hyväksymispäivä arvosana

arvostelija

Riskianalyysimenetelmät

Jouni Meriläinen

Helsinki 21. huhtikuuta 2003

Seminaariesitelmä

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Riskianalyysimenetelmät

Jouni Meriläinen

Seminaariesitelmä
Tietojenkäsittelytieteen laitos
Helsingin yliopisto
21. huhtikuuta 2003, 20 sivua

Kirjoituksessa tarkastellaan aluksi lyhyesti perinteistä riskianalyysiä ja sen ongelmia. Sen jälkeen kuvaillaan muutamaa lähestymistavaltaan erilaista, laajassa käytössä olevaa systemaattista riskianalyysimenetelmää. Kohdealueena on tietoturva, mutta menetelmät soveltuvat tietojärjestelmien yleisempäänkin riskianalyysiin. Vikapuuanalyysin (engl. *Fault Tree Analysis, FTA*) yhteydessä tarkastellaan myös erityisesti tietoturvaan tarkoitettua menetelmää, uhkapuita (engl. *threat trees*). Vika- ja vaikutusanalyysin (engl. *Failure Mode and Effects Analysis, FMEA*) ohessa kerrotaan myös kriittisyysanalyysistä (engl. *Criticality Analysis, CA*). Poikkeamatarkastelun (engl. *Hazard and Operability Studies, HAZOP*) lisäksi kuvaillaan siitä ohjelmistojen analysointia varten kehitettyä SHARD-menetelmää.

Aiheluokat (Computing Reviews 1998): K.6.5

Avainsanat: riskianalyysi, vikapuuanalyysi, FTA, uhkapuu, vika- ja vaikutusanalyysi, FMEA, kriittisyysanalyysi, poikkeamatarkastelu, HAZOP, SHARD

Sisältö

1 Johdanto

Riskillä tarkoitetaan yleiskielessä ”jonkin menetyksen, tappion tai muun epäedullisen tapahtuman mahdollisuutta, uhkaa tai vaaraa” [Haa92]. Määritelmä rajaa käsitteen ulkopuolelle epäedulliset tapahtumat, jotka tapahtuvat varmasti tai eivät varmasti tapahdu, sekä kaikki edulliset tapahtumat.

Riskienhallinnaksi (engl. *risk management*) kutsutaan väljästi ilmaistuna kaikkia niitä toimenpiteitä, joilla riskejä pyritään pitämään hyväksyttävällä tasolla. Tietoisesta riskien hallinnasta on hyötyä aina, kun on olemassa epäedullisten tapahtumien mahdollisuus — eli likimain kaikessa ihmisen toiminnassa. Mitä merkittävämmät ovat riskit, sitä tärkeämpää on, että järjestelmällinen riskienhallinta on mukana toiminnassa alusta loppuun saakka. Tietoturvan alalla riskianalyysiä voidaan käyttää ainakin tietoturvakatselmuksien yhteydessä. Riskienhallintaa ja sen mukana riskianalyysiä on syytä kuitenkin pitää osana kaikkien järjestelmien suunnittelua ja käytön aikaista muutoksenhallintaa.

Riskienhallinnan ensimmäinen vaihe on riskianalyysi (engl. *risk analysis*), joka koostuu uhkien tunnistamisesta (engl. *hazard identification*) ja niiden riskien suuruuden arvioinnista (engl. *risk estimation*). Toinen vaihe on riskien merkityksen arviointi (engl. *risk evaluation*). Viimeinen vaihe, riskien valvonta (engl. *risk control*), pitää sisällään muun muassa toimet riskien pienentämiseksi [SFS00]. Riskianalyysin tehtävä on siis tunnistaa riskit ja tuottaa tietoja niiden suuruuksista riskien merkityksen arviointia varten. Käsitteistö ja terminologia eivät ole täysin vakiintuneita.

Tässä kirjoituksessa rajaudutaan tarkastelemaan pelkästään riskianalyysiä. Luvussa 2 kuvaillaan perinteistä riskianalyysiä ja riskianalyysin yleisiä ongelmia. Luvun tarkoituksena on esitellä käsitteistöä ja tarjota vertailukohta seuraavia lukuja varten. Näissä puolestaan esitellään muutama tunnettu järjestelmällinen riskianalyysimenetelmä. Näillä pyritään välttämään perinteisen riskianalyysin ongelmia, erityisesti epävarmuutta uhkien luettelon kattavuudesta.

Luvussa 3 on aiheena vikapuuanalyysi (engl. *Fault Tree Analysis, FTA*) sekä tälle läheinen menetelmä, erityisesti tietoturvaan tarkoitetut uhkapuut (engl. *threat trees*).

Luvussa 4 kerrotaan vika- ja vaikutusanalyysistä (engl. *Failure Mode and Effects Analysis, FMEA*) sekä tämän laajennuksesta nimeltä vika-, vaikutus- ja kriittisyysanalyysi (engl. *Failure Mode, Effects and Criticality Analysis, FMECA*).

Luvussa 5 kuvaillaan poikkeamatarkastelua (engl. *Hazard and Operability Study, HAZOP*) sekä tämän pohjalta erityisesti ohjelmistojen analysointiin kehitettyä SHARD-menetelmää.

2 Perinteinen riskianalyysi

Tässä luvussa tarkastellaan perinteisen riskianalyysin menetelmiä. Aihepiiriin lasketaan tässä kuuluvaksi kaikki sellainen riskianalyysi, missä ei noudateta mitään riskianalyysin prosessin kokonaan tai merkittävältä osin kattavaa, nimettyä ja yksityiskohtaisesti kuvattua menetelmää.

Riskianalyysin ensimmäinen vaihe on uhkien tunnistaminen (luku 2.1). Toinen vaihe, riskien suuruuden arviointi, käsittää tyypillisesti arviot uhkaavan menetyksen suuruudesta (luku 2.2) ja uhan todennäköisyydestä (luku 2.3).

2.1 Uhkien tunnistaminen

Riskin käsitteeseen liittyy jokin epäedullinen tapahtuma, joka saattaa tapahtua. Tällaisten mahdollisten tapahtumien luettelon laatimista kutsutaan uhkien tunnistamiseksi. Tietoturvauhkien tapauksessa luettelon laatimisessa on syytä olla mukana yrityksestä ainakin tietojärjestelmän, suojattavan tiedon ja yritysturvallisuuden vastuuhenkilöt tai asiantuntijat [Cou77]. Uhat on eriteltävä tarkoituksenmukaisella tasolla. Hyvin laveasti määritellyt uhat (”tietoturva vaarantuu”) ovat analyysissä sellaisenaan varsin hyödyttömiä, samoin kuin niin tiukasti kaikkien parametriensä puolesta rajatut, ettei niitä voi soveltaa kuin yhteen uhan toteutumaan.

Alkeellisimmillaan uhkien tunnistamiseen ei käytetä mitään järjestelmällistä menetelmää, vaan tehtävään osallistuvat pohtivat uhkia täysin vapaasti. Ainakin mukaan luetteloon tulee ottaa jo aiemmin toteutuneiksi tiedetyt uhat. Tätä varten on organisaatiossa syytä olla dokumentoituna toteutuneet riskit [Par81, s. 126–127]. Merkittävänä ongelmana on epävarmuus epäjärjestelmällisesti laaditun uhkien luettelon täydellisyydestä. Muut vaatimukset, kuten oikeellisuus ja täsmällisyys [GCW92], tai vaikkapa todennettavuus ja keskinäinen ristiriidattomuus, lienevät helpommin saavutettavissa.

