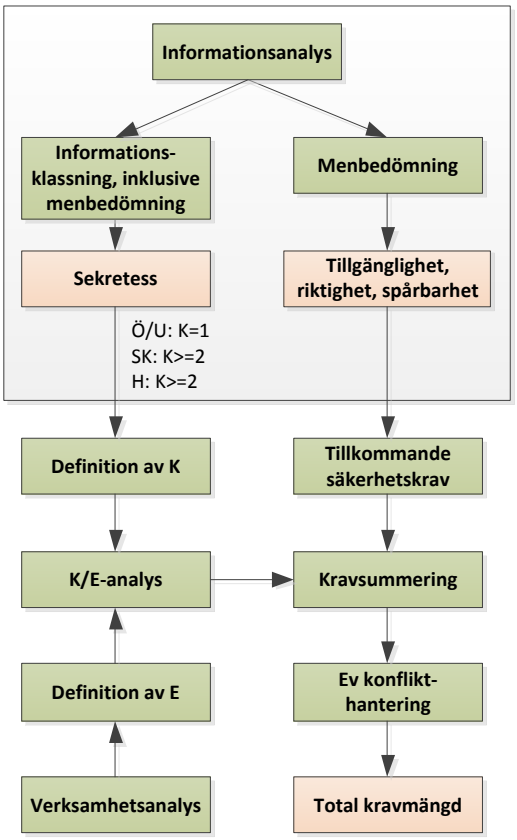
Värt att notera i denna tabell är att den generella beskrivningen bedömer den faktiska konsekvensen vid förlust av information utifrån ett generellt perspektiv, medan beskrivningen för hemliga eller utrikesklassade uppgifter enbart lutar sig på den sedan tidigare genomförda sekretessbedömningen som sker vid inplacering i informationssäkerhetsklass.

Placering i informationssäkerhetsklass sker genom en menbedömning vilket är detsamma som en förtida bedömning av konsekvensen av att informationen röjs för obehörig.

Detta innebär att konsekvensbedömningen görs av olika aktörer beroende på vilken information som finns i systemet. Bedömningen av konsekvens görs redan vid informationens inplacering i informationssäkerhetsklass då det gäller ett system med H/R-information eller högre. Konsekvensbedömningen i ett öppet system görs inte förrän vid K/E-analysen (se Figur 1).

Konsekvensbedömningen för hemlig eller utrikesklassad information tar endast sekretess i beaktande medan det generella perspektivet (i Tabell 3 ovan) skulle kunna omfatta såväl sekretess som riktighet och tillgänglighet. Även om det är uttalat att KSF endast omfattar sekretessaspekten så finns det ett antal krav i KSF som härrör såväl tillgänglighet som riktighet.

Detta kan påverka konsekvensbedömningen då det finns risk för att olika informationssystems konsekvensnivåer bedöms på olika grunder.



Figur 1 Säkerhetsanalys till K/E-analys med menbedömning

Vid bedömning av ett system som exempelvis klassats HEMLIG/RESTRICTED men där verksamheten har höga krav på att informationen i systemet är tillgänglig kan det uppstå problem. I detta fall blir det stora skillnader i kraven på sekretess och kraven på tillgänglighet vilket då medför konflikt vid bedömning av systemets konsekvens.

I KSF 3.1 [2] står att läsa att i de fall där olika bedömningar kommer i konflikt med varandra, exempelvis gällande tillgänglighet och sekretess, ska avdömning ske genom dialog med MUST. Det kan dock finnas anledning att lösa denna typ av konfliktbedömning tillsammans med verksamheten snarare än tillsammans med MUST då denna konflikt härrör faktorerna tillgänglighet och riktighet.

4.5.2 Reducering av konsekvens

För att reducera konsekvensnivån, och därmed få möjlighet att sänka kravnivån för hela systemet, krävs alltså en genomgång av den information som finns i systemet i syfte att minimera mängden känslig information i systemet med utgångspunkt i verksamhetens krav och behov.

Först och främst måste det fastsällas hur informationen i systemet är klassad. Rör det sig exempelvis om försvarssekretess eller utrikessekretess? Om ja, i vilken informationssäkerhetsklass är informationen placerad? Sedan måste en genomgång av informationen som sådan göras för att kunna avgöra om den eller de känsliga informationsmängderna verkligen behöver vara tillgängliga i det aktuella systemet. Även här behöver frågan om ”need to have” eller ”nice to have” ställas. Om det endast är “nice to have” måste denna informations tillgänglighet i systemet ställas mot den ökade kostnad det innebär att konstruera, dokumentera och vidmakthålla systemet utifrån högre ställda krav.

När det står klart vilken information som finns i systemet bör en analys genomföras för att avgöra om det går att särskilja känslig information så att den endast förekommer i vissa delar av systemet. I så fall finns möjligheten att indela systemet i olika säkerhetsdomäner, där en högre kravmängd endast appliceras på en eller ett fåtal av systemets domäner. Detta kan medföra kostnadsreduceringar för utveckling, dokumentation och vidmakthållande av systemet som helhet.

5 KRAV- OCH ARKITEKTURARBETE

I detta kapitel beskrivs generella principer och metoder för modernt arkitekturarbete, innehållande designprinciper enligt rådande *best practice* och arbetssätt inom agil/iterativ arkitekturutveckling.

5.1 Designprinciper

Vid systemutveckling eller framtagning av en IT-tjänst bör följande grundläggande principer följas:

- Minimalism
  Säkerhetsrelevanta funktioner ska inte göra något mer än nödvändigt
- Least privilege
  Användare och funktioner ska endast ha nödvändiga rättigheter
- Redundancy
  Funktioner ska ha kapacitet för att hantera hög belastning och systemfel (inklusive fysiska fel)
- Defence in depth
  Multipla funktioner ska samverka på olika nivåer för att upprätthålla säkerheten
- Self-protection
  Funktioner ska inte ha onödiga beroende till andra funktioner/domäner
- Controlled data flow
  Informationsutbyte ska följa väl definierade mönster, som är under central kontroll
- Balanced strength
  Det ska finnas en balans i styrkan hos säkerhetsfunktionerna i hela systemet
- Hardening
  Alla funktioner som inte stödjer systemets primära syfte ska vara avstängda/avinstallerade

Nedan följer en utförligare beskrivning av principerna i ovanstående lista.

Utöver dessa principer finns det andra riktlinjer, såsom:

- Economy of mechanisms
  Säkerhetslösningar måste hållas enkla, okomplicerade och begränsade till sin omfattning
- Open design
  Säkerhetslösningar måste vara öppen i design och implementation och med så få ”hemligheter” som möjligt. Dessutom ska säkerhetslösningar inte förlita sig på skydd från endast kända hot.
- Ease of use
  Säkerhetslösningar måste vara enkla (användarvänliga) och tillräckligt enkla för att undvika riskabla workarounds. Transparenta säkerhetslösningar bör eftersträvas.

