Tallinna Ülikool

Digitehnoloogiate instituut

Infotehnoloogia juhtimine

MITTEFUNKTSIONAALSETE NÕUETE MÄÄRATLEMINE TURVALISE TARKVARAARENDUSE HANKIMISEKS EESTI AVALIKUS SEKTORIS

Magistritöö

Autor: Kätlin Viik

Juhendaja: Hillar Põldmaa

Autor: ..... „ ..... „ 2017

Juhendaja: ..... „ ..... „ 2017

Instituudi direktor: ..... „ ..... „ 2017

Tallinn 2017

Autorideklaratsioon

Deklareerin, et käesolev magistritöö on minu töö tulemus ja seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

....

(kuupäev) (autor)

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina Kätlin Viik (sünnikuupäev: 19. september 1978)

1. Annan Tallinna Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Mittefunktsionaalsete nõuete määratlemine turvalise tarkvaraarenduse hankimiseks Eesti avalikus sektoris“, mille juhendaja on Hillar Põldmaa, säilitamiseks ja üldsusele kättesaadavaks tegemiseks Tallinna Ülikooli Akadeemilise Raamatukogu repositooriumis.

2. Olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tallinnas, _____

allkiri ja kuupäev

Sisukord

Lühendid ja mõisted

Aedikkäitus (inglise k *sandboxed*) – võimalikke ohtlikke toiminguid ja funktsioone tõkestav keskkond ebausaldatavaist allikaist pärit koodi või programmi käituseks.

Aktiivsusjälg – loetelu aktiivsetest alamprotsessidest uuritavas programmi täitmise punktis, võidakse kuvada veateate osana lõppkasutajale.

Analüütilise hierarhia protsess – süsteemi erinevate osade nõuete paarikaupa võrdlus tagamaks kõigi nõuete põhjendatust (praktikas vähe kasutatav, sest on väga aeganõudev).

API (inglise k *Application Programm(ming) Interface*) – reeglid ja vahendid, mida rakendusprogramm kasutab suhtluseks operatsioonisüsteemiga, andmebaasihalduse süsteemiga või muu juhtprogrammiga, samuti sideprotokolliga.

Borda meetod – iga realiseeritavate nõuete üle otsustaja hindab nõudeid ja koostab oma pingerea, kõiki pingeridasid võrreldes tekib üldine pingerida ja selle alusel prioriseeritakse nõuded.

Brauserikook (inglise k *session cookie*) – väike tekstistring, mille kirjutab veebiserver kliendi brauserisse ja mille tagastab brauser igal järgmisel pöördumisel; kasutatakse autentimiseks, järgmiste pöördumiste hõlbustamiseks jms, kuid teda saab kasutada ka kasutaja eelistusi jälgiva ja privaatsust rikkuva nuhkvarana.

CRM (inglise k *Customer Relationship Management*) – kliendihaldus; firmade tavad, strateegiad ja tehnoloogiad klientide interaktsioonide ja andmete halduseks ja analüüsimiseks kliendi kogu elutsükli kestel, kliendisuhete arendamiseks, klientide säilitamiseks ja läbimüügi tõstmiseks.

CSS (inglise k *Cascading Style Sheets*) – kaskaadlaadistik, märgistuskeelse dokumendi välisilme kirjeldamise formaalkeel

DigiDoc teek – DigiDoc teekide abil saab luua DigiDoc-ühilduvaid rakendusi.

DigiDoc veebiteenus (inglise k *DigiDocService*) – SOAP-põhine veebiteenus võimaldamaks võimalikult lihtsalt digitaalallkirjastamise ja allkirjade verifitseerimise ja Mobiil-ID funktsionaalsust siduda teiste infosüsteemidega.

ERP (inglise k *Enterprise Resource Planning*) – ettevõtte ressursside planeerimine; üks organisatsiooni oluliste tegevusprotsesside (plaanimise, tootmise, turunduse, turustuse, laonduse, rahanduse jm) integreerimise strateegiaid.

Failitee (inglise k *file path*) – ressursi inimloetav aadress failisüsteemis

HIPAA (inglise k *Health Insurance Portability and Accountability Act*) – meditsiinivaldkonna standard organisatsioonidele, mis soovivad tegeleda rakenduste loomisega Ameerika tervishoiuteenuste turule.

HTML (inglise k *Hypertext Markup Language*) – hüperteksti märgistuskeel, T. Berners-Lee 1980. a. loodud süsteem tekstile kirjatüübi ja -laadi, värvuse, graafika, hüperlinkide jms lisamiseks märgistuskoodide ("siltide") abil, veebilehtede loomise vahend.

Inkrementaalne arendus – tarkvara spetsifikatsioon, kavand ja realisatsioon on tükeldatud osadeks, mida arendatakse järjest ja kordamööda.

Iteratsioon – sammude jada korduva sooritamise protsess.

Kasutajalugude meetod – meetod, mida kasutatakse nõuete kirjeldamisel andmaks tervikpilti nõude seotusest konteksti ja teiste nõuetega. Iga nõue peab olema seotud vähemalt ühe kasutajalooga.

Kogumine (inglise k *collection*) - potentsiaalset digitõendit sisaldavate füüsiliste objektide kogumine.

Kolme klõpsu printsiip – rakenduse kasutajaliidese tehtavad toimingud tohivad üksteisest olla maksimaalselt kolme hiireklõpsu kaugusel.

Konteinerdus (inglise k *containerization*) – hajusrakenduste virtualiseerimise meetod; rakendus kapseldatakse oma töökeskkonnaga (failid, teegid jms) konteinerisse, täielikku virtuaalmasinat loomata ja operatsioonisüsteemi ühiskasutusega.

Kvantifitseerimine - kvalitatiivsete tunnuste väljendamine kvantitatiivsete näitajate kaudu, nt teadmiste või spordivõimete hindamine pallides.

Lingirivi (inglise k *breadcrumb trail*) – linkide hierarhiline jada veebilehel, kasutatakse tekstilise navigeerimisvahendina.

Läbistustestimine (inglise k *penetration testing*) – sissetungirünnete imiteerimine turvameetmete toimivuse kontrollimiseks.

Massiiv (inglise k *array*) - andmeelementide n-mõõtmeline korrastatud hulk, mille iga element on identifitseeritav üheainsa nime ja ühe või mitme indeksiga ning individuaalselt adresseeritav.

MFN – mittefunktsionaalne nõue

Ohusubjekt – organisatsiooni turvalisust ohustava või ohustada võiva sihiliku või tahtmatu intsidendi eest osaliselt või täielikult vastutav olem (isik või organisatsioon, sisemine või väline).

Ohutus (ohutusnõue) – süsteemi, toote, protsessi vms omadus mitte kahjustada oma sisemisi ega väliseid olemeid.

PDF (inglise k *Portable Document Format*) – porditav dokumendivorming; Adobe Systemsi poolt loodud omanduslik failivorming platvormist sõltumatuks ja trükindusele sobivaks dokumentide esituseks; PDF-failis on dokumendi sisu ja vormistuse kõik originaaliga identseks taasesituseks vajalikud elemendid ja parameetrid.

Primaarvõti (inglise k *primary key*) – andmebaasi kirjete ühese identifitseerimise vahend: kirjeväli (tabeliveerg) või väljade (veergude) kombinatsioon, kus on iga kirje puhul eri väärtus (mis ei tohi olla tühiväärtus); igal andmebaasitabelil on ainult üks primaarvõti.

Päringuvõltsing (inglise k *Criss-Site Request Forgery*) – rünne, mis petab laadima veebilehte, mille kaudu tehakse veebirakenduse kasutaja nimel ja õigustega mingi kahjulik toiming, näiteks ostetakse midagi

Pühademuna (inglise k *easter egg*) – dokumenteerimata funktsioon (näiteks mäng) rakendusprogrammis, käivitub mingi erilise käsu või klahvikombinatsiooniga; tavaliselt kahjutu ja mõeldud meeldiva üllatusena.

Püsikodeerimine (inglise k *hardcode*) – püsikodeeritud, püsiprogrammeeritud: riistvaras või tarkvaras muutmisvõimalusteta realiseeritud.

RESTful (inglise k *Representational State Transfer*) – RESTpõhine: nüüdisaegse veebi nõuetele sobivat arhitektuuristiili ja ühtset liidest taotlev hajustöötluse, eriti veebiteenuste programmeerimise paradigma, mis suurendab jõudlust, mastabeeritavust, muudetavust, nähtavust, porditavust, töökindlust, lihtsustab liidest.

Räsi – andmetest räsifunktsiooni abil sooritatud ühesuunalise krüpteerimisega (st erilise pakkimisega) saadud püsipikkusega väärtus.

Räsimine – tervikluse kontrolli võimaldava räsi loomine räsifunktsiooniga.

Salaami rünne – rida väikeseid ründeid korreleerub lõpptulemusena suureks ründeks rakenduse vastu.

Salgamatus (inglise k *non-repudiation*) – süsteemi, teenuse või isiku võime tõendada väidetava sündmuse või toimingu aset leidmist ja seda tekitanud olemite osalust, sh suutlikkus väärata sõnumi saatmise või saamise salgamist või näiteks mingi lausungi, dokumendi või lepingu kehtivuse eitamist.

SAP (inglise k *Systems, Applications and Products*) – Saksamaal loodud ettevõtte ressursside planeerimiseks mõeldud tarkvaralahendus, mis võimaldab oluliste ärikomponentide (nt varustamine, laoseis, turustamine, kliendihaldus, tellimuste täitmise jälgimine) integreeritud haldamist.

Semantiline – tähenduslik

Skript – väike kompileerimata käsujada, mida kasutaja sekkumiseta interpreteerib või täidab teine programm.

SLA (inglise k *Service Level Agreement*) – teenuseandja ja kliendi (tarbiva organisatsiooni või isiku) vaheline kirjalik kokkulepe, mis dokumenteerib vajalikud teenusetasemed ja nende mõõtmise viisi.

SOAP (inglise k *Simple Object Access Protocol*) – üks veebiteenuste alusstandardeid; algselt protseduuride kaugkutseks määratud protokoll platvormist sõltumatuks XML-vormingus sõnumite vahetuseks hajuskeskkonnas, eeskätt Interneti kaudu, mingit rakenduskihiprotokolli kasutades.

Sool (inglise k *salt*) – räsimisprotsessis lisatav mittesalajane (tavaliselt juhuarvuline) väärtus, näiteks parooli krüpteerimisel paroolile lisatav juhuarvuline täidis, mis raskendab sõnastikrünnet.

Sotsiaalne manipuleerimine (inglise k *Social Engineering*) – mittetehniline ründevahend, hõlmab veenvat teesklust, valesid, altkäemaksu, ähvardusi jms, peamiselt konfidentsiaalse või tundliku teabe saamiseks.

Šiffer – algoritm või algoritmipaar andmete krüpteerimiseks ja dekrüpteerimiseks.

Teek – üldiseks kasutamiseks määratud infoobjektikogu (tarkvarateek, mooduliteek, makroteek, arendusteek jms).

TLS (inglise keeles *Transport Layer Security*) – transpordikihi turve; võimaldab enne andmevahetust kliendi ja serveri vastastikku autentimist ning leppida kokku krüpteerimisalgoritmi ja võtmed.

Turvalisuse nõue – riskihalduslik tegevus teabe turvalisuse säilitamiseks vastavalt organisatsiooni tegevuse eesmärkidele, sealhulgas andmekaitse realiseerimise vahend.

Tõstutundlikkus – kirjutatud sõnade tähenduse muutus sõltuvalt suur- ja väiketähtede kasutamisest.

Urkimine – lubamatu manipuleerimine, sekkumine, muutmine, avamine (eriti riistvara sisemuse, kiipkaartide ning pakendite ja dokumentide puhul) vms rünne.

URL (inglise k *Uniform Resource Locator*) – ühtne ressursilokaator; mehhanism ressursside (nt veebilehtede) identifitseerimiseks Internetis.

UTF-8 (inglise k *Unicode Transformation Format, 8-bit*) – Unicode'i 8-bitine teisendusvorming, koodisüsteem; standardne süsteem Unicode'i märkide kodeerimiseks muutuva pikkusega koodidega.

Viie miks meetod (inglise k *5 Whys*) – iteratiivne küsitlemistehnika, mida kasutatakse põhjus-tagajärg seose määratlemiseks mingi konkreetse probleemi lõikes.

Viitpomm – loogikapomm, mis käivitub ettemääratud ajal.

VPN (inglise k *Virtual Private Network*) – virtuaalne privaatvõrk; ettevõtted kasutavad VPN-tehnoloogiat partnervõrkude ja ulatuslike sisevõrgu osade loomiseks; privaatsus ja turvalisus tagatakse side krüpteerimise ja autentimisega.

Võõrvõti (inglise k *foreign key*) – relatsioonandmebaasis ühe tabeli väli (veerg) või väljarühm, mis üheselt määrab mingi rea teises tabelis, luues seeläbi sideme nende kahe tabeli vahel.

WCAG (inglise k *The Web Content Accessibility Guidelines*) – veebisisu hõlbustuse juhised, puuetega kasutajate või piiratud võimalustega seadmete (näiteks mobiiltelefoni) hõlbustusvajadusi arvestavad veebiarenduse juhised.

Wiegersi prioriseerimise maatriks – iga nõuet hinnatakse nelja parameetriga: kliendi kasu nõude realiseerimisest, sanktsioon nõude realiseerimata jätmisel, nõude realiseerimise eeldatav maksumus ja nõudega seotud riski hindamine ning vastavalt nendele parameetritele toimub nõuete prioriseerimine.

Äpp – mobiilseadmele mõeldud ja ta spetsiifikat arvestav väike rakendusprogramm või veebipõhise teenuse link.

Ühe klõpsu printsiip – rakenduse kasutajaliideses tehtav toiming peab olema teostatav ühe hiireklõpsuga.

XSS (inglise k *cross-site-scripting*) – murdskriptimine, skriptisüst; veebisaitide nõrkusi (näiteks sisestusvälja turvakontrolli puudumist) ärakasutav koodisüsti meetod, ründab otse või teiste veebisaitide kaudu, viies kasutaja brauserisse HTML-, JavaSript-vm kahjurkoodi või kahjulikke veebilinke

Sissejuhatus

Kiirelt arenev digitaalne maailm meie ümber on huvitav, palju uusi teadmisi pakkuv ja maailmapilti avardav võimalus. Kunagi enne inimkonna ajaloos ei ole inimesed saanud nii lihtsalt ja kiiresti suhelda ja infot vahetada. Füüsilises maailmas toimetades mõtleme me pidevalt oma turvalisuse peale: lahkudes lukustame oma kodu ukse, et soovimatud inimesed eemal hoida; liigeldes järgime ühiskonnas kokkulepitud reegleid, et kõik sujuks tõrgeteta. Virtuaalses maailmas me tihti turvalisusele ei mõtle või ei tundu see lihtsalt vajalik, sest mina ise ju kasutan oma nutiseadet, arvutit jm tehnilisi vahendeid, kuidas siis saaks keegi soovimatu isik neile ligi ja vaadata minu privaatset infot. Meie kõigi käes on järjest rohkem andmeid, mida me talletame erinevates tehnilistes vahendites ja mille levimist me ilma meie loata ei soovi, seda nii kodus kui töökohal.

Kasvanud on erinevat tüüpi tarkvara loomine: ärilistel eesmärkidel loodav tarkvara, mida organisatsioonid ise või partnerite abil arendavad, erinevad nutiseadmele mõeldud arendused ehk äpid. Ärilistel eesmärkidel loodava tarkvara arendamise käigus saab partneritele teatavaks terve hulk kriitilise tähtsusega informatsiooni nii organisatsiooni äriprotsesside kui ka tehniliste lahenduste kohta. Lisaks lepingulistele kokkulepetele tuleb rakendada ka erinevaid tehnoloogilisi võimalusi, et kaitsta organisatsiooni eksisteerimiseks vajalikku informatsiooni.

Mida aeg edasi, seda aktuaalsemaks on muutunud turvaline tarkvaraarendus – iga arendusega saab arendajale teatavaks unikaalset ja olulist informatsiooni ning kasutajatelt oodatakse delikaatsete andmete jagamist tarkvara vahendusel. Üheks heaks ja järjest enam kasutust leidvaks võimaluseks on erinevate turvalisusega seotud nõuete määratlemine tarkvaraarenduse hankimisel.

Magistritöö keskne mõiste on turvalisus – turvalisus on süsteemi võime kaitsta oma objektide (ressursside ja informatsiooni) terviklust ja konfidentsiaalsust (Alliksoo, Hanson, Laur, & Oit, 2009).

Magistritöö eesmärgiks on avaliku sektori organisatsioonile mittefunktsionaalsete nõuete (edaspidi MFN) koostamine, mis sisaldab muuhulgas turvalisusega seotud aspekte ja on heaks alusmaterjaliks organisatsiooni ja/või infosüsteemi spetsiifikast lähtuva MFNi kirjeldamisel.

Lähtuvalt töö eesmärgist on autor püstitanud järgmised uurimisküsimused:

1) Kuidas defineeritakse teaduslikus kirjanduses MFNi?
2) Milliseid soovitusi annab teaduslik kirjanduse MFNide kirjeldamiseks?
3) Kuidas, tuginedes teaduslikule kirjandusele, on otstarbekas integreerida turvalisus tarkavaraarendusega?
4) Milliseid juhiseid annavad Eesti avalikus sektoris laialdast kasutust leidvad standardid – infosüsteemide kolmeastmeline etalonturbe süsteem (edaspidi ISKE) ja veebirakenduste turbe verifitseerimise standard (edaspidi ASVS) – turvalisusele tarkavaarenduses?

Töö eesmärgi saavutamiseks tehtavad tegevused:

1) vajaliku teoreetilise baasi jaoks töötab magistritöö koostaja läbi erialase kirjanduse ja teadusartiklid;
2) uurib põhjalikumalt kahe Eestis kasutusel oleva standardi soovitusi turvalisuse integreerimiseks tarkvaraarendusse;
3) defineerib avaliku sektori organisatsioonile sobiva MFNi kogumiku, kus on muuhulgas arvestatud erinevate turvalisusega seonduvate aspektidega.

Töö koosneb neljast peatükist. Esimeses peatükis uurib töö koostaja tuginedes erialakirjandusele ja teadusartiklitele, kuidas MFNi defineeritakse ja kuidas tuleks MFNi kirjeldada. Teises peatükis uurib töö koostaja erialakirjanduse ja teadusartiklite abil, milliseid soovitusi annavad erinevad raamistikud ja metoodikad turvalisuse integreerimiseks tarkvaraarendusse. Kolmas peatükk keskendub standardites ISKE ja ASVS toodud turvalise tarkvaraarenduse temaatika uurimisele.

Neljandas peatükis viib magistritöö koostaja läbi arendusuuringu, analüüsides erinevate avaliku sektori asutuste MFNi kogumeid ja erinevaid MFNe määratlevaid avaliku sektori dokumente. Magistritöö koostaja kaasab ka standardites ISKE ja ASVS toodud nõuded ja soovitused. Arendusuuring on sobiv, kuna toetab olemasoleva

parendamist ja uuele tasemele viimist. Töö lõpptulemiks on MFN loetelu, mis arvestab erinevate turvalisuse aspektidega ja on kasutatav alusmaterjalina avaliku sektori organisatsioonis tarkvaraarenduse hankimisel.

1. Mittefunktsionaalsed nõuded

Tarkvaras sisaldub hulk andmeid. Andmed on informatsiooni esitus formaliseeritud kujul, mis sobib edastuseks, tõlgenduseks või töötluseks (Buldas, Hanson, Krasnosjolov, Laur, & Veldre, 2011-2017). Andmed muutuvad informatsiooniks läbi konteksti ja inimesepoolse tõlgenduse. Informatsioonil on rida omadusi, mis oma olemuselt on kvaliteedi näitajad:

- asjakohasus (inglise k *relevance*);
- täpsus (inglise k *accuracy*);
- õigeaegsus (ka ajakohasus, inglise k *timeliness*);
- täielikkus (inglise k *completeness*);
- sidusus (ka koherentsus, inglise k *coherence*);
- vorming (inglise k *format*);
- kättesaadavus (ka hõlpsus, inglise k *accessibility*);
- ühilduvus (inglise k *compatibility*);
- turvalisus (inglise k *security*);
- õigsus (ka kehtivus, inglise k *validity*) (Miller, 1996).

Eelpool loetletud omadused on koondatavad kolme gruppi:

1) **Käideldavus** (inglise k *availability*), mis koosneb kättesaadavusest, õigeaegsusest (ajakohasusest) vormingust, ühilduvusest, asjakohasusest;
2) **Terviklus** (inglise k *integrity*), mis koosneb õigsusest, täielikkusest, täpsusest, sidususest (koherentsusest);
   Sisaldab ka infovara lubamatute muudatuste puudumist.
3) **Konfidentsiaalsus** (inglise k *confidentiality*), mis sisaldab turvalisust (Põldmaa, 2016).

Käideldavus, terviklus ja konfidentsiaalsus on infoturbe kolm komponenti. **Käideldavus** tähendab andmete takistusteta kättesaadavust volitatud kasutajatele (isikud, alamsüsteemid). Turvameetmete rakendamisel tuleb silmas pidada, et need ei muutuks ise kasutajaid takistavaks. Ülemäära rangete turvaeeskirjade puhul hakkavad kasutajad neid ignoreerima või otsima viise neist möödahiilimiseks, muutudes ise

turvariskiks. **Terviklus** tähendab, et andmeid tohivad modifitseerida ainult volitatud isikud (kasutajad). Modifitseerimine sisaldab andmete kirjutamist, muutmist, oleku muutmist, kustutamist, loomist. **Konfidentsiaalsus** tähendab, et süsteemi andmetele on ligipääs ainult volitatud kasutajatel ja protsessidel (Alliksoo, Hanson, Laur, & Oit, 2009).

Käideldavus on andmetöötluse juures kõige olulisem aspekt – kui käideldavus on rikutud, siis on raske rääkida terviklusest ja konfidentsiaalsusest. Tervikluse puhul tuleb infoturbe kontekstis jälgida, et lisaks andmetele ei muudetaks volitamata kasutaja poolt ka andmete looja, loomisaja jms andmed. Konfidentsiaalsus on ajalooliselt infoturbe kõige esimene aspekt (salakirjad, paroolid sõjaajal, tänapäeval krüptograafia) (Põldmaa, 2016).

Tagamaks andmete käideldavust, terviklust ja konfidentsiaalsust tuleb tarkvara arendamisel appi võtta mittefunktsionaalsed nõuded ja seda soovitavalt tarkvaraarenduse võimalikult varajases etapis.