Toimialalle soveltuvan tarkistuslistan (engl. *checklist*) käyttö voi auttaa laatimaan

kattavamman uhkaluettelon. Tarkistuslistat ovat helppoja käyttää ja ne voivat auttaa havaitsemaan kokonaan huomaamatta jääneitä uhkia. Ne voivat myös antaa virikkeitä oman mielikuvituksen käytölle. Boehmin esittämä lista kymmenestä merkittävimmästä ohjelmistotuotannon riskilähteestä on hyvä esimerkki tarkistuslistasta [Boe91]. TT puolestaan on julkaissut VTT:ssä laaditun tietoriskien hallinnan kysymyslistan [Teo01, s. 34–35].

Tarkistuslistoilla on myös ongelmia. Riskianalyysin kohteelle ei välttämättä ole olemassa hyvin soveltuvaa tarkistuslistaa. Toisaalta sopivassakaan tarkistuslistassa ei ole voitu ottaa huomioon tapauskohtaisia uhkia. Tarkistuslistan käytön kokemattomalle henkilölle mahdollisesti tuottama mielikuva suoritettavan analyysin perusteellisuudesta on siis erheellinen, huomauttavat Miettinen ja Kajava [MiK94, s. 17]. Guarro muistuttaa lisäksi [Gua87], että riskianalyysiä ei siksikään voi korvata pelkällä tarkistuslistan läpikäymisellä, että tämä ei tuota tietoa riskien suuruuksista.

Kattavan uhkaluettelon laatimista auttaa myös tehtävän jakaminen osatehtäviin tai eri näkökulmiin, joita kutakin mietitään erikseen. Erään luontevan jaon muodostaa suojattavien kohteiden (engl. *assets*) luettelo. Toisaalta uhat voidaan luokitella tahallisesti aiheutettuihin, tahattomiin ja luonnonilmiöiden aiheuttamiin. Ihmisten aiheuttamat voidaan edelleen jakaa luokittelemalla ihmiset tarkoituksenmukaisiin ryhmiin [Par81, s. 115, 135–138]. Eräs tapa on myös jakaa uhat kolmeen luokkaan: tiedon luottamuksellisuuteen (engl. *confidentiality*) tai eheyteen (engl. *integrity*) kohdistuvat ja palvelunestoa (engl. *denial of service*) aiheuttavat [Amo94, s. 3–6].

Luokittelun ääri-ilmiö ovat riskitaksonomiat, joissa toimialan kaikki eri tyyppiset uhat on pyritty yleistämään ja luokittelemaan hierarkkiseksi järjestelmäksi. Eräs tämän tyyppinen luokitus on SEI:n riskitaksonomia ohjelmistokehitykseen [Car93, s. A-2]. Riskitaksonomia auttaa saavuttamaan rakenteisen, toistettavan riskianalyysiprosessin ja jäsentää riskianalyyseihin osallistuvien välistä viestintää [Car93, s. 7]. Muilta osin hyvät ja huonot puolet ovat pääosin samat kuin tarkistuslistoilla tai uhkien luokittelulla, joiden molempien piirteitä taksonomioilla on.

2.2 Menetyksen suuruuden arviointi

Riskin suuruudella (engl. *risk exposure*) tarkoitetaan yleensä menetyksen suuruuden odotusarvoa [Boe91]. Tämän määrittämiseksi tulee siis selvittää kunkin uhan todennäköisyys ja menetyksen suuruus uhan toteutuessa.

Jotta riskejä voisi verrata keskenään, tarvitaan yhteinen mittayksikkö menetyksille. Raha on määritelmänsä mukaan tällainen yleinen arvon mitta. Courtneyn mielestä [Cou77] raha on peräti ainoa hyvä mittayksikkö, koska sen avulla riskejä voidaan riskien merkityksen arviointivaiheessa verrata paitsi toisiinsa myös riskien pienentämisestä aiheutuviin kustannuksiin. Menetyksen arvon ei kuitenkaan aina tarvitse olla tarkka, vaan suuruusluokka riittää varsinkin silloin, jos riskit ovat enimmäkseen aivan eri suuruusluokkaa keskenään.

Kaikille menetyksille ei ole helppoa määrittää rahallista arvoa. Voi olla vaikeaa arvioida esimerkiksi menetetyn maineen tai kilpailijalle vuotaneen tiedon aiheuttamaa rahallista menetystä. Menetys voi olla myös yhdistelmä tällaisia immateriaalisia (engl. *intangible*) menetyksiä sekä välittömiä rahallisia menetyksiä (menot) ja välillisiä rahallisia menetyksiä (menetetyt tulot) [Bri77].

Menetyksen arvon määrittäminen rahana voidaan joissain tapauksissa kokea suorastaan vastenmielisenä. Tuntuu esimerkiksi irvokkaalta mitata ihmishengen menetystä rahassa — siitäkin huolimatta, että kuoleman riskin pienentämiseen käytettävät kustannukset ovat yhteiskunnassa jatkuvasti arvioinnin kohteena. Jos rahaa ei haluta käyttää menetyksen mittayksikkönä, Parker ehdottaa [Par81] yhtenä vaihtoehtona nimeämättömän numeerisen arvoa ilmaisevan suureen käyttöä. Kaikkia riskianalyysimenetelmiä, joissa määritetään riskin suuruudelle jokin numeerinen arvo, kutsutaan kvantitatiivisiksi [Gua87].

Kvalitatiivisissa riskianalyysimenetelmissä puolestaan menetyksen suuruutta riittää kuvata sanallisesti. Kuvaus on tavallisesti jokin järjestävä luokittelu, esimerkiksi *vähäinen – kohtalainen – huomattava*. Tällainen menetysten luokittelu on helpompaa kuin numeeristen arvojen määrittäminen ja saavutettava tarkkuus voi olla joihinkin tarkoituksiin riittävä. Käytettävien luokkien merkityksestä tulee kuitenkin vallita yksimielisyys yhtäältä riskianalyysin laatimiseen osallistuvien kesken ja toisaalta riskianalyysin laatijoiden ja sen tulosten käyttäjien välillä [Gua87].

2.3 Uhan todennäköisyyden arviointi

Riskin suuruuden määrittämiseksi tulee menetyksen suuruuden lisäksi tietää uhan todennäköisyys. Tämä voi olla todennäköisyys uhan toteutumiselle kussakin määritellyssä tilanteessa, mutta hyvin usein käsitellään todennäköisyyttä uhan toteutumiselle aikayksikössä [Bri77, s. 25]. Voidaan käyttää myös luotettavuustekniikassa tavanomaista suuretta, vikojen lukumäärää aikayksikössä, tai sen käänteislukua, keskimääräistä vikaväliä [EMS79, s. 100–101] (engl. *Mean Time Between Failures, MTBF*).

Uhan todennäköisyys saatetaan tuntea tilastotietojen perusteella. Nämä tilastot saattavat olla joko riskianalyysia laativan organisaation omia, tai julkaistuja. Omien tilastojen olemassaolon edellytys on luonnollisesti, että organisaatiossa uhkien toteutumiset dokumentoidaan järjestelmällisesti. Redmill varoittaa [Red02] kuitenkin, että tilastojen soveltuvuutta tulee arvioida kriittisesti. Tarkastelun kohteena olevat olosuhteet eivät välttämättä ole samat kuin tilastoon otettujen tapausten kohdalla, eikä tilaston olosuhteita välttämättä ole dokumentoitu riittävän tarkasti.