5.1.1 Minimalism

Principen om *minimalism* innebär att säkerhetsrelevanta funktioner inte får göra mer än absolut nödvändigt, d.v.s. vara enkel, liten, och fokuserad.

Enkelhet är viktigt eftersom komplexitet tenderar att introducera fel i arkitektur, design, implementation, testning, användning och hantering av informationssäkerhetslösningar.

Komplexitet är svårt att analysera och konkretisera. Komplexitet bör således minskas med en söndra och härska strategi (s.k. *divide and conquer*) samt genom standardisering.

Även storleken på en funktion eller en tillgång (*asset*) kan vara en orsak till problem när det gäller *minimalism*, det vill säga att alltför många funktioner implementeras i en enskild systemkomponent. Det kan då vara lämpligt att dela upp funktionen eller tillgången till flera funktioner.

Geografisk spridning av dessa funktioner kan också vara ett alternativ.

Principen om *minimalism* gäller även själva systemet eftersom det inte alltid är lämpligt att skapa ett enda allomfattande system inom en organisation. Vissa komplexa eller äldre funktioner kan användas som distinkta (externa) system snarare än att ingå i det generella systemet. Genom att avskilja komplicerande faktorer kan komplexiteten minskas för helhetslösningen.

Principen om *minimalism* arbetar vanligtvis tillsammans med principen om *Least privilege*.

5.1.2 Least privilege

Principen om *Least privilege* avser att användare och funktioner endast skall tilldelas rättigheter som är absolut nödvändiga. Genom att tillämpa principen minskas den potentiella risken för sårbarheter.

En rättighet här betyder rätt för ett subjekt att utföra en funktion på ett objekt. Subjektet är en användare, eller en funktionell roll, och objektet är typiskt ett dataobjekt, t.ex. en fil. Funktionen kan vara vad som helst, men innebär ofta access, t.ex. läsa, skriva, exekvera, etc. Rättigheter kan vara ovillkorliga eller kräva ett visst sammanhang, t.ex. att de endast är implementerade på vissa terminaler eller under vissa timmar, eller bara vid anrop från vissa applikationer, eller när de tillämpas på vissa objekt.

Med tanke på att en rättighet är en <subjekt, funktion, objekt> -triplett, kan principen om *Least privilege* främjas genom att minimera subjektet, funktionen och/eller objektet.

- Minimera subjektet
  Att ha ett system som tillåter granulära användarroller, t.ex. specialiserad administratör och operatörsroller, främjar *Least privilege*. Motsatsen till detta är att endast ha en enda omnipotent administratörsroll, t.ex. Windows Administratör eller UNIX root.
- Minimera funktionen
  Principen om *Least privilege* är nära besläktad med principen om minimalism när det gäller att minska komplexiteten och storleken på en funktion. Att ha små och väldefinierade funktioner främjar båda principerna.
- Minimera objektet
  Minimering av storleken på objektet, det vill säga att öka granulariteten av funktionen, är det konventionella sättet att främja *Least privilege*. I stället för att en rättighet kontrollerar access till hela objektet, kan flera rättigheter styra accessen till olika delar av objektet.

Att identifiera den minsta uppsättning rättigheter för varje möjlig <subjekt, funktion, objekt> -triplet är en komplex och resurskrävande aktivitet. Standardisering och återanvändning, t.ex. av nod, dator och användarroller, är det mest kostnadseffektiva sättet att identifiera relevanta rättigheter.

Minimering av en funktion genom att ”kosmetiskt” ta bort kommandon från det grafiska användargränssnittet (GUI/HMI) utan också ta bort eller begränsa tillgången till den underliggande funktionaliteten (t.ex. exekverbara filer) är ett exempel på säkerhet genom s.k. *security by obscurity*. Detta tillvägagångssätt är ineffektivt om användarna kan åberopa samma funktionalitet från andra, t.ex. lågnivå-gränssnitt eller om de kan importera likvärdig funktionalitet från externa källor, t.ex. genom använd ett USB-minne för att kopiera programmet.

5.1.3 Redundancy

Principen om *redundancy* (redundans) säger att funktionerna ska ha ledig kapacitet för att hantera överbelastningar och fel på utrustningen. I korthet innebär det att viktiga komponenter inte ska äventyras av enkla incidenter, t.ex. genom att ha ”svaga” punkter eller flaskhalsar som hotar tillgängligheten.

Ordet reservkapacitet används i vid bemärkelse och indikerar att systemet inte ska planeras för exakt önskad storlek och effektivitet. Det måste finnas en del planering för det oväntade. Det måste också finnas utrymme för tillväxt och förbättring i händelse av en incident.

I likhet med principen om minimalism, gäller principen om redundans på flera olika abstraktionsnivåer. I ena änden kan ett informationssystem ha en fullt fungerande offline replika redo att snabbt ersätta det ursprungliga systemet. Ingen (akut) återställning av det ursprungliga systemet är då nödvändig. Detta är relevant för små och specialiserade system vilka utför (mycket) kritiska funktioner.

I andra änden kan individuella datorer och nätverksenheter ha redundans av kritiska komponenter, t.ex. nätverksgränssnitt, diskar, processorer och minne.

Ledig kapacitet behöver inte allokeras och/eller vara tillgänglig hela tiden. Standby-resurser kan ge nödvändig kapacitet genom failover eller hot-swap. Säkerhetskopiering och online-reparationer kan ge förnyad kapacitet genom återställning.

Slutligen kan resurskontroller och prioritering säkerställa nödvändig kapacitet där inga oallokerade eller förnybara resurser finns tillgängliga.

