MANAGING RISK

Sikkerhetspolicies i utviklingsprosjekter

Jon Ølnes, DNV Research & Innovation

Abelia-seminar, Sikkerhetspolicies – kun til pynt eller lar de seg håndheve

Oslo, 23. november 2006

Innhold

1. Sikkerhet og kvalitet
2. Sikkerhetskrav til systemer
3. Sikkerhetsarkitektur
4. Implementere sikkerhet i prosjekt og produkt
5. Konklusjoner

MANAGING RISK

Sikkerhet og kvalitet

Sikkerhetspolicy

Hvorfor sikkerhet i utviklingsprosjekt?

MANAGING RISK

1. Sikre at produktet har de nødvendige sikkerhetsegenskapene
   - Ut fra forventede eller mulige bruksområder
   - Kritiske anvendelser – samfunnskritisk, bedriftskritisk?
   - Sørge for implementasjon med kvalitet
2. Sikre utviklingsmiljø og prosjekt
   - Beskytte forretningskritisk informasjon og rettigheter?
   - Unngå leveranseproblemer
   - Unngå bakdører og bevisste svakheter i produktet?

Vanlig systemutvikling(?)

MANAGING RISK

- Få på plass funksjonaliteten først
  - Først må det virke …
- Etterpå legge på sikkerhet og «annet ekstra»
  - Optimalisere ytelse f.eks.

- Sikkerhetskrav må inn fra starten
  - Kan gi sterke føringer for design av et system
  - Eller gir føringer for funksjonaliteten

- **Sikkerhetskrav må sees på som funksjonelle krav**
  - Egenskaper ved hele løsningen, ikke bare sikkerhetsfunksjonene

Sikkerhet er ikke glasuren på bollene – den må inn i deigen

Noen utsagn om sikkerhet

MANAGING RISK

- Sikkerhet og kvalitet henger sammen
  - God kvalitet innebærer god nok sikkerhet
  - Og omvendt er en avhengig av kvalitet på sikkerhetstiltakene
- Mangel på hendelser betyr ikke god sikkerhet
  - Flaks? Som å operere uten forsikring
- Sikkerhet koster
  - Noen tiltak kan bidra positivt til resultat, men som regel en utgift
  - Kost / nytte vurderinger av sikkerhetstiltak
- Sikkerhetstiltak innebærer sikkerhetsrisiko
  - Nye komponenter som kan angripes (nøkler, passord)
  - Sum av positiv og negativ komponent, der positiv bør være klart størst
- Ingen kjede er sterkere enn det svakeste leddet

Noen spissformuleringer

MANAGING RISK

1. Har en ikke oversikt, kan en ikke sikre
   - Spaghettikonfigurasjoner gir risiko
   - Splitt og hersk for å få oversikt
   - Kritiske funksjoner separeres ut og sikres særskilt
2. Har systemet feil, har det sikkerhetsrelevante feil
   - Stabilitet, tilgjengelighet, motstå angrep
   - Systemutvikling for kvalitet gir også sikkerhet
3. Sikkerhetstiltak brytes ikke, de omgås
   - Både ved slurv og ved bevisste angrep
   - Sikkerhetsmekanismene kan være gode nok, men kvalitetsfeil gjør at en kommer rundt mekanismene
   - Sikkerhet er en egenskap ved det totale systemet, ikke bare sikkerhetsfunksjoner

Kvalitet og sikkerhet

MANAGING RISK

- Kvalitetssikring er viktigste sikkerhetstiltak
  - Unngå systemfeil og menneskelige feil
  - Unngå at det oppstår sikkerhetshull
- Sikkerhet hører hjemme i kvalitetsarbeid
  - Sikkerhet er en kvalitetsegenskap
  - Kvalitetssikringsarbeid må omfatte sikkerheten
- Dette gjelder også utviklingsprosjekter

- ISO/IEC IS17799: Code of practice for information system security
  - BS7799 Sertifisering for organisatorisk IT-sikkerhet (tilsvarer IS17799)

Sertifisering av systemer

MANAGING RISK

Common Criteria (CC): ISO 15408

1. Target of evaluation (systemet/komponenten som skal evalueres)
2. Evaluation profile (profil som det skal evalueres mot)
3. Evaluation assurance level (nivå på evalueringen, gradert 1-7)
   - EAL4 er det høyeste som kan oppnås for et ”vanlig utviklet” system
   - EAL5-6 stiller krav til utviklingsprosess og (delvis) formell spesifikasjon
   - EAL7 stiller krav til formell verifikasjon

Ulemper:

- Meget tung prosess – krever årsverk og måneder/år kalendertid
- Gir statisk system – oppgraderinger og endringer vanskelig

MANAGING RISK

Sikkerhetskrav til systemer

Hva skal produktet (kunne) gjøre?