Usein käytettävissä ei ole sopivaa tilastotietoa uhan toteutumistiheydestä. Ääritapauksessa voi arvioinnin kohteena olla tapahtuma, jota ei tiedetä ikinä tapahtuneen, mutta joka on selvästi mahdollinen [Red02]. Todennäköisyyden frekvenssitulkinnan sijaan joudutaan siksi usein käyttämään subjektiivista todennäköisyyttä. Tämä tarkoittaa, että todennäköisyytenä käytettävä arvo mittaa arvioijan uskomuksen astetta kyseessä olevan tapahtuman toteutumiseen [Gua87].

Edes asiantuntijoiden subjektiivisia arvioita ei kuitenkaan voi pitää täysin luotettavina. Fischhoff, Slovic ja Lichtenstein ovat tutkineet subjektiivista todennäköisyyden arviointia [FSL02]. Ryhmä koehenkilöitä, myös alansa asiantuntijat, arvioi uhkien luettelosta tietyille kahdelle uhalle todennäköisyydet, joiden summa oli suurempi kuin toisen ryhmän arvioima todennäköisyys uhalle, joka laveamman määritelmän takia kattoi molemmat ensimmäiselle ryhmälle eritellyistä uhista.

Jos tyydytään kvalitatiiviseen riskianalyysiin, riittää uhkien todennäköisyyksien jaottelu sanallisesti kuvattuihin luokkiin, kuten edellä menetysten suuruuksienkin [Par81, s. 144]. Siinä missä kvantitatiivinen riskianalyysi antaa riskin suuruutena menetyksen odotusarvon, siis skalaarisen arvon jossain yksikössä, liittyy kvalitatiivisessa riskianalyysissä kuhunkin uhkaan pari *(menetyksen suuruus,*

todennäköisyys). Näin on sillä oletuksella, ettei ole kehitetty laskutoimitusta, jolla saadut kaksi sanallista arvoa voidaan yhdistää yhdeksi.

Kvalitatiivisen parin *(menetyksen suuruus, todennäköisyys)* vertailu riskien merkityksen arviointivaiheessa riskin vähentämisestä aiheutuviin kustannuksiin, siis skalaarisiin rahassa ilmoitettuihin arvoihin, on selvästi mahdotonta. Myöskään riskien vertailu toisiinsa ei yleisessä tapauksessa ole helppoa. Määritellyiksi voidaan sopia tapaukset, joissa vähintään parin toinen alkio on kahdella riskillä sama, tai jollain riskillä parin kumpikin alkio on suurempi kuin toisella. Sen sijaan, varoittaa Kontio [Kon97, s. 32], ei ole mitään perustetta yleisesti olettaa, että kaksi riskiä ovat yhtäsuuria, jos niiden parien alkiot eroavat yhdellä (tai yhtä monella) luokalla toisistaan, mutta suuruusjärjestyksessä vastakkaisiin suuntiin, siis esimerkiksi *(kohtalainen, huomattava)* ja *(huomattava, kohtalainen)*. Niinpä ei voida verrata pareja, joiden alkioiden erisuuruudet ovat vastakkaisiin suuntiin.

3 Vikapuuanalyysi (FTA)

Tässä luvussa tarkastellaan lähinnä vikapuuanalyysiä (engl. *Fault Tree Analysis, FTA*). Myös uhkapuista (engl. *threat trees*) kerrotaan (luku 3.3), koska jotkin uhkapuut on määritelty vikapuiden kanssa likimain samalla tavalla.

3.1 Vikapuuanalyysin menetelmä

Vikapuuanalyysi on ylhäältä alas -tyyppinen (engl. *top-down*) eli deduktiivinen menetelmä. Siinä pohdinta alkaa lopputuloksista, vioista joita järjestelmän toiminnassa voi esiintyä. Näistä edetään syy–seuraus-ketjussa taaksepäin kohti vian mahdollisesti aiheuttaneita syitä [Fen94]. Lähestymistavan perusteella vikapuut soveltuvat tapauksiin, joissa järjestelmän vikatoimintojen joukko on tunnettu paremmin kuin sen osien vikaantumistavat, tai halutaan nimenomaan analysoida tiettyjä koko järjestelmän vikoja, esimerkiksi vain nimetyt tietoturvauhat. Vikapuuanalyysi mahdollistaa myös analyysin ulottamisen vain haluttuun syvyyteen saakka työmäärän rajaamiseksi.

Ensimmäinen tehtävä on laatia lista järjestelmän vioista eli niin kutsutuista huipputapahtumista (engl. *top event*). Tähän uhkien tunnistamisvaiheeseen menetelmä ei tarjoa mitään järjestelmällistä keinoa. Jos analyysin kohteelle on aiemmin tehty alustava vaara-analyysi (engl. *Preliminary Hazard Analysis, PHA*), voidaan

sen tuloksia käyttää tämän vaiheen apuna [LeH83]. Alustavassa vaara-analyysissä pyritään mieluiten jo suunnittelutyön varhaisessa vaiheessa tunnistamaan vaaratilanteet, joita suunniteltava järjestelmä voi käyttöön tullessaan aiheuttaa itselleen, käyttäjilleen tai ympäristölleen [GCW92].

Kukin huipputapahtuma on seuraavassa vaiheessa oman puunsa juurisolmuna. Kussakin puussa eritellään aluksi, minkä ehtojen voimassa ollessa kyseinen vika voi esiintyä. Ehdot muodostavat puun seuraavan tason ja niitä yhdistää yleensä *ja*- tai *tai*-operaattori, tai harvemmin jokin muu looginen operaattori. Näiden ehtojen ehdot puolestaan muodostavat puun seuraavan tason ja niin edelleen. Puuta jatketaan, kunnes lehtitason solmuina on sellaisia syitä, joiden todennäköisyys on helppo määrittää [LeH83]. Näitä kutsutaan perustapahtumiksi (engl. *basic event*). Joidenkin haarojen kasvattaminen voidaan kuitenkin joutua jättämään kesken, jos syiden ketjua ei esimerkiksi osata eritellä pidemmälle tai selvästi nähdään, että kyseisen haaran vaikutus puussa tulee olemaan mitätön [NRC81, s. IV-4]. Puussa käytettävät merkinnät merkityksineen on esitelty seuraavassa aliluvussa.

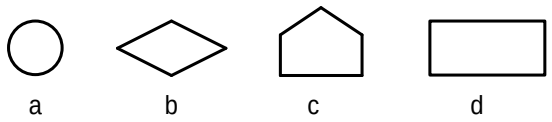
Kun puu on saatu piirrettyä, sitä tulee sieventää logiikan laskusäännöillä. Tavoitteena on ainakin poistaa puusta syyt, jotka on kokonaan sievennettävissä pois. Jäljelle jääviä syitä, jotka ovat siis aidosti huipputapahtuman ehtoja, kutsutaan perimmäisiksi syiksi (engl. *primal cause*) [LeH83]. Suuren puun sievennys ei ole aivan pieni tehtävä. Automaattisen sievennyksen algoritmien kehittämiselle haasteellista on ollut se, että sama ehto voi toistua puussa eri kohdissa [SiA96]. Sievennetyn puun lehtisolmuihin liitetään niitä vastaavat todennäköisyydet. Näistä lähtien on mahdollista laskea huipputapahtuman todennäköisyys käyttäen todennäköisyyslaskennan laskusääntöjä loogisille lausekkeille.