5.1.4 Defence-in-depth

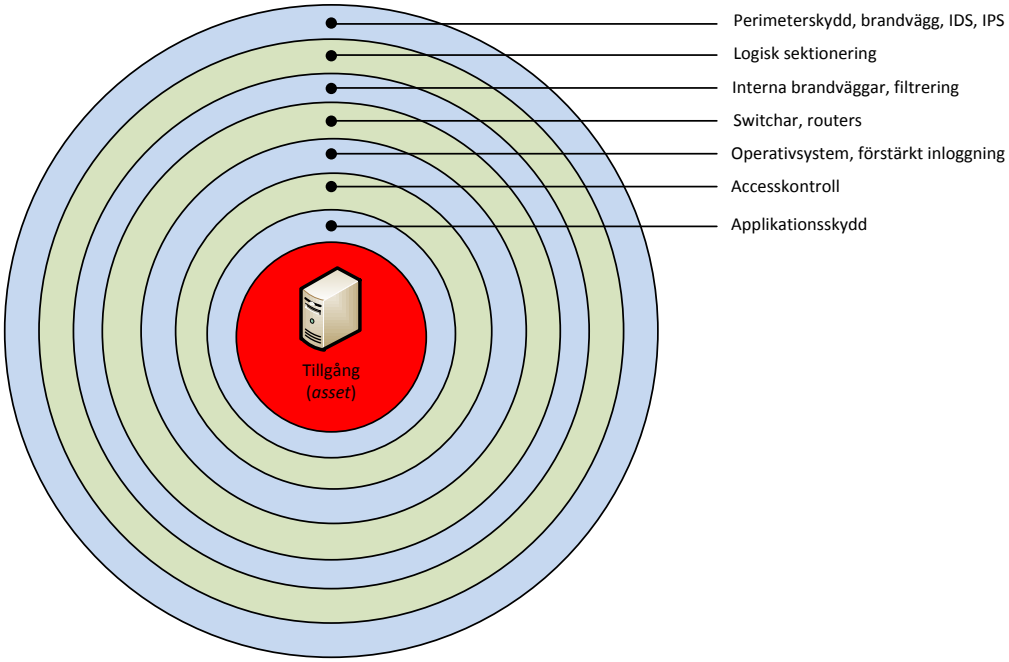
Med principen om *defence-in-depth* avses att flera funktioner ska behandla samma säkerhetsbehov, t.ex. ska det finnas skiktad säkerhet. Medan principen om redundans främjar funktioner som säkerställer kapaciteten, avser principen om *defence-in-depth* redundans för funktioner som skyddar och härdar ett IT-system. Denna princip motsvarar den svenska ”lök-modellen”.

Flera överlappande mekanismer är således bättre än en enda mekanism.

En viktig tillämpning av *defence-in-depth* är att ha flera barriärer framför de resurser som behöver skydd, så att en extern angripare (utan åtkomsträttigheter) måste bryta flera försvar för att få access till skyddsvärda tillgångar. I ett typiskt IT-system kan hindren vara filter och åtkomstkontroller längs nätverksvägarna (t.ex. i routrar och brandväggar) samt datorspecifika kontroller (t.ex. inloggningsrättigheter och tillgång objektkontroller).

En annan viktig tillämpning av *defence-in-depth* är principen om inneslutning (eng *containment*), d.v.s. att ha barriärer runt områden där skador kan uppstå för att förhindra eventuella skador från att sprida sig. I ett typiskt IT-system kan detta vara samma routrar och brandväggar som ovan men att nu styra flödet åt andra hållet.

Följande figur visar ett exempel på *defence-in-depth*:



Figur 2 Defence-in-depth (exempel)

5.1.5 Self-protection

Principen om *self-protection* anger att funktioner inte ska ha onödiga säkerhetsberoenden. Genom att tillämpa principen om *self-protection* minskar risken för en oönskad händelse att inträffa samt den potentiella omfattningen av skadan.

Principen om *self-protection* gäller många relationer, till exempel:

- Mellan nationell infrastruktur och utländsk infrastruktur
- Mellan system
- Inom system
    - Mellan interna säkerhetsdomäner och perimeterskydd
    - Mellan management och slutanvändare
    - Mellan interna domäner som utgör olika organisationer/affärsenheter
    - Mellan olika roller, t.ex. datorroller eller användarroller
    - Mellan administratör-, operatör-, och slutanvändarkonton

Kravet på *self-protection* är starkare på makronivå än mikronivå, det vill säga separation av system är viktigare än separation av zoner eller roller inom ett system.

Som det kan utläsas från ovanstående lista, kan principen om *self-protection* både tillämpas enkelriktat och dubbelriktat. Vissa förtroenden (eng *trust*) är transitiva, medan andra inte är det. Sammantaget kan dessa förtroenden bilda en hierarki av förtroenderelationer.

Kommunikationsvägar får inte undergräva *self-protection*, t.ex. kommunikation mellan systemnoder ska normalt inte passera andra noder utan hålla sig till ”neutralt” LAN/WAN.

5.1.6 Controlled data flow

Principen om *controlled data flow* avser att informationsutbytet ska följa väldefinierade flödesmönster som är under central (alternativt definierad) kontroll. Principen gäller främst nätverkskommunikation, både mellan system och inom systemen. Målsättningen är att ha en flödeskontroll som är konsekvent och effektiv i hela infrastrukturen.

Från principen om *controlled data flow* följer att:

- *Flow control policies* är en systemövergripande fråga. En risk som accepteras av en entitet är en risk som kan påverka många. En enskild entitet i ett system kan inte ensidigt ändra den beslutade policyn och därmed utsätta andra entiteter för ökad risk.
- *Flow control policies* måste formellt dokumenteras. Tillåtna dataflöden måste formellt överenskommits i avtal om samtrafik och systemspecifik säkerhetsdokumentation.
- Odefinierade policys ska som default vara förbjuden. Alla flöden måste ske i enlighet med policyn. Flöden som inte explicit har överenskommits och därmed är tillåtna är implicit förbjudna.
- *Flow control policies* behöver en stark samordning och centraliserad tillsyn. Varje organisation behöver central styrning och förvaltning av de gemensamma aspekterna av flödeskontrollpolicyn.

Flow control policies skall inte minska flexibilitet eller systemanslutningar och organisationer bör ha möjligheter att genomföra dagliga operationer inom den beslutade policyn. Det bör också vara möjligt att kunna åsidosätta mekanismer vid eventuella nödsituationer.

5.1.7 Balanced strength

Principen om en *balanced strength* bygger på att effektiviteten av säkerhetsfunktionerna skall vara motsvarande starka i hela systemet, enligt principen ”ingen kedja är starkare än sin svagaste länk”.

Ordet styrka används här i vid mening och är avsedd att täcka alla relevanta egenskaper av en säkerhetsfunktion, inklusive dess effektivitet avseende att upprätthålla säkerheten. Hot och sårbarheter realiseras inte alltid jämnt över systemet, vilket innebär olika skyddsförmågor för olika delar av systemet.

Det kan till exempel finnas ett behov av mer garantier i vissa funktioner relaterade till perimetersäkerhet och överföring av domändata. Detta bör klargöras genom riskanalyser och ska inte stå i strid med principen om en balanserad styrka. Säkerhetsfunktioner som implementeras mot samma säkerhetsmål skall naturligtvis ha samma styrka.

Principen för *balanced strength* kan vara lätt att förstå, men svårt att genomföra. Att bedöma styrkan hos en mekanism är inte trivialt, och kan innebära dyra formella utvärderingar och certifieringar, eller komplexa matematiska analyser.