1.1 Mittefunktsionaalse nõude mõiste

Juba kümme aastat tagasi juhtis Martin Glinz oma konverentsi ettekandes tähelepanu faktile, et kuigi termin „mittefunktsionaalsed nõuded“ on olnud kasutusel juba rohkem kui kakskümmend aastat, ei ole suudetud nõuete püstitamisega tegelevas kogukonnas siiani kokku leppida, mis MFNid on ja kuidas peaks nende kohta infot koguma, neid dokumenteerima ja valideerima. Samas on saavutatud ühehäälne konsensus, et MFNid on olulised ja võivad osutuda kriitiliseks eduka tarkvaraarenduse projekti läbiviimisel (Glinz, 2007).

Igas organisatsioonis on lisaks suurele hulgale andmetele rida protsesse, millele tuginedes organisatsioon igapäevaselt toimetab. Tänapäeva kiire tempo juures on infotehnoloogia oluliseks abiliseks erinevate protsesside kiiremal, kvaliteetsemal ja mugavamal läbiviimisel – see on põhjus miks luuakse tarkvara. Tarkvaraga seotud erinevatel osapooltel on rida erinevaid nõudmisi ja ootus nende nõudmiste realiseerumiseks tarkvara kaasabil.

Kõige üldisemal tasemel võib öelda, et tarkvara nõue on omadus, mis peab esitama midagi selleks, et lahendada mingi probleem reaalses maailmas. Nõude eesmärgiks võib olla mingi tegevuse või selle osa automatiseerimine, et see toetaks organisatsiooni äriprotsessi, olemasoleva tarkvara puuduste parandamine või kontrollifunktsioon seadme jaoks. Need on vaid mõned näited probleemidest, mida tarkvara võib aidata lahendada. *Software Engineering Body of Knowledge* (edaspidi SWEBOK) tarkvaratehnika juhendmaterjal liigitab nõuded toote ja protsessi nõueteks (Bourque & Fairley, 2014).

Toote (tarkvara, tarkvarateenus) nõuded spetsifitseerivad, milliseid funktsioone peab süsteem realiseerima (funktsionaalsed nõuded) ja kuidas neid funktsioone täidetakse (mittefunktsionaalsed nõuded). Protsessi nõuded määravad arenduse kitsendused (nt nõuded arhitektuurile, vahenditele või keskkonnale). Ärinõuded võivad lisaks sisaldada strateegilisi, keskkonna, maksumuse ja muid piiranguid. Eri tüüpi nõuded võivad olla omavahel sõltuvuses. Toote nõuded tulenevad enamasti ärinõuetest, protsessi nõuded nii äri- kui ka toote nõuetest (Tepandi, 2016).

Funktsionaalsed nõuded vastavad küsimusele "Mida peab tarkvara tegema?" (nt süsteem peab võimaldama kauba tellimist). MFN vastab küsimusele "Kuidas tarkvara peab vajalikke funktsioone täitma?" (nt süsteemi vastuse aeg peab jääma etteantud piiridesse (tõhusus), süsteem peab teatud ajavahemike jooksul tõrgeteta töötama (töökindlus) jne (Tepandi, 2016).

SWEBOK defineerib funktsionaalsed nõuded järgmiselt: „Funktsionaalsed nõuded kirjeldavad funktsioone, mida tarkvara peab tegema, nt vormindama teksti, moduleerima signaali. Neid nimetatakse ka võimeteks või omadusteks (Bourque & Fairley, 2014).“

MFN on SWEBOKi järgi nõuded, mis käituvad loodavat lahendust kitsendavate teguritena. MFNi defineeritakse vahel ka kitsenduste või kvaliteedi nõuetena. MFNe võib liigitada nende rakendamise eesmärgist lähtuvalt: jõudluse nõuded, ohutusnõuded, töökindluse nõuded, turvalisuse nõuded, koostalitlusvõime nõuded jm tarkvarale esitatavad nõuded (Bourque & Fairley, 2014).

Sarnaselt Martin Glinzile jõuti ka Texase Ülikooli uuringus järelduseni, et mingit ametlikku ühest definitsiooni MFNi kohta ei olegi võimalik välja tuua nagu ei ole võimalik ka defineerida lõplikku MFN nimekirja. Uuring keskendus erinevate MFNi kohta pakutud definitsioonide analüüsimisele, jõudes sarnaselt SWEBOKile järeldusele, et funktsionaalsed nõuded vastavad küsimusele „Mida tarkvara peab tegema?“. MFNi kasutatakse samal ajal kirjeldamaks kui hästi tarkvara neid tegevusi tegema peab. Olulisel kohal on tarkvara kvaliteet, st MFNi peamine eesmärk on tagada soovitud kvaliteediga tarkvara. MFN õigustab tarkvara disaini otsuseid ja seab piirangud nõutud funktsionaalsuse realiseerimisele. Uuringu läbiviijad pakuvad omapoolse erinevaid definitsioone kokkuvõtva sõnastuse MFNi kohta: MFN on kas süsteemi oluline omadus või süsteemile rakenduv piirang (Chung & do Prado Leite, 2009).

Sarnaselt eelnevalt väljatoodud definitsioonidele sõnastab ka Rootsis Linköpingsi Ülikoolis läbiviidud uuring funktsionaalse nõude kui nõude, mis defineerib mida süsteem peab tegema, sisaldades komponendi ja teda ümbritseva keskkonna vastastikuse mõju hinnangut. Funktsionaalne nõue peab olema testitav, st on võimalik testimisel demonstreerida, et püstitatud nõue on täidetud. MFN kirjeldab kuidas tarkvara funktsioone täidab – MFN annab süsteemile piirangud, milles süsteem peab saavutama soovitud funktsiooni täitmise (Gustavsson & Willander, 2005).

Oma käsitluses toovad Chung ja do Prado Leite välja kaks olulist aspekti MFNi kohta, mida tuleks alati silmas pidada, kui MFNi sõnastama asutakse. Esiteks, MFNi defineerima asumisel tuleks esialgu püüda sõnastada nõuded üldiselt (mittekonkreetse sõnastusega nõuded) ja järk-järgult täiendada neid nõudeid konkreetsemaks ja detailsemaks. Antud lähenemine peaks tagama MFNi kasutuselevõtul soovitud kvaliteedi saavutamise, sest kõikide nõuete kohta on teada miks nad nõuete loendis kajastuvad. Teiseks, MFNi puhul on kriitilise tähtsusega nõude sisu defineerimine tagamaks, et nii äripool kui arendaja saaksid ühtemoodi aru nõude sisust. Ainult siis on võimalik tagada, et loodav tarkvara töötab vastavalt soovitule (Chung & do Prado Leite, 2009).

Vaatamata sellele, et mitmekümne aasta jooksul ei ole jõutud MFNiga tegelevates ringkondades kokkuleppele, kuidas täpselt MFNi defineerida, võib siiski eelnevalt väljatoodule tuginedes väita, et MFNi defineeritakse sarnaselt.

1.2 Mittefunktsionaalsete nõuete kirjeldamine

Nii nagu üldiselt tarkvaraarenduse nõuete määratlemisel, tuleks ka MFNi määratlemisel alustada nõuete (vajaduste) sondeerimisest (Green & Stellmann, 2005); (Bijan, Yu, Stracener, & Woods, 2012), mis sisaldab erineval viisil teabe kogumist ja analüüsi esialgsete nõuete püstitamiseks.

Peale nõuete (vajaduste) sondeerimist jõutakse tavaliselt tarkvaraarenduse käigus nõuete kirjeldamiseni. Esmalt kirjeldatakse funktsionaalsed nõuded (vajadused), mis on reeglina äripoolele lihtsamini mõistetavad ja nende defineerimine ei ole nii keeruline. MFNi defineerimisel peab kaasatud osapool suutma juba mõelda kuidas peaks tarkvara toimima ja/või millised on tarkvara toimimise piirangud (st mida tarkvara ei tohiks teha) (Green & Stellmann, 2005).

Nõuete määratlemise tegevused sisaldavad intervjuusid erinevate osapooltega, kasutajate jälgimist töö tegemise ajal ning intervjuude ja vaatluste käigus kogutud andmete analüüsimist ja verifitseerimist konkreetsete kasutajate peal (Green & Stellmann, 2005).

MFNi tuleks määratleda nii täpselt kui võimalik, tihti kasutatakse selleks kvantifitseerimist. Kui vähegi võimalik peaks MFN sisaldama konkreetseid mõõdikuid, millele tarkvara peab vastama (nt maksimaalne lubatud sekundite arv mingi tegevuse sooritamiseks; tundide arv, mille jooksul süsteem peab kättesaadav olema jne) (Green & Stellmann, 2005).

Enamasti on väljatoodud rida karakteristikuid, mida hästi defineeritud nõue peab sisaldama. Samas puudub sellise nõude loomise protsess ja ka mõõdik, mis ütleks, et nõue on nüüd vastavalt protsessile ja karakteristikutele loodud (valmis) (Bijan, Yu, Stracener, & Woods, 2012).

Erinevad autorid on uurinud ja pakkunud välja nimekirja karakteristikutest, mida hea nõue peaks sisaldama. Hästi defineeritud nõue on:

- täielik,
- ühetähenduslik,

- arusaadav,
- eristatav,
- unikaalne,
- jälitatav,
- ei sisalda vastuolusid,
- teostatav,
- kontrollitav,
- jagamatu,
- õigesti tuletatud (Bijan, Yu, Stracener, & Woods, 2012).

Kui eeltöö nõuete kirjeldamiseks on tehtud, st on kogutud piisav informatsioon, tuleb asuda nõudeid defineerima.

Nõuete (vajaduste) sondeerimiseks ja määratlemiseks on erinevaid viise (nt viie miksi meetod (inglise k *5 Whys*), kasutajalugude meetod (inglise k *Use Case Modeling*)). On ka erinevaid viise kuidas prioriseerida nõuded (nt analüütilise hierarhia protsess (inglise k *Analytical Hierarchy Process*), Wiegersi prioriseerimise maatriks (inglise k *Wiegers' Prioritization Matrix*), Borda meetod (inglise k *Borda's Method*)). Vajaduste nõueteks defineerimiseks vajalike meetodite leidmine on keeruline (Bijan, Yu, Stracener, & Woods, 2012).

Green ja Stellmann soovitavad oma raamatus „*Applied Software Project Management*" kasutada iga üksiku nõude nii funktsionaalse kui MFNi defineerimiseks järgmist tabelit:

Tabel 1. Nõuete kirjeldamise tabel (Green & Stellmann, 2005)

Nimetus	Nõude nimetus ja number
Kirjeldus	Nõude lühikirjeldus
Põhjendus	Põhjendus, miks antud nõue on vajalik
Nõude sisu	Tarkvara toimimise kirjeldus, st mida tarkvara peab tegema
Seosed	Kasutajalood ja teised funktsionaalsed- ja mittefunktsionaalsed nõuded, mis on antud nõudega seotud

Kogutud informatsiooni alusel nõuete defineerimiseks (nõuete transformeerimiseks) võib lisaks eelpool toodule, aluseks võtta erinevaid meetodeid. Järgnevalt on tutvustatud mõnesid neist:

- Arhitektuuri kompromissi analüüs – (inglise k *Architecture Tradeoff Analysis*) – meetodit kasutatakse nõuete selgitamiseks ja kinnitamiseks kaasates loodava lahenduse võimalik arhitektuur;
- Kiire prototüüpimine – (inglise k *Rapid Prototyping*) – arendajad teevad lõppkasutajatega pidevaid intervjuusid ja saadud info põhjal loovad süsteemi(osade) prototüüpe ning selle põhjal toimub nõude valideerimine, ranget nõuete defineerimist ei toimu ja ei teki ka nõuete loetelu;
- Plankeel[1] - (inglise k *Planguage*) – Tom Gilbi poolt loodud mõiste iseloomustamaks planeerimise (inglise k *planning*) ja keele (inglise k *language*) kooskasutamist. Gilbi järgi peavad nõuded sisaldama konteksti (nt omanik, versioon, olemus jms). Plankeel meetod aitab kaasata nõudega seotud olulise info, selgitab nõuet ja õigustab nõude sisaldumist spetsifikatsioonis;
- Lean meetod – (inglise k *Lean System Engineering*) – ei ole protsess nõuete kirjeldamiseks, vaid on pigem mõtteviis, kuidas nõuded kirja saada. Lean meetod kasutab nõudeid väärtuse defineerimiseks ja keskendub pigem kvaliteedi tagamisele nõuete kirjeldamisel, et minimeerida mitteproduktiivsete lahenduste loomist;
- Six Sigma disaini meetod – (inglise k *Design for Six Sigma*) – äriprotsesside juhtimise meetod, mis nõuete defineerimise kontekstis peaks aitama tagada lõppkasutajate soovidest lähtuva kasutatava disain loomise. Fookus on ühekordsel lahenduse disainimisel – ideaalne disain saavutatakse esimesel korral (Bijan, Yu, Stracener, & Woods, 2012).

Bijan, Yu, Stracener ja Woods (2012) jõuavad oma uuringus järeldusele, et kuigi on palju erinevaid meetodeid, metodoloogiaid jm abistavad materjale, ei ole siiski võimalik välja tuua ühtegi konkreetset, mis oleks hästi kasutatav just süsteemi nõuete

[1] Plankeel on magistritöö koostaja poolt inglisekeelse termini „*planguage*“ põhjal tuletatud eestikeelne vaste.

defineerimiseks. Välja saab tuua mõned ühised jooned erinevate meetodite ja metodoloogiate vahel:

- tuleb olla pidevas dialoogis lõppkasutajaga,
- tuleb püüda nõuded kirja panna selgelt ja üheselt mõistetavalt (umbmäärasus on nõuete puhul suur probleem),
- nõuded peavad suutma kaasas käia muutuvate vajadustega, st nõudeid tuleb hoida ajakohasena,
- nõuete omavahelised seosed peavad olema selged (üks nõue võib sõltuda teisest),
- nõuete kogumi hindamiseks puudub vahend (Bijan, Yu, Stracener, & Woods, 2012).

Bijan, Yu, Stracener ja Woods toovad oma uuringu lõppjäreldusena välja, et nõuete sondeerimiseks ja dokumenteerimiseks on hulgaliselt soovitusi, aga puudu on juhistest, kuidas lõppkasutaja soovid formuleerida konkreetseteks, üheselt mõistetavateks ja täielikeks nõueteks (Bijan, Yu, Stracener, & Woods, 2012).

Kui on teada MFNi mõiste sisu ja erinevad viisid selle kirjeldamiseks, tuleb leida moodus, kuidas lähtuvalt organisatsiooni tarkvaraarendamise protsessist integreerida turvalisus protsessi erinevatesse etappidesse.

2. Turvalise tarkvaraarenduse raamistike ja metodoloogiate lühitutvustus

On olemas terve rida erinevaid protsessikirjeldusi, standardeid, raamistikke ja metodoloogiad, mis kõik on mõeldud kasutamiseks abivahendina turvalise tarkvara arendamisel. Kõik need erinevate põhimõtete ja parimate praktikate kogumid annavad hea lähtekoha erinevate nõuete, sh MFNi, püstitamisel.

Eelnevalt on töös tutvustatud MFNi mõistet ja erinevaid nõuete kirjeldamise võimalusi. Käesolev peatükk annab lühiülevaate mõnedest raamistikest ja metodoloogiatest, mille järgimine võib aidata kaasata turvalise tarkvaraarenduse eesmärgi saavutamisele.

2.1 Tarkvaraarenduse elutsükli meetod

Tarkvaraarenduse elutsükli meetod (inglise k *Software Development Lifecycle*, edaspidi SDL) on Microsofti poolt loodud arendusprotsessiga seostatud turvategevuste kogum, mis on väljatöötatud tarkvara arenduseks olukordades, kus loodav tarkvara peab vastu pidama erinevatele turvarünnetele (Davis, 2005). SDL on loodud põhimõttel, et see oleks kasutatav ükskõik millise suurusega organisatsioonis, võimaldades selle rakendamist ka väikestes organisatsioonides (Microsoft, 2010).

SDL lisab kõikidele tarkavaraarenduse etappidele vajalikud tegevused ja tulemused saavutamaks turvalist tarkvara. Seega võib öelda, et SDL on kogum kohustuslikke turvategevusi, mis on väljatoodud nende tegemise järjekorras ja grupeeritud integreeritud tarkvara arenduse etappidesse (Microsoft, 2010).

Microsofti tarkvaraarenduse etapid koos nendesse integreeritud turvategevustega on kujutatud alljärgneval joonisel (Joonis 1):

Koolitus	Nõuete püstitamine	Disain	Teostus	Verifitseerimine	Rakendmaine	Tagasiside
Turvanõuete koolitus	Püstita turvanõuded Loo kvaliteedi nõuded/vigade tõkked Turva- ja privaatsusriskide hindamine	Püstita disainiõuded Analüüsi ründepindasid Ohtude modelleerimine	Kasuta kinnitatud aredusvahendeid Eemalda ebaturvalised funktsioonid Staatiline analüüs	Dünaamiline analüüs Hägutestimine (suitsutestid) Ründepindade läbivaatus	Loo intsidendihalduse plaan Lõplik turvatestimine Väljalaske arhiveerimine	Käivita intsidendihalduse plaan

Joonis 1. Microsofti tarkvaraarenduse etapid integreeritud turvategevustega (Microsoft, 2010)

Joonisel 1 kujutatud nõuete püstitamise, disaini, teostuse, verifitseerimise ja rakendamise etapid on Microsofti arendusprotsessi põhietapid. Koolitus ja tagasiside toetavad ja täiendavad põhietappe.

Saavutamaks Microsofti poolt aktsepteeritud SDL rakendatust tarkvaraarenduse projektis, peavad arendustiimi poolt olema läbitud kõik koolituse, nõuete püstitamise, disaini, teostuse, verifitseerimise ja rakendamise 16 turvategevust. SDL põhimõtete järgimine tagab Microsofti kogemusel kõrge töökindluse ja turvalise tarkvara (Microsoft, 2010).

2.2 Tarkvara veatu konstrueerimise metodoloogia

Tarkvara veatu konstrueerimise (inglise k *Correctness by Construction*, edaspidi CbyC) metodoloogia on Praxis Critical Systemsi poolt väljatöötatud meetod arendamaks kõrge terviklusetasemega tarkvara. Antud meetodi abil on edukalt arendatud ohu- ja turvakriitilisi süsteeme. Meetod tagab minimeeritud vigade arvuga tarned, nõudes pidevat vigade tuvastamist ja kõrvaldamist kogu tarkvara arenduse protsessi jooksul. CbyC aluspõhimõte on, et vigu ei tule mitte raporteerida, vaid need tuleb avastamisel koheselt likvideerida (Davis, 2005).

CbyC seitse põhimõtet on järgmised:

1) Ole valmis nõuete muutumiseks arendamise käigus – rakenda pigem rangemaid põhimõtteid, et vältida tulevikus kallist ja ebavajalikku ümbertegemist;

2) Ole alati teadlik sellest, mida sa testid – on kaks erinevat lähenemist testimisele: ühel juhul sa testid selleks, et arendada veatu tarkvara ja teisel juhul sa testid tõestamaks, et tarkvara töötab veatult;
3) Eemalda teadaolevad vead enne testimist – testimine iseenesest on kulukas ja sellest veel kulukam on tarnida vigast tarkvara ning lasta tellijal vigu raporteerida ja neid siis parandada;
4) Arenda lihtsalt verifitseeritavat tarkvara;
5) Arenda inkrementaalselt;
6) Pea meeles, et teatud tarkavara arenduse osad ongi rasked – ükski vahend ega meetod ei tee tarkvara arendust lõpuni lihtsaks, enamus vahendeid ja meetodeid lahendavad kerged probleemid, võimaldades keskenduda rasketele;
7) Tarkvara ei ole iseenesest kasulik – nõuetele vastavalt arendatud tarkvara on kasutuskõlblik ainult selle juurde kuuluvate juhendite, protsesside, disaini dokumentatsiooni, hästi kommenteeritud lähtekoodi ja läbimõeldud testjuhtudega. Kõik eelpool nimetatu on oluline osa tarkvara arenduse protsessist ja neid tuleb vastavalt ka käsitleda (Davis, 2005).

2.3 Turvalisuse integreerimine agiilsetesse tarkvaraarendamise metodoloogiatesse

Agiilseid tarkvaraarendamise metodoloogiaid on palju (nt SCRUM, ekstreemprogrammeerimine, funktsioonikeskne arendus) ja nad on oma olemuselt erinevad. Siiski on neil mitmeid ühiseid jooni: ajaliselt lühikesed arenduse iteratsioonid; vähene eeldisain; pidevalt arenev disain ja arhitektuur; kollektiivne koodi omandus, mille erinevad osad on igaühe poolt vabalt muudetavad; otsesuhtlus; minimaalne või üldse puuduv dokumentatsioon (kood on dokumentatsiooni põhimõte) ja järk-järguline testjuhtude genereerimine. Mitmed agiilse arenduse põhimõtted on vastuolus traditsioonilise turvalise tarkvaraarendamise protsessiga, nt disainil põhinev arenduseelne ohtude modelleerimine on tugevalt vastuolus pidevalt muutuva disaini põhimõttega agiilses arenduses (Davis, 2005).

Eelnevale tuginedes võib väita, et turvalisuse integreerimine agiilsesse tarkvaraarendamisse on keeruline ja nõuab erinevat lähenemist võrreldes selgelt eristuvate etappidega arendusmetoodikatega.

Turvalisuse integreerimise võimalusi agiilse tarkvaraarenduse protsessi on oma artiklis „*Towards Agile Security Assurance*“ tutvustanud Beznosov ja Krutchten. Uuringu peamine eesmärk oli välja selgitada, kuidas oleks võimalik ühendada agiilse arendamise ja turvalisuse tagamise põhimõtteid (Beznosov & Kruchten, 2004).

Oma artiklis klassifitseerisid nad olemasolevad turvalisuse tagamise tegevused nelja kategooriasse: agiilsete meetoditega sobituvad tegevused (2 tk), tarkvara arenduse etappidest sõltumatud tegevused (8 tk), tegevused, mis on automaatsed või pool-automaatsed (4 tk) ja tegevused, mis on oma olemuselt agiilsete meetoditega kokku sobimatud (12 tk)[2] (Davis, 2005).