Vikapuuanalyysi vaikuttaa ensi katsomalta aukottoman varmatoimiselta riskianalyysimenetelmältä. Sen tuottamien tulosten tarkkuutta rajoittavat tietysti lehtitason todennäköisyyksien epätarkkuudet. Toisaalta voidaan tehdä herkkyysanalyysi kunkin lehtitason solmun vaikutuksesta huipputapahtumaan [LeH83]. Tällöin ei huipputapahtumalle lasketun todennäköisyyden epätarkkuus ole suoraan ongelma. Toisaalta lehtitason todennäköisyyksien välillä saattaa olla riippuvuuksia, jotka ovat loogisilla lausekkeilla ilmaistavia monimutkaisempia. Myös tämä rajoittaa vikapuun tulosten luotettavuutta.

3.2 Vikapuun merkinnät

Tässä aliluvussa esitellään vikapuissa käytettävät merkinnät. Nämä perustuvat NUREG-0492-käsikirjaan [NRC81, s. IV-3]. Myös muita merkintätapoja on olemassa, esimerkiksi SFS-IEC 60300-3-9 [SFS00, s. 44] käyttää tässä esiteltävien amerikkalaisten loogisten porttipiirien symbolien tilalla eurooppalaisia.

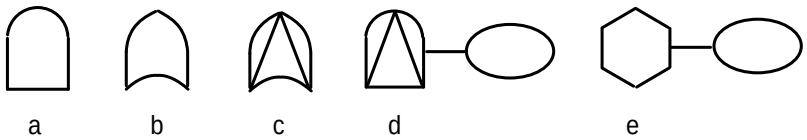
Vikapuun lehtitason solmuja on kolmenlaisia. Nämä on esitelty kuvassa 1. Perustapahtumat ovat sellaisia, joita ei enää tarvitse eritellä pidemmälle. Niiden todennäköisyys on siis riittävän uskottavasti määritettävissä. Perustapahtumia merkitään ympyrällä. Erittelemättömät tapahtumat (engl. *undeveloped events*) ovat sellaisia, joita ei joko osata eritellä pidemmälle tai joiden vaikutus puussa arvioidaan niin vähäiseksi, ettei niiden käsittelyyn kannata nähdä vaivaa. Erittelemättömien tapahtumien merkintä on suunnikas (salmiakki). Puuhun voidaan myös merkitä järjestelmässä normaalisti ilmeneviä tapahtumia (engl. *external event*), jotka eivät siis ole vikoja, mutta voivat osallistua jonkin vian syntyyn. Symbolina on ”talo”. Vikapuun huipputapahtumaa ja sisäsolmuja eli välitapahtumia (engl. *intermediate event*) puolestaan merkitään suorakulmioilla.



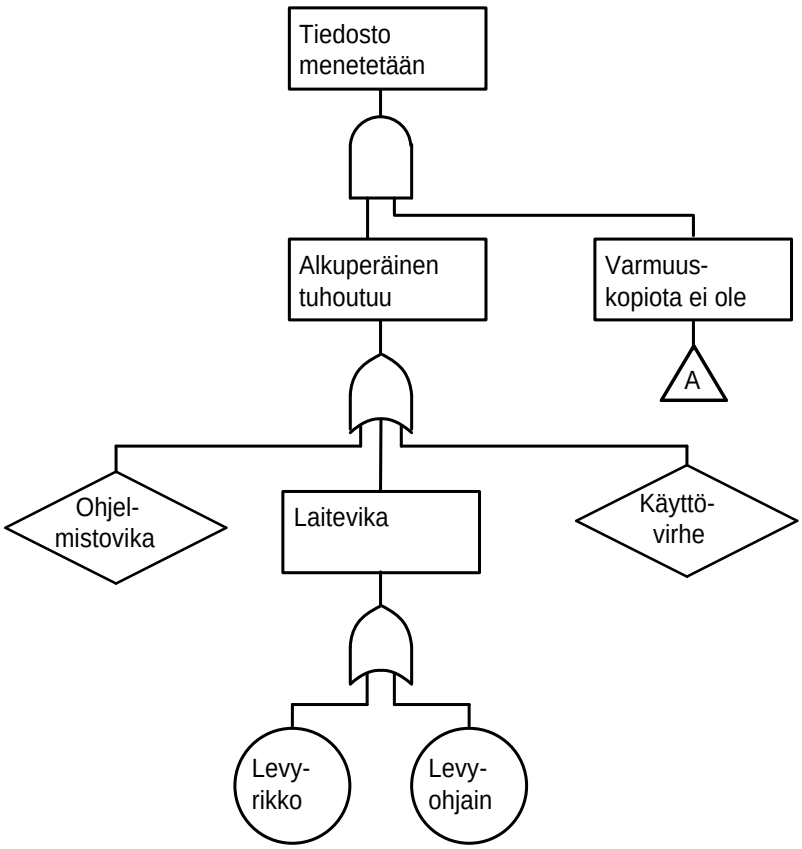
Kuva 1. Vikapuun merkintöjä: a on perustapahtuma, b on erittelemätön tapahtuma, c on normaali tapahtuma ja d on huippu- tai välitapahtuma.

Loogisia operaattoreita puun tasojen välillä merkitään digitaalielektroniikan porttipiirien symboleilla, jotka on esitetty kuvassa 2. Tavanomaisten loogisten operaatioiden lisäksi on oma merkintänsä priorisoidulle *ja*-operaattorille (engl. *priority and*). Tämä toimii kuten normaali *ja*, mutta alemman tason tapahtumien pitää tapahtua tietyssä järjestyksessä, joka ilmoitetaan symboliin liittyvässä soikiossa. Kuusikulmiolla merkitään puolestaan esto-toimintoa (engl. *inhibit*). Siinä alemman tason syy johtaa ylemmän tason vikaan, mikäli symboliin liittyvässä soikiossa oleva ehto on voimassa. Jotta vikapuun rakenne säilyisi selkeänä puurakenteena, ei porttipiirejä saa yhdistellä monimutkaisempien loogisten lausekkeiden esittämiseksi, vaan välissä täytyy olla aina välitapahtuma.

Kuvassa 3 havainnollistetaan pienen vikapuun avulla, miten symbolit liitetään toisiinsa. Puista tulee helposti varsin suuria, kun monimutkaisen järjestelmän vikojen syitä lähdetään huolella erittelemään. Tästä syystä on myös oma merkin-



Kuva 2. Vikapuun merkintöjä: a on *ja*-operaatio, b on *tai*-operaatio, c on poissulkeva *tai*-operaatio, d on priorisoitu *ja*-operaatio ja e on *esto*-toiminto.



Kuva 3. Esimerkki vikapuusta. Tiedosto menetetään, jos sekä alkuperäinen että varmuuskopio tuhoutuvat. Varmuuskopion tuhoutumisen alipuu on merkitty esitettäväksi muualla. Alkuperäinen voi tuhoutua ohjelmistovian tai laitevian tai käyttövirheen seurauksena. Ohjelmistovikaa tai käyttövirhettä ei osata eritellä enempää. Laitevika on joko levyrikko tai levyohjaimen vikaantuminen, joiden kummankin todennäköisyys lienee arvioitavissa.

tänsä, tasasivuinen kolmio, alipuun esittämiselle erikseen, esimerkiksi eri sivulla. Saman symbolin avulla vältetään puussa monessa paikassa toistuvien alipuiden piirtäminen moneen kertaan.