Evaluering enligt Common Criteria (ISO/IEC 15408) är ett utbrett och erkänt sätt att mäta styrkan på säkerhetsfunktioner. En blandning av certifierad och icke certifierad mjuk-/hårdvara för att uppnå en viss säkerhetsnivå är ofta nödvändigt, men kan lätt skapa obalans i säkerheten. Obalansen minskas genom att välja icke certifierad programvara som integrerar med den certifierade programvaran och återanvänder certifierade säkerhetsfunktioner. Att använda icke certifierad/auktoriserad programvara av okänt ursprung, t.ex. vissa typer av *public domain* programvara, ökar obalansen.

Ofta kan en tillgång nås via olika vägar, vilka är skyddade av olika säkerhetsfunktioner. Ett enkelt exempel är den fysiska sökvägen och den infologiska vägen till en tillgång. Alla sådana vägar måste

skyddas av funktioner med motsvarande styrka. Det kan till exempel vara obalanserad säkerhet att bara ha en fyrsiffrig accesskod på fysiska dörrar och ett 15 tecken långt datorlösenord.

Ibland kan flera infologiska vägar leda till samma tillgång. Olika applikationer kan exempelvis processa och publicera samma data. Alla säkerhetsfunktioner på applikationsnivå måste då ha samma styrka som de underliggande säkerhetsfunktionerna för att inte obalans ska uppstå.

Äldre systemen (arv) och dåliga systemkonstruktioner kan införa applikationer som duplicerar säkerhetsfunktioner som redan finns i basplattformen. Alla dubbla säkerhetsfunktioner måste då ha samma styrka som de grundläggande säkerhetsfunktionerna för att inte skapa obalans.

5.1.8 Hardening

Med härdning avses processen att säkra ett system genom att minska dess exponering av eventuella sårbarheter. Ett system har en större sårbarhetsyta desto fler funktioner den uppfyller; generellt kan sägas att en enda funktion gör systemet säkrare än då flera funktioner exponeras. Metoder för att minska tillgängliga attackvektorer innefattar typiskt avlägsnande av onödiga program, onödiga användarnamn eller inloggningar och inaktivering eller borttagning av ej använda tjänster (*services*).

Det finns härdningsskript och verktyg som kan inaktivera onödiga funktioner konfigurationsfiler eller utföra diverse andra skyddsåtgärder.

Det finns olika metoder för härdning Unix och Linux-system. Det kan handla om, bland andra åtgärder, med implementation av en patch till kärnan (*kernel*), såsom t ex Exec Shield eller PaX, stänga öppna nätverksportar och installera intrångsdetektionssystem, brandväggar och intrångsförebyggande system. Motsvarande metoder finns även för Windowsbaserade system.

Härdning omfattas av, men begränsas inte till, följande aktiviteter:

- Definiera systemets syfte och begränsa/minimera mjuk- och hårdvara
- Dokumentera minimikrav avseende mjukvara, hårdvara och tjänster i systemet
- Installera den minsta mängd mjukvara, hårdvara och tjänster som behövs för att uppfylla kravbilden
- Använd en dokumenterad installationsprocess
- Installera nödvändiga patchar
- Installera de nyaste (och mest säkra) versionerna av nödvändiga applikationer
- Konfigurera åtkomstkontroll utifrån principen *deny all – grant minimum*
- Konfigurera säkerhetsfunktionerna på ett lämpligt sätt
- Aktivera säkerhetsloggning
- Testa systemet för att säkerställa en korrekt konfiguration/installation
- Ändra alla default-lösenord

5.2 Kravfördelning/-allokering

Kravfördelning/kravallokering är en viktig aspekt i STAU2, där säkerhetsfunktioner ska implementeras genom mekanismer. I detta skede kan designprinciperna ovan vara till stöd när det gäller att hitta en balanserad lösning. Detta kan innebära att ett system inte själv behöver implementera samtliga säkerhetsfunktioner, utan använda externa system/mekanismer i den totala kravuppfyllnad. I KSF 3.1 benämns detta som krav på omgivningen och förtecknas i ITSS. Exempel på detta är:

- Användning av kryptologiska funktioner
- Logganalysverktyg
- Intrångsskydd (typiskt en brandvägg)

Vid användning av externa system som en del av säkerhetslösningen bör nyttjandet av GFE eftersträvas. Här finns också en vinst i att använda certifierade lösningar (till exempel granskade med hjälp av standarden Common Criteria).

I samband med kravallokeringen är det viktigt att bibehålla spårbarhet i kravarbetet. Krav på säkerhetsfunktioner ska ges en unik etikett och därefter kopplas mot en (eller flera) mekanismer. Även mekanismerna ska vara möjliga att identifiera, oavsett om de är implementerade i SiF eller finns i externa system.

5.3 Agil arkitektur

Agila systemutvecklingsmetoder har under det senaste årtiondet rönt stora framgångar. Grundtankarna bakom agil utveckling bygger på ett iterativt arbetssätt med ett mycket nära samarbete mellan utvecklaren och beställaren eller mottagaren. Det bygger också på en decentraliserad syn på beslutsfattande. Flexibilitet, tydlig vision och samarbete är nyckelord för ett framgångsrikt agilt projekt.

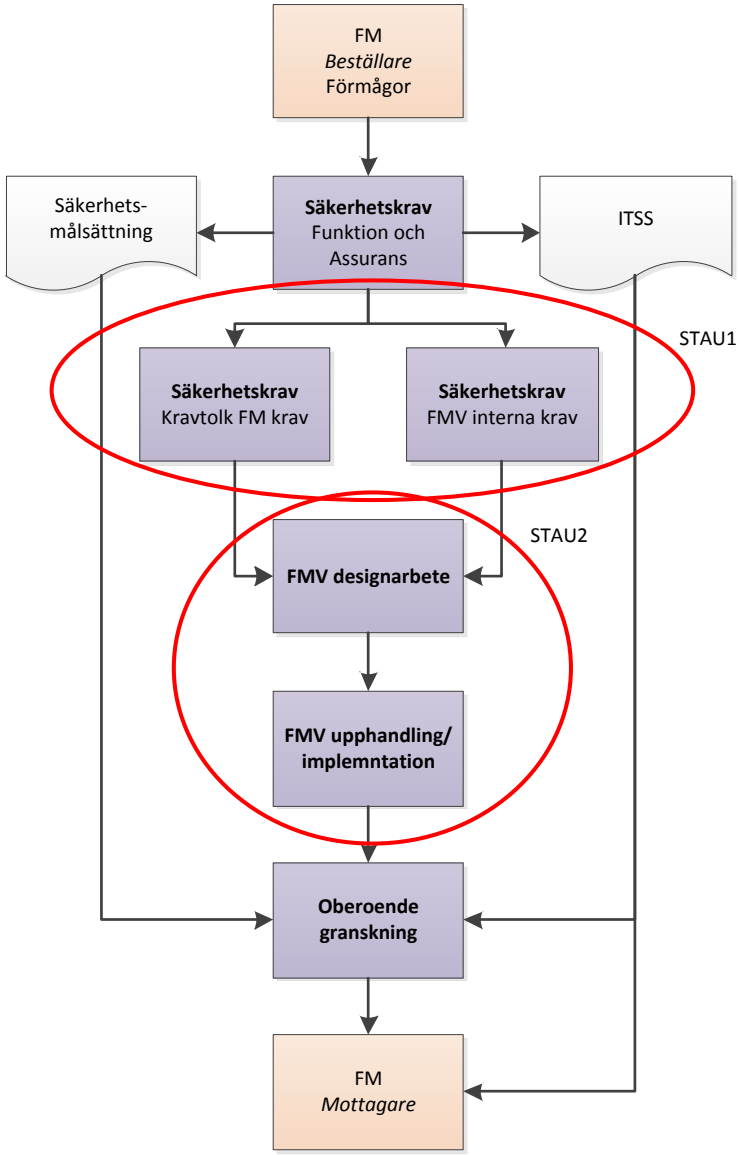
Att gå från att arbeta i traditionella projekt till agil utveckling är en stor omställning för alla discipliner, inklusive kravarbete och arkitektur. Ett agilt arbetssätt innebär exempelvis att mängden förarbete innan utvecklingen startar minimeras. Innan första utvecklingsiterationen fokuserar respektive disciplin på strategier och riktlinjer som påverkar hur produkten/systemet ska utvecklas under arbetet. Efter det sker uppföljning, lärdomar, förbättring och kontinuerlig utveckling beroende på vad som ska levereras i respektive iteration. Detaljer utformas inför eller under respektive iteration.