Soovitused antakse Beznosovi ja Kruchteni poolt peamiselt tegevustele, mis ei ole agiilsete meetoditega sobituvad. Automaatsed ja pool-automaatsed testid on kindlasti tulevikku silmas pidades agiilse arenduse juures kasutust leidvad, sest peaksid tagama kiiremini parema turvalisusega tarkvara (nt testid nõrkuste tuvastamiseks, läbistustestid jm). Peamiseks probleemiks mittesobituvate turvalisuse tagamise tegevuste juures on agiilse arenduse praktiliselt puuduv dokumentatsioon ja väliste ekspertide kaasamine erinevates arendamise etappides. Lahenduseks pakutakse erinevate turvatestide pisteliselt läbiviimist juba esimeste iteratsioonide juures, vähendamaks hilises etapis selguvaid olulisi disaini ja arhitektuuri puudusi, mille kõrvaldamine nõuab täiendavat ajalist ja finantsilist ressurssi. Soovitatakse ka turvaekspertide kaasamist arenduse alguses, kasvõi osalise ajaga, et tagada kogu arendustiimi parem turvateadlikkus ja turvalisuse aspektist lähtudes paremad disaini ja arhitektuuri otsused. Osalise ajaga arendamist nõustav turvaekspert suudab uuringu läbiviijate hinnangul olla mitme agiilse arendusmeeskonna tugiisik üheaegselt ja seega peaks suudetama hoida ka arenduse kulud mõistlikul tasemel (Beznosov & Kruchten, 2004).

[2] Konkreetsete turvalisuse tagamise meetodite loeteluga saab tutvuda viidatud allikas, antud töös neid ei käsitleta.

2.4 Infoturbe ühiskriteeriumite raamistik

Käesolev peatükk kirjeldab Kanada, Prantsusmaa, Hollandi, Ühendkuningriigi ja Ameerika Ühendriikide poolt 1996. aastal väljatöötatud turvalisuse hindamise standardit (inglise k *Common Criteria for Information Technology Security Evaluation*, edaspidi CC).

Standardi pakutav raamistik võimaldab tarkvara kasutajatel spetsifitseerida tarkvara turvanõudeid, arendajatel neid nõudeid realiseerida ning sõltumatutel testimislaboritel nõudeid hinnata (Tepandi, 2016).

CC on üles ehitatud kolmeosalisena:

- Tutvustav osa annab ülevaate ajaloost, standardi eesmärgist ja loob arusaamise üldisest turvalisuse hindamise kontseptsioonist ja põhimõtetest, lisaks tutvustab hindamise mudelit.
- Teine osa keskendub funktsionaalsetele turvanõuetele keskendudes eelkõige süsteemi lõppkasutaja poolt püstitatavatele turvanõuetele. Teine osa annab nimekirja erinevatest funktsionaalsetest turvanõuetest, mille hulgast süsteemi lõppkasutaja saab teha endale sobiva valiku.
- Standardi kolmas osa koosneb turvalisuse hindamise nõuetest valideeritavale süsteemile, sisaldades erinevaid meetodeid tagamaks soovitud turvalisust. Osa sisaldab ka seitset eeldefineeritud komplekti tõendusnõuded, mida tuntakse ka hindamistagatiste tasemete nime all (inglise k *Evaluation Assurance Levels*) (Davis, 2005).

Seitse hindamise taset on järgmised:

1) Tase 1 (EAL1) – funktsionaalsuse ja liideste spetsifikatsioon ja funktsionaalselt testitud (sõltumatu osapoole vastavustestimine);
2) Tase 2 (EAL2) – struktuurselt testitud (lisaks eelmisele nõutud nt arendajate poolne kaastöö koodipõhisel testimisel ja vigade otsingul);
3) Tase 3 (EAL3) – metoodiliselt testitud ja kontrollitud (lisaks eelmistele tuleb nt rahuldada spetsifikatsiooni adekvaatsuse/katvuse kriteeriumid ning testida disaini);

4) Tase 4 (EAL4) – metoodiliselt disainitud, testitud ja läbi vaadatud (lisaks eelmistele mitmesugused analüüsid ja läbivaatused);
5) Tase 5 (EAL5) – poolformaalselt disainitud ja testitud (lisaks eelmistele tuleb osaliselt rakendada formaalseid disaini meetodeid);
6) Tase 6 (EAL6) – poolformaalselt verifitseeritud disain ja testitud (lisaks eelmistele nõutakse osalises või täismahus formaalsete disaini ja testimise meetodite kasutamist);
7) Tase 7 (EAL7) – formaalselt verifitseeritud disain ja testitud (lisaks eelmistele nõutakse täiendavate formaalsete disaini ja verifitseerimise meetodite kasutamist) (Tepandi, 2016).

Turvatasemetest esimene on kõige nõrgem ja seitsmes kõige tugevam. Nõudmised formaalsete arenduse meetodite olemasolule esitatakse alates viiendast turvaklassist. Esimese nelja klassi kohta on ühtlustatud arusaamine ja tunnustamise kokkulepped mitmete maade vahel. Viiendast seitsmenda klassini ei ole praegu üldist kokkulepet turvaklasside sertifitseerimise tunnustamise kohta (Tepandi, 2016).

Kui on teada, milline on MFN oma olemuselt, kuidas seda kirjeldada (st mida silmas pidada) ja millised on erinevad lähenemised nõuete rakendamiseks tarkvaraarenduse juures, siis parema teadlikkuse eesmärgil võib lisaks uurida ka standardeid. Uurida tuleks standardeid, mille eesmärk on anda soovitusi, kuidas erinevaid tarkavaraarenduse käigus esilekerkivaid küsimusi lahendada ja millistele teemadele tähelepanu pöörata, et tarkvaraarenduse protsess ja loodav tarkvara vastaksid turvalisuse nõuetele. Järgnev peatükk magistritöös tutvustab kahte Eesti avalikus sektoris enim järgimist leidvat standardit.

3. Ülevaade standarditest

Tulenevalt asjaolust, et erinevad turvanõuete teemalised uuringud ei ole avalikult kättesaadavad, sest sisaldavad liiga tundlikku informatsiooni, ei ole magistritöö koostamisel tuginetud varasemalt läbiviidud uuringutele. Vabalt kättesaadavad on vaid uuringud, mis keskenduvad indiviidi käitumise uurimisele olukorras, kus organisatsioonis on rakendatud erinevaid infoturbe nõudeid ja muudetud nende järgimine töötajatele kohustuslikuks. Antud valdkond aga ei haaku käesoleva magistritöö teemaga.

Töö koostamisel on uuritud turvastandardite soovitusi ning toetutakse kahele Eesti avalikus sektoris viimasel ajal enim järgimist leidvale standardile:

- infosüsteemide kolmeastmeline etalonturbe süsteem – ISKE,
- veebirakenduste turbe verifitseerimise standard – ASVS.

3.1 ASVS standardi tutvustus

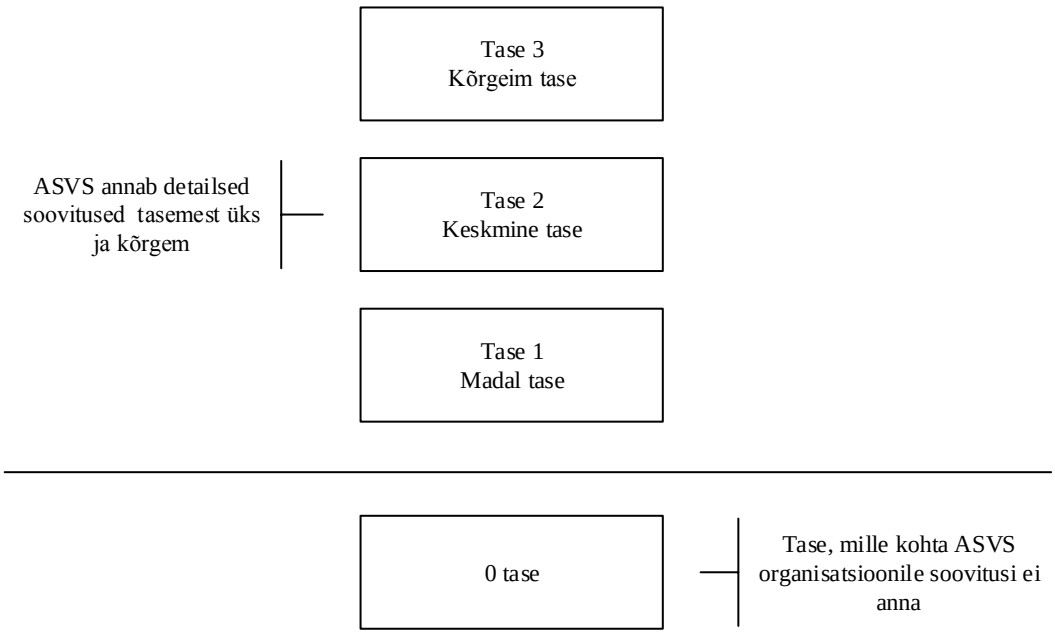
Open Web Application Security Project (edaspidi OWASP) ASVS versioon 3.0.1 on kogukondliku töö tulemusena valminud standard kirjeldamaks turvanõuete ja -kontrollide raamistikku funktsionaalsetele ja mittefunktsionaalsetele turvanõuetele veebirakenduste disainimisel, arendamisel ja testimisel (Manico, Cuthbert, & van der Stock, 2016).

ASVS standardil on kaks peamist eesmärki:

1) Aidata organisatsioonidel luua ja töös hoida turvalisi rakendusi;
2) Võimaldada turvalisusteenusel, turbetööriistade pakkujatel ja tarbijatel ühildada oma nõudeid ja pakkumisi (Manico, Cuthbert, & van der Stock, 2016).

3.1.1 ASVS turvalisuse tasemed

ASVS defineerib kolm turvalisuse verifitseerimise taset, kusjuures iga järgnev tase kirjeldab turvanõudeid detailsemalt (rohkem süvitsi). Standardi järgi on võimalikke tasemeid organisatsioonis neli. 0 taset (inglise k *cursory level*) standard ei kirjelda, eeldades sellel tasemel organisatsiooni enda poolt erinevate kordade ja piirangutega kehtestatud turvanõuete olemasolu (elementaarsed turvanõuded), nt uste lukustamine, arvutite kasutamise kord jms. ASVS erinevad tasemed on kujutatud järgneval joonisel (Joonis 2):



Joonis 2. ASVS 3.0.1 turvalisuse tasemed (Manico, Cuthbert, & van der Stock, 2016)

ASVS kolm taset on järgmised:

1) Tase 1 ehk madal tase (inglise k *opportunistic level*) – mõeldud igasugusele rakendusele;
2) Tase 2 ehk standard tase (inglise k *standard level*) – mõeldud rakendustele, mis sisaldavad tundlikku informatsiooni, mis vajab kaitsmist;
3) Tase 3 ehk kõrgeim tase (inglise k *advanced level*) – mõeldud kriitilistele rakendustele, mis teostavad kõrge väärtusega transaktsioone (nt pankade rakendused), sisaldavad tundliku iseloomuga andmeid (nt meditsiinilised- ja militaarandmed) ning muud rakendused, mis nõuavad kõrgeimat usaldatavust (Manico, Cuthbert, & van der Stock, 2016).

Iga tase sisaldab turvanõuete nimekirja, kus iga turvanõude juures on kirjeldatud antud turvanõude spetsiifikale vastavad võimalikud lahendused arendajatele tarkvara arendustes kasutamiseks, st turvanõuded, mis tuleb tarkvara arendajal arendusse koodi tasemel sisse kirjutada (Manico, Cuthbert, & van der Stock, 2016).

3.1.1.1 ASVS tase üks

Rakendus vastab tasemele üks kui rakendus on kaitstud enamlevinud ja lihtsalt leitavate turvanõrkuste suhtes. OWASP on väljatöötanud nimekirja kümnest enamlevinud veebirakenduse turvariskist „OWASP Top 10“[3], dokument on kõigile huvilistele veebis vabalt kättesaadav (Manico, Cuthbert, & van der Stock, 2016).

Kümme enamlevinud turvariski „OWASP Top 10“ järgi on:

1) Süstimisründed – ründaja lisatud sisu võib rakenduses sundkäivitada soovimatuid käske või võimaldama volitamata ligipääsu andmetele;
2) Autentimis- ja seansihalduse vead – autentimis- ja seansihaldus ei ole korralikult rakendatud, andes võimaluse ründajal salasõnu, võtmeid või seansitunnuseid omastada;
3) Murdskriptimine – vead, mis tekivad, kui rakendus kuvab veebilehitsejas infot ilma korraliku valideerimise või tagasilükkamiseta. Vead võimaldavad ründajal veebilehitsejas käivitada skripte, mille abil saab kasutajaseansse üle võtta, veebilehti näotustada või kasutaja pahaloomulisele veebilehele ümber suunata;
4) Objektide ebaturvaline otseviitamine – arenduse käigus tuleb jälgida, et ei jääks avalikuks viide sisemisele rakendusdokumendile (nt fail, kaust, andmebaasi võti), mille abil ründaja saab andmetele volitamata ligipääsu;
5) Vigased turvaseaded – heatasemeline turvalisus nõuab turvalise seadistuse määratlemist, paigaldamist ja haldamist. Vältida tuleks vaikimisi seadeid ning kasutusel olevat tarkvara tuleb regulaarselt uuendada;
6) Tundlike andmete kaitseta jätmine – tundlikud andmed (nt krediitkaardi- ja autentimisandmed) nõuavad erilist kaitset, nt krüptimine ja erimeetmed andmevahetuses;

[3] OWASP Top 10 (inglisekeelne dokument): https://storage.googleapis.com/google-code-archive-downloads/v2/code.google.com/owasptop10/OWASP%20Top%2010%20-%202013.pdf

7) Puuduv pääsukontroll funktsionaalsuse tasemel – enamasti kontrollivad veebirakendused ligipääsu õigusi enne funktsionaalsuse näitamist veebilehitsejas, samasugune kontroll tuleks teha ka serveris, et vältida funktsionaalsusele volitamata ligipääsu;
8) Päringuvõltsimine – rünne, mis petab laadima veebilehte, mille kaudu tehakse veebirakenduse kasutaja nimel ja õigustega mingi kahjulik toiming;
9) Teadaolevalt auklike komponentide kasutamine – tarkvarakomponent (teek, raamistik) vajab töötamiseks täisõigusi. Aukliku komponendi kasutamine nõrgendab rakenduse kaitsemeetmeid ja võib kaasa tuua tõsise andmekao või serveri ülevõtmisega lõppeva ründe;
10) Kontrollimata ümbersuunamised ja edastamised – veebirakendused suunavad kasutaja korralikult kontrollimata infoga sihtlehtedele, mis võimaldab ründajal suunata kasutajaid õngitsus- või pahavarasaitidele või kasutada edastusi volitamata ligipääsuks (Nigul, 2014).

Esimese taseme rakendamisel saab rakendusi enamasti kontrollida automaatsete tööriistade ja/või -testidega. Lihtsamate turvanõrkuste tuvastamine võib toimuda ka manuaalselt ilma lähtekoodi sisenemiseta. ASVS esimene tase on miinimum, mida iga oma rakenduste turvalisusest huvitatud organisatsioon peaks järgima (Manico, Cuthbert, & van der Stock, 2016).

Esimese taseme ohud on peamiselt ründed füüsiliste isikute poolt, kes kasutavad lihtsalt leitavaid ja ärakasutatavaid turvanõrkusi. Esimese taseme turvanõrkuste ründeid ei teosta reeglina füüsilised isikud, kes on valmis panustama hulgaliselt aega ja vahendeid konkreetse rakenduse ründamiseks. Esimene tase sobib siiski üksikuna rakendamiseks ainult juhul, kui rakenduses olev informatsioon ei ole unikaalne ja ärikriitiline ning ei vaja kaitset. Tase üks on tavaliselt rakendatud taseme kaks või kolm raames, mitte eraldiseisvana. Kui rakenduses sisalduv ja töödeldav informatsioon vajab kaitset, siis tase üks ei ole enam sobilik ja organisatsioon peab rakendama vähemalt taset kaks (Manico, Cuthbert, & van der Stock, 2016).

3.1.1.2 ASVS tase kaks

Rakendus on kaitstud tasemel kaks kui on adekvaatselt rakendatud erinevad meetmed ja abinõud kaitsmaks rakendusi enamike teadaolevate tänapäevaste riskide suhtes.

Taseme kaks rakendamine tagab, et turvalisus on kaasaegne, efektiivne ja katab kogu rakendust. Teise taseme turvalisus peab olema arendatud rakendustesse kõikidel organisatsioonidel, mis tegelevad tehingutega erinevate äride vahel, sh meditsiiniline info, ärikriitiliste funktsioonide tagamisega ja tundliku informatsiooni töötlemisega (Manico, Cuthbert, & van der Stock, 2016).

Teise taseme puhul on ründe sooritajad heade teadmiste ja oskustega ning kõrgelt motiveeritud rünnakut edukalt sooritama. Reeglina ei ole tegu juhusliku rünnakuga, vaid planeeritud ja oskuslikult läbiviidud rünnakuga konkreetse rakenduse ründamiseks, kasutades ära rakenduse nõrkusi (Manico, Cuthbert, & van der Stock, 2016).

3.1.1.3 ASVS tase kolm

Taset kolm rakendatakse eriti kõrge turvavajadusega rakenduste puhul, enamasti vajavad kolmanda taseme turvalisust militaar-, tervise-, julgeoleku- ja kriitilise infrastruktuuri rakendused. Organisatsioonid vajavad kolmandat taset kriitiliste funktsioonide tagamiseks, mille katkemisel võib organisatsioon sattuda raskustesse oma ülesannet täitmisel või üldse lakata olemast (Manico, Cuthbert, & van der Stock, 2016).

Tase kolm eeldab põhjalikku analüüsi, arhitektuuri süüvimist, kodeeringut ja testimist. Turvaline rakendus on läbimõeldult jaotatud mooduliteks (lihtsustamaks elastsuse, skaleeritavuse ja turvakihtide määratlemist). Turvalisus (süvakaitse) on tagatud mooduli põhiselt (iga moodul on füüsiliselt ja/või võrgu ühendusega eraldatud) ja kõik on nõuetekohaselt ning piisava detailsusega dokumenteeritud. Kontrollid on kohustuslikud tagamaks konfidentsiaalsust (nt krüpteerimine), terviklust (nt transaktsioonid, sisendi valideerimine), käideldavust (nt koormustaluvus), autentimist (kaasaarvatud süsteemidevaheline autentimine), salgamatust, autoriseerimist ja auditeerimist (logimine) (Manico, Cuthbert, & van der Stock, 2016).

3.1.2 ASVS rakendamine praktikas

ASVS soovitab standardi erinevate tasemete rakendamisel lähtuda organisatsiooni tegevusalast. Organisatsioonid on standardis tegevusalade kaupa jagatud nelja gruppi

ja igale grupile on tulenevalt tema spetsiifikast kirjeldatud tasemed üks kuni kolm. Organisatsioonide grupid tegevusalade kaupa jagatuna on järgmised:

1) Finants ja kindlustus – antud tegevusalade puhul on rünnakud tihti ajendatud rahalise kasu saamise eesmärgist. Tavaliselt on rünnaku eesmärk delikaatsete andmete või konto mandaatide saamine, mida kasutatakse pettuse toimepanemiseks või võimendamaks rakendusse programmeeritud raha liikumise funktsionaalsuse kasutamist.

   Tase 1 soovitused kuuluvad rakendamisele rakendustes, mis on võrgu kaudu ligipääsetavad.

   Tase 2 soovitused kuuluvad rakendamisele rakendustes, mis sisaldavad delikaatseid andmeid (nt krediitkaartide numbrid), delikaatseid isikuandmeid, mille abil saab piiratud koguses ja piiratud viisidel raha liigutada.

   Tase 3 soovitused kuuluvad rakendamisele rakendustes, mis sisaldavad suurel hulgal delikaatset informatsiooni või võimaldavad suurtes summades ja/või kiireid (ilma viivituse ja/või täiendava kontrollita) raha ülekandeid.
2) Tootmine, transport, tehnoloogia, kommunaalteenused, infrastruktuur ja kaitsetööstus – esmapilgul ei tundu neil tegevusaladel midagi ühist olevat. Potentsiaalsed ohuallikad, kes võiksid ründeid sooritada, on nende kõigi puhul aja, oskuste ja vajalike vahendite poolest ühesuguselt motiveeritud ja varustatud. Tihti ei ole sensitiivne info ja süsteemid lihtsalt leitavad ning nende tuvastamiseks on vaja siseringi kuuluvate isikute abi ja sotsiaalse manipuleerimise tehnikate kasutamise oskust. Rünnakutes võivad osaleda organisatsiooni sisesed liikmed ja välised osalised, võimalik on ka nende kahe osapoole koostegutsemine. Rünnakute eesmärgiks võib olla intellektuaalse omandi üle kontrolli saavutamine, tagamaks strateegilist või tehnoloogilist eelist. Alahinnata ei tasu ka eesmärki halvata, mõjutada või häirida tundlike rakenduste funktsionaalsust. Valdavalt soovitakse rünnakuga siiski omandada delikaatset infot ja andmeid (paroolid, ligipääsu koodid jms). Sellisel juhul kasutatakse andmeid peamiselt identiteedi varguseks, alusetuteks makseteks või erinevate petuskeemide rahastamiseks.

   Tase 1 soovitused kuuluvad rakendamisele rakendustes, mis on võrgu kaudu ligipääsetavad.

   Tase 2 soovitused kuuluvad rakendamisele rakendustes, mis sisaldavad ainult sisemiseks kasutamiseks mõeldud informatsiooni või informatsiooni töötajate

kohta, mida on võimalik kasutada töötajate sotsiaalseks manipuleerimiseks. Tase 2 soovitused on lisaks mõeldud rakendustele, mis sisaldavad organisatsiooni jaoks vähem olulist informatsiooni intellektuaalomandi ja ärisaladuse kohta.

Tase 3 soovitused kuuluvad rakendamisele rakendustes, mis sisaldavad organisatsiooni jaoks üliolulist intellektuaalomandi ja ärisaladuse informatsiooni ning avaliku sektori riigisaladusi sisaldavad rakendused. Informatsioon, mida need rakendused loovad ja talletavad, on organisatsiooni püsimiseks ja eduks kriitilise tähtsusega. Siia alla kuuluvad ka rakendused, mis on ellu kutsutud kontrollimaks kriitilise tähtsusega rakenduste toimimist (nt erinevad kontrollisüsteemid).