Sikkerhetskrav

MANAGING RISK

1. Hvilke sikkerhetskrav skal systemet (kunne) oppfylle?
   - Finnes det lovpålagte krav (personopplysninger, gradert informasjon, SOX)?
   - Stiller kunder/oppdragsgivere spesifikke krav?
   - Hva mener vi selv er viktig?
2. Hvilke (sikkerhets-)krav stiller dette til utviklingsprosessen?
   - Viktigste er å unngå feil – god kvalitet på utviklet system
   - Dokumentasjonskrav til utviklingsløpet (og til ferdig system)
   - Tilrettelegging for sikkerhetstesting (og kanskje evaluering og sertifisering)

- Med høye sikkerhetskrav til system må en stille krav til:
  - Utviklingsprosess
  - Sikkerhet for systemene som brukes i utvikling
  - Sikkerhetspolicy og organisasjon for utviklingsprosjektet

Risikoanalyse, scenarioer

MANAGING RISK

- Sikkerhetskrav må bunne i en risikoanalyse
  - Metodisk tilnærming nødvendig
- Scenariotilnærming for anvendelser av systemet
  - Definer scenario – omgivelser, informasjon, brukere
  - Definer ønsket oppførsel av systemet, inkludert ytelse
  - Trusler – hva kan tenkes å skje?
  - Risiko – hva kan forårsake en hendelse, og hva er sannsynlighet?
  - Sårbarhet – hva kan konsekvensene være?

Sikkerhet i kravspesifikasjon

MANAGING RISK

- Tilgjengelighetskrav
  - Overbelastning og blokkering – ytelse + motstå feil og bevisste angrep
  - Design av sikkerhetsfunksjoner – en del løsninger kan kvele ytelsen
- Systemintegritet
  - Krav til korrekt operasjon / motstå angrep for å modifisere system
- Autentisering og aksesskontroll
  - Bruk plattform heller enn å implementere eget!
  - Men kan gi problemer for plattformuavhengige systemer
- Integritet / konfidensialitet av informasjon
  - Autentisering og aksesskontroll er som regel viktigst
  - Støtte for kryptografisk beskyttelse under overføring
- Sporbarhet
  - Logging og sikring av logger (autentisering + integritet)
- Oppretting etter hendelse (feil eller angrep)
- Sikkerhet i systemadministrasjon
  - Minst samme sikkerhet som systemet selv
  - Risiko for bakdører til kompromittering av systemet

Dekkes krav av plattform eller system?

MANAGING RISK

- Sikkerhetskrav oppfylles av systemet eller plattform/omgivelser
  - I hvor stor grad ønsker en å stole på plattform/omgivelser?
  - Er plattformuavhengighet en utfordring?
  - Hva kreves av plattform og omgivelser?
    - Fysisk sikkerhet
    - Autentisering, autorisasjon og aksesskontroll(?)
    - Krypto, logging, sikker kommunikasjon?
    - Systemadministrasjon?
  - Hva må en ha i tillegg?
- Dokumentere krav og forutsetninger
- Dokumentere avhengigheter og samspill mellom system og plattform
- Kundenes sikkerhetspolicyer – fleksibilitet og forutsetninger
  - Hva skal kunne støttes?
  - Hva er konfigurerbart i systemet?

MANAGING RISK

Sikkerhetsarkitektur

Hvordan implementere sikkerhet

Arkitektur er tvingende nødvendig

MANAGING RISK

- Har en ikke oversikt, kan en ikke sikre
- God, ryddig systemarkitektur er en forutsetning for sikre systemer
- Spesifikke sikkerhetstjenester realiseres gjennom sikkerhetskomponenter
- Sikkerhetsarkitektur viser sikkerhetskomponenter i systemet og hvordan disse brukes av resten av systemet
- Skill helst ut sikkerhet i egne komponenter – kan beskyttes særskilt
- Sørg for at sikkerhetskomponentene brukes – alltid!