Fördelningen vad gäller delaktigheten från respektive disciplin är jämnare sett till projektets totala levnadstid. På samma sätt deltar kunden/verksamheten under hela projektet och inte bara i början och i slutet. Alla IT-projekt behöver delaktighet från verksamheten dels för att säkerställa att den IT-lösning som utvecklas stödjer verksamheten men också för att utveckla verksamheten tillsammans med IT. I ett traditionellt projekt görs mycket av det arbetet i förstudie och uppstart för att sen utvärderas under acceptanstesterna. Vid det laget är det dock svårt att påverka produktens utformning. Därför fokuserar agila projekt på att involvera verksamheten under hela projektet. Förstudien och uppstarten blir således mindre detaljerade än vad som är brukligt i traditionella projekt, dock inte mindre viktiga. Utvärdering sker löpande, vilket ger verksamheten större möjligheter att påverka produktutvecklingen men också anpassa sin verksamhet i ett tidigt skede om så skulle behövas.

6 KRAV OCH ARKITEKTUR I ISD-PROCESSEN

6.1 Kravreduceringsprocessen i ISD

Nedanstående figur [Figur 3] beskriver kravflödet från *Beställare* (FM) behov av förmågor via FMV krav- och designarbete och åter till leverans till FM. Styrande dokument avseende informationssäkerhet är Säkerhetsmålsättningen och IT-systemet Säkerhetsspecifikation (ITSS).

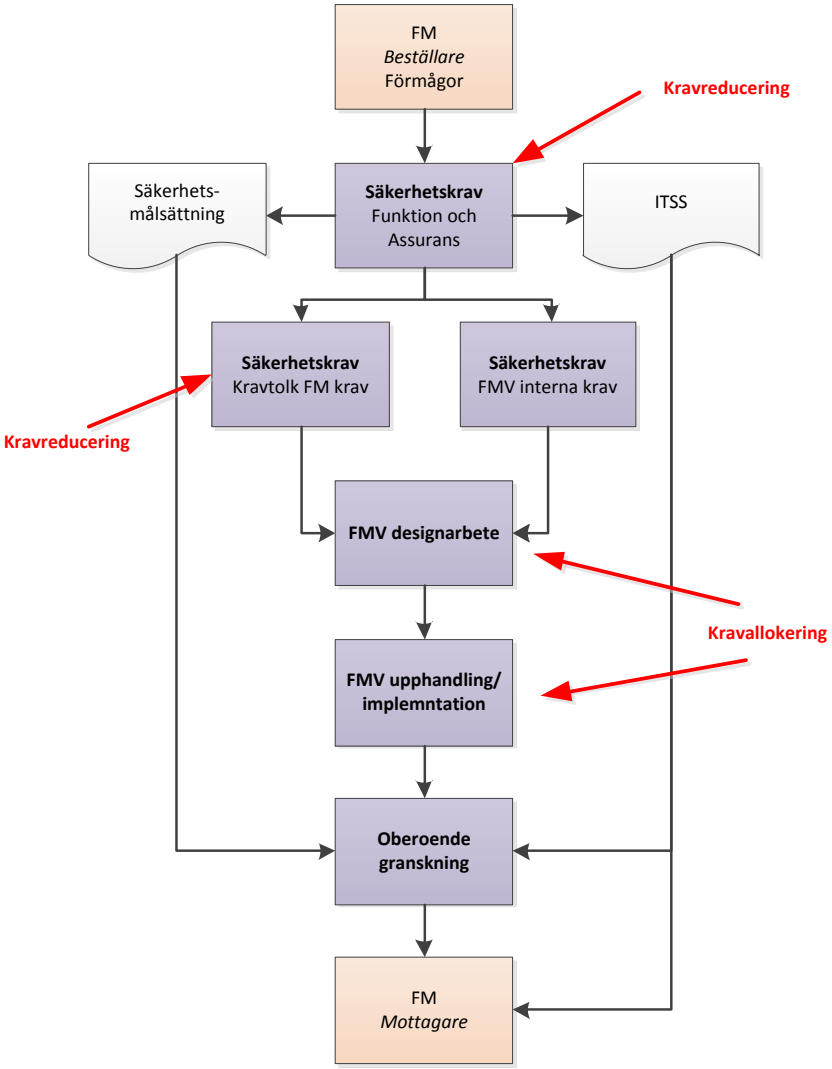


Figur 3 Kravflöde

Kravreducering genomförs i de initiala stegen i processen. Det första steget är inom beställarens område då denna ska kravställa nödvändiga förmågor hos systemet (SiF). Det andra kravreduceringssteget sker hos utföraren och/eller koordinatorn som kravställer och upphandlar SiF. FMV producerar TS och SoW. De röda ringarna i figuren ovan visar var kravarbetet sker kopplat till respektive STAU.

Vid flera leverantörer är behovet av en integratör speciellt viktigt, då det kan finnas övergripande säkerhetskrav som inte direkt kan appliceras på en enskild underleverantör.

Nedanstående [Figur 4] visar var i processen kravreducering kan vara möjlig.