3) Tervishoid – enamasti on sihiks delikaatne info, mida kasutatakse identiteedi varguseks, alusetuteks makseteks või erinevate petuskeemide rahastamiseks. Erinevates riikides võivad lisaks olla nõuded spetsiaalsete tervishoiustandardite rakendamiseks, nt HIPAA Ameerika Ühendriikides.

   Tase 1 soovitused kuuluvad rakendamisele rakendustes, mis on võrgu kaudu ligipääsetavad.

   Tase 2 soovitused kuuluvad rakendamisele rakendustes, mis sisaldavad vähesel määral delikaatset meditsiinilist informatsiooni, personaliseeritavaid andmeid ja maksete andmeid.

   Tase 3 soovitusi rakendatakse rakendustes, mis kontrollivad meditsiiniseadmeid ja meditsiinilisi andmeid, mille tõrgete ja ebatäpsuste tulemusena on ohus inimelud. Siia alla kuuluvad ka erinevad makse- ja kassasüsteemid, mis sisaldavad suurel hulgal andmeid ülekannete kohta, mida on võimalik kasutada erinevate pettuste toimepanemiseks. Antud taseme alla kuuluvad ka kõik taseme 3 rakenduste administreerimisliidesed.

4) Jaekaubandus, toitlustus ja majutus – peamiseks nende tegevusvaldkondade organisatsioonide rakenduste rünnakute motiiviks on maksete ja ülekannete sooritamiseks vajaliku informatsiooni või siis isikuandmete hankimine, mis oleksid kasutatavad erinevate pettuste läbiviimisel. Vähem tõenäoline, aga mitte võimatu on ka intellektuaalse omandi ja konkurentide kohta käiva informatsiooni hankimiseks sooritatavad ründed, mis tagaksid organisatsioonile parema positsiooni turul või läbirääkimiste laua taga.

   Tase 1 soovitused kuuluvad rakendamisele rakendustes, mis on võrgu kaudu ligipääsetavad.

Tase 2 soovitused kuuluvad rakendamisele erinevates ärirakendustes (nt ERP, CRM). Taset 2 tuleb rakendada ka vähesel ja mõõdukal määral maksete informatsiooni sisaldavates rakendustes ja tellimuste haldamise rakendustes (nt e-poe funktsionaalsus, kus kajastub tellimuste ja nendega seotud maksete info).
Tase 3 soovitused kuuluvad rakendamisele makse- ja kassafunktsioonide rakendustes, mis sisaldavad suurel hulgal ülekannete andmeid, mida on võimalik kasutada erinevate pettuste toimepanemiseks. Antud taseme alla kuuluvad ka kõik taseme 3 rakenduste administreerimisliidesed. Tase 3 kaitset peab rakendama ka rakendustele, mis sisaldavad suurel hulgal delikaatset informatsiooni (nt täies mahus säilitatavad krediitkaardi numbrid, isikukood, ema neiupõlve nimi jms) (Manico, Cuthbert, & van der Stock, 2016).

3.1.3 Loodud tarkvara ASVS standardile vastavuse hindamine

OWASP organisatsioon ei sertifitseeri müüjaid, kontrollijaid ega tarkvara ennast, mis tähendab, et ametlikult ei saa olla ühelgi organisatsioonil OWASPi sertifikaati. Organisatsioonil võivad olla põhjalikud teadmised ja oskused nii OWASPi rakendamiseks kui ka rakendatuse kontrollimiseks, aga need ei ole kunagi sertifitseeritud (Manico, Cuthbert, & van der Stock, 2016).

ASVSile vastavuse kontrollimisel tuleb silmas pidada erinevaid aspekte. Kontrollida tuleb mitte ainult piiratud ligipääsuga rakendusi vaid, ka neid rakendusi, mis on vaba ja piiramata ligipääsuga. Eriti juhul kui organisatsioon soovib rakendada standardit tasemetel kaks või kolm. Erinevatele nõuetele vastavuse kontrolli tulemustes tuleb fikseerida nii hästi rakendatud (kontrollimisel positiivse tulemuse saanud), kui ka nõuded, mille rakendamine ebaõnnestus. Erinevate läbistustestide tulemusena, tuleb esitada nii läbitud kui läbimata nõuete loetelu. Kõikidele probleemsetele kohtadele tuleb pakkuda lahendus standardiga kooskõlla viimiseks (Manico, Cuthbert, & van der Stock, 2016).

Kõik andmed esialgse testimise, ettepanekute, paranduste ja järeltestimise kohta tuleb säilitada, et oleks vajadusel võimalik tõestada iga nõude testimist (selle läbiviimist) ja saadud tulemusi (Manico, Cuthbert, & van der Stock, 2016).

Kui organisatsioon soovib, et tema rakendus vastaks ASVS nõuetele, ei piisa ainult automaatsetest läbistustestidest. Automaatsed läbistustestid on sobivad esimesele tasemele vastavuse testimiseks. Enamik keskmise ja kõrgema taseme nõudeid ei ole kontrollitavad automaatsete läbistustestide abil. Standard juhib tähelepanu ka asjaolule, et manuaalsete ja automaatsete läbistustestide vahel on joon järjest enam hägustumas, sest järjest rohkem kasutatakse erinevaid kombinatsioone manuaalsetest ja automatiseeritud testidest (Manico, Cuthbert, & van der Stock, 2016).

3.1.4 ASVS standardi kontrollnõuete gruppide lühiülevaade

OWASP standardi kontrollnõuded on jagatud 16 gruppi. Iga kontrollnõuete grupp sisaldab eesmärkide tutvustust, nõuete loetelu (koos viitega, mis tasemele nõue kohaldub) ja viiteid lisamaterjalidele ning võimalikele tööriistadele. Kontrollnõuete grupid on järgnevad[4]:

1) Arhitektuur, disain ja ohtude modelleerimine

   Tuleb jälgida, et rakendus vastaks tasemest tulenevatele nõuetele. Tasemel üks tuleb kontrollida, et rakenduses kasutatud komponendid on identifitseeritud ja nende olemasolu on põhjendatud. Tasemel kaks tuleb jälgida, et arhitektuur on defineeritud ja loodud kood vastab defineeritud arhitektuurile. Tasemel kolm peab olema tagatud ajakohane ja tõhusalt rakendatud arhitektuur ja disain.
2) Autentimine

   Autentimine on kinnituse andmine olemi väidetava identiteedi või tunnusomaduse õigsusele. Tuleb tagada, et teabe saatja oleks üheselt tuvastatud, ainult volitatud isikud saaksid ennast autentida ja mandaat on edastatud turvaliselt.
3) Seansi haldus

   Seansi haldus on üks veebirakenduse põhikomponente, mille abil rakendus loob, kontrollib ja säilitab kasutaja suhtlust sisselogimisest väljalogimiseni. Tuleb tagada, et seansid oleks kasutaja kaupa unikaalsed ning oletamine ja jagamine ei oleks võimalikud. Kui seanss ei ole rakenduse kasutamiseks vajalik peab see olema tühistatud ning tegevusetuse ajal peab toimuma seansi aegumine.
4) Ligipääsu kontroll

[4] Antud töös kasutatud gruppide numeratsioon ei vasta standardi gruppide numeratsioonile.

Autoriseerimine tagab, et ressurssidele pääsevad ligi ainult need kasutajad, kellel on selleks vastavad õigused. Tagada tuleb rakendust kasutama asuvate isikute kehtiv õigus rakendust kasutada. Kasutajatele kehtivad läbimõeldud ja põhjendatud rollide ja õiguste kogumid. Rollide ja õiguste metaandmed on kaitstud viisil, et neid ei ole võimalik taasesitada ega manipuleerida.

5) Pahatahtliku sisendi käsitlemine

   Kõige levinum veebirakenduse turvanõrkus, mille puhul rakendus ei suuda piisaval tasemel kontrollida kliendilt või teiselt keskkonnalt tulevaid sisendandmeid enne nende kasutamist. Peamine veebirakenduse nõrkuste (nt süstimisründed, murdskriptimine, failisüsteemide ründed, puhvri ületäitumine) põhjustaja. Tuleb tagada, et kogu rakendusele edastatav sisend on valideeritud, st korrektne ja otstarbekas. Väliseid andmeid ei tohi kunagi usaldada ja neid tuleb vastavalt käsitleda.

6) Krüptograafiliste turvameetmete olemasolu

   Tagada tuleb, et vigade korral lõpetavad kõik krüptograafilised moodulid töö turvaliselt ja vigadega käsitlemine on reglementeeritud. Juhuarvude tekitamise vajaduse korral tuleb leida sobilik juhuarvude generaator. Ligipääs võtmetele peab olema korraldatud turvaliselt.

7) Vigade haldus ja logimine

   Vigade halduse ja logimise peamine eesmärk on tagada kasutaja, administraatori ja kasutajatoe vajalik reageerimine. Eesmärgiks tuleb võtta kõrge väärtusega logiandmete kogumine. Kõrge väärtusega logid sisaldavad enamasti tundlikke andmeid ja seetõttu tuleb nende käitlemisel rakendada asukohamaal kehtivaid andmete kaitsmisega seotud seadusi ja direktiive. Seadustest tulenevad enamlevinud piirangud logimisel: tundlike andmete kogumine ja logimine võib toimuda põhjendatud vajaduse korral; kõikide logitud andmete käitlemisel ja kaitsmisel tuleb lähtuda andmete turvaklassist tulenevatest turvanõuetest; tagatud peab olema logide põhjendatud eluiga, st neid ei säilitata igavesti, vaid minimaalselt vajaliku aja jooksul.

8) Andmete kaitse

   ASVS eeldab, et andmete kaitse toimub usaldatavas süsteemis, mida on tugevdatud ja millel on piisavad kaitsemehhanismid. Rakendused peavad eeldama, et kõikide kasutajate seadmed on mingis osas kompromiteeritud (ebaturvalised).

Kui rakendus edastab tundlikke andmeid sellistele seadmetele, peab olema tagatud talletatud andmete krüpteeritus ning andmed peavad olema kaitstud omavolilise omandamise, muutmise ja avalikustamise eest. Antud osa standardist määratleb rakenduses kajastuvate andmete konfidentsiaalsuse, tervikluse ja kättesaadavuse tagamise nõuded.

9) Kommunikatsioon (side)

   Tundlike andmete edastamisel tuleb kasutada TLS protokolli. Rakenduses tuleb alati kasutada tugevaid algoritme ja šifreid.

10) HTTP turvaline konfiguratsioon

    Rakenduse serveris rikkumise või ründe võimalus on minimeeritud, ei kasutata serveri vaikimisi konfiguratsiooni. HTTP päises on vastustel sisu tüübis kasutusel turvaline märgistik.

11) Kahjurvarad

    Kahjurkood on haruldane ja seda on väga raske tuvastada. Manuaalne rida realt teostatav koodi ülevaatus väga kogenud koodi läbivaataja poolt ei pruugi tuvastada loogikavigu isegi juhul kui on teada, et kood sisaldab kahjurit. Antud osa standardist saab rakendada ainult juhul, kui on täielik ligipääs lähtekoodile, sh võimalikult paljudele kolmandate osapoolte teekidele. Pahatahtliku tegevuskoha ilmsikstulekul ei tohi kahjurvara mõjutada ülejäänud rakendust. Viitpommide või muude ajal põhinevate ründevahendite sisaldumine rakenduses peab olema välistatud. Rakenduses peab olema välistatud tagauste, pühademunade, salaami rünnete või loogika vigade olemasolu, mille abil on ründajal võimalik tegutseda.

12) Äriloogika

    Äriloogika peab olema järjestikune ja läbimõeldud, sisaldama piisavalt piiranguid tuvastamaks ja ära hoidmaks ründeid (nt pidevad väikestes summades ülekandeid, miljoni sõbra liitmine rakenduse kasutamiseks ühe toiminguga). Kõrgeima väärtusega äriloogika puhul peavad olema läbimõeldud võimalikud kuritarvitamise juhtumid ja ohusubjektid ning olema korraldatud kaitse teeskluse, urkimise, salgamise, informatsiooni paljastamise ja õiguste vallutuse rünnete vastu.

13) Failid ja ressursid

    Tagatud peab olema ebausaldusväärsetest failidest pärinevate andmete turvaline käsitlemine. Ebausaldusväärsetest allikatest pärinevad failid tuleb talletada

väljaspool veebi juurkataloogi ja nendel peavad olema rakenduses piiratud õigused.

14) Mobiil

Antud nõuete grupp on ainult mobiilirakendusele kohalduv. Mobiilirakendustele tuleb kohaldada samal tasemel turvakontrolle nagu serveritele, rakendades turvakontrolle usaldusväärses keskkonnas. Tundlike infovarade talletamisel seadmetes tuleb järgida turvalisuse nõudeid. Tundlike andmete edastamine mobiilseadmetest tuleb lahendada TLS põhimõtteid silmas pidades.

15) Veebiteenused

Tuleb tagada, et rakendus, mis kasutab RESTful – või SOAP-põhiseid veebiteenuseid omab veebiteenuse piisavat autentimise, sessioonihalduse ja autoriseerimise taset. Tagatud peab olema kõikide parameetrite, mis liigutavad andmeid madalamalt tasemelt kõrgemale, sisendandmete kõlblikkuse kontroll. Peab olema tagatud SOAP veebiteenuse kihi algtasemel koostalitlusvõime, et soodustada rakendusprogrammiliidese (API) kasutust.

16) Konfiguratsioon

Tagatud peab olema ajakohaste teekide ja platvormide kasutamine ning vaikimisi turvaline konfiguratsioon. Konfiguratsioon peab olema piisavalt tugevdatud, et kasutaja poolt algatatud vaikimisi konfiguratsiooni muudatused ei tekitaks ega paljastaks alussüsteemide nõrkusi või vigu (Manico, Cuthbert, & van der Stock, 2016).

3.2 ISKE standardi tutvustus

Infosüsteemide kolmeastmeline etalonturbe süsteem (edaspidi ISKE) on Eestis riigi infosüsteemi infoturbe standard. ISKE väljatöötamisel ja arendamisel on aluseks võetud Saksamaa BSI (saksa k *Bundesamt für Sicherheit in der Informationstechnik*, inglise k *Federal Office for Information Security*) avaldatav infoturbe standard – *IT Baseline Protection Manual* (saksa k *IT-Grundschutz*). 2004. aastast kasutusele võetud ISKE rakendamise eesmärk on tagada infosüsteemides töödeldavatele andmetele piisava tasemega turvalisus. Süsteem on loodud eelkõige riigi ja kohaliku omavalitsuse andmekogude pidamisel kasutatavatele infosüsteemidele ning nendega seotud

infovaradele turvalisuse tagamiseks. Andmete turvalisus tähendab, et on saavutatud kolm eesmärki**:** teabe käideldavus, teabe terviklus, teabe konfidentsiaalsus (Riigi Infosüsteemi Amet, 2017).

ISKE ei ole mõeldud riigisaladust töötlevate infosüsteemide turbeks (Vabariigi Valitsus, 2009).

Üheski infosüsteemis ei ole olemas täielikku turvalisust, st täielikku käideldavust (K), terviklust (T) ja konfidentsiaalsust (S). Millistele infoturbe aspektidele tuleb konkreetsete andmete korral tähelepanu pöörata, oleneb konkreetsest infosüsteemist ja selle otstarbest, st käideldava info väärtusest. Enamasti võetakse arvesse turvalisuse kõiki kolme komponenti, kuid erinevate kaaludega. Organisatsioonis nõutav infoturbe tase sõltub organisatsiooni ülesannetest, õigusaktidest ja eeskirjadest, organisatsiooni tegevuse sisemisest korraldusest, infosüsteemide ja ka teenuseandjate ja koostöö- või lepingupartnerite tagatud või nõutud turvatasemest jms (Riigi Infosüsteemi Amet, 2017).

Turvameetmete süsteemi rakendamine seisneb infoturbe eesmärkidele vastavate turvaklasside määramises ja nendele vastavate turvameetmete valimises vastavalt ISKE rakendamisjuhendile ja nende rakendamises ning rakendamise auditeerimises (Vabariigi Valitsus, 2009).

Meetmestik on ehitatud kihilisena, nii et keskmine tase (tase M) saadakse teatud meetmete lisamise teel madala taseme (tase L) meetmetele ja kõrge tase (tase H) teatud meetmete lisamisel madala ja keskmise taseme meetmetele (Riigi Infosüsteemi Amet, 2017).

ISKE nõudmisi peab arvesse võtma enne uute infosüsteemide arendusega alustamist või olemasolevatesse muudatuste tegemist, sest infosüsteemide tagantjärele kohendamine kehtivatele nõuetele vastavaks võib osutuda väga keeruliseks (Riigi Infosüsteemi Amet, 2017).

3.2.1 Andmete turvaklassi ja turbeastme määramine

Andmete vajaliku turvaklassi peab määrama andmete omanik. ISKE kasutab turbetasemete määramiseks neljapallilist skaalat. Rakendades kolmele

turvaeesmärgile neljapallilist skaalat määratletakse turvaosaklassid, mille tähised koosnevad turvaeesmärgi tähisest ja turvataseme väärtusest (Riigi Infosüsteemi Amet, 2017).

Andmete turvaklass on kolme turvaosaklassi konkreetne kombinatsioon. Nende kõikvõimalike kombinatsioonide arv on 4x4x4 ehk erinevaid turvaklasse on 64. Andmete turvaklassi moodustub turvaosaklasside tähistest nende järjestuses K-T-S (Riigi Infosüsteemi Amet, 2017).

Käideldavus (K):

- K0 – käideldavus väiksem kui 90% aastas ja maksimaalne lubatud ühekordse katkestuse pikkus teenuse töö ajal üle 24 tunni (st ühekordse katkestuse pikkus tohib olla suurem kui 24 tundi)*;
- K1 – käideldavus suurem või võrdne 90% ja väiksem kui 99% aastas ning maksimaalne lubatud ühekordse katkestuse pikkus teenuse töö ajal kuni 24 tundi (st ühekordne katkestuse pikkus tohib olla kuni 24 tundi)*;
- K2 – käideldavus suurem või võrdne kui 99% ja väiksem kui 99,9% aastas ning maksimaalne lubatud ühekordse katkestuse pikkus teenuse töö ajal kuni 4 tundi (st ühekordse katkestuse pikkus võib olla vahemikus väiksem või võrdne 4 tunniga ja suurem kui 1 tund)*; suurem või võrdne kui 99% ja väiksem kui 99,9% aastas ning maksimaalne lubatud ühekordse katkestuse pikkus teenuse töö ajal kuni 4 tundi (st ühekordne teenuse katkestuse võib olla 1 – 4 tundi)*;
- K3 – käideldavus suurem ja võrdne kui 99,9 % aastas ja maksimaalne lubatud ühekordse katkestuse pikkus teenuse töö ajal 0 sekundit kuni 1 tund (st ühekordse katkestuse pikkus võib olla väiksem või võrdne 1 tunniga)* (Riigi Infosüsteemi Amet, 2017).

* Maksimaalne lubatud katkestuste arv, maksimaalne lubatud summaarne katkestuste aeg ja muud detailsemad teenustaseme mõõdikud ning teenuse osutamise tingimused (nt päringutele vastamise aeg, planeeritud hooldustööde tegemise aeg, nõutav rikete kõrvaldamise aeg, riketest teavitamise kontaktid, varundamise tingimused jmt) kirjeldatakse ja lepitakse kokku asutuse teenustaseme lepetes (SLAdes) (Riigi Infosüsteemi Amet, 2017).

Terviklus (T):

- T0 – info allikas, muutmise ega hävitamise tuvastatavus ei ole olulised; info õigsuse, täielikkuse ja ajakohasuse kontrollid pole vajalikud;
- T1 – info allikas, selle muutmise ja hävitamise fakt peavad olema tuvastatavad; info õigsuse, täielikkuse, ajakohasuse kontrollid erijuhtudel ja vastavalt vajadusele;
- T2 – info allikas, selle muutmise ja hävitamise fakt peavad olema tuvastatavad; vajalikud on perioodilised info õigsuse, täielikkuse ja ajakohasuse kontrollid;
- T3 – infol allikal, selle muutmise ja hävitamise faktil peab olema tõestusväärtus; vajalik on info õigsuse, täielikkuse ja ajakohasuse kontroll reaalajas (Riigi Infosüsteemi Amet, 2017).

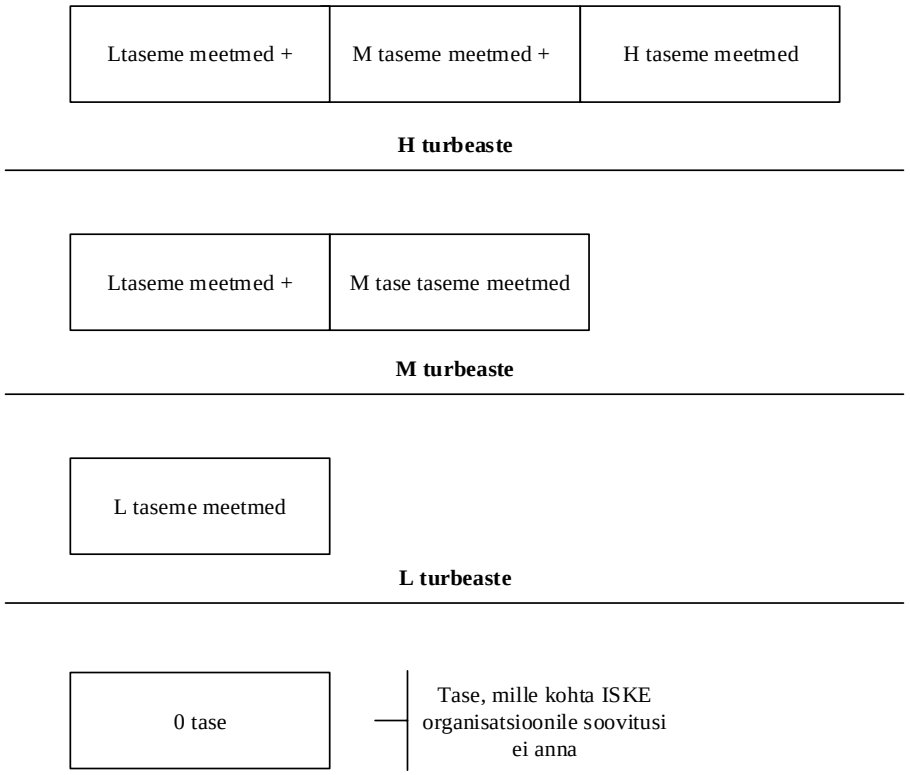
Konfidentsiaalsus (S):

- S0 – avalik info: juurdepääsu teabele ei piirata (st lugemisõigus kõigil huvitatutel, muutmise õigus määratletud tervikluse nõuetega);
- S1 – info asutusesiseseks kasutamiseks: juurdepääs teabele on lubatav juurdepääsu taotleva isiku õigustatud huvi korral;
- S2 – salajane info: info kasutamine lubatud ainult teatud kindlatele kasutajate gruppidele, juurdepääs teabele on lubatav juurdepääsu taotleva isiku õigustatud huvi korral,
- S3 – ülisalajane info: info kasutamine lubatud ainult teatud kindlatele kasutajatele, juurdepääs teabele on lubatav juurdepääsu taotleva isiku õigustatud huvi korral (Riigi Infosüsteemi Amet, 2017).

