TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Informaatikainstituut

IDU70LT

Marek Lendsaar, 130460IABMM

INFOSÜSTEEMI TEENINDUSVÕIMEKUSE TÕSTMINE EKSAMITE INFOSÜSTEEMI NÄITEL

Magistritöö

Juhendaja: Enn Õunapuu

PhD

dotsent

Tallinn 2016

Autorideklaratsioon

Kinnitan, et olen koostanud antud magistritöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Marek Lendsaar

09.05.2016

Annotatsioon

Käesolev magistritöö käsitleb Haridus- ja Teadusministeeriumi haldusalas töötava Eksamite infosüsteemi esmaste jõudlust tõstvate lahenduste juurutamist. Antud tegevuse eesmärk on luua esmased võimalused infosüsteemi teenindusvõime tõstmiseks, suurendada veakindlust ning seeläbi tagada haridusvaldkonnas riiklike strateegiliste eesmärkide täitmise.

Töö peamine uurimiseesmärk on infosüsteemi teenindusvõime tõstmine ilma rakendustarkvara muutmiseta. Alamprobleemidena uuritakse süsteemi töö mõõtmise vahendeid, rakenduskihi teenindusvõimekuse tõstmise võimalusi sisu mahu töötlemise ja koormuse hajutamise näol ning mõõdetakse ja võrreldakse erinevate muudatuste tulemusi.

Eesmärgi täitmiseks lisatakse infosüsteemi mitme rakendusserveri vahel koormust jaotav komponent, mille rollis katsetatakse kolme erinevat tarkvara: HAProxy koormusjaoturit, Nginx veebiserverit ja Apache veebiserverit. Viiakse läbi eksperimente sobivaima seadistuse leidmiseks ning infosüsteemi testitakse funktsionaalsuse säilimise kontekstis, kus lisandunud komponendi rolli täidavad vaheldumisi eelpool mainitud tarkvarad.

Töö tulemusena on välja toodud sobivad lahendused rakendus- ja andmebaasiserverist koosneva infosüsteemi täiendamise viisid, mis võimaldaksid jagada kliendi päringuid mitme teenindava rakendusserveri vahel, lõhkumata senise infosüsteemi funktsionaalset toimist.

Töös välja pakutud lahendused põhinevad avatud lähtekoodiga tarkvaradel ning on sobilikud rakendamiseks piiratud ressursside tingimustes.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 51 leheküljel, 5 peatükki, 15 joonist, 8 tabelit.

Abstract

The service capability enhancement of information system in the example of Examination Information System

The current master's thesis concerns core performance-enhancing solutions of information systems of exams administered by the Education and Science Ministry. The purpose of the thesis is to create primary opportunities for enhancing service capability and fault tolerance, thereby ensuring the fulfilment of national strategic objectives in the field of education.

The primary objective is to enhance service capability without altering application software. In addition, the means of measuring operation systems are examined by enhancing the capacity and load-spreading of application-layer service. The discrepancies in results are measured and compared.

To reach the objective a component to balance the load is added between several application servers in the information system. Three different types of software are tested: HAProxy load balancer, Nginx web server and Apache web server. Experiments to find the most appropriate setting are carried out and the information system is tested in the context of preserving functionality. Previously listed software act as additional components.

The adequate solutions for upgrading the information systems consisting of the implementation- and database server are presented in the results of the thesis. These allow distribution of a number of inquiries by a client between several application servers in service without interrupting the functionality of the current information system.

The solutions offered in this thesis are based on open source software and are suitable for implementing in conditions of limited resources.

The thesis is in Estonian and contains 51 pages of text, 5 chapters, 15 figures, 8 tables.

Lühendite ja mõistete sõnastik

CA	Sertifitseerimisasutus (*cerfification authority*)
CN	Üldnimi (*common name*)
CRL	Sertifikaatide tühistusnimekiri (c*ertificate revocation list*)
DER	Sertifikaadi binaarkodeeringu vorm
CLU	Klastri sõlm (*cluster node*)
Digi-ID	ID-kaardiga analoogne kiipkaart ainult elektroonilises keskkonnas kasutamiseks.
Digipädevus	Enesekindel, kriitiline ja loov IKT kasutamine töö, tööhõive, õppimise, puhkuse, kaasatuse ja/või ühiskonnaelus seotud eesmärkide saavutamiseks. [1]
DN	Eristustunnus (*distinguished name*)
DNS teenus	Domeeninimede teenus (*domain name system*)
EIS	Eksamite infosüsteem
HTTP	Hüperteksti edastusprotokoll
HTTPS	Hüperteksti edastusprotokoll üle turvasoklite kihi
IKT	Info- ja kommunikatsioonitehnoloogia
LB	Koormusjaotur (*load balancer*)
LTS	Pikaajalise toega versioon (*long term support*)
OCSP teenus	Sertifikaadi kehtivuse kontroll reaalajas (*online certificate status protocol*)
PEM	Sertifikaadi ASCII kodeeringu vorm
URL	Internetiaadress (*uniform resource locator*)
TLS	Transpordikihi turbeprotokoll (*transport layer security*)
SSL	Turvasoklite kiht (*secure sockets layer*)
SK	AS Sertifitseerimiskeskus, ID-kaardi sertifikaatide väljastaja
VHOST	Virtuaalne veebiserveri kuulatav liides (*virtual host*)
WSGI	Veebilüüsi liides (*web server gateway interface*)

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Eesti elukestva õppe strateegia 2020 kui Eesti tähtsamaid haridusvaldkonna arenguid suunav dokument on aluseks aastatel 2014 kuni 2020 hariduse eelarve kujundamisel, defineerides viis strateegilist eesmärki ja nende mõõdikud, mille läbi saaks toimuda liikumine teadmiste- ja innovatsioonipõhise ühiskonna suunas. [2]

Elukestva õppe strateegia neljas strateegiline eesmärk „Digipööre elukestvas õppes“ näeb ette järgeva: õppimisel ja õpetamisel rakendatakse kaasaegset digitehnoloogiat otstarbekamalt ja tulemuslikumalt, paranenud on kogu elanikkonna digioskused ning tagatud on ligipääs uue põlvkonna digitaristule. [2] Sellest tulenevalt koostati Haridus- ja Teadusministeeriumi poolt „Digipöörde programm“, mille minister Jevgeni Ossinovski 8. aprillil 2015 kinnitas. Digipöörde programm kujundab tervikliku lähenemise digipädevuse arendamiseks ja digivõimaluste eesmärgipäraseks kasutamiseks õppeprotsessis ning toetab sellega muutunud õpikäsitluse rakendamist. Programmi eesmärgi saavutamiseks on vaja muuhulgas luua toetav keskkond ning koostada ja teha kättesaadavaks kvaliteetne õppevara, et omandatavaid teadmisi ja oskusi oleks võimalik viia paremini kooskõlla edasiõppimise eelduste ja tööturu vajadustega. [1]

Eksamite infosüsteemi (edaspidi EIS) kui haridusvaldkonna ühte suurimat töövahendina kasutatavat tehnilist lahendust on mainitud elukestva õppe strateegias integreeritavate süsteemide hulgas ning infosüsteemile on määratud konkreetsed ülesanded ka digipöörde programmi dokumendis. Olulise eesmärgina peab EIS-is olema loodud valmisolek erinevate testide, nagu riigieksamid ja tasemetööd, läbi viimiseks digitaalsel kujul, sisaldades endas nii ülesannete esitlust, tulemuste kogumist, analüüsi, tagasisidestamist ning muud riikliku hariduspoliitika kujundamiseks vajaliku lähteandmestiku kogumist.

1.1 Taust ja probleem

Eksamite infosüsteem sai alguse Riiklikus Eksami- ja Kvalifikatsioonikeskuses ellu viidava ÕKVA programmi („Õppe kvaliteedi parendamine õppeasutuste sise- ja õpitulemuste välishindamissüsteemi arendamise kaudu“) raames, kus üheks eesmärgiks oli õpitulemuste välishindamisel kasutatavate testide vastavusse viimine testiteooriale. Huvigruppidele tuleb luuga ligipääs kvaliteetsetele näidisülesannetele ning on loodud ka võimalus eksamite elektroonseks läbiviimiseks. [3]

Eesmärgi täitmise võimalusena nähti Riiklikus Eksami- ja Kvalifikatsioonikeskuses uue infosüsteemi loomist, mis ühelt poolt võtaks üle olemasoleva riigieksamite infosüsteemi funktsionaalsuse ning teiselt poolt lisaks uued, seni infosüsteemidega katmata tehnilised lahendused asutuse põhiprotsesside toetamiseks. EIS-i loomiseks koostati tehniline kirjeldus, mis oli ühtlasi aluseks hanke läbiviimisel. Kombineeritud tegevuste tulemusena sai haridusvaldkond e-hindamist võimaldava, riigi infosüsteemi kuuluva ja tänaseni kasutuses oleva veebipõhise infosüsteemi.