Figur 4 Kravreducering

Kravreducering kan dels ske inom FM försorg och dels inom FMV försorg. I fråga om FM kravreducering kan det vara aktuellt att granska exponering och konskevens, för att etablera en adekvat kravnivå.

Inom FMV sker reduceringsarbetet genom val av mekanismer, t ex godkända säkerhetsfunktioner (så som kryptologiska funktioner) samt val av arkitektur (t.ex. genom val av MOTS eller GFE). Allokering av krav på omgivande miljö är också ett sätt att reducera den specifika kravställningen på aktuellt SiF.

Rollen som integratör är viktig, oavsett om det är FMV eller en underleverantör som har denna. Det finns t.ex. krav i ITSS som inte kan uppfyllas av (under-)leverantörer, utan som måste hanteras av *Utföraren* (FMV). Detta kan exempelvis röra sig om krav på miljön, där en enskild leverantör endast har en begränsad del av säkerhetslösningen. Ett annat exempel är krav på acceptanskriterier vid en leverans, där leverantören endast är delaktig.

6.2 Agilt arbetssätt inom ISD

För att kunna avgöra i vilken utsträckning reducering av kraven i KSF 3.1 är möjlig, behövs ett tätt samarbete mellan FMV som designansvarig och verksamheten i form av FM. För att kunna avgöra vilka förändringar som är möjliga att göra behöver FMV efter genomgång av krav och behov från FM återkoppla till verksamheten. Detta måste ske för att få svar på vilka förändringar kopplat till exponering och konsekvens som är möjliga att genomföra och med bibehållen funktionalitet. Alternativt måste verksamheten godkänna avkall på funktionalitet för att exempelvis sänka kostnader kopplat till utveckling, granskning och test. Det kommer således krävas någon form av iterativt arbete mellan dessa båda aktörer för att få svar på dessa frågor.

Agil utveckling är, till skillnad från System Engineering (SE) med iterativt arbetssätt, just nu inte ett vedertaget arbetssätt inom ISD-processen och FMV utveckling. FM är beställare och mottagare, FMV har designansvaret och utvecklingen utförs av en eller flera underleverantörer. Detta gör att ett så pass nära samarbete mellan samtliga dessa parter som den agila utvecklingsmetodiken kräver blir svårt att upprätthålla. Oavsett detta finns det dock vinster att hämta i den agila metodikens iterativa arbete när det kommer till ISD-processens hantering av krav och arkitektur. Läs mer kring förslag på förändrad metodik under rubrik 7.2.

6.3 Inkrementellt och Iterativt arbetssätt inom ISD

Under ett längre, eller mer komplext, projekt är det hög sannolikhet att Beställaren kan komma att förändra systemets målbild, t ex på grund av förändrad användning eller målmiljö. Sättet att hantera större förändringar är att arbeta i mindre steg.

Ett inkrementellt arbetssätt innebär att systemet utformas stegvis, och utformas allt eftersom fler och fler krav ska implementeras. När ett systems vision, krav och funktioner bryts ner till mindre delar är det viktigt att det görs på rätt sätt. I en flerlagersstruktur, där ett lager t ex kan vara ett grafiskt gränssnitt eller datalager som kommunicerar med en databas, anses dessa lager vara horisontella. För att kunna implementera ett inkrement av ett system (eller produkt) krävs det att systemet/produkten delas veritkalt, dvs en funktionsmässig uppdelning där varje funktion skär genom flera/alla lager.

Vid ett inkrementellt systemarbete är det möjligt att lagga till och ta bort funktioner under tiden som systemet utvecklas.

Iterativ systemutveckling, och kanske främst iterativ kravspecificering, innebär repetition av samma process. Detta arbetssätt möjliggör förfining av krav som redan har specificerats.

Just kravspecificering inom ISD-processen, först i STAU1 men särskilt i STAU2, kan med fördel genomföras med en iterativ arbetsprocess. Från den totala kravfångsten i STAU1 kan sedan kraven allokeras till specifika systemkomponenter eller till den omgivande miljön. I denna process erhålls också stöd för den arkitektuella systemlösningen.

7 SLUTSATSER

Mycket av arbetet med att analysera KSF 3.1 och dess påverkan på ISD-processen har för projektgruppen varit att ge sig ut på otrampad mark. Inriktningen på arbetet har till stor del utkristalliserat sig under arbetets gång. Gruppen har varit i stort behov av att bolla idéer och tankar såväl med varandra som med utomstående individer med spetskompetens inom olika områden för att hämta inspiration och finna nya infallsvinklar.

Under rubrik 3 listades följande frågeställningar, kring vilka mycket av diskussionerna i arbetet har förts:

- Vilka motiv finns till att sänka kravnivån?
- Vad är vinsten med att sänka kravnivån från Hög till Utökad respektive från Utökad till Grund?
- Hur sänks exponeringsnivån?
- Hur sänks konsekvensnivån?
- Hur uppnås spårbarhet i kravarbetet?
- Är det alltid önskvärt att sänka kravnivån på ett system?
- Vad måste uppmärksammas på vid kravreducering?
- Hur behöver ISD-processen förändras för att ta höjd för kravreduceringsarbete?

Nedan sammanfattas kort projektgruppens samlade syn kring dessa frågor.

Vilka motiv finns till att sänka kravnivån?
De primära motiven till att reducera kraven för ett system är av besparingsskäl. Om det finns möjlighet att sänka kraven för ett system från hög till utökad eller ännu hellre från hög till grund så finns stora ekonomiska fördelar att uppnå. Att uppfylla de krav som stipuleras i kravnivå Hög är kostnadsdrivande.

Ett annat motiv till kravreducering är att finna tidsmässiga fördelar i de fall då verksamheten har stort behov att snabbt få tillgång till systemet i fråga. En lägre kravnivå kan medföra att konstruktion, test och granskning av systemet tar mindre tid i anspråk.

I de fall då den föreslagna designen för ett system omfattar COTS/MOTS/GOTS kan det vara värt att se över kravmassan för att eventuellt kunna reducera den. Ovan nämnda typer av ”hyllprodukter” kan vara svåra att påverka då de ofta är standardiserade och låsta exempelvis vad gäller vilka olika konfigurationer som tillåts. Detta kan i sin tur försvåra eller till och med omöjliggöra kravuppfyllnad på de högre nivåerna.

Projekt som bedrivs med stöd av ISD-processen ska arbeta efter att uppnå tillräcklig säkerhet genom designprinciper och analyser/bedömningar av exponeringsfaktor och konsekvensnivå.

Vad är vinsten med att sänka kravnivån från Hög till Utökad respektive från Utökad till Grund?
De största vinsterna med kravreducering är av naturliga skäl kostnadsbesparingar (både i samband med upphandling/anskaffning och under systemets vidmakthållande), minskad kravställning och, i bästa fall, ett mer lättanvänt system.