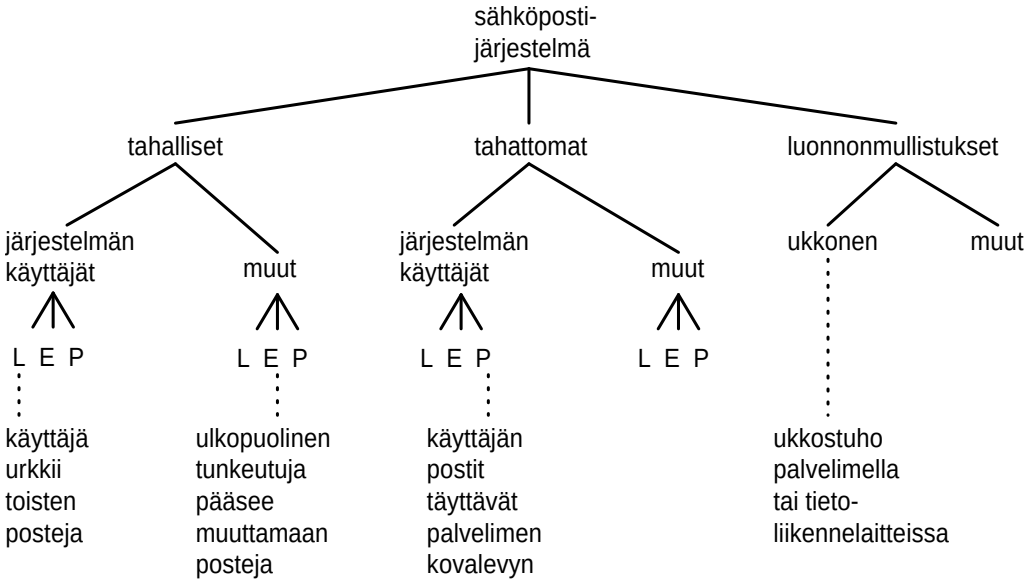
3.3 Uhkapuut

Amoroso esittää [Amo94, s. 15–28] kaksi erilaista uhkapuuksi kutsuttavaa riskianalyysipuuta tietoturvan arviointiin. Näistä toinen muistuttaa hyvin paljon vikapuuta. Tällaisen uhkapuun juurena on jokin tietoturvauhka. Sen lapsisolmuina olevia aliuhkia yhdistää looginen operaattori kuten vikapuissa siten, että uhan toteutuminen on aliuhkien toteutumisten looginen funktio.

Riskin suuruutena Amoroso käyttää uhan vakavuuden ja hyökkääjältä vaadittavien ponnistusten osamäärää. Tavanomaiseen riskin määritelmään verrattuna Amoroso käyttää siis riskin todennäköisyyden tilalla hyökkääjältä vaadittavien ponnistusten käänteislukua. Amoroso huomauttaa, että laskutoimituksia tällaisessa uhkapuussa voi tehdä myös sanallisilla arvoilla. Esimerkiksi voidaan ottaa uhka, joka toteutuu, jos jompi kumpi sen kahdesta aliuhasta toteutuu. Toisen aliuhan hyökkääjältä vaatimat ponnistukset ovat ”huomattavat” ja toisen ”kohtalaiset”. Tällöin ilmeisesti hyökkääjä valitsee helpomman tavan, eli koko uhan hyökkääjältä vaatimat ponnistukset ovat ”kohtalaiset”.

Amoroso esittää myös toisenlaisen uhkapuun. Tässä puuta käytetään uhkien tunnistamiseen. Menetelmänä ovat perättäiset luokittelut eri kriteerien mukaan. Puun juurena on esimerkiksi koko järjestelmään kohdistuvien uhkien joukko, jota ei analyysin alussa tunneta. Kullakin puun tasolla ylemmän tason uhkien joukko jaetaan kahteen luokkaan: johonkin vapaasti valittavaan luokkaan ja sen komplementtiin. Tällaisen luokituksen tarkoitus on varmistua siitä, ettei luokituksen ulkopuolelle jää mitään uhkia. Vaihtoehtoisesti luokitustapana käytetään ylemmän tason uhkien joukon jakoa kolmeen luokkaan: luottamuksellisuus, eheys ja palvelunesto.

Uhkien sanallisia kuvauksia liitetään tällaisen puun lehtisolmuiksi vasta siinä vaiheessa, kun kaikki oleelliset luokitteluperusteet on käytetty. Menetelmän tarkoituksena on saada aikaan mahdollisimman kattava uhkaluettelo jatkamalla järjestelmällistä analyysiä mahdollisimman pitkään eli siirtymällä vapaamuotoisiin sanallisiin kuvauksiin mahdollisimman myöhään. Tällaisesta uhkapuusta on esimerkki kuvassa 4.



Kuva 4. Esimerkki uhkapuusta. Sähköpostijärjestelmän uhat on jaettu eri luokkiin. L on tässä lyhenne luottamuksellisuuteen, E eheyteen ja P palvelunestoon liittyville uhkille. Uhkien sanallisia kuvauksia on laitettu vain muutama esimerkin vuoksi.

4 Vika- ja vaikutusanalyysi (FMEA)

Tässä luvussa tarkastellaan vika- ja vaikutusanalyysiä (engl. *Failure Mode and Effects Analysis, FMEA*) sekä lyhyesti myös sille läheistä menetelmää, vika-, vaikutus- ja kriittisyysanalyysiä (engl. *Failure Mode, Effects and Criticality Analysis, FMECA*). Lisäksi tarkastellaan, miten vika- ja vaikutusanalyysiä on sovellettu ohjelmistopohjaisten järjestelmien analysointiin.

Vika- ja vaikutusanalyysi on alhaalta ylös -tyyppinen eli induktiivinen menetelmä, toisin kuin edellä esitelty vikapuuanalyysi [Fen94]. Suomenkielisen terminologian epätäsmällisyyden ei pidä antaa hämätä: tässä luvussa ”vialla” tarkoitetaan järjestelmän komponentin vikaantumista (engl. *failure*) kun edellä samalla sanalla tarkoitettiin tarkasteltavan järjestelmän vikatoimintoa (engl. *fault*). Tärkeät järjestelmät suunnitellaan usein niin, että minkään yksittäisen komponentin vikaantuminen ei aiheuta koko järjestelmän viallista toimintaa [Woo94].

4.1 FMEA-lomake

Chrysler Corporation, Ford Motor Company ja General Motors Corporation [CFG95] ovat laatineet vika- ja vaikutusanalyysin käsikirjan, jota QS-9000-laatujärjestelmää noudattavien toimittajien edellytetään käyttävän. Oleellisesti samaa menetelmää käytetään niin tuotteen suunnittelusta peräisin olevien (*"design FMEA"*) kuin sen tuotantoprosessiinkin liittyvien (*"process FMEA"*) riskien analysointiin.

Analyysin ensimmäinen vaihe on sen kohteena olevan järjestelmän jakaminen osajärjestelmiinsä ja hierarkkisesti edelleen aina osiinsa saakka. Jakaminen voidaan lopettaa, kun on päästy sellaisten osien tasolle, joiden vikaantumismuodot eli eri tavat vikaantua on helposti nähtävissä. Järjestelmälle itselleen ja sen osajärjestelmille ja näiden komponenteille täytetään tämän jälkeen FMEA-lomake kullekin. Lomakkeella on sen täyttäjän tietojen lisäksi analyysin kohteen tunnistetiedot, mukaan lukien kohteesta vastaava taho.

Lomakkeella on perustietojen jälkeen nimetty tarkasteltava osa ja kuvattu sen normaali toiminta. Tämän jälkeen esitetään osan mahdolliset eri vikaantumismuodot. Kullekin vikaantumismuodolle määritellään vakavuusaste S (engl. *severity*) asteikolla 1–10, missä luokat on sanallisesti määritelty siten, että 1 tarkoittaa ”ei vaikutusta” ja 10 tarkoittaa ”vakava arvaamatta ilmenevä vaara”.