Aastast 2014 on toimunud järkjärguline digitaalse eksamivormi laialdasem kasutuselevõtt. Aastast aastasse on laiendatud nii kaasatud õppeainete hulka kui ka teste lahendavate õpilaste valimit. 2016. aasta maikuus sooritavad kuuendate klasside valimis olevad õpilased elektroonilised tasemetööd järgmistes õppeainetes: eesti keel / eesti keel teise keelena, matemaatika ja ühes õppeaines, mis avalikustatakse vahetult enne testi toimumist. Tehniliselt tähendab see testimist võimaldavale infosüsteemile aastast aastasse kasvavat koormust. Hanke spetsifikatsioonist lähtuvalt oli Eksamite infosüsteem mõeldud kuni 400 üheaegase tavapärase käitumisega testisooritaja teenindamiseks. Vastav võimekus saavutati lihtsa kahest serverist koosneva konfiguratsiooniga, milles üks seade täitis andmebaasiserveri ja teine rakendusserveri rolli. Teenindusvõimekuse tõstmiseks tuleb teha infosüsteemis muudatusi piiratud ressursside situatsioonis ning vastavate võimaluste leidmisele, analüüsile ja rakendamisele käesolevas magistritöös keskendutaksegi.

6. klasside tasemetööd toimuvad 2016. aasta maikuus täies mahus elektrooniliselt. Selleks ajaks peab olema tagatud Eksamite infosüsteemi piisav võimekus vastavalt korraldusprotsessile kliente teenindada. Infosüsteemi võimalikke täiendusvõimalusi otsitakse, analüüsitakse ja rakendatakse Sihtasutus Innove õiguste, võimaluste ja volituste

piires. Parendatud infosüsteemi tööst on kasu nii Innovele, kes suudab riigi ees halduslepinguga võetud kohustusi täita kui ka Haridus- ja Teadusministeeriumile hariduspoliitika strateegiliste otsuste testimisel.

Tööde teostamise ajavahemik jääb suures mahus 2016. aasta esimesse kvartalisse.

1.2 Ülesande püstitus

Eksamite infosüsteem koosneb tehnilises mõttes riistvarast ja sellega seotud infrastruktuurist, tarkvaraplatvormist ning rakendustarkvarast. Selleks, et Innove saaks nii põhitegevusi kui lepinguga võetud kohustusi edukalt täita ja tagatud oleks hariduspoliitika valdkondlike strateegiliste eesmärkide täitmine, tuleb EIS-i võimekus viia sobivale tasemele. Antud töö käsitleb infosüsteemi jõudluse tõstmist eelkõige riistvara ja süsteemitarkvara vaatenurgast, kus tähtsaimaks uurimisobjektiks on infosüsteemile sobiva arhitektuuri ja komponentide leidmine, mille puhul infosüsteem suudaks teenindada 1000 üheaegset kasutajat.

Aktsepteeritava tulemuseni jõudmiseks tuleb edukalt lahendada järgnevad ülesanded:

1) Tekitada konkreetne skaala süsteemi töö mõõtmiseks ja muudatuste mõju hindamiseks ning mõõta süsteemi jõudlus enne muudatuste tegemist.
2) Leida, läbi proovida, analüüsida ja rakendada lahendus, mis võimaldaks infosüsteemi klientide päringuid ja sellega kaasnevat koormust mitme serveri vahel hajutada;
3) Võimalusel leida, läbi proovida ja rakendada lahendused, mis võimaldaksid vähendada rakendusserveri hooldustöödest tingitud katkestuste mõju kasutajatele.

Lahenduse väljatöötamisel tuleb arvestada, et infosüsteemi rakendustarkvara töö loogikas suuri muudatusi teha ei saa. Tarkvara funktsionaalsus peab säilima ka pärast täienduse rakendumist.

1.3 Metoodika

Tegemist on praktilise uurimusega, kus hindamiskriteeriumiteks on eelkõige lahenduse toimivus ja kuluefektiivsus. Tegevusuurimuse (*Action Research*) käigus leitakse

võimalikud lahendused, teostatakse eksperimente ja analüüsitakse tulemusi, mille põhjal selekteeritakse sobivaim võimalik lahendus.

Eksamite infosüsteemis esinevate analoogsete probleemidega on ühel või teisel viisil seotud kõik suuremate infosüsteemide pidajad ning sageli sõltub lahendus nii projekti visioonist – millisele teenustaseme tingimustele süsteem vastama projekteeritakse, kui ka kasutada olevatest tehnilistest võimalustest. EIS kasutab klientide teenindamisel levinud standardseid lahendusi ja see loob teoreetiliselt mahuka eelduste pagasi süsteemi probleemide lahendamiseks, sest veebipõhised infosüsteemid on tänapäeval laialdaselt levinud ning suure tõenäosusega võiks mõni mujal maailmas välja töötatud stsenaarium sobida ka antud süsteemis rakendamiseks.

Eksamite infosüsteemi jõudluse tõstmiseks luuakse Proxmox riistvara virtualiseerimise platvormile infosüsteemi arenduskeskkond, mis peegeldab võimalikult täpselt toodang-keskkonna tehnilisi parameetreid ja funktsionaalsust. Virtuaalne riistvara annab arenduskeskkonna tarbeks vajaliku paindlikkuse, võimaldades operatiivselt vahetada infosüsteemi teenindavaid virtuaalservereid, hoida nendest mitut versiooni ning vajadusel luua koopiaid.

Erinevate materjalide põhjal leitakse potentsiaalsed lahenduskäigud infosüsteemi jõudluse tõstmiseks, need realiseeritakse eksperimentidena arenduskeskkonnas erinevate virtuaalserverite peal ning testitakse ja hinnatakse kitsendavate tingimuste ja täiendava kasu kontekstis, sealhulgas pööratakse tähelepanu ka võimalikule lisaväärtusele, mida ühe või teise tarkvara kasutamine infosüsteemi käideldavusele lisab. Parimat lahendust rakendatakse infosüsteemi toodang- keskkonnas.

Uurimust saab kasutada analoogsetes keskkondades analoogse protsessi läbiviimiseks. Töös esitatavad konfiguratsiooninäited on mõeldud töötama Linux operatsiooni-süsteemidel, eelkõige Ubuntu Server 14.04 LTS, kuid sobiva rakendustarkvara olemasolul ja konfiguratsiooni vähesel muutmisel võivad lahendused toimida ka teistel platvormidel.

1.4 Ülevaade tööst

Magistritöö koosneb sissejuhatusest, neljast sisulisest peatükist, eesti- ja inglisekeelsest kokkuvõttest, kasutatud kirjanduse loetelust ja lisadest.

Esimeses sissejuhatavas peatükis selgitatakse üleüldist taustsüsteemi ning Eksamite infosüsteemi (EIS) kujunemise põhjuseid. Kirjeldatakse ühiskondlike arenguid, millest lähtuvalt on magistritöös käsitletav probleem tekkinud, püstitatakse reaalsed uurimisülesanded ja pakutakse välja probleemide lahendamise metoodika.

Teises peatükis antakse ülevaade infosüsteemist, millega seotud muutuste võimalusi antud magistritöös uuritakse. Lugeja saab ülevaate EIS-i tehnilisest ülesehitusest, peatükis kirjeldatakse rakendustarkvara arendusplatvormi, käitavat süsteemitarkvara koos juurde kuuluvate lisatarkvarade ja moodulitega ning kasutatavat riistvara.

Magistritöö kolmandas peatükis kirjeldatakse peamisest uurimisülesandest lähtudes infosüsteemi kaudu edastatava sisu optimeerimise vajalikkust ja võimalusi.