Kravreducering kan dock ge effekter som ökat beroende till omgivande system, vilket i sig kan öka komplexiteten för systemlösningen som helhet.

En övergripande beskrivning av vinsterna med kravreducering finns i kapitel 4.3, vilket bygger på sammanställningen i referens [4].

Kort kan skillnaderna i kravnivåer sammanfattas som:

- Behörighetskontroll; Ökade krav på autenticering och separation av roller
- Säkerhetsloggning; Ökade krav avseende logganalys
- Skydd av kommunikation
- Härdning (inklusive integritetskontroll)

De största vinsterna genom kravreducering ligger dock inom assuransområdet.

Hur sänks exponeringsnivån?
Reducering av exponering kan ske genom att förändra systemets arkitektur eller genom att på annat sätt förändra systemets tekniska lösning. Genom att kravställa driftmiljön eller begränsa systemets användning kan också exponeringen för systemet som helhet sänkas. Om dessa typer av åtgärder vidtas för att sänka exponeringen så förändras inte systemet som sådant. Däremot sänker de förändrade kraven på omgivningen och/eller användarna exponeringen för systemet.

Hur sänks konsekvensnivån?
Konsekvensnivån baseras på den information som finns i systemet. Sänkning av konsekvens uppnås genom att reducera mängden känslig/hemlig information i systemet. Detta kan exempelvis göras genom att plocka bort information som endast bedöms vara ”nice-to-have” eller genom att dela in systemet i olika delsystem eller säkerhetsdomäner och på så sätt isolera den känsliga informationen.

Är det alltid önskvärt att sänka kravnivån på ett system?
Det kan finnas tillfällen då sänkning av kravnivån inte är önskvärd, exempelvis då systemet är tänkt att kommunicera med andra system med en viss sekretessnivå. Ett exempel: System X ska utvecklas och kommer att vara i behov av att kommunicera med System Y. System Y innehåller H/S-information. Om System X genomgår en kravreduceringsprocess kan detta komma att innebära att System X bara blir tillåtet att kommunicera med System Y med ett antal godkända säkerhetsmekanismer (eller i värsta fall inte alls). Om dessa typer av säkerhetsmekanismer av olika anledningar inte är tillämpliga i System X kan denna situation vara ett exempel då kravreducering inte är önskvärd.

Vad måste uppmärksammas vid kravreducering?
Vid en eventuell reducering av kravnivå är det viktigt att beakta den färdiga systemlösningen. Risker finns att IT-systemet inte kan användas på rätt sätt om funktionalitet eller information reducerats bort. En checklista med viktiga kontrollpunkter som bör beaktas vid kravreducering finns under rubrik 7.3.3.

Hur behöver ISD-processen förändras för att ta höjd för kravreduceringsarbete?
För att ISD-processen ska kunna förhålla sig till och på ett effektivt sätt kunna arbeta med KSF 3.1 kommer det att behövas iterationer inom arkitekturarbetet. En analys kring hur arbetet med arkitektur och kravreducering påverkar ISD-processen finns under rubrik 7.2.

7.1 Arkitektur

Informationssäkerhetskraven för ett system härstammar huvudsakligen från Säkerhetsmålsättningen och ITSS. Kraven kan vid behov behöva anpassas för det specifika systemet och därefter allokeras, antingen på SiF eller på omgivande, externa, säkerhetslösningar.

I denna process är spårbarheten viktig, då granskningen i STAU4 måste kunna påvisa kravuppfyllnad baserad just på Säkerhetsmålsättningen och ITSS.

Spårbarheten kan uppnås på olika sätt men typiskt används en tabell där kopplingen mellan säkerhetsmål SM.x uppnås genom funktion X och Y som implementeras av mekanism Q och Z. Kopplingen mellan säkerhetsmål och funktionella krav är, liksom kopplingen mellan funktioner och mekanismer, av typen [1..n].

Krav, både funktionella och assurans, ska vara enkla, entydiga och unika.

7.1.1 Checklistor för arkitektur och säkerhetsfunktioner

Följande checklistor härstammar från designprinciperna i kapitel 5.1 och ska ses som en generisk kontroll. Checklistorna är på intet vis uttömmande, utan ska ses som stöd och orientering.

Checklistorna bygger på designprinciperna och FM krav, d.v.s. KSF, och är uppdelade med fokus på säkerhetsfunktioner respektive arkitektur.

Säkerhetsfunktionerna är generiska, men i huvudsak avses följande:

- Behörighetskontroll
- Säkerhetsloggning
- Intrångsskydd
- Intrångsdetektering
- Skydd mot skadlig kod
- Skydd mot röjande signaler
- Skydd mot obehörig avlyssning

7.1.1.1 Checklista avseende säkerhetsfunktioner

Pos	Text	Referens
1.	Är säkerhetsfunktionen enkel och minimal?	Kapitel 5.1.1
2.	Har säkerhetsfunktionen onödiga beroende till andra funktioner?	Kapitel 5.1.5
3.	Följer säkerhetsfunktionen en vedertagen standard?	Kapitel 5.1.1
4.	Är säkerhetsfunktionen geografiskt begränsad?	Kapitel 5.1.1
5.	Har användare endast tilldelats absolut nödvändiga rättigheter?	Kapitel 5.1.2
6.	Är subjekten i systemet minimerade (d.v.s. har granulära accessrättigheter)?	Kapitel 5.1.2
7.	Finns det flera roller med fullständiga rättigheter i systemet? I så fall varför?	Kapitel 5.1.2
8.	Är säkerhetsfunktionerna i systemet minimerade och väl definierade?	Kapitel 5.1.2
9.	Är objekten i systemet minimerade och/eller differentierade?	Kapitel 5.1.2
10.	Är access till objekt i systemet reglerat enligt beslutad rättighetsfördelning?	Kapitel 5.1.2
11.	Är systemet härdat?	Kapitel 5.1.4, 5.1.8

7.1.1.2 Checklista avseende arkitektur

Pos	Text	Referens
1.	Är externa gränssnitt (t.ex. USB och Ethernet) inaktiverade om de inte behövs?	Kapitel 5.1.2
2.	Finns redundans för systemkritiska komponenter? Elkraft? Kommunikationskanaler? Hårdvara? Mjukvara?	Kapitel 5.1.3
3.	Finns rutiner för sökerhetskopiering av kritiska tillgångar (objekt)?	Kapitel 5.1.3
4.	Används överlappande säkerhetsfunktioner (*defence-in-depth*)?	Kapitel 5.1.4
5.	Är skyddsvärda tillgångar identifierade och skyddade?	Kapitel 5.1.4
6.	Har systemet multipla skyddslager?	Kapitel 5.1.4
7.	Är systemet zon-indelat med hänsyn till olika informationstillgångar?	Kapitel 5.1.5
8.	Är det interna och externa informationsflödet identifierat, konsekvent och effektivt?	Kapitel 5.1.6
9.	Finns det en dokumenterad, överenskommen och implementerad flödeskontrollpolicy?	Kapitel 5.1.6
10.	Är samtliga säkerhetsfunktioner i balans?	Kapitel 5.1.7

7.2 Processförändring

I traditionellt projektarbete (som generellt sett härstammar från projektmetoder som vattenfallsmodellen eller RUP) så sker arbetet sekventiellt. En omfattande kravspecifikation levereras tillsammans med en arkitektur och design som beskriver hur implementationen ska genomföras. Förhoppningsvis har antaganden och teorier stämt och inga komplikationer uppstår. Om projektet upptäcker en felaktighet eller något som behöver justeras behöver samma arbetsflöde repeteras igen.