Turvaklasside määramise käigus tuleb hoolikalt jälgida süsteemide vahelisi seoseid, hoolitsedes selle eest, et oluliste infosüsteemidega seotud muude süsteemide liiga nõrk turve ei ohustaks oluliste süsteemide turvalisust. Turvaklass on aluseks andmetele ja muudele infovaradele meetmete määramisel (Riigi Infosüsteemi Amet, 2017).

Turvaklass on aluseks turbeastme määramisele. ISKEs on kirjeldatud kolm turbeastet: madal – L, tuleneb inglisekeelsest sõnast „*Low*“, keskmine – M, tuleneb inglisekeelsest sõnast „*Medium*“, kõrge – H, tuleneb inglisekeelsest sõnast „*High*“.

ISKE turbeastmed on kujutatud järgneval joonisel (Joonis 3):



Joonis 3. ISKE turbeastmed

3.2.2 Tarkvaraarenduse seisukohast oluliste moodulite ülevaade

Kui kõigi infovarade nõutav turbeaste on määratud, tuleb leida igale infovarale vastavad tüüpmoodulid ISKE tüüpmoodulite turvaspetsifikatsioonide kataloogist B. Tüüpmoodulid jagunevad viieks:

- B1 Üldkomponendid
- B2 Infrastruktuur
- B3 IT süsteemid
- B4 Võrgud
- B5 Rakendused

Tüüpmoodulite spetsifikatsioonid sisaldavad loetelu rakendamisele kuuluvatest turvameetmetest.

Eestis kehtiva ISKE standardi viimasesse versiooni (versioon 8.02) on lisatud uus moodul B 5.27 „Tarkvaraarendus“, mille eesmärgiks on pakkuda organisatsioonile tuge tarkvaraarenduse läbiviimisel. Järjest enam on organisatsioonides tüüptarkvara

asemel kasutusel organisatsiooni spetsiifikast ja –protsessidest lähtuva spetsiaalselt organisatsiooni vajaduste jaoks loodud tarkvara. Lisaks organisatsiooni vajadustest lähtuva tarkvara arendamisele organisatsioonis, leiab järjest enam kasutust ka välistelt partneritelt erinevate teenuste tellimine. Väljasttellimise (inglise k *outsourcing*) turvaliseks läbiviimiseks annab erinevaid juhiseid moodul B 1.11 Väljasttellimine.

Meetmed jagunevad:

- kohustuslikud meetmed – ISKE rakendajale järgimiseks kohustuslik;
- „z“ meetmed ehk soovituslikud meetmed – vajavad rakendamist eelkõige kõrgema turbeastme puhul;
- „w“ meetmed ehk selgitavad meetmed – meetme eesmärk on aidata mõista ja rakendada mõnda teist meedet.

Järgnevalt vaatleb töö koostaja mõlema mooduli peamisi ohte ja meetmeid, mis võivad olla abiks tarkvaraarenduse seisukohast.

3.2.2.1 ISKE moodul B 5.27 Tarkvaraarendus

Tarkvaraarenduse moodul vaatleb erinevaid aspekte, millega organisatsioonid peavad sisese tarkvara kasutamise korral arvestama. Selleks võtab moodul vaatluse alla organisatsiooni ettevalmistuse, realiseerimise ja kasutuselevõtu erinevad tahud, keskendudes tarkvaraarenduse infoturbe asjakohastele aspektidele.

Peamised ohud tarkvaraarenduse puhul jagunevad ISKEle tuginedes järgmiselt:

- Vääramatu jõud – IT-süsteemi ühe komponendi väljalangemine võib põhjustada kogu IT-käituse hädaolukorra, mis tähendab oluliste tööprotsesside seiskumist. Kogu süsteemi avariini viivad sagedamini IT-süsteemi kesksete komponentide (nt kohtvõrguühenduse server, võrguühenduselemendid) rikked;
- Töökorralduslikud puudused – antud ohtude grupp on kõige mahukam käsitledes organisatsiooni puudulikke reegleid ja protseduure, nt turvameetmete ebapiisav järelevalve, testimine tootmisandmetega, õiguste volitamata kasutamine, õige arenduskeskkonna valik ja selle ebapiisav turvamine jne;

- Inimvead – antud ohtude grupp keskendub inimese tegevuse tõttu tekkivatele ohtudele, nt hooletus turvameetmete suhtes, väär pääsuõiguste haldus, inimtegevuse tulemusel tekkiv andmete konfidentsiaalsuse ja tervikluse kadu jne;
- Tehnilised rikked – ohtude osa keskendub puudulike või puuduvate autentimise ja krüpteerimise lahenduste võimalikele ohtudele, nt nõuded paroolidele ei ole kehtestatud või need ei ole piisavalt turvalised, aegunud protokolli versioonide kasutamine, ebaturvaliste krüptograafiliste algoritmide kasutamine, ebaturvaline võtmete haldus jne;
- Ründed – võimalikud ründed on seotud andmete manipuleerimise, pahavara levitamise, võltsitud sertifikaatide kasutamise, erinevate teenustõkke rünnete jms (Riigi Infosüsteemi Amet, 2017).

Kõigist eelpool kirjeldatud ohtudest lähtuvalt on tarkvaraarenduse jaoks välja töötatud terve rida meetmeid, mille rakendamise abil on võimalik minimeerida tekkivat kahju või seda sootuks ära hoida. Järgnevalt on tutvustatud mooduli B 5.27 Tarkvaraarendus soovitatavaid meetmeid. Koos tarkvaraarenduse mooduliga tuleb organisatsioonis läbiviidavate arenduste spetsiifikast lähtudes pidada silmas osaliselt ka järgnevate moodulite meetmeid:

- Moodul B 1.10 Tüüptarkvara – mooduli töökorralduslikud meetmed võivad anda lisandväärtust;
- Moodul B 5.25 Rakendused – moodul sisaldab erinevaid tarkvara rakendamise protseduure, eriti tuleb tähelepanu pöörata tarkvara kasutamise, kasutamisest kõrvaldamise ja hädaolukordadeks valmisoleku meetmetele;
- Moodul B 5.21 Veebirakendused – meetmed tuleb läbi vaadata juhul, kui arendatakse veebirakendust (Riigi Infosüsteemi Amet, 2017).

Tarkvaraarenduse turvaliseks läbiviimiseks esitab ISKE standard järgnevate meetmete rakendamise soovituse:

- Rollide ja vastutuste ühene määratlemine – kõik projekti juures töötavad isikud peavad teadma, millised on nende vastutusala ülesanded ja kes on kontaktisik väljaspool tema ülesannete vastutusala.

  Kõikides projektides peab olema nimetatud isik, kellel lasub koguvastutus tarkvaraarendusprotsessis vajaliku turbe eest: antud isik määrab kõik

turvameetmed, kontrollib nende rakendamist ning on kontaktisik seoses meetmeid puudutavate küsimustega.

Infoturbe olekut tuleb igas projektis regulaarsete ajavahemike järel kontrollida.

- Protsessimudeli valik – tagamaks arendusprotsessi reguleeritud kulgu, tuleb valida sobiv protsessimudel (nt koskmudel, spiraalmudel, prototüüpimine).
- Vastavusnõuete järgimine – tarkvara arendamisel ja rakendamisel tuleb arvestada, et ei minda vastuollu seadusandlusega (nt autoriõigus, litsentsitingimuste täitmine, standarditest kinnipidamine).
- Arenduskeskkonna valimine (soovituslik meede) – tarkvaraprojekti nõuetekohane teostamine eeldab sobivat arenduskeskkonda. Esmajoones on sobiliku keskkonna valiku aluseks ettenähtud programmeerimiskeel ja kavandatav rakendustüüp. Lisaks tuleks silmas pidada ka soetamis- ning käitamiskulusid.
- Arendustööriistade valik – arendamiseks tuleb kasutada tõestatud turbefunktsioonidega tööriistu, mida ei ole võimalik kahjustada (nt peidetud turvaaukude ja tagauste tekitamiseks).
- Täiendavate tööriistade soetamine (soovituslik meede) – tarkvaraarenduse jaoks vajalikud täiendavad tööriistad (nt graafikaprogramm) ja lisaseadmed (nt kiipkaardilugeja). Soetamisel tuleb kontrollida pakkujaid ja arvesse võtta vastavale kasutuseesmärgile kehtivaid turbenõudeid.
- Arenduskeskkonna turvaline kasutamine – arendus- ja testkeskkond peavad alati olema eraldiseisvad töökeskkonnast[5]. Arendus- ja testkeskkonnas ei tohi kasutada töökeskkonna andmeid, tagatud peab olema kõikide andmete (ka arendusandmete) käideldavus, terviklus ja konfidentsiaalsus. Mittevajalikud andmed tuleb programmipakettidest enne töökeskkonda viimist eemaldada. Kogu arendusega seotud suhtlus (tellija, arendaja, testija) peab toimuma krüpteeritud andmeühenduste kaudu. Kasutusel peab olema ajakohane viirusetõrje.
- Turvaline süsteemikujundus – sisendandmeid tuleb enne edasitöötlemist põhjalikult kontrollida ja need valideerida. Süsteemikomponentide vahel tuleb andmeid edastada krüpteeritult. Tarkvaral ja süsteemil peab olema turvaline tüüpkonfiguratsioon. Vigade või süsteemi komponentide tõrke korral ei tohi avaldada andmeid (nt versiooni numbreid või failiteed). Tarkvara kasutamine peab olema lubatud võimalikult väheste kasutajaõigustega.

[5] ISKE nimetab töökeskkonda „tootmiskeskkond“

- Turvalistest protseduuridest kinnipidamine – oluline on tarkvara versioonikontroll, st tuleb dokumenteerida erinevate versioonide erisused ja tagatud peab olema varasema versiooni taastamise võimalus (regulaarne andmevarundus). Tervikluse tagamiseks tuleb teostada regulaarset koodi ülevaatust sõltumatu arendaja poolt, st kontrolliv arendaja ei tohi ise olla koodi kirjutanu.
- Turvafunktsioonide rakendamine (soovituslik meede) – täiendavate turvafunktsioonide integreerimine rakendusprogrammide alla (nt pääsuõiguste reguleerimine, logimine). Rakendusele esitatavate täiendavate nõuetega tuleb arvestada juba planeerimisel ja arendamisel, sest nende hilisem realiseerimine on tavaliselt liiga suurte kulude tõttu võimatu.
- Tarkvaralitsentsid terminaliserveri keskkonnas (soovituslik meede) – terminaliserveril kasutatavad programmid peavad olema kooskõlas omandatud litsentsidega ning sobival viisil protokollitud.
- Tarkvaraarenduse põhjalik dokumenteerimine – tarkvaraarendus peab olema dokumenteeritud, võimaldamaks tuvastada turvalisusega seotud aspekte ja tagamaks tarkvara hooldamist. Dokumentatsioonis saab eristada projekti dokumentatsiooni (nõuded arendatavale süsteemile, hankeprotsessi dokumentatsioon, leping(ud), projekti protsessi dokumentatsioon) ja süsteemi dokumentatsiooni (süsteemi standard, süsteemi arhitektuur, liideste määratlused (ka organisatsiooni sisesed), kodeerimissuunised, koodi kommentaarid, konfiguratsiooni dokumendid, muudatuste dokumendid, kvaliteedi tagamise ja testimise dokumendid, paigalduse ja kasutuselevõtu dokumendid (sh administraatori juhend, kasutusjuhendid pea- ja lõppkasutajale).
- Projektimeeskonna koolitamine – projektimeeskonda (sh arendajaid) peaks koolitama enne arendustega alustamist järgmistel teemadel: nõuete analüüs, projektihaldus üldiselt ja konkreetselt süsteemiarenduse korral, riskihaldus tarkvaraarenduses, kvaliteedihaldus ja selle tagamine, tarkvaraarenduse mudelid ja meetodid, muudatuste haldus, infoturve, asutuse turvanõuded, turvalisuse aspektid konkreetsetes valdkondades.
- Tarkvara turvaline installeerimine - töökeskkonnas turvalise installeerimise eelduseks on loodud tarkvara funktsionaalsuse edukas testimine eelnevalt kindlaks määratud testimisprotseduuri järgi. Järgneb käitamisprotseduuri testimine usaldusväärsuse suhtes. Installeerimisprotsessi jaoks peab olema

installeerimisplaan, mis kirjeldab kõiki läbiviidavaid etappe ja vaatleb võimalikke veaallikaid või kõrvalekaldeid erinevate sihtsüsteemide vahel. Pärast installeerimist kontrollitakse dokumenteeritud testimisplaani järgi, kas paigaldus on õige ning antakse süsteem kasutusse.

- Turvalisust mõjutavate paikade ja uuenduste kiire paigaldamine – administraatorid peaksid erinevate infoallikate abil ennast kursis hoidma turvaaukudega seotud infoga. Uuendused ja paigad, mida paigaldatakse peavad pärinema usaldusväärsest allikast ja olema enne paigaldust testitud. Ebaõnnestunud paigalduse korral peab saama paigaldamisele eelnenud süsteemi olukorda taastada (varukoopiad). Süsteemis läbiviidavaid muudatusi tuleb dokumenteerida (kes tegi, millal tegi, mida tegi, miks tegi).
- Konfiguratsioonimuudatuste teostamine – kasutajaid tuleb teavitada muudatuste läbiviimisest. Süsteemi muudatused tuleb dokumenteerida, muudetavatest failidest tuleb teha varukoopiad, et tagada võime süsteemi muutmisele eelnenud olukorra taastamiseks.
- Regulaarne tervikluse kontroll (soovituslik meede) – tervikluse kontrolli tuleb teha regulaarselt, nt igal öösel. Jälgida tuleb, et kontrollsumma fail ja kontrolli programm on manipulatsioonide eest kaitstud. Peab olema määratletud käitumine, kui on avastatud tervikluse kadu.
- Valmisolek hädaolukorraks – tarkvaraarenduse jaoks tuleb kindlaks määrata üksikasjalik andmevarundus- ja taastamise kontseptsioon. Andmevarundus- ja taastamise kontseptsiooni tuleb testida ja testimise tulemused dokumenteerida. Tarkvaraarenduse juurde ei kuulu üksnes dokumendid, programmid ja süsteemid, vaid ka inimeste oskusteave. Kui teadmised, mis on vajalikud rakenduse väljatöötamiseks, hooldamiseks või edasiarenduseks, on koondunud ühe isiku kätte, võivad selle suure sõltuvusega kaasneda väga tõsised probleemid.
- Regulaarne andmevarundus – andmekadude vältimiseks tuleb andmeid regulaarselt varundada, rakendada võib täielikke andmevarundusi kombineeritult inkrementaalsetega. Enamikes arvutisüsteemides on võimalik selleks rakendada automaatselt toimivaid varundamisprotsesse. Tarvis on määratleda reeglid, milliseid andmeid tuleb erinevatel ajahetkedel varundada. Varukoopiad tuleb regulaarselt luua vähemalt nendest andmetest, mida ei ole võimalik muu info põhjal tuletada. Konfidentsiaalseid andmeid tuleks enne varundamist krüpteerida,

kusjuures tuleb arvestada, et andmeid peab olema võimalik dekrüpteerida pikema aja möödumisel. Kasutajad peavad olema teadlikud kehtivatest andmevarunduse põhimõtetest.

- Andmete taastamise harjutamine – andmete taastamist tuleb pisteliselt, kuid vähemalt pärast igat andmevarundusprotseduuri sisseviidud muudatust, testida. Harjutamise eesmärgiks on saada kinnitus, et andmeid on võimalik täies mahus taastada (Riigi Infosüsteemi Amet, 2017).

3.2.2.2 ISKE moodul B 1.11 Väljasttellimine

Järjest enam tellivad organisatsioonid tarkvaraarenduse ja sellega seotud teenused kas täielikult või osaliselt välistelt partneritelt. Sellisel juhul saab ISKEt rakendav organisatsioon toetuda ISKE moodulile B 1.11 Väljasttellimine. Väljasttellimine võib puudutada nii riist- ja tarkvara kasutamist kui ka teenuseid. Väljasttellimise korral tuleb suurt tähelepanu pöörata turvaaspektide ja lepinguliste kokkulepete kujundamisele teenuste tellija ja pakkuja vahel. Meetmete rakendamise seisukohast ei ole oluline kas teenuse pakkumine toimub tellija või teenuse pakkuja ruumides.

Peamised ohud väljasttellimise puhul on ISKEs kirjeldatud järgmiselt:

- Vääramatu jõud – teenusepakkuja laivõrgu (inglise k *Wide Area Networks*, WAN) tõrge võib kaasa tuua pikemaajalise (mõne päevase) teenuse katkestuse. Probleemiks on see juhul, kui laivõrgu kaudu ühendatud IT-süsteemides käitatakse ajatundlikke rakendusi.
- Organisatsioonilised puudused – kõige suurem hulk ohte väljasttellimise puhul varitseb just selles ohtude grupis. Väljasttellimine tuleb väga hoolikalt planeerida ja läbi mõelda enne vastava teenuse tellimist – nii sisenemine teenusesse (millised on nõuded teenusepakkujale ja tema poolt osutatavale teenusele, kuidas kontrollitakse nõuete järgimist jne) kui ka sellest väljumine (kas ja millistel tingimustel on võimalik teenuse tagasi toomine oma organisatsiooni).
- Inimvead – peamised ohud seotud info ja andmete konfidentsiaalsusega, nt valesti hallatud pääsuõigused, konfidentsiaalsed dokumendid ei ole piisavalt turvatud ning elektrooniline andmete liigutamine ei ole piisavalt turvatud. Inimvigade alla kuulub ka isiklike meili- ja dokumendi süsteemide (nt isiklik meiliboks, *Google Drive*, *Dropbox* jms) kasutamine tööalaste andmete käitlemiseks.

- Tehnilised rikked – ohtudena käsitletakse autentimise puudulikku korraldust (nt salasõnade haldus, krüpteerimata autentimisandmete salvestamine jms), krüpteerimisvahendi töökindluse tagamist, välise teenusepakkuja süsteemide riket, aga ka välise teenusepakkujaga suhtluse turvalisus (nt eraldi püsiliinid, VPN-ühendused, juurdepääsud kaughooldusele jms) (Riigi Infosüsteemi Amet, 2017).

Kirjeldatud ohtudest lähtuvalt on väljasttellimise jaoks välja töötatud meetmed, mille rakendamise abil on võimalik minimeerida tekkivat kahju. Järgnevalt tutvustab töö koostaja mooduli B 1.11 Väljasttellimine soovitatavaid meetmeid.

Väljasttellimise projekt koosneb erinevatest etappidest, mille realiseerimisel tuleks lähtuda järgmisest:

- Strateegia määramine – planeerimine peab olema põhjalik ja arvestama kõikvõimalike aspektidega: majanduslikud (tasuvusanalüüs), tehnilised (teostatavuse uuring, raamtingimused), organisatoorsed (paindlikkus, sõltuvus teistest, tulevikuplaanid). Lisaks tuleb kõikide eelnevate aspektide juures silmas pidada, et turvalisus oleks tagatud organisatsiooni poolt nõutud tasemel. Välise teenusepakkuja puhul tekib enamasti huvide konflikt, mida ei tohi alahinnata: ühest küljest peab teenuseid osutama kuluefektiivselt, et suurendada oma kasumit, teisalt ootab tellija maksimaalset kvaliteeti, paindlikkust ja kliendisõbralikkust. Kõik teenused, mida pakkujalt soovitakse, tuleb kirjeldada. Midagi ei tohi pidada iseenesest mõistetavaks, vältimaks hilisemaid vaidlusi ja olukorda, kus osutatav teenus ei vasta organisatsiooni vajadustele.
- Turvanõuete spetsifitseerimine – turvanõuded peavad olema püstitatud piisavalt konkreetselt, et potentsiaalsel pakkujal oleks võimalik organisatsioonile sobivat teenust pakkuda. Turvanõuded peavad hõlmama nii nõudeid välisele teenusepakkujale, kui tema poolt kasutatavatele tehnilistele vahenditele. Unustada ei tohi ka nõudeid oma organisatsioonile. Silmas tuleb pidada, et turvanõuete koostamine ei ole ühekordne tegevus, vaid pidev kahe partneri vaheline protsess tagamaks vajalikku turvalisusse taset organisatsioonis. Piisavaks võib turvanõudeid lugeda, kui teenusepakkujat kohustatakse täitma turbeastet, mis vastab ISKE turbeastmele.

- Teenuse tarnija valimine – meede esitab soovitused hanke korraldamisel kvalifitseerimisnõuete määratlemiseks ja nendel põhinevate kohustuste kirjeldamiseks tulevasele teenusepakkujale.
- Leping teenusepakkujaga – Antud meede keskendub teenustaseme lepingule (SLA) ja annab nimekirja, mille abil jälgida, et kõik oluline saaks SLAs kajastatud. Detailsuse aste sõltub alati konkreetsest tellitavast teenusest ja sellega seotud osadest.
- Turvakontseptsiooni koostamine – turvakontseptsiooni koostamine on tellija ja teenusepakkuja tihedas koostöös valmiv detailne dokument, mis muuhulgas sisaldab valmisolekut hädaolukorraks. Enamasti kestab kontseptsiooni koostamine kuni teenuse täieliku liikumiseni teenusepakkuja juurde. Silmas tuleb pidada, et erinevad turvakontseptsioonid (nt testimise-, side-, riist- ja tarkvara turvakontseptsioon) ei läheks üksteisega vastuollu. Võimaluse korral soovitab meede kasutada turvakontseptsiooni valideerimiseks kolmandat osapoolt.
- Migratsioon ehk üleviimine – meede rõhutab detailset üleviimise planeerimise ja erinevate detailide põhjaliku läbimõtlemise vajadust. Mõlema osapoole personal peab olema vajalikul tasemel ettevalmistatud. Määratletud peavad olema protseduurid, mis on ajutised ja kehtivad vaid üleviimise ajal, st peale üleviimise etappi nad kaotavad kehtivuse.
- Jooksev töö – tegevus, mis käivitub peale edukat teenuse üleviimist teenusepakkuja juurde. Peamiselt juhib meede tähelepanu, et olemas peab olema käitamise plaan ja teenusepakkujat tuleb regulaarselt kontrollida, et kokkulepitud nõuetest peetakse kinni. Võimalusel tuleks aeg-ajalt kaasata kontrolli funktsiooni täitma sõltumatu kolmas osapool. Lisaks tuleb tagada, et erinev dokumentatsioon oleks pidevalt uuenev ja ajakohane (Riigi Infosüsteemi Amet, 2017).