Neljandas peatükis keskendutakse sügavamalt reaalse infosüsteemi muudatuse ja käesoleva magistritöö ühe alameesmärgi tarbeks sobivate lahenduste leidmisele, nende läbi proovimisele, tulemuste kaardistamisele ja sobivaima lahenduse valimisele.

Viiendas peatükis võrreldakse rakendatud muudatusi infosüsteemi jõudluse paranemise kontekstis. Infosüsteemi komponente testitakse ja analüüsitakse saadud tulemusi.

2 Olukorrakirjeldus

Vastavalt spetsifikatsioonile on Eksamite infosüsteemi loodud veebipõhise testimis-keskkonnana, mis toetab:

1) üksikülesannete koostamist, eeltestimist, eeltesti analüüsimist ja ülesannete haldamist;
2) ülesannete turvalist säilitamist;
3) eksamitööde koostamist, dokumenteerimist ja arhiveerimist;
4) eksamite tehnilist korraldamist, sh eksamitele registreerimist, eksamitunnistuste arvestust ja eksamitööde digitaliseerimist;
5) eksamitööde hindamist ja dokumenteerimist, sh hindamise standardiseerimist;
6) eksamite sooritamist e-keskkonnas;
7) ülesannete pangas olevate avalikustatud ülesannete ja nende hindamisjuhendite kasutamist eksamiteks valmistumisel;
8) eksamitulemuste avalikustamist. [4]

EIS-i rakendustarkvara põhiliseks programmeerimiskeeleks on Python. Kasutajaliidese komponendid on loodud arendusraamistikus Pyramid, kasutajaliidese vormid Mako mallidena. Vajadusel on kasutatud ka JavaScripti ning erandjuhtudel Java appleteid. Andmevahetuskihiks on universaalne SQLAlchemy, mis võimaldab pärast mõningast seadistamist kasutada erinevaid andmebaasisüsteeme.

Rakendusserverina kasutatakse Eksamite infosüsteemis Apache veebiserverit koos *python*i toe loova *mod_wsgi* mooduliga. Serveri ja kliendi vaheline andmesidekanal peab olema krüpteeritud. Selleks on rakendusserveris seadistatud mod_ssl, mille täiendavaks ülesandeks on ID-kaardi autentimiseks sobiliku keskkonna loomine, sobivate kliendisertifikaatide sertifitseerimisahelate (aktsepteeritakse infosüsteemi enda ja AS Sertifitseerimiskeskus poolt väljastatud sertifikaate) ja tühistusnimekirjade (CRL) kontroll. Kindlustamaks kõikide kliendipäringute turvaline töötlemine, suunatakse krüpteerimata HTTP päringud esmasel süsteemi poole pöördumisel *mod_rewrite* abil koheselt turvakanalisse.

EIS-i rakendusserveri süsteemitarkvaraks on kasutatud algusest peale OpenSUSE nimelist Linuxi distributsiooni. OpenSUSE oli algusaastatel mugav, lihtsasti hallatav,

piisavas koguses erinevaid võimalusi pakkuv ning ID-kaardi tarkvara poolt toetatud. Paraku lõpetati aastal 2014 OpenSUSEle ID-kaardi komponentide ametlik tootmine. Samuti puudub distributsioonil sarnaselt Debianil baseeruvatele operatsioonisüsteemidele pikaajalist tuge (LTS) pakkuv versioon, mis kokkuvõttes muutis rakendusserveri operatsioonisüsteemi haldamise probleemseks ja aeganõudvaks. Alates 2016. aastast võetakse järk-järgult kasutusele Ubuntu Server LTS süsteemitarkvara, millel põhinevad ka mitmed riiklikul tasemel arendatavad infosüsteemi komponendid, nagu X-tee turvaserver ja jälgimisjaam.

Andmebaasiserverina kasutati alguses platvormide ühtlustamise põhjendusel Microsoft SQL 2008 serverit, kuid laienemisvajaduste ilmnemisel loobuti PostgreSQL tarkvara kasuks. PostgreSQL on kasutusel senini, aja jooksul on värskendatud jooksvalt tarkvara versiooni.

Keskset infosüsteemi käitavad kaks Dell PowerEdge 1950 serverit. Tehniliste näitajate poolest on tegemist pigem vana riistvaraga. Rakendust teenindaval serveril on üks Intel Xeon E5320 4-tuumaline protsessor ja 8 GB mälu, andmebaasiserveril kaks 4-tuumalist protsessorit ning 16 GB mälu. Riistvara ressursid on täielikult operatsioonisüsteemi käsutuses ja rakendustarkvara poolt kasutatavad.

Eksamite infosüsteemi planeerimisel arvestati e-testide laialdasema kasutuselevõtu võimalusega ja koolides piisava internetiühenduskiiruse puudumise ohus loodi kohaliku serveri kasutamise võimalus. Tegemist on USB mälupulgalt käivitatava operatsioonisüsteemiga, milles sisalduv funktsionaalsus võimaldab ühel kooli arvutiklassis asuval arvutil täita lokaalse serveri rolli. Kohalik server laeb kesksest serverist aeglase internetiühenduse kaudu eksamitöö paketi, serveerib eksamitööd läbi kohaliku kiire võrgu, kogub kokku eksaminandide vastused ning lõpuks laeb eksamitöö tulemused taas üle aeglase välisühenduse kesksesse serverisse hindamiseks. Lahendus võimaldab olulisel määral vähendada lisaks võrgule ka teiste EIS-i ressursside koormatust. Kohaliku serveri miinuseks on täiendav seadistamiskoormus koolidele ja seepärast pole Innove antud võimalust seni peetud erinevate üleriigiliste testide läbi viimisel kasutanud.

3 Süsteemi töö mõõtmise vahendid

Süsteemi töö mõõtmiseks on võimalik kasutada tarkvaras Apache Jmeter, millega tuleb koostada testid, mõõtmaks süsteemi jõudlust ja simuleerimaks kasutajate käitumist süsteemis. Süsteemis suuremaid muudatusi tegemata on automaatselt võimalik testida ülesannete komponentide edastamist ja avalike ülesannete või konkreetse eksamitöö lahendamist. Eksamitöö lahendamisel lisandub läbiviimise loogikast tulenevad nõuded – eksaminandid sooritavad eksami kindlas ajavahemikus ning eksamitööd on võimalik lahendada ainult üks kord. Simuleerimaks võimalikult täpselt kasutaja käitumist, tuleb antud loogikat arvestada ka infosüsteemi automaattestimisel.

Infosüsteemi serveriplatvormi koormamise tehnilisest poolest lähtuvalt jaguneb süsteemi töö mõõtmine täiendavalt veel veebilehe staatiliste komponentide ja dünaamiliste komponentide edastamise kiiruse mõõtmiseks. Staatilised komponendid paigaldatakse installatsiooniprotseduuri käigus rakendusserveri kõvakettale, need on osaks kasutajaliidesest ning neid ei ole võimalik infosüsteemi kasutajaliidese vahenditega muuta. Antud hulka kuuluvad .png, .gif, .jpg pildifailid, .css kujundusfailid, .js JavaScripti failid ning .jar Java komponendid.

Dünaamilised komponendid tekitatakse eksamitöö ülesannete loomisel kasutajaliidese vahenditega ja neid säilitatakse infosüsteemi andmebaasis. Nendeks võivad olla eksamitöös kasutust leidvad erinevad html, pildi, heli ja videofailid.

Eksamitöö lahendamise protsessis on suurim vahe staatiliste ja dünaamiliste komponentide vahel see, et infosüsteem ei piira staatilistele komponentidele ligipääsu vastupidiselt dünaamilistega, mille puhul tuleb väga selgelt kontrollida kasutajatele edastatavat sisu ja näidata konkreetsele kasutajale tema rollist lähtuvaid selgelt piiritletud elemente.