Tämän jälkeen lomakkeella luetellaan kunkin vikaantumismuodon mahdolliset aiheuttajat tai syntytavat. Kullekin näistä ilmoitetaan esiintymistodennäköisyys O (engl. *occurrence*) asteikolla 1–10. Arvo 1 merkitsee äärimmäisen epätodennäköistä ja arvo 10 lähes väistämätöntä. Luokat 1–10 on sanallisen kuvauksen lisäksi määritelty numeerisina todennäköisyyden arvoalueina.

Kunkin vikaantumismuodon ja kunkin tämän aiheuttajan tai syntytavan osalta ilmoitetaan myös, millä menetelmillä vikaa tällä hetkellä pyritään valvomaan. Vielä ilmoitetaan, millä todennäköisyydellä D (engl. *detection*) puheena olevalla syntytavalla tullutta vikaa eivät sen nykyiset vastatoimet estä, vaan vika jää huomaamatta. Asteikolla 1–10 arvo 1 annetaan vioille, jotka huomataan lähes varmasti, ja arvo 10 vioille joita on lähes mahdoton huomata tai joille ei vastatoimia ole käytössä.

Varsinaisen riskianalyysin viimeinen vaihe on riskin suhteellisen suuruuden määritys. Tämä FMEA:n riskiprioriteettiarvo (engl. *Risk Priority Number*) saadaan tulona $RPN = S \cdot O \cdot D$. Tuloksen arvoalue on siis 1–1000. FMEA-menetelmä jat-

kuu riskianalyysivaiheen jälkeen vielä niin, että kullekin riskille määritetään toimenpide-ehdotus ja arvioidaan, mihin arvoon RPN pienenisi toimenpiteen johdosta.

Vika- ja vaikutusanalyysi soveltuu järjestelmiin, jotka osataan jakaa sellaisiin komponentteihin, joiden vikaantumismuodot on kuvattavissa. Koko järjestelmää koskevat uhat tullevat kattavasti tunnistettua, kun käydään läpi kaikkien osien eri vikaantumismuodot. Koko järjestelmän riskien suuruuksien laskenta osien RPN-arvojen perusteella jää pitkälti analysoijan oman päättelykyvyn varaan.

Menetelmän hyviä puolia tietoturvan kannalta on, että riskin suuruuden arvioinnissa otetaan huomioon, voiko vika syntyä niin, ettei sitä huomata. Monissa tietoturvauhkissahan tekijänä pidetään toimintansa salaamaan pyrkivää ilkeämielistä hyökkääjää.

4.2 Kriittisyysanalyysi

Standardi MIL-STD-1629A [DoD80] määrittelee vika-, vaikutus- ja kriittisyysanalyysin (FMECA). Sen alkuosa on jokseenkin tavanomainen FMEA-riskianalyysi. Tämän perään tehdään kriittisyysanalyysi (engl. *Criticality Analysis, CA*), jonka tarkoituksena on selvittää järjestelmän kunkin osan kriittisyys järjestelmän toiminnan kannalta. Standardin muita osia ei tässä tarkastella.

Kriittisyysanalyysin ensimmäinen vaihe on vikaantumismuodon kriittisyyslukujen laskeminen. Vikaantumismuodon m kriittisyysluku (engl. *failure mode criticality number*) jollekin osalle ja vakavuusluokalle saadaan neljän luvun tulona $C_m = \beta\alpha\lambda_p t$. Vaikutustodennäköisyys β (engl. *failure effect probability*) on ehdollinen todennäköisyys sille, että vikaantumismuoto johtaa tarkastelun kohteena olevan vakavuusluokan vaikutukseen ehdolla, että vikaantumismuodon vika on tapahtunut. Osan vikatiheys λ_p (engl. *part failure rate*) ilmaistaan vikoina ajan t yksikköä kohti. Vikaantumismuotosuhde α (engl. *failure mode ratio*) on se osuus λ_p:stä, joka koskee tarkasteltavana olevaa vikaantumismuotoa, eli $\alpha\lambda_p$ on tarkasteltavana olevan vikaantumismuodon vikojen esiintymistiheys osassa. Aika t ilmaisee, kuinka pitkän aikaa tarkasteltavana oleva osa on toiminnassa. Jos vikatiheys on suhteutettu johonkin muuhun kuin aikaan, korvataan aika t samalla suureella. Vikaantumismuodon kriittisyysluku on siis osan tarkasteltavana olevan vakavuusluokan ja vikaantumismuodon vikaantumisten lukumäärän odotusarvo osan toiminta-aikana.

Osan r kriittisyysluku (engl. *item criticality number*) jollekin vakavuusluokalle saadaan summana $C_r = \sum_m C_m$ osan vikaantumismuodon kriittisyysluvuista C_m yli kaikkien vikaantumismuotojen m, jotka ovat kyseistä vakavuusluokkaa. Osan kriittisyysluku on siis osan tarkasteltavana olevan vakavuusluokan kaikkien eri vikaantumismuotojen vikaantumisten kokonaislukumäärän odotusarvo osan toiminta-aikana. Osien kriittisyyslukuja saman vakavuusluokan sisällä vertaamalla saadaan käsitys eri osien kriittisyydestä järjestelmässä toisiinsa verrattuna.

4.3 Ohjelmistojen analysointi

Vika- ja vaikutusanalyysi on alunperin suunniteltu järjestelmille, jotka koostuvat kuluvista, vikaantuvista komponenteista. Luotettavuustekniikalle tyypilliseen tapaan komponenttien keskimääräinen vikaantumistiheys on yleensä tunnettu. Vika- ja vaikutusanalyysi soveltuu myös sellaisten järjestelmien analysointiin, jotka sisältävät vikaantuvia komponentteja, vaikka ne eivät koostuisikaan pelkästään vikaantuvista komponenteista.

Menetelmän soveltuvuus ohjelmistollisiin järjestelmiin ei ole yksiselitteinen. Tiukasti tulkittuna ohjelmisto ei voi vikaantua — algoritmi on joko oikea tai virheellinen, mutta käytön aikana se ei kulu eikä satunnaisesti vikaannu. Ohjelmiston toiminta on joka kerta samanlainen, jos syöte ja muut olosuhteet on mahdollista saada täsmälleen samoiksi [Red02].

Toisaalta laitteistossa voi olla pysyviä tai ohimeneviä satunnaisia vikoja, jotka vaikuttavat ohjelmiston toimintaan. Ohjelmistossa voi myös olla piileviä virheitä, jotka esiintyvät vain tietyissä olosuhteissa, joita ei tunneta, ja näin ne voivat näyttää satunnaisesti esiintyviltä. Jos nimittäin tarkalleen tiedettäisiin, missä olosuhteissa vika esiintyy, virhe luultavasti korjattaisiin. Vikaantumismuotoja ei siis tarkalleen tunneta [HaH02, s. 23]. Goddard esittää [God00] näkemyksenään, ettei ole suurtakaan merkitystä, mikä kunkin ohjelmistovian syy lopulta on, laitteistovika vai piilevästi virheellinen ohjelmisto. Riittää että vika esiintyy sillä tavoin satunnaisen tapaisesti, että sitä voidaan satunnaisilmiönä menestyksellisesti mallintaa.

Vika- ja vaikutusanalyysi on mahdollista tehdä ohjelmistolle. Vaikein vaihe on vikaantumismuotojen määritys. Koska osien vikaantumismuotoja ei tunneta, joudutaan käyttämään jotain yleistä ohjelmistojen vikaantumismuotojen luokitusta,

esimerkiksi ”ohjelma kaatuu”, ”ohjelma pysähtyy”, ”ohjelma ei pysähdy”, ”ohjelman tulosteet ovat vääriä” ja niin edelleen. Tämän jälkeen on mahdollista analysoida, mitä vaikutuksia ohjelmiston eri osien vikaantumismuodoilla olisi koko ohjelmiston toimintaan [HaH02, s. 14, 23–24].