Det är en logisk arbetsgång men i stora projekt tenderar varje fas att bli för omfattande vilket ökar risken för försenad tidplan och/eller förhöjda kostnader om gjorda antaganden om framtiden visar sig vara felaktiga. Det är mer eller mindre på detta sätt som utvecklingsarbetet inom FM och FMV har bedrivits. Projektgruppens uppfattning, efter att ha analyserat KSF 3.1 och ISD-processen, är att det skulle finnas en hel del att vinna på att använda en iterativ metodik (motsvarande agil utveckling) för kravhantering.

Agila projekt vill undvika långa, omfattande och dokumentationsintensiva faser. Genom att sätta en övergripande strategi, vision och leveranspunkter påbörjas utvecklingen tidigt. Genom att lära utifrån den produkt som utvecklas kan respektive disciplin göra ett bättre arbete.

Den stora skillnaden mellan agil utveckling och den mer traditionella vattenfallsutvecklingen är att förarbetet inom krav, test och arkitektur minimeras. Istället påbörjas en iterativ produktutveckling tidigt i projektet.

För att ISD-processen ska kunna förhålla sig till och på ett effektivt sätt kunna arbeta med KSF 3.1 kommer det att behövas iterationer inom arkitekturarbetet. För att detta ska fungera krävs att samtliga involverade aktörer är införstådda med hur arbetet måste bedrivas. Dessutom är det av stor vikt att FMV ges mandat att fullt ut driva arbetet med krav och arkitektur.

Projektgruppen föreslår att FMV aktivt tar ett ytterligare kliv in i koordinatorrollen genom att anpassa ISD-processen vad gäller arbetet med kravreducering och arkitektur. För att detta ska ge de ekonomiska och tidsmässiga fördelarna som projektgruppen föreslår så är det viktigt att även FM är införstådda i FMV intentioner. FM måste ges möjligheten att lämna över ansvaret till FMV för framtagning av den totala kravmängden för ett system tillsammans med dess slutliga arkitektur så att inte dubbelarbete förekommer.

Primärt är det arbetet med STAU2, TS och SoW som skulle komma att påverkas av projektgruppens föreslagna processförändring. Det ISD-metodunderlag som härrör dessa underlag bör ses över och kompletteras med ett resonemang kring kravreducering, dess vara eller icke vara samt hur arbetet med kravreducering kan bedrivas. De i detta underlag föreslagna checklistor bör också inarbetas i ISD-metodunderlaget för att stötta och i viss mån inspirera projekt i arbetet med arkitektur och kravreducering för att tillgodose en sund arkitektur på lägsta möjliga kravnivå utan att säkerheten äventyras.

Här bör det också belysas att kravreducering, utifrån KSF 3.1, sker genom reducering av exponering och/eller konsekvens allternativt allokering av säkerhetsfunktioner till omgivande miljö. Till detta bör det kopplas en riskanalys, för att säkerställa att de ursprungligt kravställda förmågorna kan upprätthållas.

7.3 Riskhantering

Som beskrivits ovan innebär kravreducering att ett systems tilltänkta funktionalitet, användning och arkitektur ses över och förändras i den mån det är möjligt för att uppnå en lägre kravnivå. I samband med att förändringar i systemet genomförs, eller föreslås, kan det också komma att introduceras risker i den nya systemlösningen som inte funnits i den gamla. Det är därför viktigt att samtidigt med förändringar av arkitektur också analysera vilka ytterligare risker som kan ha kommit att uppkommit.

7.3.1 Riskanalys i designfasen

Samtidigt som kravreduktion introduceras i ISD-processen bör även en metod för riskanalys i denna fas av arbetet införas. Det måste analyseras om förändringen i arkitektur, teknisk lösning eller användningssätt har introducerat nya risker. Den nya systemlösningen kan exempelvis:

- ge access till systemet för fler aktörer
- tillåta eller begränsa en alternativ driftmiljö
- möjliggöra andra kommunikationsvägar
- begränsa tillgång till viss information

Redan genomgångna risker bör också ses över igen då förändringarna i systemet även kan ha ökat sannolikheten för dessa.

Det är inte säkert att ovanstående exempel introducerar nya risker men oavsett hur den nya lösningen ser ut är en riskanalys nödvändig, om än i enkel form, för att kunna påvisa medvetenhet kring tillkommande risker och en dokumenterad bedömning och plan för hantering och/eller acceptans av dem.

7.3.2 Residuala risker

I ISD-processen bör även ett resonemang föras kring riskhantering och residuala risker. Det finns ett antal frågor kopplat till detta. Några exempel är:

- Vem fattar beslut kring vilka residuala risker som är acceptabla?
- Vem tar ansvar för och ”äger” dessa risker?
- Hur bör/måste verksamheten arbeta med dessa risker under systemets livscykel?

Denna typ av frågor bör diskuteras och svaren förankras med FM och inarbetas i ISD-metodstöd.

7.3.3 Checklista avseende kravreducering och riskanalys

Denna checklista sammanfattar de viktiga aspekter som projektgruppen anser bör beaktas vid kravreducering. Checklistan ska inte ses som fullständig utan som en första ansats till ett bra metodstöd vid kravreducering samt vid efterföljande riskanalys och hantering av residuala risker.

Pos	Text
1.	Har systemdesignen förändrats så att FM krav på förmågor inte kan uppnås?
2.	Vid behov, har implementationen av externa systemkomponenter säkerställts?
3.	Används GFE för att reducera den totala kravbilden på SiF? Om ja; är dessa rätt använda?
4.	Har eventuell kravreducering medfört att möjligheterna avseende informationsutbyte med andra system inte kan uppfyllas i nödvändig grad?
5.	Har en ny riskanalys genomförts efter genomgången kravreducering?
6.	Har den nya riskanalysen även omfattat tidigare analyserade risker för att utröna om sannolikheten för dessa påverkats?
7.	Har eventuella residuala risker accepterats och riskanalysen undertecknats av [ansvarig beslutsfattare]?