3.1 Eksamitöö esitamise jõudluse mõõtmine

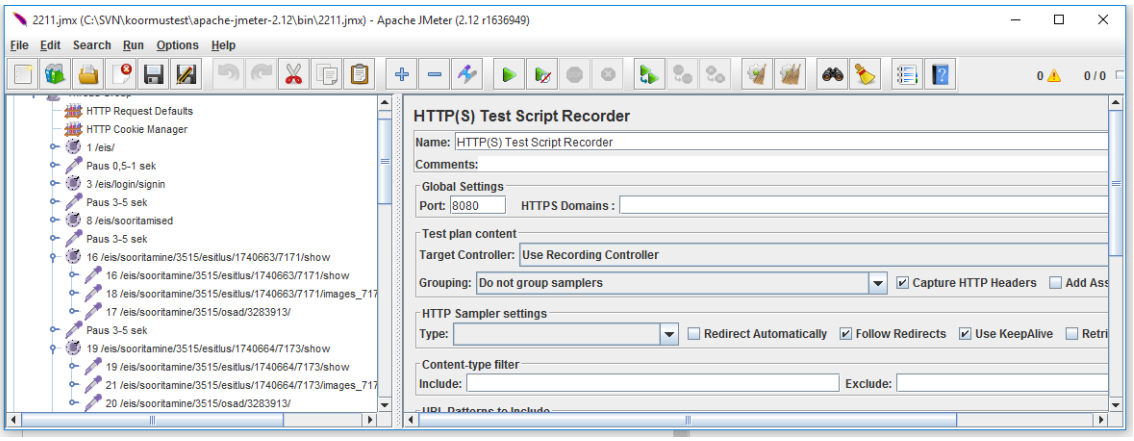
Apache Jmeter on jõudlustestimiseks mõeldud Java rakendus, mis algselt oli mõeldud ainult veebirakenduste testimiseks, kuid aja jooksul on võimalusi lisandunud ning praegusel hetkel pakutakse oluliselt laiemat testimise funktsionaalsust. [5]

Apache JMeter tarkvaraga koostatud koormustest on korduvkasutatav ja seda ei ole vaja muuta, kui infosüsteemi käitumine ei muutu. Koormustest koosneb kahest põhidetailist, testiplaanist (*Test Plan*) ja töölauast (*WorkBench*), mille alla lisatakse rakenduse vahenditega koormustestimiseks vajalikud parameetrid. Testiplaan on võimalik kokku panna täielikult käsitööna, kuid tegemist on veebilehe paljudest detailidest lähtuvalt äärmiselt ajamahuka protsessiga. Seetõttu kasutame JMeter tarkvara traditsioonilist *proxy* serveri funktsionaalsust ja salvestame testiplaani läbi Mozilla Firefox veebilehitseja klõpsides soovitava testistsenaariumi käsitsi läbi.

Joonis 1. Andmesalvestus JMeteri abil

Tulemuseks on andmevahetuse salvestus, mis sisaldab endas nii kasutaja tegevusena tekkinud sisendandmed, serveripoolset vastust kui ka HTTP päises edastatavaid parameetreid, nagu näiteks serveri vastuskood. Näiteks on võimalik eristada ajutist suunamist (kood 302) ja püsivat suunamist (kood 301) ning seda jõudlustesti hilisemal käivitamisel kontrollida.

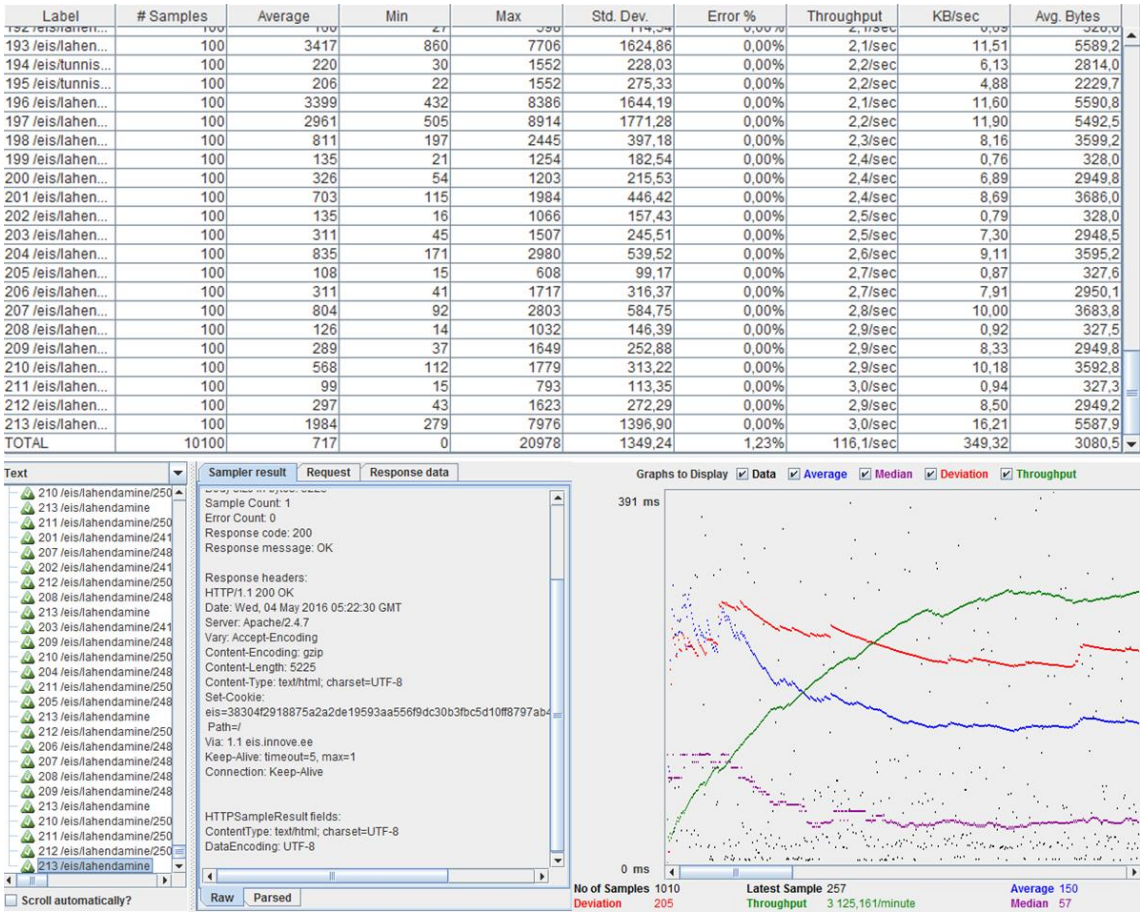
Kasutaja reaalse käitumise jäljendamiseks tuleb lisada sobivatesse kohtadesse pausid, mis võiksid parima tulemuse saamiseks olla võimalikult sarnased aegadega, mida kasutaja infosüsteemi hiireklikkide vahel teeb. EIS-i kontekstis kasutatakse klikkide vahelist aega tõenäoliselt ülesande lahendamiseks ja seda on võimalik ülesande eeltestimisel mõõta.



Joonis 2. Osa salvestatud testiplaanist

Pauside lisamise järel on testiplaan valmis ning seda saab kasutada infosüsteemi dünaamilise sisu edastamise jõudluse mõõtmiseks.

Apache JMeter väljastab testide lõppedes valitud moodulitest lähtuva väljundi, millest on võimalik välja valida sobivad parameetrid muudatuste mõju hindamiseks. Alljärgnev pilt kuvab tulemusi selliselt, nagu JMeter need testi käigus väljastas.



Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
193 /eis/lahen...	100	3417	860	7706	1624,86	0,00%	2,1/sec	11,51	5589,2
194 /eis/tunnis...	100	220	30	1552	228,03	0,00%	2,2/sec	6,13	2814,0
195 /eis/tunnis...	100	206	22	1552	275,33	0,00%	2,2/sec	4,88	2229,7
196 /eis/lahen...	100	3399	432	8386	1644,19	0,00%	2,1/sec	11,60	5590,8
197 /eis/lahen...	100	2961	505	8914	1771,28	0,00%	2,2/sec	11,90	5492,5
198 /eis/lahen...	100	811	197	2445	397,18	0,00%	2,3/sec	8,16	3599,2
199 /eis/lahen...	100	135	21	1254	182,54	0,00%	2,4/sec	0,76	328,0
200 /eis/lahen...	100	326	54	1203	215,53	0,00%	2,4/sec	6,89	2949,8
201 /eis/lahen...	100	703	115	1984	446,42	0,00%	2,4/sec	8,69	3686,0
202 /eis/lahen...	100	135	16	1066	157,43	0,00%	2,5/sec	0,79	328,0
203 /eis/lahen...	100	311	45	1507	245,51	0,00%	2,5/sec	7,30	2948,5
204 /eis/lahen...	100	835	171	2980	539,52	0,00%	2,6/sec	9,11	3595,2
205 /eis/lahen...	100	108	15	608	99,17	0,00%	2,7/sec	0,87	327,6
206 /eis/lahen...	100	311	41	1717	316,37	0,00%	2,7/sec	7,91	2950,1
207 /eis/lahen...	100	804	92	2803	584,75	0,00%	2,8/sec	10,00	3683,8
208 /eis/lahen...	100	126	14	1032	146,39	0,00%	2,9/sec	0,92	327,5
209 /eis/lahen...	100	289	37	1649	252,88	0,00%	2,9/sec	8,33	2949,8
210 /eis/lahen...	100	568	112	1779	313,22	0,00%	2,9/sec	10,18	3592,8
211 /eis/lahen...	100	99	15	793	113,35	0,00%	3,0/sec	0,94	327,3
212 /eis/lahen...	100	297	43	1623	272,29	0,00%	2,9/sec	8,50	2949,2
213 /eis/lahen...	100	1984	279	7976	1396,90	0,00%	3,0/sec	16,21	5587,9
TOTAL	10100	717	0	20978	1349,24	1,23%	116,1/sec	349,32	3080,5