Eelnevad peatükid andsid ülevaate ASVS standardist ja tarkvaraarendusega enamseostuvatest moodulitest ISKE standardis, et MFNi kirjeldamisel toetuda kahes standardis väljatoodud parimale praktikale.

4. Avaliku sektori organisatsiooni mittefunktsionaalsete nõuete määratlemine tarkvaraarenduseks

Antud magistritöö eesmärk on välja töötada MFN, kus oleks arvestatud erinevate turvalisuse aspektidega, et tagada turvaline tarkvaraarendus avaliku sektori organisatsioonis. Turvalisuse peale tuleb mõelda juba tarkvara hankimise etapis, sest hilisem turvalisuse integreerimine tarkvaraarendusse on komplitseeritud (hanke lähteülesandest/seadusandlusest tulenevalt) ning ressursikulukas. Võimalik, et teatud juhtudel ei ole tagantjärele turvalisuse integreerimine isegi võimalik, nt loodud lahenduse arhitektuur ja/või disain ei võimalda seda.

Magistritöö koostamise käigus uuris töö koostaja erinevaid riigihangete registrist kättesaadavaid MFNi dokumente (Riigi Infosüsteemi Amet (edaspidi RIA), Registrite ja Infosüsteemide Keskus (edaspidi RIK), Keskkonnaministeeriumi Infotehnoloogiakeskus (edaspidi KEMIT)), lisaks olid töö koostajal kasutada mõned avalikult mittekättesaadavad avaliku sektori asutuste MFNid. Olemasolevate dokumentide analüüsile lisaks toetus töö koostaja peatükis 3. „Ülevaade standarditest“ tutvustatud standardites toodud erinevatele soovitustele. Uuringu tulemiks on MFNi kogum, mida saab kasutada alusena tarkvaraarenduse hankimisel MFNi kirjeldamisel Eesti avaliku sektori organisatsioonis. Vajadusel saab iga organisatsioon loodud MFNi täiendada oma organisatsiooni-, tarkvara- ja äriprotsesside spetsiifiliste nõuetega.

4.1 Üldised põhimõtted

MFNi loendi ees peab olema selgitus, kuidas tellija näeb ette nõuete rakendamist ja üldised juhised arenduspartnerile (pakkujale) etteantud nõuetega arvestamisel (sissejuhatav osa).

Magistritöö koostaja poolt pakutav sõnastus:

Dokument määratleb mittefunktsionaalsed nõuded (*organisatsiooni nimi*) ja tema haldusala asutuste (*juhul kui organisatsioonil on haldusala*) uutele infosüsteemidele ja nendega seotud dokumentatsioonile.

Nõuded on kohustuslikud ka olemasolevate infosüsteemide lisaarendustele ja versiooniuuendustele mahus, mis on lisaarenduse ja versiooniuuenduste käigus võimalik.
Kui mõnda nõuet ei ole pakkuja arvates võimalik või otstarbekas täita, tuleb selle mittetäitmise fakt ja põhjendus pakkumuses välja tuua.

Nõuded on kohustuslikuks täitmiseks kõikidele tulemitele.

Kõik erikokkulepped fikseeritakse tellijaga kirjalikku taasesitamist võimaldaval viisil.

Ülaltoodud sõnastus peaks tagama, et alates MFNi kehtestamisest on nad kohustuslikud antud organisatsiooni kõikidele tarkvaraarendustele, sh olemasoleva tarkvara muutmisele/täiendamisele ja uutele versioonidele üleminekul. Teatud olukordades ei pruugi olla võimalik või vajalik osade nõuete rakendamine, seetõttu tuleb nõuete kehtestamisel jätta võimalus ka põhjendatud juhtudel nõuete mitterakendamiseks. Lõplik otsus nõuete rakendamise kohta peab jääma avaliku sektori organisatsioonile endale.

MFNi igale nõudele tuleks lisada ka nõude eest vastutaja ja etapi või tegevuse nimetus, millal nõuet tellija poolt valideeritakse.

4.2 Vastavus standarditele ja seadusandlusele

Käesolev osa kirjeldab erinevaid standardeid ja seadusandlust, millega tuleb arvestada avalikus sektoris tarkvaraarenduste teostamisel. Nõudes peab sisalduma standardi või seaduse (määruse jms) nimetus ning avalikult kättesaadav viide vastavale materjalile. Kui on asjakohane, siis ka täpsustav informatsioon, millisele osale täpselt tuleb tähelepanu pöörata.

Kindlasti tuleb antud alapeatükis välja tuua, kui arenduse valmimisel soovitakse infosüsteemi hinnata mingile standardile vastavalt (nt vastavalt Infoturbe

ühiskriteeriumitele, inglise k *Common Criteria*). Juhul, kui loodavat süsteemi sertifitseeritakse valmimisel, tuleb arenduse algusest peale planeerida kõik vastavalt standardi nõuetele.

Magistritöö koostaja poolt pakutav nõuete loetelu ja sõnastus on järgnev:

- **Nõue 1:** Lahendus peab vastama dokumendis „Riigi infosüsteemi koosvõime raamistik“ kirjeldatud põhimõtetele.
  Märkus: Raamistik on leitav:
  https://www.mkm.ee/sites/default/files/riigi_it_koosvoime_raamistik.pdf
- **Nõue 2:** Lahendus peab vastama dokumendis „Veebide koosvõime raamistik“ kirjeldatud põhimõtetele.
  Märkus: Raamistik on leitav:
  https://www.mkm.ee/sites/default/files/veebide_raamistik.pdf
- **Nõue 3:** Lahendus peab vastama „Eesti avaliku teabe masinloetava avalikustamise roheline raamat“ sõnastatud põhimõtetele ja nõuetele.
  Märkus: Dokument on leitav:
  https://opendata.riik.ee/sites/default/files/manuals/avaliku-teabe-masinloetava-avalikustamise-roheline-raamat-20141125_0.odt
- **Nõue 4:** Lahendus peab vastama „Aadressiandmete süsteem“ määruses kehtestatud aadressiandmete nõuetele.
  Märkus: Määrus on leitav https://www.riigiteataja.ee/akt/113102015002
- **Nõue 5:** Lahenduses tuleb tegevusalade määramiseks kasutada Eesti Majanduse Tegevusalade Klassifikaatorit (EMTAK).
  Märkus: Klassifikaator on leitav:
  https://emtak.rik.ee/EMTAK/pages/klassifikaatorOtsing.jspx
- **Nõue 6:** Lahendus peab vastama „Avaliku teabe seaduses“ kehtestatud teabe avalikustamise ja juurdepääsu võimaldamise nõuetele.
  Märkus: Seadus on leitav:
  https://www.riigiteataja.ee/akt/114032011019?leiaKehtiv
- **Nõue 7:** Lahendus peab vastama „Riigi infosüsteemi haldussüsteem“ (edaspidi RIHA) esitatud nõuetele ja reeglitele.
  Märkus: Määrus on leitav: https://www.riigiteataja.ee/akt/12933746?leiaKehtiv

- **Nõue 8:** Lahendus peab vastama „Infosüsteemide turvameetmete süsteem“ esitatud nõuetele.

  Märkused: Määrus on leitav: https://www.riigiteataja.ee/akt/13125331?leiaKehtiv Täpne rakendamise ulatus lepitakse kokku lähtuvalt konkreetsest infosüsteemist.
- **Nõue 9:** Lahenduses peab isikuandmete töötlemine toimuma vastavalt „Isikuandmete kaitse seadus“ kehtestatud nõuetele.

  Märkus: Seadus on leitav: https://www.riigiteataja.ee/akt/12909389?leiaKehtiv
- **Nõue 10:** Lahenduses peavad X-tee teenused olema realiseeritud vastavalt RIA poolt kirjeldatud nõuetele.

  Märkus: Nõuded on leitavad: https://www.ria.ee/ee/xtee-juhendid.html
- **Nõue 11:** ID-kaardi allkirjastamislahenduse kasutamisel eelistatakse DigiDoc veebiteenuse kasutamist.

  Märkus: Dokumentatsioonis peab olema väljatoodud DigiDoc-teekide versioonid ja kasutuskohad.
- **Nõue 12:** Kuupäeva ja aja (kuupäev, kellaaeg, ajaintervall) talletamisel teksti kujul tuleb aluseks võtta ISO 8601 standardis kirjeldatud põhimõtted.

  Märkus: Põhimõtted on leitavad: https://en.wikipedia.org/wiki/ISO_8601 või http://www.w3.org/TR/NOTE-datetime
- **Nõue 13[6]:** Veebirakenduse kasutajaliides peab vastama vähemalt WCAG 2.0 tasemele AA.

  Märkus: Soovituste kogum on leitav: http://www.w3.org/TR/WCAG20/
- **Nõue 14[7]:** Veebipõhine kasutajaliides peab ühilduma HTML (*lisada soovitud versiooni nr, nt HTML 5*) ja CSS (*lisada soovitud versiooni nr, nt CSS 3*) standarditega.

  Märkus: Valideerimiseks kasutatav validaator on leitav http://validator.w3.org/

Antud nõuete loend on koond üldistest avaliku sektori tarkvaraarendusi reguleerivatest normdokumentidest, lisada tuleb organisatsiooni spetsiifikast lähtuvad nõuded, nt ISKE puhul tuleks kindlasti välja tuua loodava infosüsteemi turvaosaklasside hinnang, et arendaja (pakkuja) saaks sellega arvestada juba pakkumuse tegemisel.

[6] Nõue on püstitatud „Veebide koosvõime raamistikus“ ja on siin välja toodud, sest kajastub kõikides avaliku sektori organisatsioonide analüüsitud MFN dokumentides.
[7] Nõue on püstitatud „Veebide koosvõime raamistikus“ ja on siin välja toodud, sest kajastub kõikides avaliku sektori organisatsioonide analüüsitud MFN dokumentides.

Otstarbekas on välja tuua viide organisatsiooni IT profiilile. IT profiil võib olla osa MFNte dokumendist või eraldiseisev dokument, millele MFN viitab ja mis lisatakse tarkvaraarenduse hanke dokumentatsioonile.

Avalikus sektoris leiab järjest enam kasutust OWASPi poolt koostatud 10 enamlevinud turvaohu vastu testimise nõude lisamine MFNi hulka. See aitab ühtlasi kaasa turvanõuetele vastava infosüsteemi loomisele, sest esitades tellijana nõude, et loodav lahendus peab edukalt läbima turvaohtude testi, tuleb arendajal juba arendama asumisel järgida ASVS standardis toodud soovitusi turvaohtude vältimiseks. Kindlasti ei piisa vaid nõude esitamisest, organisatsioonis peab olema ka kompetents testitulemuste hindamiseks ja vajalike muudatuste kommunikeerimiseks.

ASVS standardi erinevate nõuete sisseviimine oli magistritöös tegemise käigus vaadeldud dokumentides väljatoodud kahes MFNi dokumendis. Üks MFN dokument kirjeldas ASVS taseme, millele loodav tarkvaraarendus peab vastama. Lisainformatsioonina oli märgitud, et tellija võib vastavust testida ning arendaja peab olema valmis lähtuvalt testitulemustest arendatud tarkvara täiendama. Teisel juhul oli nõue kirjeldatud väga üldisel tasemel. Nõue oli järgnev: „Turvatestide läbiviimisel peab rakendama OWASP ASVS metoodikat (OWASP *Application Security Verification Standard* 2014. https://www.owasp.org/index.php/ASVS) (Keskkonnaministeeriumi Infotehnoloogiakeskus, 2016).“

Tulenevalt asjaolust, et ISKE ei ole sobilik riigisaladust töötlevate andmekogude turbeks (nt militaarvaldkond), võiks just nendes organisatsioonides kaaluda ASVS standardi järgimist, sest ASVS on mh sobilik ka militaarvaldkonnale.

4.3 Arhitektuur

Arhitektuuri nõuded tuleb püstitada, et tagada infosüsteemi tellija soovidele vastav tarkvaraarendus. Arhitektuur on alus, millele infosüsteem luuakse.

Magistritöö koostaja poolt väljapakutud arhitektuuriga seotud nõuete loetelu on järgmine:

- **Nõue 15:** Realiseeritud lahendus peab tellija poolt kinnitatud arhitektuurile vastavalt töötama tellija poolt nõutud funktsionaalsete ja mittefunktsionaalsete nõuete ulatuses.
- **Nõue 16:** Kõik infosüsteemis kasutatavad komponendid peavad olema tuvastatavad, põhjendatud ja dokumenteeritud.
- **Nõue 17:** Kõik välised komponendid (nt välised süsteemid, teegid), millele toetudes loodav lahendus töötab, peavad olema tuvastatavad ja dokumenteeritud.
- **Nõue 18:** Liidesed väliste süsteemidega peavad olema standardsed (allutatud sarnastele reeglitele) ja liidestamise detailid peavad olema dokumenteeritud.
- **Nõue 19:** Rakenduse liidesed peavad olema tõrkekindlad.
  Märkused: Lõppkasutaja peab saama jätkata rakenduse kasutamist ulatuses, mis on protsessiliselt võimalik.
  Süsteem peab tõrke korral võimalikult lühikese aja jooksul väljastama asjakohase veateate.
- **Nõue 20:** Infosüsteemide platvormid (nt rakendusserver, andmebaas) ja topoloogia peavad olema tellijaga kooskõlastatud enne reaalse tarkvaraarenduse algust.
- **Nõue 21:** Kõikide arendamisel kasutatavate komponentide (rakenduse, andmebaasi, kolmanda osapoole omad) eluea lõpp (inglise k *End-of-Life*, EOL) ei tohi teadaolevalt olla vähem kui 2 aastat.
- **Nõue 22:** Infosüsteemi ülesehituses peaks kasutama kolmekihilist arhitektuuri: andmekiht, kontrollerkiht (äriloogika) ja esitluskiht.
- **Nõue 23:** Infosüsteem peab olema üles ehitatud nii, et eessüsteemid (inglise k *front end*) ja tagasüsteemid (inglise k *back end*) on arhitektuuriliselt selgelt lahutatud.
- **Nõue 24:** Andmebaasid ja rakendused peavad kasutama UTF-8 kodeeringut.
- **Nõue 25:** Andmebaasi objektide nimetused peavad olema sisulised ja andma aimu nende otstarbest
- **Nõue 26:** Rakendus peaks olema 64-bitine[8].
- **Nõue 27:** Lahendused peavad olema projekteeritud laiendatavana ja edasi arendatavana.

[8] Nõuete kirjeldamisel on eelkõige lähtutud põhimõttest, et nõudeid kasutatakse uue loodava infosüsteemi puhul. Magistritöö koostaja hinnangul ei ole käesoleval ajahetkel enam 32-bitise rakenduse loomine otstarbekas.

- **Nõue 28:** Rakendust peab saama liigutada ilma ümberprogrammeerimiseta erinevate domeenide ja domeenisaitide vahel.
- **Nõue 29:** Rakendusel peab olema haldusliides.
  Märkus: Haldustoimingute tegemine otse andmebaasis peab olema viidud miinimumini.
- **Nõue 30:** Kui rakendused saadavad e-kirju, siis peavad nad selleks kasutama välist e-mailiserverit.
- **Nõue 31:** Rakendus peab suutma kasutada keskkonnamuutujaid (nt kuu, päev, serveri nimi).
- **Nõue 32:** Keskkonnapõhiseid muutujaid peab saama seadistada konfiguratsioonifailist.
- **Nõue 33:** Ebaõnnestunud logimiste arvu peab saama piirata ajaühiku kohta ja IP-aadressist lähtuvalt.
  Märkus: Muudatusi (logimiste arv, ajaühik) peab saama seadistada konfiguratsioonifailis.
- **Nõue 34:** Klientrakendus ei tohi teostada otsepöördust andmebaasi poole.
  Märkus: Võib kasutada rakendusserverit või adapterit.
- **Nõue 35:** Rakenduse failid, mis ei tohi olla kasutajale nähtavad, peavad olema kaitstud (rakenduse kasutajale mittekättesaadavates) kaustades.
- **Nõue 36:** Sorteerimisreeglistik peab vastama eesti tähestikule, tõstutundlikkus peab olema väljalülitatud. Accent peab olema sisselülitatud
- .
- **Nõue 37:** Kuvatava lehekülje ettevalmistamine serveris normaalkoormusel peab jääma alla … (*organisatsiooni poolt soovitud aeg*). Erandiks on PDF formaadis dokumendi, aruannete ja otsingutulemuste genereerimine, mille maksimaalne genereerimisaeg võib olla … (*organisatsiooni poolt soovitud aeg*).
- **Nõue 38:** Loodav lahendus peab võimaldama serveri poolt lõppkasutajale tagastatavate andmeobjektide arvu piirangut ja/või mahukate andmekomplektide lehekülgjaotust (inglise k *pagination*).

Avaliku sektori organisatsioonidel on järjest enam olemas asutuspõhine autentimislahendus. Selle olemasolu korral peaks vastava lahenduse kasutamise nõue sisalduma arhitektuuri nõuetes. Üks võimalikke sõnastusi:

- **Nõue 39:** Kui rakenduse toimimiseks on vajalik kasutajate, rollide ja õiguste haldamine, tuleb selleks kasutada (*organisatsiooni nimi*) vastavat lahendust (*autentimislahenduse nimi*).

Olenevalt tellija soovist võib olla põhjendatud ka nt failisüsteemis ühte kausta salvestatavate failide arvu määratlemine, võimalik sõnastus nõudena:

- **Nõue 40:** Ühes kaustas salvestatavate failide arv ei tohi olla suurem kui 10 000.

4.4 Turvalisus, sh infoturve

ASVS standard annab hulgaliselt erinevaid turvalise tarkvaraarendusega seonduvaid soovitusi nt autentimise, seansi halduse ja ligipääsu kohta. Erinevaid turvalisust tagavaid meetmeid kirjeldab ka ISKE.

Lisaks on antud peatüki nõuete kirjeldamisel toetutud erinevate avaliku sektori asutuste MFNi dokumentidele.

Magistritöö koostaja poolt väljapakutud turvalisust käsitlevate nõuete loetelu on järgmine:

- **Nõue 41:** Välistele kasutajatele mõeldud rakendused peavad võimaldama sisselogimist ainult ID-kaardi, mobiil-ID ja SmartIDga.
  Märkus: Kasutajanime ja parooliga autentimine ei ole lubatud.
- **Nõue 42:** Rakenduse autentimise jõustamine peab toimuma serveri poolel.
- **Nõue 43:** Ebaõnnestunud autentimine peab lõppema viisil, mis ei jäta ründajale võimalust rakendusse sisse tungida.
- **Nõue 44:** Autentimist võimaldav informatsioon (nt autentimissaladused, API võtmed, salasõnad) ei tohi sisalduda lähtekoodis ega võrgus olevates lähtekoodi repositooriumites.
- **Nõue 45:** Rakenduse kasutajale kuvatavad URLid ei tohi sisaldada isikuandmeid.
- **Nõue 46:** Rakendusel peab olema konfigureeritav kasutajaseansi aegumise aeg.
- **Nõue 47:** Süsteemist väljumine peab toimuma kasutajale üheselt arusaadaval ja turvalisel viisil. Seansist väljumine toimub kahel viisil: kasutaja seansi lõpetab

süsteem, sest seanss on olnud pikem kui süsteemile seadistatud vaikimisi limiit või kasutaja lõpetab seansi omal soovil.

- **Nõue 48:** Kasutajal peab olema igal süsteemi kasutamise ajahetkel võimalik seanss omal soovil lõpetada.
- **Nõue 49:** Iga eduka süsteemi sisselogimise (autentimise) korral tuleb alati luua unikaalne seansi identifikaator (inglise k *session ID*).
- **Nõue 50:** Seansi identifikaator ei tohi kajastuda ressursilokaatoris (URLis), veateadetes ega logides.
- **Nõue 51:** Seansi identifikaator peab olema piisava pikkusega, juhuslik ja unikaalne kogu aktiivse seansi jooksul.
- **Nõue 52:** Süsteem ei tohi võimaldada samaaegset konkurentset seanssi.
- **Nõue 53:** Süsteem ei tohi võimaldada kasutajale ligipääsu süsteemi toimimise informatsioonile (nt failide täisnimed).
- **Nõue 54:** Rakendus ja selle komponendid peavad võimaldama keskkondade lahusust (nt arendus-, test- ja toodangu keskkond).
- **Nõue 55:** Andmebaasis olevate rakenduste kontod peavad omama ainult minimaalselt rakenduse tööks vajalikke õigusi.
- **Nõue 56:** Krüptoalgoritmide ja räsifunktsioonide kasutamisel tuleb järgida RIA veebilehel avaldatud krüptograafiliste algoritmide elutsükli uuringu värskeimas versioonis toodud soovitusi ja põhimõtteid.

  Märkus: Uuring on leitav: https://www.ria.ee/ee/kruptouuringud.html
- **Nõue 57:** Rakenduses peab olema tagatud võimekus välja vahetada aegunud ja ebaturvalisi krüptoalgoritme.

  Märkus: Krüptoalgoritmide väljavahetamine peab olema dokumentatsioonis kirjeldatud.
- **Nõue 58:** Kõik paroolid ja salasõnad peab rakendus salvestama kas räsituna ja soolatuna või krüpteeritud kujul.

  Märkus: Krüpteerimise kasutamisel peab protseduur olema kirjeldatud ISKE meetmes M 4.401 „Konfidentsiaalsete andmete kaitse veebirakenduses“ toodud nõuetest lähtuvalt.
- **Nõue 59:** Kõik võtmed ja salasõnad peavad olema asendatavad ja nad tuleb toodangu keskkonna installatsiooni ajal luua või asendada.