Designregler – tillitsmodell

MANAGING RISK

- Definer hvilke komponenter som er tiltrodd
  - ... og for hvilke sikkerhetsmessige formål
  - Ikke stol på noen andre komponenter for formålet
  - ”Eksplisitt tillit” er nødvendig for sikkerhet
  - Manglende modellering av tillit gir usikre systemer
- Sørg for at komponentene fortjener tillit
  - Kvalitet og funksjonalitet
  - Gode nok sikkerhetsfunksjoner
  - Beskytt dem mot angrep
- Sørg for at komponentene alltid brukes
  - Ingen ”snarveier” eller ” komponentinterne” løsninger
  - Sikkerhetsfunksjonene skal ikke kunne omgås
- Dette dokumenteres i arkitektur

Tillitsmodell for brukere og roller

MANAGING RISK

- Vanlig feil: For dårlig beskyttelse mot brukere
- Skal systemet designes slik at en stoler på brukerne?
  - Eller bare brukere i visse roller?
  - Eller kombinasjoner av personer/roller for visse oppgaver?
  - Systemet må hindre brukere i å gå ut over sine rettigheter
  - Og sikkerhetsfunksjonene må støtte dette

Komponentbasert utvikling

MANAGING RISK

- Moderne systemutvikling er basert på komponenter og gjenbruk – bygger videre på andres arbeid
  - Sikkerhetskrav må reflekteres i krav til komponentene
  - Sikkerhetskrav kan være utslagsgivende for valg av komponenter
  - Dokumentasjon, testing etc. opp mot sikkerhetskrav

Spesifikasjon og implementasjon

MANAGING RISK

- En sikker komponent skal:
  - Fylle sin spesifikasjon helt korrekt
  - Ikke gjøre noe mer enn det – ingen ekstra funksjoner
- Spesifikasjon av grensesnitt og sikkerhet
  - Få med sikkerhetsegenskaper som deler av dette
- Spesifikasjon av innmat og sikkerhet
  - Hvordan utføres oppgaven, gjøres det noe ekstra?
  - Krav til komponenter som skaffes fra andre kilder

Gjenbruk av komponenter

MANAGING RISK

- Hva er opphavet – er dette til å stole på?
  - Hva er kvaliteten – holder dette?
  - Hva er sikkerhetsegenskapene?
- ”Svart boks” – bare grensesnitt teller
  - Mangler spesifikasjon av ”det indre”
  - Er det noen sideeffekter?
- Sikkerhetstesting er meget vanskelig
  - Kan teste funksjonaliteten
  - Vanskelig å teste for ”ekstra funksjonalitet” (ikke-funksjonalitet)
  - Vanskelig å simulere angrep på komponentnivå

MANAGING RISK

Implementere sikkerhet i prosjekt og produkt

(og kvalitet)

Sikkerhetspolicy for utvikling

MANAGING RISK

1. Sørge for at sikkerhetskrav blir implementert i systemet
   - Med tilstrekkelig kvalitet
2. Sørge for sikkerheten i utviklingsmiljø og prosjekt
   - Som del av kvalitetssystemet
   - For å sikre dette ”som et hvilket som helst annet miljø”

Prosjektledelsen må ta ansvar

MANAGING RISK

- Legg sikkerhet inn i tilbud/kostnadskalkyler fra starten
  - Gir ofte føringer for arkitektur
  - Legg inn ressurser til sikkerhetsmessig risikoanalyse
  - Beregn omfanget av å implementere sikkerhetsfunksjoner
- Hvis det skal kuttes, vurder sikkerhet på linje med andre funksjoner
  - Vanligst i dag: Opprettholder funksjonalitet, kutter sikkerhet
  - Sikkerhetskrav må betraktes som funksjonelle krav
- Få sikkerhet inn i kvalitetsplaner, sjekklister og testplaner
  - Sørg for at alle beslutninger har et sjekkpunkt på sikkerhetsrelevans
  - ”Ikke relevant for sikkerheten” skal være et bevisst valg
  - Oppfyllelse av sikkerhetskrav skal testes som andre funksjonelle krav

Prosjektsikkerhetsansvarlig som rolle

MANAGING RISK

- Utnevn sikkerhetsansvarlig – som i enhver annen organisasjon
  - Bør rapportere til prosjektleder
  - Kan kombineres med andre roller i prosjektet
  - Opprett et ”team sikkerhet” om nødvendig (store prosjekter)
  - Gjerne representanter fra forskjellige team i prosjektet (forankring)
- Involvert i alle beslutninger og avsjekk av sjekklister
  - Skal verifisere sikkerhetsrelevans av beslutninger
  - Skal godkjenne sjekklister med hensyn på punkter om sikkerhet
- Sikkerhetsarkitektur må inn fra starten
  - Sikkerhetsansvarlig trenger ikke å være arkitekt selv – kan være egen person