Joonis 3. Apache JMeter testitulemuste väljundi näide

3.2 Komponendi esitamise jõudluse mõõtmine

Komponendi esitamise jõudluse mõõtmine erineb eksamitöö jõudluse mõõtmisest eelkõige selle poolest, et kui eksamitöö puhul simuleeritakse kasutaja käitumist, siis komponendi esitamise jõudluse mõõtmise eesmärk on koormata võimalikult ühtlaselt servereid.

Komponendi esitamise jõudluse mõõtmiseks kasutatakse taas Apache JMeter tarkvara, kuid testiplaan on oluliselt erinev kasutajakäitumise jõudlustestist. Kasutajakäitumist

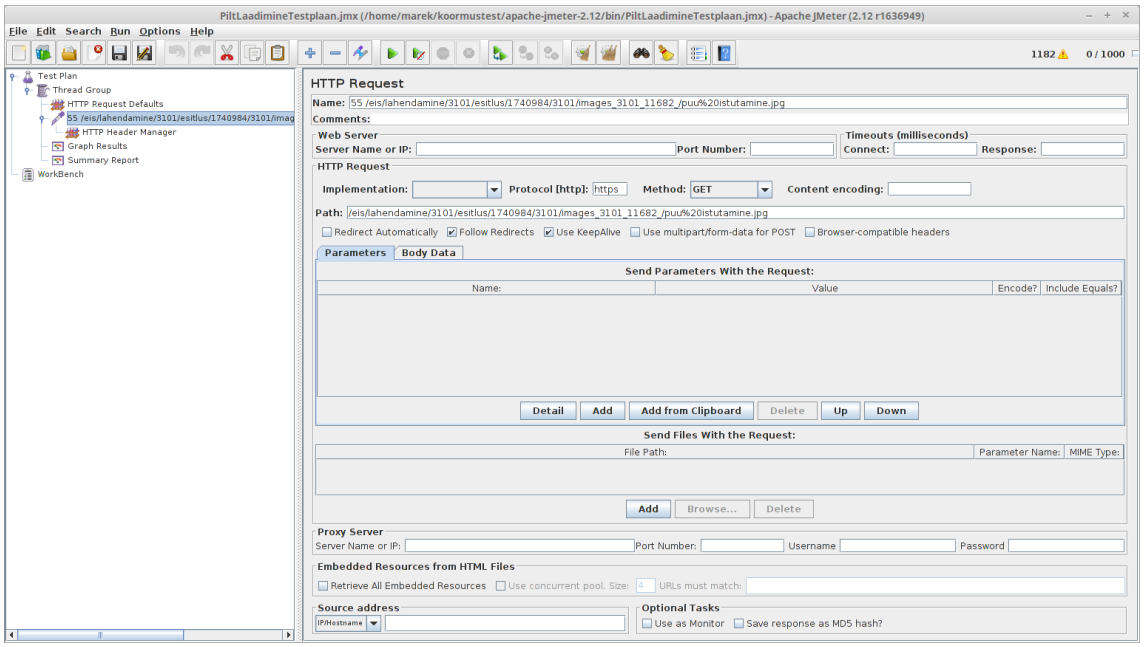
simuleeriv jõudlustest sisaldab kõiki päringu alusel kasutajale saadetavaid faile. Nende hulgas võib olla erinevaid HTML, CSS, pildi- ja helifaile, mida rakendusserver pärib andmebaasist või serveerib staatilise sisu kaustast. Failide mahud ja töötluspõhimõtted on erinevad, seetõttu on nende leidmise ja esitamise aega on väga raske ennustada. Mõõtmisel toimub suur hajuvus ning testitulemused ei ole võrreldavad.

Võrreldavate tulemuste saamiseks tuleb piirata muutuvate parameetrite arvu ja seda on kõige lihtsam teha laadides andmebaasist korduvalt ühte konkreetset faili:

`http://10.101.120.133/eis/lahendamine/1924/esitlus/6524716/1924/images_1924_29679_/Euroopa_kliimavootmed.jpg`

Tegemist on avalikuks lahendamiseks mõeldud geograafia ülesandega „Kliimavöötmed Euroopas“. Faili näol on tegemist Euroopa kliimakaardiga, kuhu lahendaja peab märkima kliimavöötmete nimed. Faili maht on 85 KB.

Apache JMeteri testiplaan koosneb lõimede grupist, HTTP päringute vaikesätetest, ühest HTTP päringust ning andmete kogumise laienditest, millest üks joonistab graafilise väljundi ning teine kogub andmed tabelisse. Testiplaani sisu illustreerib joonis nr. 4.



Joonis 4. Üksikkomponendi jõudlustesti testiplaan

4 Rakenduskihi teenindusvõimekuse tõstmine

Kasutaja töötab infosüsteemiga läbi veebipõhise kasutajaliidese. Veebilehitseja kaudu saadetakse infosüsteemi suunas päring, mille vastuse pärib infosüsteem vastavalt kasutajaliideses defineeritud äriloogikale andmebaasist, töötleb seda ning edastab arvutivõrgu kaudu kasutajale. Iga päringu teenindamiseks on infosüsteemi esmaseks kasutatavaks ressursiks serverite arvutusvõimsus, operatiivmälu ja võrguühendus teenusepakkuja infrastruktuuriga. Ressursivajadus sõltub päringute töötlemise kiirusest, päringu vastuste mahust ja kasutajate arvust. Infosüsteemi arhitektuuri muutmata on võimalik teenindada suuremat arvu kliente infosüsteemi füüsilist jõudlust tõstes ja/või edastatavat andmemahtu vähendades.

E-eksamite, e-tasemetööde ja muude e-testide eelist nähakse õppevara mitmekesistamises ja uute võimaluste tekkes. [1] E-testide arendamisest lähtuvalt püütakse leida nii põnevamaid ülesannete koostamise viise kui ka haaravamat sisu. Uuendusena võimaldab e-ülesanne kasutada mitut tüüpi multimeediat, mida paberi peal varem teostada ei olnud võimalik, nagu liikuvad pildid ja videod, kuulamise ja kõnelemise ülesanded. Optimaalse kvaliteedi parima praktika kujundamine ja selle integreerimine infosüsteemi koosseisu, on kindlasti üks ülesannetest, mis e-testide arendusmeeskonnal tulevikus ees seisab.

Praegusel hetkel tähendab e-testide katseline läbi viimine mitte kõige optimaalsemaid protsesse nii korraldusest kui infosüsteemi võimalustest ja võimekusest lähtuvalt. Seetõttu on vajalik infosüsteemi tehniline ülesehitus muuta selliseks, mis võimaldaks kiiresti lisada vajaminevat ressurssi. Kasutajatelt tulevaid päringuid peab saama hajutada ja koormust vajadusel ümber suunata ja mitme rakendusserveri vahel jagada. Infosüsteemi funktsionaalsus ei tohi seejuures kannatada, toimima peab jääma Eestit muust maailmast eristav isikutuvastus ID-kaardiga.