- **Nõue 60:** Kaitsmata avalik võrguliiklus ei ole lubatud, avalik võrguliiklus peab olema krüpteeritud.
- **Nõue 61:** Rakendusse ja andmetele tohib olla ligipääs ainult dokumenteeritud ja kirjeldatud teid mööda ja dokumenteeritud autentimisprotseduure kasutades.
- **Nõue 62:** Salvestunud andmed peavad olema taastatavad.
- **Nõue 63:** Rakendus tohib kasutada ainult brauserikooke (inglise k *session cookie*), muud koogid on keelatud.
- **Nõue 64:** Rakendus ei tohi teostada X-tee päringut otse kasutaja arvutist.
- **Nõue 65:** Rakendusserver ja andmebaasiserver peavad olema võimelised töötama eraldi serveritel.
- **Nõue 66:** Veebirakenduse kõik viited failidele ja kataloogidele peavad olema ilma absoluutse failiteeta.
- **Nõue 67:** Kui rakenduse poolt töödeldavate andmete ISKE konfidentsiaalsuse klass on 2 või kõrgem, peab rakendus sisaldama lahendust, mis suudab toodangu andmetest genereerida testandmed, mis ei sisalda konfidentsiaalset informatsiooni.
- **Nõue 68:** Välistele kasutajatele mõeldud veebilehega rakendused, mille ISKE turvaklass on M või L, peavad olema kaitstud keelatud päringute eest.
- **Nõue 69:** Kui rakendus võimaldab mitteautenditud kasutajal edastada andmeid, tuleb need andmed puhastada XSS filtriga.

Krüptograafiliste võtmete halduse (eeldus krüptograafiliste turvamehhanismide kasutamiseks) kohta annab juhiseid ISKE meede M 2.46 „Krüpteerimise õige korraldus“. Meede sisaldab juhiseid võtmete loomise, funktsioonide eraldamise, jagamise (sh väljavahetamise), installeerimise ja salvestamise, arhiveerimise, vahetamise ja hävitamise kohta.

Krüptograafiliste võtmete haldamise poliitika olemasolu, kehtestamise ja jõustamise nõue sisaldub ka ASVS standardi kontrollgrupis „Krüptograafiliste turvameetmete olemasolu“, nõue nr 7.9.

Lisaks annab ISKE meede M 2.164 „Sobiva krüptoprotseduuri valimine“ juhised, kuidas langetada oma organisatsiooni spetsiifikast lähtuv sobilik valik.

Kõrge tervikluse turvaosaklassiga (T3) rakenduste puhul võib olla vajadus määratleda ka ajatempli kasutamine. Üks võimalikke lahendusi on infoturbefirma Guardtime

poolt väljatöötatud ajatemplisüsteem, mis on krüptograafiliste meetoditega tekitatud andmekogum, mille abil tõestatakse, et digitaalne dokument eksisteeris teatud ajahetkel (Buldas, Hanson, Krasnosjolov, Laur, & Veldre, 2011-2017).

Järjest enam leiavad kasutust ka pilveteenused. Vaadeldud MFNides ei sisaldunud pilveteenusele esitatavaid nõuded, aga MFNi koostamisel avaliku sektoris ei tohiks enam välistada pilveteenust kasutavaid lahendusi. Pilveteenus peaks olema MFNi dokumendis kajastatud minimaalselt allpool toodud nõudena:

- **Nõue 70:** Loodava lahenduse realiseerimiseks võib kasutada riigipilve tehnoloogiaid, järgima peab RIA poolt koostatud dokumenti „Nõuded riigipilvele“.
  Märkused: Dokument „Nõuded riigipilvele“ on leitav: https://www.ria.ee/riigiarhitektuur/wiki/lib/exe/fetch.php?media=an:riigipilve_alusnouded.pdf
  Informatsioon riigipilve kohta on leitav: http://riigipilv.ee/

Riigipilve teenused on arendatud ISKE nõudeid järgides, seega on tagatud ka avaliku sektori organisatsioonile kohustuslik ISKE rakendatus.

4.5 Lähtekood

Lähtekoodi kohta on analüüsitud MNFides väga erineva detailsusega nõudeid lähtuvalt organisatsiooni spetsiifikast. Leidub ka ühiste joontega nõudeid, mida toetavad töös vaadeldud standardid.

Standarditest ISKE juhib peamiselt tähelepanu sellele, mis ei tohiks sisalduda koodis ning kus koodi hoiustama peaks – ISKE erinevad koodidega seotud ohud.

ASVS standard keskendub pigem kahjuliku koodi vältimise protseduuride kirjeldamisele kontrollgrupis nr 5 „Pahatahtliku sisendi käsitlemine“. Kontrollgrupp nr 5 sisaldab juhiseid vältimaks rakendust kahjustada võivate sisendite sattumise rakendusse.

Magistritöö koostaja poolt väljapakutud lähtekoodi puudutavate nõuete loetelu on järgmine:

- **Nõue 71:** Rakenduse lähtekoodi hoitakse (*organisatsiooni nimi*) repositooriumis.
- **Nõue 72:** Rakenduse lähtekood peab olema 30% ulatuses koodi mahust kommenteeritud detailsusega, mis võimaldab erialast ettevalmistust omaval tarkvaraarendajal teostada süsteemi edasiarendust.
- **Nõue 73:** Rakenduse lähtekood peab olema kommenteeritud inglise või eesti keeles.
- **Nõue 74:** Lähtekoodi kommentaarid peavad olema selged, arusaadavad ja sisuliselt kirjeldama vastavat koodi, mille juures nad on.
- **Nõue 75:** Muutujate, tüüpide ja funktsioonide nimed peavad olema sisulised ja andma aimu nende otstarbest.
- **Nõue 76:** Koodis kasutatavad konstandid ja lühendid tuleb kirjutada suurtähtedega.
- **Nõue 77:** Üleantav kood ei tohi sisaldada paroole (ka siis kui need on koodist väljakommenteeritud), mida kasutati arenduse käigus. Kõik arenduse käigus kasutatud paroolid tuleb asendada fraasiga „<*password*>“.
- **Nõue 78:** Koodis kasutatavaid konstante ei tohi selle kasutamise kohta väärtusena püsikodeerida – need tuleb defineerida muutujatena ja kasutada läbi nende.
- **Nõue 79:** Koodis defineeritud andmetüübid peavad olema nimetava käände ainsuses. Andmemassiivid tuleb nimetada nimetava käände mitmuses (nt *collectionid*, *arrayd*).
- **Nõue 80:** Lähtekood peab olema kompileeritav tellija poolt määratletud kompilaatori(te)ga.

  Märkus: Tellija poolt aktsepteeritud kompilaator(id) on … (*loetelu kompilaatoritest ja viide, kus sellega tutvuda saab*)
- **Nõue 81:** Lähtekood peab olema valideeritav tellija poolt määratud validaatori(te)ga.

  Märkus: Tellija poolt aktsepteeritud validaator(id) on … (*loetelu validaatoritest ja viide, kus sellega tutvuda saab*)
- **Nõue 82:** Kasutuses mitteolev kood tuleb rakendusest eemaldada.

Lähtekoodi nõuetes võiks kajastuda ka organisatsiooni arenduste spetsiifikast lähtuvad programmeerimiskeelte nõuded. Neid ei ole otstarbekas määratleda antud magistritöö käigus, sest igal organisatsioonil on oma infosüsteemidest lähtuv spetsiifika.

Alljärgnevalt on toodud mõned näited kirjeldamaks, mida magistritöö koostaja eelpool kirjapanduga silmas peab:

- **Nõue 83:** Python rakenduse kood peab olema kirjutatud vastavalt „Style Guide for Python Code“ põhimõtetele.
  Märkus: Põhimõtete dokument on leitav: https://www.python.org/dev/peps/pep-0008/
- **Nõue 84:** JAVA rakenduse kood peab olema kirjutatud vastavalt „SUN Java Code Convention“ põhimõtetele.
  Märkus: Põhimõtete dokument on leitav:
  http://java.sun.com/docs/codeconv/html/CodeConvTOC.doc.html
  Lisainfo JAVA kohta võib leida: http://geosoft.no/development/javastyle.html

4.6 Andmebaas

Antud alateema üldiste nõuete kirjeldamine on üks keerulisemaid, sest enamasti tekivad nõuded andmebaasile ikkagi organisatsioonis kasutusel oleva põhjalt. Mõned üldised nõuded, mis peaksid tagama turvalise tarkvaraarenduse tulemi on siiski antud alapeatükis püstitatud.

Magistritöö koostaja poolt väljapakutud andmebaasi nõuded on järgnevad:

- **Nõue 85:** Andmebaasi tabelid ja väljad peavad olema kommenteeritud.
- **Nõue 86:** Andmebaasi väljapikkused peavad olema väljendatud sümbolites (tähemärkide arv).
- **Nõue 87:** Andmebaasi objektide nimetused peavad olema inglise keeles.
- **Nõue 88:** Andmebaasi objektide nimetused tohivad sisaldada ladina tähestiku väiketähti „a-z“, numbreid „0-9“ ning alakriipsu „_“.
  Märkus: Andmebaasi objekti nimetus ei tohi alata numbriga
- **Nõue 89:** Andmebaasi objekti nimed peavad olema semantilised.

- **Nõue 90:** Igas andmebaasi tabelis peab olema defineeritud üks primaarvõti (inglise k *Primary Key*).
- **Nõue 91:** Ühest andmetabelist teise viitamisel tuleb kasutada võõrvõtit (inglise k *Foreign Key*). Võõrvõtme nimi peab seostuma tabeli ja väljaga, millele see viitab.
- **Nõue 92:** Kõik võõrvõtmed peavad olema indekseeritud.
- **Nõue 93:** Kui rakenduse versioon nõuab andmebaasi muudatusi, peavad üleantava koodiga kaasas olema andmebaasi paigalduse skriptid.

Toodud loetelu ei saa käsitleda lõplikuna ja kindlasti on põhjendatud organisatsioonispetsiifiliste nõuete lisamine. Küll aga annab antud loetelu teatava lähtekoha ja on terviku seisukohalt oluline komponent.

4.7 Logimine ja monitooring

Mida, kuidas ja mis ulatuses logida ja monitoorida oleneb suuresti organisatsioonist ning kasutusel olevatest infosüsteemidest.

ASVS standardil on eraldi kontrollgrupp „Vigade haldus ja logimine“, mis käsitleb logimisega seotud nõudeid.

ISKEs on logimise teemalisi meetmeid rohkelt, erinevad meetmed keskenduvad konkreetse tarkvara või lahenduse (nt meede M 4.81 „Võrgutoimingute audit ja logimine“ või meede M 4.270 „SAP logimine“), aga ka üldise IT-süsteemide logimise põhimõtete tutvustamisele (meede M 2.500 „IT-süsteemide logimine“). Palju kasulikke suuniseid annab ka meede, mis käsitleb andmekaitset logimisprotseduurides (meede M 2.110 „Andmeprivaatsuse suunised logimisprotseduurides“), meede sisaldab ka monitooringut. ISKE kohaselt haldustoimingute logimine vastab süsteemi monitooringule ja kasutaja toimingute logimine aitab kaasa protseduuride monitooringule (Riigi Infosüsteemi Amet, 2017).

Magistritöö koostaja poolt väljapakutud logimise ja monitooringu nõuded on järgnevad:

- **Nõue 94:** Süsteemi muudatused ning rakenduse ja kasutajate tegevused logitakse seostatuna muudatuse/tegevuse teostanud konkreetse füüsilise isiku ja tema rolliga.
- **Nõue 95:** Logid peavad olema kirjutatud inglise keeles.
- **Nõue 96:** Logimine peab olema konfigureeritav ning kasutada tuleb standardseid logiformaate, et võimaldada hilisem logianalüsaatorite kasutamine.
- **Nõue 97:** Andmebaasi logidest saadetakse reaalajas koopia failisüsteemi logisse, mis peab sisaldama ka logimisfunktsionaalsuse aktiveerimise ja deaktiveerimise infot (nt aeg, kasutaja jm).
- **Nõue 98:** Failisüsteemi logimisel peavad logid olema katalogiseeritud, üldlevinud faililaiendiga (nt .log, .txt, .xml) ja roteeruvad.
- **Nõue 99:** Rakendusserveri standardväljundisse logimine on keelatud.
- **Nõue 100:** Rakendus ei tohi väljastada veateateid või aktiivsusjälgi, mis sisaldavad seansi identifikaatorit või isikuandmeid.
- **Nõue 101:** Turvalist sisselogimise mehhanismid peavad olema võimelised logima nii õnnestunud, kui ka ebaõnnestunud sisselogimise katseid.
- **Nõue 102:** Logid peavad olema kaitstud lubamatu ligipääsu ja muutmise eest.
- **Nõue 103:** Kui rakenduse ISKE konfidentsiaalsuse turvaosaklass on 2 või kõrgem (S2, S3) ja/või tervikluse turvaosaklass on 2 või kõrgem (T2, T3), peab rakendus logima andmete loomist, muutmist (sh kustutamist) ja vaatamist.

  Märkus: Turvaosaklasside S3 ja T3 korral peab rakendus logima ka administraatorite ja haldurite poolt tehtavaid andmete muudatusi (sh otse andmebaasis) ja vaatamisi.

Rakenduse puhul on otstarbekas määratleda kas monitooring on ainult passiivne, ainult aktiivne või monitooritakse rakendust mõlemal viisil. Rakenduse passiivse monitooringu puhul kontrollib rakendus ennast ise ja annab monitooringu süsteemile vigadest teada ning eesmärk on kasutajate toimingute õnnestumise ja ebaõnnestumise fikseerimine. Rakenduse aktiivse monitooringu korral kontrollib monitooringu süsteem rakendust ning eesmärk on rakenduse toimimiseks tähtsate komponentide staatuse kontroll, et tagada rakenduse töökindlus.

Monitooringu puhul on otstarbekas tuua välja ka organisatsiooni logimise tasandid (ühe nõudena). Näide logimise tasandite nõudest:

- **Nõue 104:** Logimise tasandid on:
  DEBUG – arendamise etapis süsteemi olekut kirjeldav informatsioon (kasutatakse ainult arenduskeskkonnas)
  INFO – kasutaja päringute informatsioon, kasutaja infoteated (nt „Andmed salvestatud“, „Andmed muudetud“)
  WARNING – kasutajale kuvatav valest sisendist tekkiv oodatud viga (nt „Vigaselt sisestatud andmed“)
  ERROR – süsteemi vead, mis tekivad kasutaja sisendist (nt vigase andmebaasipäringu tekkimine)
  FATAL – rakenduse toimimise kriitilised vead, mis takistavad rakenduse tavapärast toimimist (nt vigane konfiguratsioon) (Riigi Infosüsteemi Amet, 2015).

4.8 Konfiguratsioon

Teatud konfiguratsiooniga seotud nõuete määratlemist soovitavad mõlemad magistritöös uuritud standardid. ISKE sisaldab erinevaid meetmeid ja soovitusi konfiguratsiooni kohta, neist kaks üldisemat määratlevad vastavalt konfiguratsiooni dokumenteerimise (meede M 2.25 „Süsteemi konfiguratsiooni dokumenteerimine“) ja turvalise aluskonfiguratsiooni (meede M 4.237 „IT-süsteemi turvaline aluskonfiguratsioon“) põhimõtted. ASVS sisaldab kahte kontrollgruppi, mis määratlevad konfiguratsiooniga seotud nõudeid: „HTTP turvaline konfiguratsioon“ ja „Konfiguratsioon“.

Vaadeldud avaliku sektori asutuste MFNi sisaldas eraldi konfiguratsiooni peatükki ainult KEMITi MFNi dokument, samas erinevad nõuded konfiguratsioonile sisaldusid kõikides MFNi dokumentides.

Magistritöö koostaja poolt väljapakutavad konfiguratsiooni käsitlevad nõuded on järgnevad:

- **Nõue 105:** Kõik komponendid peavad olema ajakohase turvakonfiguratsiooniga ja versiooniga.
- **Nõue 106:** Komponentide vaheline (nt rakendusserveri ja andmebaasi serveri) suhtlus peab olema krüpteeritud.

- **Nõue 107:** Komponentide vahelise ühenduse jaoks tuleb kasutada minimaalselt vajalike õigustega kontot.
- **Nõue 108:** Rakendus peab olema aedikkäideldud, konteinerdatud või muul viisil isoleeritud, et takistada ründajal rakenduse kasutamist teise rakenduse ründamiseks.
- **Nõue 109:** Rakenduse konfiguratsiooniparameetrid ei tohi muutmisel vajada uuesti kokku kompileerimist.

  Märkused: Eraldi konfiguratsioonifail võib olla kasutusel logimise ning arendaja ja administraatori vastutusala parameetrite jaoks.
- **Nõue 110:** Konfiguratsiooniparameetrite nimed peavad olema sisulised.

  Märkus: Kui sisulist nime ei ole võimalik kasutada, siis peab kasutatava nime kõrval olema seletus.
- **Nõue 111:** Konfiguratsioonifailid peavad olema rakendusserveri tüübile vastavalt vaikimisi kaitstud.
- **Nõue 112:** Samasisulisi konfiguratsiooni parameetreid ei tohi korduvalt kasutada, lubatud on kirjeldada ainult üks kord.

4.9 Kasutajaliides

Kasutajaliidese nõuded varieeruvad suuresti lähtuvalt sellest, kas loodav infosüsteem on mõeldud organisatsioonisiseseks või –väliseks kasutuseks. Enamasti luuakse tänapäeval rakendused suurele kasutajagrupile ja avalikus sektoris on kasutajad nii organisatsioonisisesed (ametnikud) kui ka –välised (teised avaliku sektori organisatsioonid, ettevõtted, kodanikud). Lisaks on kasutajal veel erinevaid rolle, nt tööl on kasutaja ametnik, kes peab tööülesannete täitmiseks sooritama erinevaid tegevusi, töövälisel ajal aga kodanik, kes soovib infosüsteemi abil teostada isiklikke toiminguid. Kõiki neid erinevaid aspekte tuleb infosüsteemi arendama asudes kasutajaliidese puhul arvesse võtta.

Magistritöö koostaja poolt väljapakutavad kasutajaliidese nõuded on järgnevad:

- **Nõue 113:** Kasutajaliidese kõik disainiotsused peavad olema tellijaga kooskõlastatud.

- **Nõue 114:** Kasutajaliidese kõik osad ja teated peavad olema eestikeelsed.
- **Nõue 115:** Kasutajaliidese ülesehitus peab toetama kasutaja intuitiivset käitumist ka esmakordsel kasutamisel (vigade tegemine peab olema raske).
- **Nõue 116:** Peale kasutaja sisselogimist rakendusse kuvatakse päises sisseloginud kasutaja nimi ja rolliinfo.
- **Nõue 117:** Kui ühele kasutajatunnusele on määratud mitu rolli, kuvatakse kasutajale rollivaliku andmeväli.
- **Nõue 118:** Kasutajaliides peab alati küsima kinnitust andmete kustutamise ja massmuutmise kohta.
- **Nõue 119:** Kasutajal peab olema võimalik rakenduses tegevus pooleli jätta ja hiljem jätkata samast kohast ilma kohustuseta algusesse liikuda.
- **Nõue 120:** Kasutajaliides peab veatult toimima brauseritega, mida toetab eID baastarkvara.au

  Märkus: eID poolt toetatav brauserite nimekiri on leitav:
  http://www.id.ee/?id=33993
- **Nõue 121:** Kui kasutajaliides ei ühildu kasutatava brauseriga, peab kasutaja saama vastavasisulise teavituse.
- **Nõue 122:** Veebilehitseja navigatsiooninupud peavad käituma rakenduses analoogiliselt klassikalise veebilehitsemisega (nt veebilehitseja „Tagasi“ nupp navigeerib kasutaja eelmisele kuvatud lehele).
- **Nõue 123:** Kasutajaliides peab olema lähtuvalt äriloogikast navigeeritav ning võimaldama andmete sisestamist ja kasutamist ainult klaviatuuri kasutades.
- **Nõue 124:** Rakenduse kasutamisel tuleb välistada hiire ja klaviatuuri vältimatu kordamööda kasutus.
- **Nõue 125:** Kasutajaliidese toiminguni navigeerimiseks peab kehtima kolme klõpsu printsiip, väljalogimiseks ühe klõpsu printsiip.
- **Nõue 126:** Interaktiivse vormi puhul ei tohi lehe värskendamisega tegevust korrata (nt faili teistkordselt laadida, saadetud andmeid uuesti saata).
- **Nõue 127:** Kui vorm koosneb mahukatest andmeväljadest, peab kasutajaliides eeldefineeritud ajavahemike järel salvestama välja sisu, et vältida sisestatud andmete kadumist.
- **Nõue 128:** Vormide puhul peab konkreetsel väljal olles kuvama kasutajale juhised, mis kujul informatsiooni väljale sisestada tuleb.

- **Nõue 129:** Andmete sisestamisel peab rakendus alati kontrollima, et sisestatud andmed vastavad välja tüübile.
- **Nõue 130:** Rakendus peab võimalikult palju informatsiooni automaatselt eeltäitma (nt kirje sisestamise kuupäev).
- **Nõue 131:** Kasutajaliidese tausta, menüüde ja tekstide värvid peavad olema muudetavad ilma rakenduse lähtekoodi muutmata.
- **Nõue 132:** Kasutajaliidese peab olema tõlgitav teise keelde ilma rakenduse lähtekoodi muutmata.
- **Nõue 133:** Vältida tuleb kuvasid, mis eeldavad info lugemiseks kerimist vertikaalselt (paremale-vasakule).
- **Nõue 134:** Kui rakenduses teostatav päring on pikem kui kolm sekundit, peab kasutajat sellest visuaalselt teavitama (nt ekraanil on liivakella kujutis; kuvatakse teade, et päringut teostatakse).
- **Nõue 135:** Kasutajal peab süsteemis liikudes igal lehel (va esileht) olema võimalik näha navigatsiooniriba ehk lingirivi (inglise k *breadcrumb trail*) ning iga lingirivis sisalduv pealkiri peab viima vastavale lehele.
- **Nõue 136:** Iga lehevahetuse peab muutma lingirivi (navigatsiooniriba).
- **Nõue 137:** Rakendusel peab olema haldusliides.
- **Nõue 138:** Rakenduse esilehel peab olema võimalus halduri poolt lisada kasutajale mõeldud teavitusi ja informatsiooni, mis peavad olema hästi märgatavad.
- **Nõue 139:** Kasutajaliides peab teavitama kasutajat ette seansi aegumisest.

Eelpool kirjeldatud nõuded on üldised ja peaksid olema sobilikud kasutamiseks igale avaliku sektori organisatsioonile. Lisaks on veel mitmeid kasutajaliidesega seotud nõudeid, mis on organisatsiooni spetsiifilised ja need tuleb kindlasti lisada MFNi loendisse.