Sikkerhetskrav ut til alle utviklere

MANAGING RISK

- Hver enkelt utvikler har ansvar:
  - Design og implementasjon i henhold til arkitektur
  - Verifisere hvilke sikkerhetskrav som ”treffer” – oppfylle kravene
- Designfeil: En komponent har en kritisk rolle uten at den skal ha det
  - Utvikler kjenner ikke arkitekturen – dupliserer kritisk funksjonalitet
  - Kravspesifikasjon for komponent oppfyller ikke sikkerhetskrav
- Implementasjonsfeil
  - Samme type feil oppstår under koding – spesifikasjoner følges ikke
  - Utvikler legger til funksjonalitet (gjerne ”bare” internt i komponent) istedenfor å bruke funksjonalitet fra andre komponenter
- Vanlig feiloppfatning: Sikkerhetsfunksjonene tar seg av all sikkerhet
  - Funksjoner er nødvendig, men tetter ikke et hullete system
  - Utviklerne skal ikke tro at sikkerhet bare angår sikkerhetskomponentene!

Involver hele teamet aktivt

MANAGING RISK

- Modell 1: Sentralstyrte sikkerhetskrav og påbud om oppfyllelse
  - Fordel: Kan ha god sikkerhetskompetanse og miljø sentralt
  - Erfart ulempe: Krav som ikke er forankret/forstått, virker negativt på utviklerne
  - Konflikt sikkerhetsteam – utviklere er vanlig
  - Uten nitid oppfølging risikerer en at kravene neglisjeres
- Modell 2: Delegert utvikling av sikkerhetskrav, delegert ansvar
  - Involver utviklere (alle eller de mest sentrale) i risikoanalysen
  - Sikkerhetsansvarlig spesifiserer overordnede sikkerhetskrav
  - Sikkerhetsansvarlig kvalitetssikrer (sikkerhets-)arkitektur
  - Utviklere/team spesifiserer detaljerte krav sammen med sikkerhetsansvarlig
  - Sikkerhetsansvarlig brukes som ressursperson av utviklerne
  - Testplaner for sikkerhet utvikles på samme måte
  - Ulempe: Svakere fagmiljø hvis utviklerne ikke er sikkerhetskompetente
- Dette er bare mulig dersom prosjektledelsen aktivt støtter arbeidet

Personalbehandling

MANAGING RISK

- Menneskene er største trussel og største ressurs
  - Ansatte må akseptere seg selv som en potensiell risiko
  - Ansatte må se sin rolle i sikkerhetsarbeidet og hva hver enkelt kan bidra med
- Skap en prosjektkultur for sikkerhet og kvalitet
  - ”Her tar vi ikke snarveier, men følger rutinene”
  - Nyansatte fanges opp av bedriftskulturen
- Ha en personalpolitikk som sikrer fornøyde og lojale ansatte!