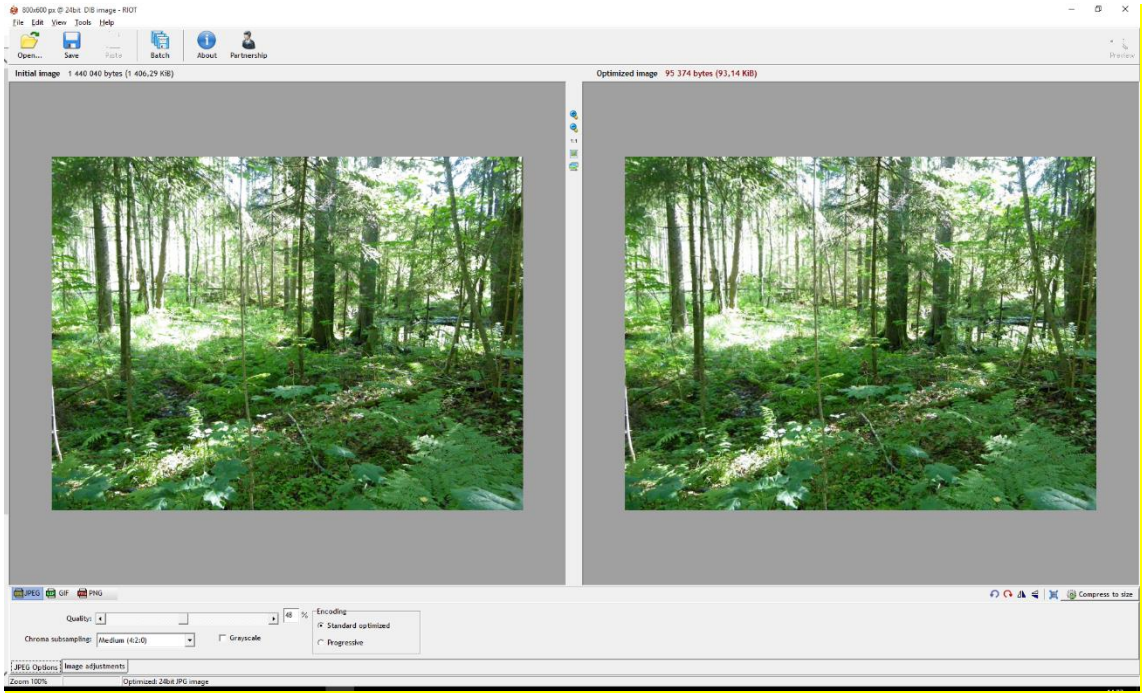
4.1 Sisu mahu töötlemine

Arvutite vahendusel toimuv haridustaseme mõõtmine on nii Eesti Vabariigis kui ka Sihtasutuses Innove väga värske protsess. Töid koostavad spetsialistid ei ole varem pidanud andmemahuga arvestama ja sellest tulenevalt puuduvad protsessis ka sisufailide optimeerimistegevused. Praegusel hetkel imiteeritakse veel pabertesti ülesandeid, kus

multimeediana on kasutusel pildi- ja helifailid, kuid esimesi katsetusi tehakse ka videofailide esitamise ja heli salvestamisega.

Innove arvutite kettatõmmise ettevalmistamisel on püütud arvestada ainespetsialistide võimaliku vajadusega töödelda erinevaid multimeedia faile. Seetõttu on seadmetesse paigaldatud muuhulgas vabavaralised tarkvarad IrfanView ja Audacity.

Pildifaili mahu vähendamiseks avame selle IrfanView tarkvaraga, salvestamiseks kasutame sisseehitatud tööriista spetsiaalselt veebis avaldamiseks mõeldud piltide töötlemiseks.

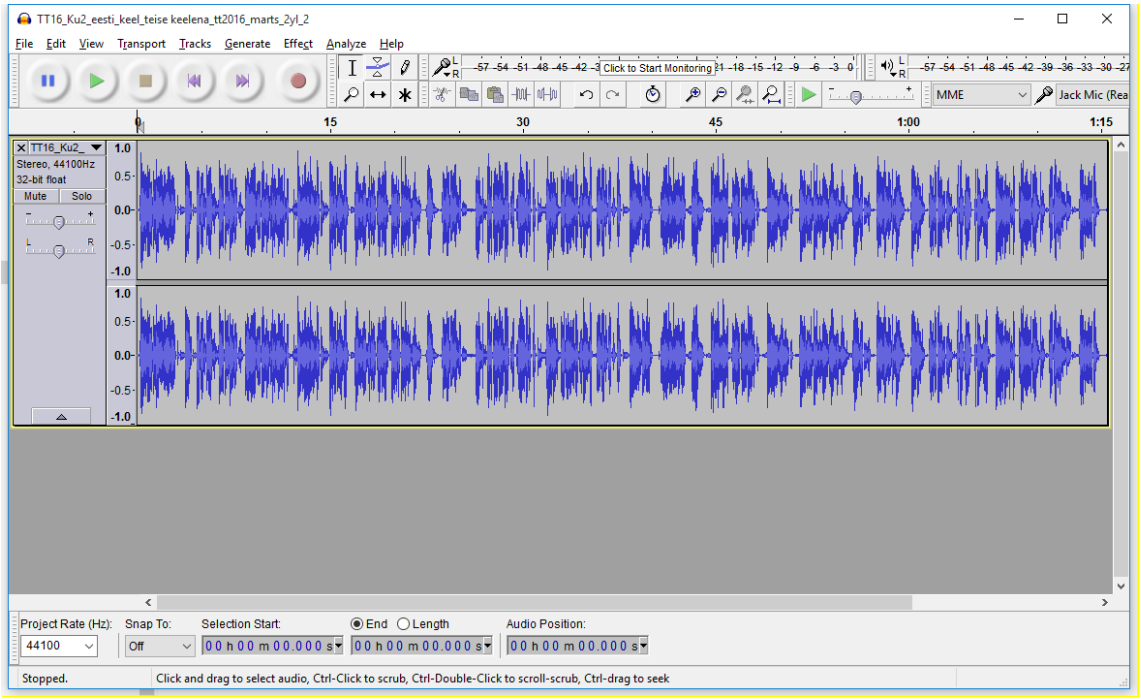


Joonis 5. Pildifaili redigeerimine IrfanView tarkvaraga

Erineva arvutikasutusoskusega töötajatel on väga lihtne heebliga kvaliteediparameetrit muutes leida minimaalne aktsepteeritav kvaliteet. Eraldusvõime osas üle 800 pikslise küljemõõduga pilte ei ole lubatud ülesannetes tarvitada, seega on otstarbekas ka külgede suhe sobivaks seadistada. Käesoleval juhul õnnestus redigeerides 1406,29 KB fail vähendada mahuni 93,14 KB.

Helifaili vähendamiseks avame selle Audacity tarkvaraga ja salvestame endise 320 Kbps kvaliteediparameetri asemel 145-185 Kbps näitajaga. Tulemusi võrreldes ei suutnud käesoleva töö autor vähimalgi määral kuuldavat kvaliteedilangust tabada. Tõenäoliselt

võiks kvaliteedinäitajat veelgi allapoole tuua, kuid selleks oleks vaja mõnda teist tööriista – Audacity versioon 2.1.2. madalamat bitikiirust vaikesätetena ei võimaldanud.



Joonis 6. Helifaili redigeerimine Audacity tarkvaraga

Kõikide helifailide töötlemine annab alljärgneva tabeliga illustreeritud tulemuse.