Paljudel organisatsioonidel on väljatöötatud visuaalne identiteet (inglise k *Corporate Visual Identity*, CVI), mida tuleb arvesse võtta MFNi kirjeldamisel. Visuaalne identiteet sisaldab reeglina tervet hulka erinevaid elemente (logo, värvid, kirjastiilid, fotod, ikoonid, dokumentide kujundus, signatuurid jms), mida peab arvesse võtma tarkvaraarenduse teostamisel. Nõudena võiks ta olla esitatud järgnevalt:

- **Nõue 140:** Kogu loodava rakenduse kasutajaliidese visuaal peab vastama organisatsiooni visuaalsele identiteedile.

Kasutajate tehnilised vahendid on väga erinevad ja analüüsitud MFNi dokumentides oli mitmel korral välja toodud ka monitoride resolutsioonide nõuded. Kõige otstarbekam tundub magistritöö koostajale kirjeldada MFNi dokumenti monitori resolutsioonide kohta kaks nõuet:

- **Nõue 141:** Avalikuks kasutuseks loodav rakendus peab oleme kasutatav resolutsioonidega … (resolutsioonide loetelu, mida loodav rakendus peab toetama; nt 1024x768)
- **Nõue 142:** Sisemiseks kasutuseks loodav rakendus peab olema kasutatav resolutsioonidega … (resolutsioonide loetelu, mida loodav rakendus peab toetama; nt 1920x1200)

Sisemiseks kasutuseks on monitori resolutsiooni nõuet kindlasti lihtsam kirjeldada, sest on teada, millised monitorid organisatsioonis kasutusel on ja/või mida plaanitakse lähiajal soetada.

Eraldi on otstarbekas määratleda kasutajatele kuvatavate teadete nõuded, mis siiski suuresti sõltuvad loodavast rakendusest. Näitena võib tuua RIA MFNi kasutajale kuvatavate teadete jaotuse:

- Vormisisesed teated – kuvatakse välja juures, kus viga tekkis;
- Vormi üldised teated – kuvatakse sisuploki ülaosas;
- Lehe üldised teated – üldine teavitus lehe kohta, kus kasutaja viibib;
- Serveri teated – kuvatakse eraldi lehel, kasutajal peab olema võimalik liikuda tagasi eelmisele- või esilehele (Riigi Infosüsteemi Amet, 2015).

Lisaks on järgnevalt toodud üldisemad veateateid puudutavad nõuded, mis võiksid magistritöö koostaja arvates sisalduda MFNi kasutajaliidese osas:

- **Nõue 143:** Teade peab olema kirjutatud selges, lõppkasutajale arusaadavas keeles ja olema lühike ning eestikeelne.
- **Nõue 144:** Veateade peab sisaldama probleemi kirjeldust, vea koodi ja lahendust või infot, mis juhendab kasutajat edaspidiseks vea vältimiseks.

- **Nõue 145:** Süsteem peab asendama vaikimisi veateate lehekülje, kuid säilitama algse HTTP vastuskoodi.

Kindlasti on lähtuvalt arendatavast rakendusest võimalik defineerida veel mitmeid MFNe, mis seostuvad kasutajaliidesega. Antud peatükis käsitletud nõuded peaksid andma piisava lähtekoha.

4.10 Testimise nõuded

MFNi dokumendis on oluline määratleda nõuded testimisele. Nõuetega on võimalik seada teatud piirid testimise läbiviimisele ja ulatusele.

Mõningad võimalikud testimisega seonduvad konkreetsed nõuded on toodud alljärgnevalt:

- **Nõue 146:** Rakenduse kõik üleantavad versioonid peavad olema enne tellijale üleandmist täies mahus (testitakse kõiki funktsionaalseid ja mittefunktsionaalseid nõudeid) testitud. Tellija nõudmisel tuleb arendajal koos rakenduse üleandmisega esitada testitulemuste raport.
- **Nõue 147:** Uue rakenduse ja olemasoleva rakenduse uue versiooni igakordsel üleandmisel tellijale, peab kaasas olema skript analüüsi käigus kokkulepitud jõudlustestide teostamiseks.
- **Nõue 148:** Jõudlustestid peavad olema läbiviidud vähemalt kahekordse eeldatava koormuse varuga.
- **Nõue 149:** Rakenduse koormuse testimiseks tuleb luua testandmete kogum.

Arvestades pidevalt muutuvat keskkonda, äriprotsesse ja ärinõudeid, oleks tõenäoliselt testimise puhul mõistlikum püstitada testimise nõue üldisel tasemel. Üldisel tasemel testimise nõude püstitamiseks on väga sobiv kasutada lähtekohana ASVS standardit. ASVSist lähtuvad nõuded võiksid olla järgmised:

- **Nõue 150:** Loodav rakendus peab olema enne tellijale tarnimist testitud OWASP Top 10 väljatoodud turvanõrkuste vastu. Testimistulemuste raport tuleb esitada tellijale rakenduse üleandmisel.

- **Nõue 151:** Loodava rakenduse turvalisuse tase peab vastama ASVS standardi tasemele 2.

Nõude nr 151 puhul tuleb kindlasti määratleda, kas vastavust peab kontrollima arendaja ja esitama vastavasisulise raporti rakenduse tarnimisel või korraldab tarnitud rakenduse testimise tellija (testib ise või laseb seda teha sõltumatul kolmandal osapoolel). Lisaks peaks MFNis olema kirjeldatud (nt märkusena nõude juures), kuidas toimub testitulemusena selgunud puuduste kõrvaldamine.

4.11 Dokumentatsioon

Avalikus sektoris on kohustuslik järgida ISKE nõudeid ja dokumentatsiooni kohta annab soovitusi magistritöö käigus läbitöötatud materjalidest ainult ISKE. Täpsemalt on mooduli B 5.27 „Tarkvaraarendus“ all meede M 2.574 „Tarkvaraarenduse põhjalik dokumenteerimine“. ISKE jagab tarkvaraarenduse dokumentatsiooni kolmeks: projekti dokumentatsioon, projektijuhtimise dokumentatsioon, süsteemi dokumentatsioon.

ASVS standard ütleb küll, et dokumenteerimine on oluline, aga samas ei sisaldu selles konkreetset kontrollgruppi, mis annaks soovitusi dokumentatsiooni osas ehk eeldatakse, et iga organisatsioon püstitab nõuded lähtuvalt oma töökorraldusest ja vajadusest ja täiendavaid juhiseid anda ei ole otstarbekas.

Kahes analüüsitud avaliku sektori organisatsiooni MFNis on välja toodud eraldi nõuded dokumentatsioonile: ühel juhul on üldised nõuded dokumendile (nt dokument peab olema eesti keeles) ja loetelu dokumentidest, mis kindlasti peavad tarkvaraarendusega kaasas käima (nt kasutajate kasutusjuhendid, arhitektuurilised mudelid, nõuded riistvarale); teisel juhul on toodud väga detailne nõuete loetelu (nt värvide, fontide ja muude kujunduselementide kasutamine peab olema piisav ja adekvaatne) koos teatud dokumentide näidistega (nt installeerimise juhendi näidis).

Ühe MFNi puhul oli töökorraldust käsitlevate nõuete hulgas mainitud mõned dokumentatsiooniga seotud aspektid (nt üleantavad dokumendid peavad sisaldama

sisseviidud muudatusi nii, et on väljatoodud muutunud ja lisandunud osa (võrreldes viimati üleantuga)).

Avaliku sektori asutustel on otstarbekas viidata erinevatele seadusest tulenevatele kohustustele, millega kaasnevad tarkvaraarenduse dokumentatsiooni nõuded (nt ISKE, RIHA dokumentide nõuded).

Magistritöö koostaja poolt väljapakutud dokumentatsiooniga seotud nõuete loetelu on järgmine:

- **Nõue 152:** Kogu rakenduse dokumentatsioon peab olema kirjeldatud korrektses eesti keeles.
  Märkused: Erandiks võib olla kolmanda osapoole komponentide dokumentatsioon (dokumentatsioon, mis pole kirjutatud tellija jaoks).
  Erandina käsitletakse ka väliste partneritega seotud projektdokumentatsiooni. Kõik erandid tuleb kirjalikku taasesitamist võimaldaval viisil kooskõlastada tellijaga enne dokumentatsiooni koostamist.
- **Nõue 153:** Dokumentatsioon peab sisaldama versiooni numbrit, muutmise kuupäeva, autori nime ja olema koostatud selge struktuuriga.
  Märkus: Iga dokumendi versiooni kõik uuendused (võrrelduna eelmise kehtinud versiooniga), peavad olema visuaalselt eristatavad.
- **Nõue 154:** Lahenduse dokumentatsioon peab sisaldama RIHA määrusest tulenevat kohustuslikku informatsiooni.
  Märkus: RIHA määrus on leitav:
  https://www.riigiteataja.ee/akt/13147268?leiaKehtiv
- **Nõue 155:** Lahenduse dokumentatsioon peab sisaldama ISKE standardi meetmes M 2.574 „Tarkvaraarenduse põhjalik dokumentatsioon“ nõutud dokumente.
  Märkus: Meede on leitav:
  https://iske.ria.ee/8_02/ISKE_kataloogid/7_Kataloog_M/M2/M_2.574
- **Nõue 156:** Kõik dokumendis viidatud ja seotud teised dokumendid, peavad olema tellijale edastatud enne dokumendi heakskiitmist tellija poolt.
- **Nõue 157:** Kõik tellijale esitatud dokumendid peavad olema redigeeritavad enamlevinud redaktoritega (nt Microsoft Office, OpenOffice).

Teatud arenduste puhul võib osutuda vajalikuks ka konkreetsete dokumentides kajastamist vajavate osiste loetlemine. Alljärgnevalt mõned näited KEMITi MFNi dokumendist:

- **Nõue 158:** Süsteemianalüüsi dokumendis peavad olema detailselt kirjeldatud tarkvaralahendusele esitatavad funktsionaalsed nõuded, andmemudeli elemendid (domeenimudel, loogiline andmemudel jms), kasutajate ja süsteemide interaktsioonid (jadadiagrammid, kasutuslood, olekumudelid jms) ning teised loodava lahenduse funktsionaalsust puudutavad aspektid.
- **Nõue 159:** Tarkvara disaini dokumentatsioon peab kirjeldama erinevate all-süsteemide, moodulite ja teiste komponentide struktuuri ja omavahelisi sõltuvusi, valitud ülesehitust (sh andmemudeli struktuuri) ja korduvkasutatavaid komponente, nõudeid infrastruktuurile ja muid loodava lahenduse tehnoloogilisi aspekte.
- **Nõue 160:** Lõppkasutaja juhendis peavad olema kirjeldatud nõuded andmeformaatidele ja andmereeglid, funktsionaalsuste kasutamise viisid, võimalike enamlevinud vigade ja tõrgete lahendamise teed (Keskkonnaministeeriumi Infotehnoloogiakeskus, 2016).

RIK on lahendanud konkreetse dokumentatsiooni loetelu ja erinevate dokumentide sisu nõuded MFNi lisaga „RIK dokumentatsiooniplaan“, mis sisaldab üleantavate dokumentide loetelu ja nende nõutud sisu detailkirjeldust (Registrite ja Infosüsteemide Keskus, 2013).

Kumba kahest eelpool toodud valikust eelistada on tegelikult iga organisatsiooni valik. Magistritöö koostajale tundub paremini kasutatav ja lihtsamini hallatav RIKi poolt kasutatav lahendus, sest see on hästi struktureeritud eraldiseisev dokument, mida arendaja saab lihtsalt kasutada. See peaks olema ka tarkvaraarendust tellivale organisatsioonile mugavam, sest ajakohasust on lihtsam tagada – tegu ei ole mõne nõudega üldises MFNi loendis, vaid konkreetse MFNi lisaga, mis käsitleb dokumentatsioonile esitatavaid nõudeid.

Olenevalt organisatsioonist ja senisest kogemust võib lisada MFNi loendile ka näidisdokumente, mida arendaja saab kirjeldamisel aluseks võtta. See tagab mõlemale

poole selguse, mida oodatakse ja aitab vältida vaidlusi tulemi vastuvõtmisel. Dokumentide näidised peaksid olema esitatud MFNi dokumendi lisana.

Kokkuvõte

Magistritöö on kirjutatud teemal „Mittefunktsionaalsete nõuete määratlemine turvalise tarkvaraarenduse hankimiseks Eesti avalikus sektoris“. Töö eesmärgiks oli avaliku sektori organisatsioonile MFNi koostamine, mis arvestaks turvalisuse aspektiga tarkvaraarenduses ja oleks alusmaterjaliks organisatsiooni või infosüsteemi spetsiifikast lähtuva MFNi kirjeldamisel.

Töö koosnes neljast peatükist. Esimene osa tutvustas erinevaid MFNi mõistete määratlusi ja seisukohti kuidas MFNe peaks kirjeldama. Teine peatükk keskendus turvalise tarkvaraarenduse raamistike ja metodoloogiate tutvustamisele. Kolmas peatükk tööst tutvustas Eesti avalikus sektoris enim järgimist leidvaid standardeid ASVS ja ISKE. ASVS annab juhiseid kuidas tagada veebirakenduse turvalisus. ISKE on Eesti riigi infoturbe standard ja avaliku sektori organisatsioonidele järgimiseks kohustuslik.

Teaduslikule kirjandusele tuginedes võib väita, et ühest definitsiooni MFNi kohta ei ole võimalik välja tuua, kuigi MFNi defineerimisega on tegeletud aastakümneid. Teadlased on saavutanud konsensuse teemal, milline on MFNi olemus: MFN defineerib viisid, kuidas loodav tarkvara peab temale püstitatud funktsionaalseid nõuded täitma. MFNi peamine eesmärk on tagada soovitud kvaliteediga tarkvara.

Nõude kirjeldamisel tuleb kaasata lõppkasutaja, nõue tuleb kirjeldada üheselt mõistetavalt, ta peab olema ajakohane ja tagatud peab olema nõuete omavaheliste seoste selgus. Nõuete kirjeldamisel võib aluseks võtta erinevaid meetodeid, samas ei ole võimalik teaduslikule kirjandusele tuginedes ühtegi paremini rakendatavat meetodit välja tuua – lähtuda tuleb siiski iga organisatsiooni spetsiifikast ja leida oma organisatsioonile sobivaim.

Turvalisuse integreerimiseks tarkvaraarendusega on hulk erinevaid raamistikke ja metodoloogiaid, neist mõnda (tarkvaraarenduse elutsükli meetod, tarkvara veatu konstrueerimise meetod, infoturbe ühiskriteeriumite raamistik) tutvustas ka käesolev töö. Lisaks sisaldas magistritöö ka erinevaid soovitusi turvalisuse integreerimiseks agiilsetesse arendusmetoodikatesse.

Magistritöös käsitletud standardid ASVS ja ISKE kirjeldavad hulgaliselt nõudeid ja soovitusi, mis aitavad tagada turvalise tarkvaraarenduse. Mõlemal standardil on turvanõuded ja –soovitused jagatud kolme erineva taseme vahel. Sobiva tasemega nõuete rakendamine sõltub konkreetse organisatsiooni infosüsteemidele esitatavatest nõuetest.

Empiirilise osa magistritööst moodustas neljas peatükk, milles magistritöö koostaja analüüsis erinevate avaliku sektori organisatsioonide MFN dokumente ning dokumente, mis reguleerivad tarkvaraarendusele esitatavaid nõuded avalikus sektoris.

Magistritöö koostaja kirjeldas Eesti avaliku sektori organisatsioonile sobivad, turvalisuse erinevaid aspekte arvestavad, mittefunktsionaalsed nõuded. MFNi nõuded jagas töö koostaja 11 gruppi: üldised põhimõtted, vastavus standarditele ja seadusandlusele, arhitektuur, turvalisus, lähtekood, andmebaas, logimine ja monitooring, konfiguratsioon, kasutajaliides, testimine ning dokumentatsioon. Magistritöö koostaja arvates on töö tulem hea alusmaterjal MFNi kirjeldamisel Eesti avaliku sektori organisatsioonile. Nõuete kasutamisel tuleb lisada organisatsiooni spetsiifikat sisaldavad nõuded ja iga nõude juurde kirjeldada vastutaja ning tarkvaraarenduse etapp või tegevus, mille käigus nõude täitmist valideeritakse.

Magistritöö käigus loodud MFNis ei sisaldu nõudeid pilvetehnoloogiate kasutamiseks, va nõue 70, mis ütleb, et pilvetehnoloogiate kasutamisel tuleb lähtud riigipilvele esitatud nõuetest. Pilvetehnoloogiate kasutuselevõtuga seotud nõuded on üks võimalikke uurimissuundi edaspidiseks.

Mobiilsete seadmete järjest suureneva levikuga, suureneb erinevate äppide arendus. Seoses sellega oleks tulevikus kasulik uurida, millisel määral erinevad ja/või kattuvad nõuded turvalise arenduse osas ning milliseid tehnilisi lahendusi kasutades tagada äppide ja nendes sisalduvate andmete turvalisus.

Summary

Based on actuality of the topic the author has chosen following thesis: „Defining Non-functional Requirements for Secure Software Development Procurement in Estonian Public Sector".

The key word of the thesis is security – security is system's capability to maintain its objects (resource and information) integrity and confidentiality.

Thesis consists of four sections, the first section introduces notion of the non-functional requirement and how it should be described. The second section of the work gives an overview of different process models and methodologies which help to integrate security into software development process. The third section introduces main two standards – Application Security Verification Standard and IT Baseline Protection Manual – which are followed by Estonian public sector in software development. Application Security Verification Standard gives instructions how to develop secure applications. IT Baseline Protection Manual is official information security standard of Estonia and is mandatory to follow by all Estonian public sector organizations.

The goal of the thesis was to define non-functional requirements for Estonian public sector organization for secure software development.

Empirical part of the thesis was section four, where the author analysed several public sector organization's non-functional requirements documents and other Estonian public sector documentation which regulates different requirements for software development.

The author described a basic list of non-functional requirements for Estonian public sector organization, which takes into consideration several aspects of security in software development. Requirements were divided into 11 groups: general principles, accordance to standards and legislation, architecture, security, source code, database, logging and monitoring, configuration, user interface, testing and documentation.

The author of the master thesis achieved the goal.

Kasutatud kirjandus

Alliksoo, K., Hanson, V., Laur, M., & Oit, M. (2009). *Turvarisk.* Tallinn: Cybernetica AS.

Beznosov, K., & Kruchten, P. (September 2004. a.). Towards Agile Security Assurance. doi:10.1145/1065907.1066034

Bijan, Y., Yu, J., Stracener, J., & Woods, T. (Mai 2012. a.). System Requirements Engineering - State of the Methodology. doi:10.1002/sys.21227

Bourque, P., & Fairley, R. E. (2014). Guide to the Software Engineering Body of Knowledge. Allikas: http://www.computer.org/portal/web/swebok/v3guide

Buldas, A., Hanson, V., Krasnosjolov, J., Laur, M., & Veldre, A. (2011-2017). Allikas: Andmekaitse ja Infoturbe leksikon: http://akit.cyber.ee/

Chung, L., & do Prado Leite, J. C. (2009). On Non-Functional Requirements in Software Engineering. Dallas, Texas. Allikas: https://pdfs.semanticscholar.org/2d1e/79e057a9111ea6863378ffeca526a4e41c5f.pdf

Davis, N. (Detsember 2005. a.). Secure Software Development Life Cycle Processes: A Technology Scouting Report. Allikas: https://www.sei.cmu.edu/reports/05tn024.pdf

Glinz, M. (2007). On Non-Functional Requirements. (lk 21-26). Delhi: IEEE Computer Society. doi:http://doi.ieeecomputersociety.org/10.1109/RE.2007.45

Green, J., & Stellmann, A. (2005). *Applied Software Project Management.* Allikas: http://www.academia.edu/download/35512087/Applied_Software_Project_Management_abrir_con_chrome.pdf

Gustavsson, J., & Willander, J. (08 2005. a.). Security requirements - A field study of current pracitce. Liköpings, Rootsi. Allikas:

https://pdfs.semanticscholar.org/3bbf/467c259cf87970329ad313a93c9ae8887acd.pdf

Keskkonnaministeeriumi Infotehnoloogiakeskus. (Juuni 2016. a.). *Riigihangete register*. Allikas: https://riigihanked.riik.ee/register/fsdownload?fileId=E8D88CAB-5DAE-C885-C630-4A987CFE173F

Manico, J., Cuthbert, D., & van der Stock, A. (Juuli 2016. a.). Application Security Verification Standard. Allikas: https://www.owasp.org/images/3/33/OWASP_Application_Security_Verification_Standard_3.0.1.pdf

Microsoft. (November 2010. a.). Simplified Implementation of the Microsoft SDL. Allikas: https://www.microsoft.com/en-us/download/details.aspx?id=12379

Miller, H. E. (Märts 1996. a.). The Multiple Dimensions of Information Quality. doi:DOI: 10.1080/10580539608906992

Nigul, T. (September 2014. a.). OWASP - ega ometi o-herilane. Tallinn, Eesti. Allikas: https://blog.ria.ee/owasp-ega-ometi-o-herilane/

Põldmaa, H. (2016). *Infoturbe haldus.* Loeng, Tallinn.

Registrite ja Infosüsteemide Keskus. (Märts 2013. a.). Allikas: http://adr.rik.ee/rik/dokument/4062858

Riigi Infosüsteemi Amet. (Juuli 2015. a.). *Riigihangete register.* Allikas: https://riigihanked.riik.ee/register/fsdownload?fileId=DE3F1A00-7647-28F7-4E52-FC1019CC8AEE

Riigi Infosüsteemi Amet. (2017). ISKE portaal. Eesti. Allikas: https://iske.ria.ee/8_02/

Riigi Infosüsteemi Amet. (Jaanuar 2017. a.). ISKE rakendusjuhend. Eesti. Allikas: https://iske.ria.ee/8_02/?action=AttachFile&do=get&target=ISKE rakendusjuhend ver. 8.00.pdf

Tepandi, J. (21. 12 2016. a.). Tarkvara protsessid, kvaliteet ja standardid. Tallinn. Allikas: http://tepandi.ee/tks-loeng.pdf

Vabariigi Valitsus. (25. Jaanuar 2009. a.). Infosüsteemide turvameetmete süsteem. Eesti. Allikas: https://www.riigiteataja.ee/akt/13125331?leiaKehtiv