Personellsikkerhet og administrasjon

MANAGING RISK

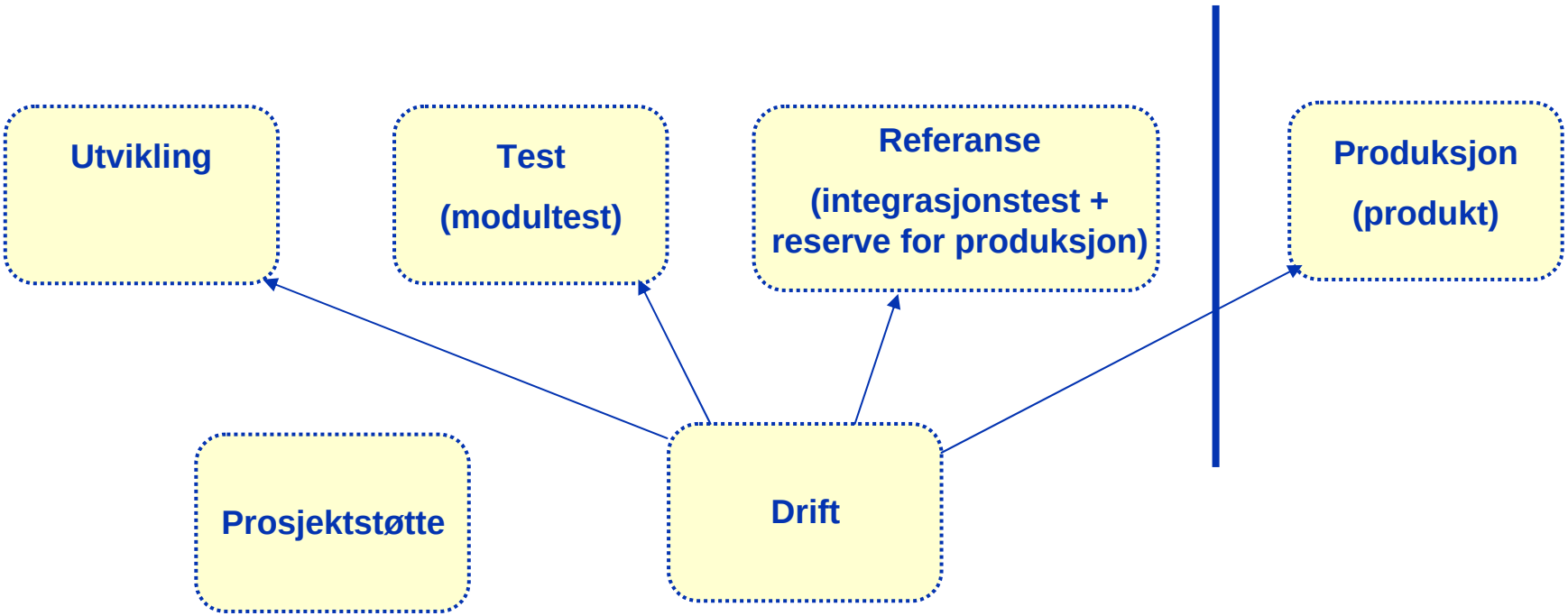
- En korrupt utvikler kan gjøre svært mye skade
  - Sørge for at en kan stole på dem
  - Eller ha systemer som kan oppdage og fange opp tilfeller
  - Forskjellige roller/personer for utvikling, transport, testing, godkjenning?
  - Logging og konfigurasjonsstyring
- En korrupt systemadministrator kan gjøre svært mye skade
  - Tilgang til det meste....
- Sikkerhet i utviklingsprosjekter er mest personellsikkerhet
  - Siden utviklere nødvendigvis må ha høye rettigheter i systemer
  - Administrative rutiner og personellsikkerhet, rolleseparasjon

Utviklingsprosessen

MANAGING RISK

- Sikkerhet krever skikkelig prosess og metodikk
  - Siden sikkerhet er en kvalitetsegenskap
  - Oversikt: Abstraksjonsnivå for modellering (for tillit)
  - Velg skikkelige verktøy
  - .... som utviklerne må kunne ordentlig
- Utviklingsprosess, designkrav, kvalitetskrav
  - Generelle krav til komponenter
  - Sikkerhetsarkitektur – definere tillit for å motstå trusler
  - Sikkerhetstjenester og –mekanismer
- Endringsprosesser er viktige
  - Konsekvenser for tillit ved nye / endrede komponenter?
  - Tillegg: Overholde tillitsmodell og sikkerhetskrav
  - Endringer i tillitsmodell, skifte av tiltrodde komponenter?

Eks.: Systemlandskap for utvikling



- Utvikling: Høy dynamikk – regler og roller for transport til test (kort vei til test)
- Test: Egne testere? Tilbakeføring til utvikling. Regler/roller for transport.
- Referanse: Kloning av produksjonsmiljø – også for beredskap? Ekte data?
- Produksjon: Skarpt atskilt – hos kunde, tilgjengelig for sluttbrukere, ekte data

Hva med verktøy?

MANAGING RISK

- Spesifikasjoner eller kode kompileres
  - Må stole på utviklingsverktøyene – sikkerhet og kvalitet
  - En angriper kan gjøre mye gjennom en kompilator....
  - Hva er kvaliteten på verktøyene, og generert kode?
- Sikkerhetsevaluering, verifikasjon, testing
  - Må stole på verktøy også her

Konklusjoner

Konklusjoner

MANAGING RISK

- Sikkerhet er en kvalitetsegenskap
  - Dersom systemet har feil, har det sikkerhetsfeil også
- Sikkerhetskrav er funksjonelle krav
  - Som må støttes av det totale systemet
  - Risikoanalyse for hvilke sikkerhetskrav systemet skal kunne støtte
- Dernest trengs sikkerhetsfunksjoner
  - Som ikke skal kunne omgås
  - Og sikkerhetsarkitektur for oversikt og tillitsmodellering
- Forankring av sikkerhetsarbeidet i utviklingsprosjekt
  - Prosjektledelsen må ta ansvar
  - Rolle som sikkerhetsansvarlig
  - Engasjer alle utviklere i risikoanalyse, krav og oppfølging
- Skikkelig utviklingsmetodikk er nødvendig
  - Rolleseparasjon og inndeling i systemer
  - Gode verktøy som utviklerne må kunne

www.dnv.com

Jon.Olnes@dnv.com

+47 478 46 094