Tabel 1. Helifailide töötlemise tulemus

Optimeerimata	Optimeeritud	Failinimi
375,36 KB	229,51 KB	TT14_Ku1_0.näidis(1).mp3
666,33 KB	228,05 KB	TT16_Ku11_eesti_keel_teise keelena_.mp3
620,41 KB	266,66 KB	TT16_Ku12_eesti_keel_teise keelena _2.mp3
633,67 KB	240,79 KB	TT16_Ku13_eesti_keel_teise keelena_.mp3
867,35 KB	324,74 KB	TT16_Ku14_eesti_keel_teise keelena_.mp3
5166,33 KB	1849,86 KB	TT16_Ku2_eesti_keel_teise keelena_.mp3
8329,43 KB	3139,61 KB	
8,13 MB	3,07 MB	

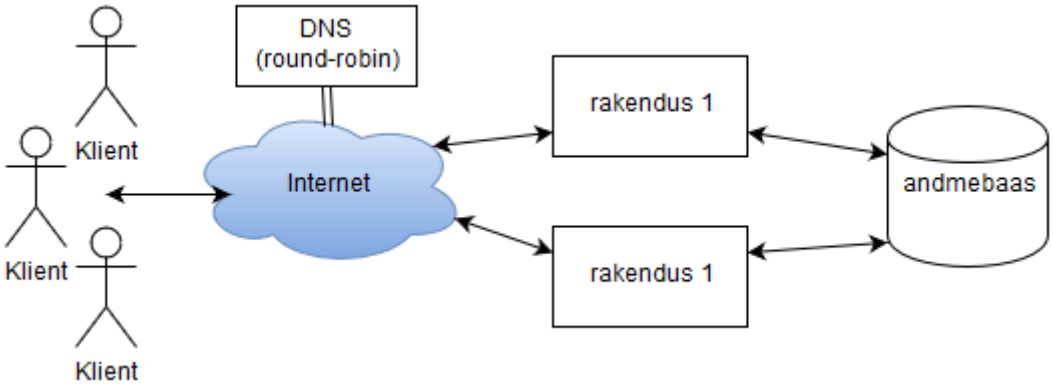
Helifailide ümber salvestamine annab ligikaudu 2,5 kordse säästu, pildifailide puhul, mille arv testitöös on ligikaudu 40 ning mida seetõttu pole ka tabelina kujutatud, vähenes andmemaht kokkuvõttes ligikaudu 9 korda – esialgsest 12,74 megabaiti vähenes 1,42-ni. Summeerituna oleks iga eksaminandi puhul pidanud süsteem edastama 20,88 MB, mida õnnestus vähendada 4,49 megabaidini ehk ligikaudu 5 korda. Mõnekümne kliendi

teenindamiseks taolist optimeerimist ette võtta ei oleks mõtet, kuid näiteks 400 lahendaja puhul võib see halbade juhuste kokkulangemisel tähendada infosüsteemi välisühenduse täielikku ummistumist.

4.2 Koormuse jaotamine rakendusserverite vahel

Sisu mahu optimeerimine ei lahenda teenindusvõimekuse probleemi täielikult, sest kliendi päringuid jääb teenindama üks rakendusserver. Rakendusserveri ressurss on infosüsteemi teenindava füüsilise riistvara ressursiga piiratud see seab ka infosüsteemi üleüldisele jõudlusele kindlapiirilise piirangu.

Antud probleemi on võimalik lahendada nii kliendi kui ka teenusepakkuja poolelt. Lihtsaimaks viisiks mitut rakendusserverit kasutada ja koormust mitme teenindava serveri vahel jagada, on pakkuda läbi domeeninimede süsteemi (DNS) klientidele mitu teenusele vastavat IP aadressi, mille järjekorda päringute järel muudetakse (*round-robin dns*).



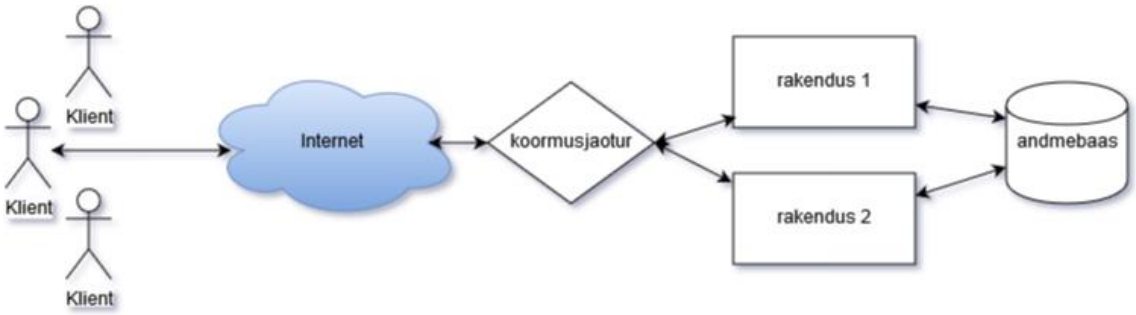
Joonis 7. DNS'i kaudu toimiv koormusjaotus

Lahenduse negatiivseks pooleks on suurenev koormus teenusepakkuja nimeserverile ja kohustus tagada mitme rakendusserveri üheaegne töötamine, mistõttu pole see Eksamite infosüsteemi puhul praeguses etapis aktsepteeritavaks lahenduseks. Antud lahendust võib kaaluda tulevikus mitmes erinevas asukohas EIS-i töös hoidmiseks, kuid selle tarbeks tuleb leida viis, kuidas erinevates andmekeskustes asuvaid andmebaase piisava kiirusega sünkroonis hoida ning väline DNS server asutuse haldusesse võtta.

Teenusepakkuja poolel koormusjaoturi funktsiooni täitvaid tooteid on turul saadaval mitmeid. Olemas on nii kommertslahendusi kui ka vabavaralisi. Tuntuimate ärilahenduste tootjatena võib nimetada näiteks F5, Riverbed, sobivaid lahendusi leiab kindlasti ka Cisco tooteportfellist. Riigihangete registri päringutest lähtuvalt võivad neid või analoogseid lahendusi kasutada riigisektorist näiteks Siseministeeriumi Infotehnoloogia- ja Arenduskeskus (riigihanke viitenumber: 171116), Riigi Infosüsteemi Amet (riigihanke viitenumber: 142676), Eesti E-tervise Sihtasutus (riigihanke viitenumber: 134663).

Leidub ka litsentsitasudeta kasutatavaid lahendusi, mida antud töös alljärgnevalt vaadeldakse.

Kui traditsiooniliselt kasutatakse terminit *proxy* infosüsteemi komponendi puhul, mis on vaheliideseks kliendi ja internetist teenust pakkuva serveri vahel, siis käesoleva töö kontekstis viitame sama terminiga *reverse proxy* nimelisele lahendusele, mida võib eesti keelde tõlkida kui lüüsi või koormusjaoturit. *Reverse proxy* lahendust visualiseerib joonis nr 7.



Joonis 8. Lüüsi kaudu toimiv koormusjaotus

Sõltumata sellest, kas tegemist on tasulise või tasuta lahendusega, lisatakse infosüsteemi ülesehitusse spetsiaalne komponent - lüüs, mille peamiseks ülesandeks on kasutajalt tulevate päringute vastu võtmine, nende edastamine administraatori poolt defineeritud algoritmi alusel rakendusserveritele ning vastuse vahendamine kasutajale.

Kõikide valikuvariantide sobivuse hindamise tingimusteks on lahenduse koormuse jaotamise funktsionaalsus, krüpteeritud andmesidekanali loomise võimekus kliendi ja Eksamite infosüsteemi vahel ning EIS-i funktsionaalsuse säilimine.

Eksamite infosüsteemi lähtekoodis kasutatakse erinevaid Apache veebiserveri keskkonnamuutujaid erinevatest kanalitest süsteemi sisenevate kasutajate tuvastamiseks kui ka eksamikeskkonna turvalisuse tagamiseks. Vajaminevad muutujad on välja otsitud Eksamite infosüsteemi lähtekoodist, selgitused pärinevad Apache veebiserveri dokumentatsioonist. [7]

Tabel 2. EIS-is kasutatavad SSL muutujad

Muutuja nimetus	**Selgitus**
SSL_CLIENT_I_DN	Kliendi sertifikaadi väljastaja DN
SSL_CLIENT_I_DN_CN	Kliendi sertifikaadi väljastaja CN (CN väärtus DN-ist)
SSL_CLIENT_VERIFY	Kliendi sertifikaadi kontrolli tulemus
SSL_CLIENT_S_DN	Kliendi sertifikaadi DN
SSL_CLIENT_S_DN_CN	Kliendi sertifikaadi CN (CN väärtus DN-ist)
SSL_CLIENT_S_DN_O	Kliendi sertifikaadi väljastaja (O väärtus DN-ist)
SSL_CLIENT_M_SERIAL	Kliendi sertifikaadi seerianumber
SSL_CLIENT_CERT	PEM kodeeringus kliendi sertifikaat

Selleks, et süsteem saaks õigesti toimida, peavad vastavad väljad olema koormusjaoturi poolt väärtustatud andmetega sellises formaadis, mida tagasüsteem töödelda oskab. Alljärgnevas tabelis on välja toodud muutujatele vastavate väljade väärtuste näidised.