5 Poikkeamatarkastelu (HAZOP)

Tämän luvun aiheena on poikkeamatarkastelu (engl. *Hazard and Operability Study, HAZOP*). Tämä on alunperin kemian teollisuuden käyttöön kehitetty menetelmä [SFS00, s. 36]. Lisäksi kerrotaan poikkeamatarkastelun pohjalta ohjelmistojen analysointiin kehitetystä SHARD-menetelmästä.

5.1 Tavallinen poikkeamatarkastelu

Poikkeamatarkastelussa, kuten vika- ja vaikutusanalyysissä, järjestelmä jaetaan ensin osajärjestelmiin ja edelleen osiinsa. Kunkin osan kohdalla tarkastellaan kaikkien eri prosessimuuttujien poikkeamia oikeasta arvosta. Tällaisia prosessimuuttujia ovat kemian tekniikassa esimerkiksi lämpötila, paine, virtausmäärä ja koostumus. Menetelmässä kartoitetaan prosessimuuttujan poikkeaman haitallisia vaikutuksia osan, osajärjestelmän ja koko järjestelmän toimintaan. Poikkeamatarkastelu on siis luonteeltaan alhaalta ylös -tyyppinen eli induktiivinen [SFS00, s. 36].

Poikkeamia pyritään hahmottamaan seitsemän avainsanan avulla: ”ei” (tai ”ei mitään”), ”enemmän”, ”vähemmän”, ”lisäksi”, ”osittain”, ”päinvastoin”, ”muu kuin”. Esimerkiksi avainsanan ”enemmän” tapauksessa pohditaan, mitä jokin määrällinen lisäys jossakin kohdin järjestelmää voi aiheuttaa järjestelmässä, esimerkiksi oikeaa arvoa suurempi virtaus tai paine. Kunkin poikkeaman osalta ilmoitetaan mahdolliset syyt, jotka voivat johtaa siihen. Sen jälkeen poikkeaman seurauksia järjestelmässä tarkastellaan kunkin syyn osalta erikseen [SFS00, s. 36].

Poikkeamatarkastelu soveltuu järjestelmiin, jotka on mallinnettavissa komponentteina, joissa esiintyy poikkeamia normaalisuureista. Koko järjestelmää koskevat uhat tullevat kattavasti tunnistettua, kun käydään läpi erilaiset poikkeamat kunkin osan kohdalla ja niiden vaikutukset. Järjestelmän tai osajärjestelmän riskien suuruuksien määritykseen osien riskien suuruuksien perusteella menetelmä ei tarjoa juurikaan tukea. Poikkeamatarkastelu lienee muunnettavissa helpoh-

kosti tietoturvan riskianalyysiin soveltuvaksi, koska useimmat tietoturvauhat on mallinnettavissa poikkeamiin tiedon siirtymisessä.

5.2 Ohjelmistojen analysointi

Fenelon *et al.* ovat kehittäneet poikkeamatarkasteluun pohjautuvan ohjelmiston vaarojen analyysi- ja poistomenetelmän (engl. *Software Hazard Analysis and Resolution in Design, SHARD*) [Fen94]. Tässä tieto- ja ohjausvuot ovat korvanneet kemian prosessisuureet. Myös muut ovat kehittäneet poikkeamatarkastelun sovituksia ohjelmistopohjaisille järjestelmille [LaG97, Sch97].

SHARD-menetelmässä on hylätty kaikki poikkeamatarkastelun tavalliset avainsanat. Niiden sijaan on poikkeamatarkastelun ohjeeksi otettu ohjelmistovikojen luokittelu seuraaviin kuuteen luokkaan. Ohjelmisto voi jättää kokonaan tuottamatta halutun palvelun, tai se voi tuottaa palvelun aiheettomasti. Ohjelmiston tuottama palvelu voi olla etuajassa tai myöhässä. Palvelun antama arvo voi olla hieman väärä tai huomattavasti väärä [Fen94]. Tämän luokittelun tulkinta kussakin analyysissä riippuu järjestelmän toiminnasta tai mallinnuksesta. Esimerkiksi pakettikytkentäisen tietoliikennelaitteen tapauksessa palvelun puuttuminen voi tarkoittaa paketin katoamista ja aiheeton palvelu ylimääräistä kopiota paketista.

Järjestelmän mallintamiseen käytettävä menetelmä voidaan valita tilanteen mukaan, mutta Fenelon *et al.* ovat käyttäneet esityksessään [Fen94] MASCOT3-menetelmää (engl. *Modular Approach to Software Construction, Operation and Test*). Samaa käyttävät Fenelonin kanssakirjoittajat McDermid ja Pumfrey samaa SHARD-menetelmää käsittelevässä kirjoituksessaan [McP94].

MASCOT3-mallinnusmenetelmä on alunperin kehitetty suurten tosiaikaohjelmistojen kehityksen työnjaon määrittelyyn. MASCOT3 tarjoaa keskenään samanmerkityksiset kuvalliset ja tekstuaaliset merkinnät ohjelmiston prosesseille ja näiden väliselle tiedonsiirrolle samoin kuin järjestelmän hierarkkiselle jaolle osajärjestelmiinsä [Gri92].

SHARD-analyysi [McP94] alkaa kunkin tarkasteltavaksi otettavan vuon nimeämisellä järjestelmässä. Tämän jälkeen laaditaan MASCOT3-malli järjestelmälle. Seuraavaksi valitaan poikkeamille sopivat avainsanat ja sovelletaan niitä kuhunkin järjestelmän osaan. Poikkeamien mahdolliset syyt selvitetään deduktiivisella päättelyllä. Näiden syiden mahdolliset seuraukset järjestelmässä päätellään induktiivisesti. Tämän jälkeen menetelmä jatkuu muilla riskienhallinnan vaiheilla.

6 Yhteenveto

Tässä kirjoituksessa on aluksi esitelty perinteisen riskianalyysin ominaisuuksia ja ongelmia. Näistä uhkien tunnistamisvaiheen merkittävin ongelma on epäilys tuotetun uhkien luettelon puutteellisuudesta. Riskien suuruuksien määritys menetyksen odotusarvona herättää epäilyn varsinkin riskille arvioidun todennäköisyyden arvon oikeellisuudesta.

Perinteisen riskianalyysin ongelmia pyritään välttämään käyttämällä järjestelmällisiä riskianalyysimenetelmiä. Näistä kolme yleisessä käytössä olevaa menetelmää on esitelty tässä kirjoituksessa, samoin joitakin niille läheisiä menetelmiä. Esitellyt kolme menetelmää ovat kaikki ainakin joiltain osiltaan sangen yksityiskohtaisesti määriteltyjä, ja ne on konkreettisuutensa ja erilaisten lähestymistapojensa takia valittu esimerkeiksi tähän kirjoitukseen.

Kaikki kolme esiteltyä menetelmää pyrkivät tarkan hierarkkisen jaottelun avulla tekemään analyysistä jollain tavalla kattavan. Vikapuu esittää tarkan erittelyn tarkasteltavaksi valitun uhan mahdollisista syistä (ylhäältä alas, deduktiivisesti) ja näistä syistä lasketun todennäköisyyden uhalle, muttei tarjoa apua uhkien tunnistamiseen. Vika- ja vaikutusanalyysi ja poikkeamatarkastelu tutkivat järjestelmän osien toimintaa ja pyrkivät siitä päättelemään (alhaalta ylös, induktiivisesti) koko järjestelmän käyttäytymistä. Nämä tarkastelut voivat johtaa varsin kattavaan uhkien tunnistamiseen, mutta järjestelmän tason riskien todennäköisyyksien laskentaan ei menetelmissä juuri puututa.

Vaikka esitetyt menetelmät on tässä jaoteltu deduktiivisiin ja induktiivisiin, käytännössä niitä sovellettaessa tehdään jatkuvasti tarkistavaa ja täydentävää päättelyä kumpaankin suuntaan. Myös lähestymistavaltaan kokonaan toisenlaisia järjestelmällisiä menetelmiä on olemassa. Ainakin LRAM [Gua87] ja Riskit [Kon97] vaikuttavat käytännössä toimivilta.

Perinteinen riskianalyysi ja sen monet sovellukset perustuvat (ääneen lausuttuun tai lausumattomaan) oletukseen riskien satunnaisuudesta. Tietoturvassa on kyse laitteista, ohjelmista ja ihmisistä. Laitteiston vikaantumiset ovat suurelta osin aidosti satunnaisia. Ohjelmistojen ja ihmisten toiminta taas ei ole satunnaista, kuten Redmill huomauttaa [Red02]. Satunnaisuusoletukseen perustuvien riskianalyysien soveltuvuus tietoturvaan jää siis käytännön kokemusten perusteella arvioitavaksi.

Lähteet

Amo94 Amoroso, E.G., *Fundamentals of Computer Security Technology*. Prentice-Hall International Inc., Englewood Cliffs, NJ, USA, 1994.

Boe91 Boehm, B.W., Software risk management: principles and practices. *IEEE Software* 8, 1 (January 1991), 32–41.

Bri77 Briscoe, G.J., *Risk management guide*. United States Energy Research and Development Administration, ERDA 76-45/11, SSDC-11, UC-41, Washington, DC, USA, 1977.

Car93 Carr, M. et al., *Taxonomy-based risk identification*. The Software Engineering Institute (SEI), Carnegie Mellon University, Technical Report CMU/SEI-93-TR-006, ESC-TR-93-183, Pittsburgh, PA, USA, 1993. `http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.006.html` [24.2.2003].

CFG95 *Potential Failure Mode and Effects Analysis (FMEA) reference manual*. Chrysler Corporation, Ford Motor Company, General Motors Corporation, 1995.

Cou77 Courtney, R.H. jr., Security risk assessment in electronic data processing systems. *AFIPS Conference Proceedings, National Computer Conference*, Dallas, TX, USA, June 13–16, 1977, 97–104.

DoD80 *Procedures for performing a Failure Mode, Effects, and Criticality Analysis*. Military standard MIL-STD-1629A, Department of Defense, Washington, DC, USA, 1980.

EMS79 Ervamaa, J., Mankamo, T., Suokas, J., *Luotettavuustekniikka*. Insinööritieto Oy, Helsinki, 1979.

Fen94 Fenelon, P. et al., Towards integrated safety analysis and design. *ACM SIGAPP Applied Computing Review* 2, 1 (March 1994), 21–32.

FSL02 Fischhoff, B., Slovic, P., Lichtenstein, S., Fault Trees: Sensitivity of Estimated Failure Probabilities to Problem Representation. *Journal of Experimental Psychology: Human Perception and Performance* 4, 2 (1978), 330–344.

GCW92 Gowen, L.D., Collofello, J.S., Wallis, F.W., Preliminary Hazard Analysis for Safety-Critical Software Systems. *Eleventh Annual International Phoenix Conference on Computers and Communications*, Scottsdale, AZ, USA, April 1–3, 1992, 2.5.3.1–2.5.3.8.

God00 Goddard, P.L., Software FMEA Techniques. *Proceedings Annual Reliability and Maintainability Symposium*, Los Angeles, CA, USA, January 24–27, 2000, 118–123.

Gri92 Griffiths, M.P., MASCOT 3. *IEE Tutorial Colloquium on Formal Methods and Notations Applicable to Telecommunications*, March 19, 1992, London, United Kingdom, 5/1–5/4.

Gua87 Guarro, S.B., Principles and Procedures of the LRAM Approach to Information Systems Risk Analysis and Management. *Computers & Security* 6, 6 (December 1987), 493–504.

Haa92 Haarala, R. et al. (toim.), *Suomen kielen perussanakirja, toinen osa L–R*. Kotimaisten kielten tutkimuskeskus, Helsinki, 1992.

HaH02 Haapanen, P., Helminen, A., *Failure mode and effects analysis of software-based automation systems*. Report STUK-YTO-TR 190, Säteilyturvakeskus, Helsinki, 2002. `http://www.stuk.fi/julkaisut/tr/stuk-yto-tr190.html` [25.2.2003].

Kon97 Kontio, J., *The Riskit method for software risk management, version 1.00*. Institute for Advanced Computer Studies and Department of Computer Science, University of Maryland, Technical Report CS-TR-3782, UMIACS-TR-97-38, College Park, MD, USA, 1997. `http://soberit.hut.fi/~jkontio/riskittr.pdf` [24.2.2003].

LaG97 Lawrence, J.D., Gallagher, J.M., A proposal for performing software safety hazard analysis. *Reliability Engineering and System Safety* 55, 3 (March 1997), 267–282.

LeH83 Leveson, N.G., Harvey, P.R., Software Fault Tree Analysis. *The Journal of Systems and Software* 3, 2 (June 1983), 173–181.

McP94 McDermid, J.A., Pumfrey, D.J., A development of hazard analysis to aid software design. *COMPASS '94 Proceedings of the Ninth Annual*

Conference on Computer Assurance, June 27 – July 1, 1994, Gaithersburg, MD, USA, 17–25.

MiK94 Miettinen, J.E., Kajava, J., *Tietoriskien arviointi*. Oulun yliopisto, Tietojenkäsittelyopin laitos, Research Papers Series A-20, Oulu, 1994.

NRC81 *Fault Tree Handbook*. U.S. Nuclear Regulatory Commission, NUREG-0492, Washington, DC, USA, 1981. `http://www.nrc.gov/reading-rm/doc-collections/nuregs/staff/sr0492/sr0492.pdf` [24.2.2003].

Par81 Parker, D.B., *Computer security management*. Reston Publishing Company, Reston, VA, USA, 1981.

Red02 Redmill, F., Exploring subjectivity in hazard analysis. *Engineering Management Journal* 12, 3 (June 2002), 139–144.

Sch97 Schubach, S., A modified computer hazard and operability study procedure. *Journal of Loss Prevention in the Process Industries* 10, 5–6 (1997), 303–307.

SFS00 *Luotettavuusjohtaminen, osa 3: käyttöopas, luku 9: teknisten järjestelmien riskianalyysi*. Standardi SFS-IEC 60300-3-9. Suomen standardisoimisliitto SFS ry, Helsinki, 2000.

SiA96 Sinnamon, R.M., Andrews, J.D., Fault Tree Analysis and binary decision diagrams. *Proceedings Annual Reliability and Maintainability Symposium*, Las Vegas, NV, USA, January 22–25, 1996, 215–222.

Teo01 *Käytännön tietoturvallisuusopas PK-yrityksille*. Teollisuuden ja työnantajain keskusliitto, Helsinki, 2001. `http://www.tt.fi/arkisto/getoriginal.pl?ft_cid=1444` [2.4.2003].

Woo94 Wood, A., Availability Modeling. *IEEE Circuits and Devices Magazine* 10, 3 (May 1994), 22–27.