Tabel 3. Rakendusserveri poolt oodatavad väärtused

Muutuja nimetus	**Oodatava väärtuse näide**
SSL_CLIENT_I_DN	emailAddress=pki@sk.ee,CN=ESTEID-SK 2011,O=AS Sertifit-seerimiskeskus,C=EE
SSL_CLIENT_I_DN_CN	ESTEID-SK 2011
SSL_CLIENT_VERIFY	*NONE, SUCCESS, GENEROUS, FAILED*:põhjus
SSL_CLIENT_S_DN	*serialNumber*=38305166518,GN=MAREK,SN=LENDSAAR, CN=LENDSAAR\,MAREK\,38305166518,OU=*authenti-cation*, O=ESTEID,C=EE
SSL_CLIENT_S_DN_CN	LENDSAAR,MAREK,38305166518
SSL_CLIENT_S_DN_O	ESTEID
SSL_CLIENT_M_SERIAL	61CABB328C380ECA539AB0E26E334765

SSL_CLIENT_CERT	-----BEGIN CERTIFICATE----- I5GAQEwCAYGBACORgEEMB8 GA1UdIwQYMBaAFHtq8lVQXLjZegiHQa76ois9W1d2MEAGA1U dHwQ5MDcwNaAzoDGGL2h0dHA6Ly93d3cuc2suZWUvcmVwb3 NpdG9yeS9jcmxzL2VzdGVpZDIwMTEuY3JsMA0GCSqGSIb3DQ EBBQUAA4IBAQBNgdOdvzy15m2MgqRAqiWR0xdyOk9GlS0kcj 2/ZGZ8e6z70gE1ns4p0kNUhcW9O+D7qROURy/Kc0nsihpEmJWE SdgPj4RWw8TGPiIPie2ELkXIebykTCPBHLczUf33nsM7uTg0B4 MOe1eJB6dyvjmOYIr7W59pnwRTkHUc8J5Xu9JWswKWj0/QpJd yHz72b7nT65hOb7V9wPueVTHZT1KZGYAzmhFIky5DSBgDU5v +a20Crdex92JSIJxb87r0T3UycrZ7NWNHc6YLwTO/r2vHXNcshvd IX6MvRu22icsp1ZaPaplacNXxT7VLa9EUk/uNzEchaqRmwOqlw GheESlB -----END CERTIFICATE-----

Lisaks sertifikaatide probleemile, on koormust jaotava komponendi infosüsteemi lisamise negatiivseks pooleks ka täiendava rikkepunkti lisandumine, kuid kui ülesehituselt lihtsama ja väga harva muudatusi nõudva koormusjaoturi taasteprotsess peaks olema oluliselt kiirem, kui keeruka rakendusserveri taastamine, mis võimaldaks koormusjaoturi kasutamine viia ühe rakendusserveri rikke mõju kliendile märkamatuks. Võimaliku lahenduse leidmiseks vaadeldakse kolme vabavaralist tarkvara.

4.2.1 HAProxy

HAProxy on tasuta ja suure jõudlusega TCP ja HTTP põhinevatele rakendustele *proxy* funktsiooni pakkuv tarkvara. HAProxy võib täita TCP proxy, HTTP lüüsi (gateway, reverse proxy), SSL töötleja (SSL ühenduste algatamine, lõpetamine, ümber suunamine), koormusjaoturi ja mitmeid teisi ülesandeid. [7] Seejuures rõhutatakse HAProxy dokumentatsioonis, et rakendus ei ole vahemäluna toimiv proxy (*caching proxy*) ega ka veebiserver. Selle kirjelduse järgi on HAProxy potentsiaalselt sobiv kandidaat täitma EIS-i koormusjaoturi rolli ning tarkvara eeldatavalt väike ressursinõudlikkus ja lihtne konfiguratsioon teevad temast esmase eelistuse.

Ootuspäraselt on HAProxy koormusjaoturis vastava funktsionaalsuse konfigureerimine lihtsaks tehtud. Defineerida tuleb kaks sektsiooni - *frontend* ja *backend*. Esimene neist kirjeldab, kuidas päringuid vastu võetakse ja kuidas need tagasüsteemi (*backend*) edastatakse ning teine defineerib koormusjaotuse algoritmi, sihtserverid ning vajadusel täiendavad parameetrid. [8]

Dokumentatsiooni järgi on HAProxy tehniliselt võimeline teenindama nii HTTP kui HTTPS ühendusi. Turvalise kanali kasutamiseks tuleb tarkvara konfiguratsiooni *frontend* sektsioonis seadistada proxy kuulama 443 porti ja konfigureerida viide sertifikaadile. Sertifikaat ja privaatvõti peavad olema PEM formaadis ning asuma samas failis, tarkvara tuleb suunata õiget faili lugema direktiiviga *crt*. Aktsepteeritud CA'te nimekiri seadistatakse direktiiviga *ca-file*, kus kõik aktsepteeritud juursertifikaadid on PEM formaadis antud faili lisatud, vahesertifikaadid eespool ja juursertifikaadid lõpus.

Kliendi sertifikaatide kehtivust on võimalik kontrollida tühistusnimekirjade alusel. HAProxy toetab PEM formaadis tühistusnimekirjasid, mida saab eristada DER formaadist lugedes tekstiredaktoriga faili sisu. PEM formaadis CRL algab kirjega „-----*BEGIN X509 CRL*-----" ja lõppeb „-----*END X509CRL*-----".

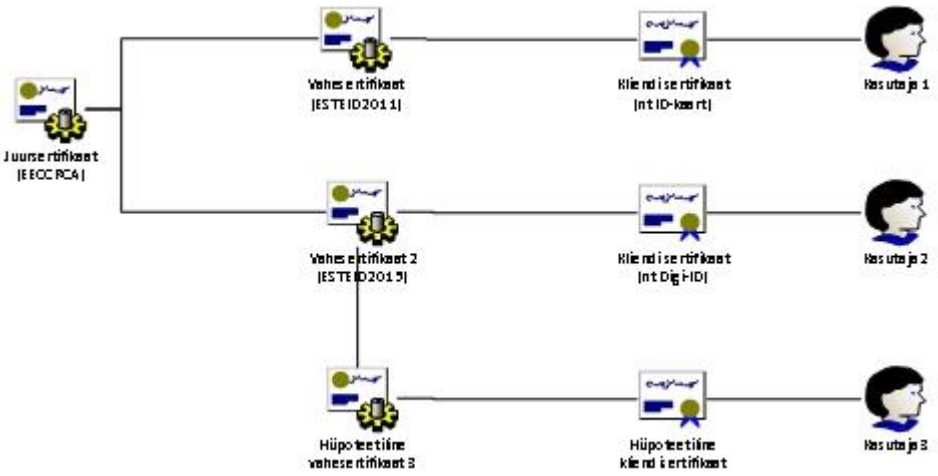
Sertifitseerimiskeskuse väljastatud tühistusnimekirjad on teistsugused, proovime vabavaralise openssl'iga konverteerida need sobivaks. Linuxi konsoolilt käivitatav käsk näeb välja järgnev:

```
openssl crl -inform DER -in esteid2011.crl -outform PEM -out esteid2011.pem
```

Olles konverteerinud kõik SK väljastatud tühistusnimekirjad sobivasse formaati, tuleb järgneva sammuna need ühte faili kokku tõsta ning *crl-file* direktiiviga panna tarkvara õiget faili lugema.

Mitmete katsete tulemusena tühistusnimekirjade kontroll tööle ei hakanud. Vea põhjust jälitades selgub, et kui sertifikaati kontrollida aegunud CRLi vastu, siis teavitatakse kasutajat brauseri kaudu kehtivuse lõpetanud sertifikaadist (ilmselt mõeldud CRL'i) ning koormusjaoturi logisse tuleb teade, et kliendi sertifikaati ei usaldata. Sellest järeldub, et mingisugune CRLi lugemine siiski toimub. Kui tühistusnimekirjad uuendada ning uuesti kasutajatuvastust proovida, teavitati brauseri kaudu kasutajat ilmnenud tundmatust veast ja logisse kirjutati, et kliendi CA sertifitseerimisjada ei õnnestunud kontrollida (*SSL client CA chain cannot be verified*).

Varasema kogemuse põhjal on osades tarkvarades oluline sertifikaatide kindel järjekord. Sertifitseerimisjada peaks sisaldama loogilises järjekorras kõiki sertifikaate, mis võimaldavad kliendi autentsust tuvastada.



Joonis 9. Sertifitseerimisraja tutvustus

Joonisel nr. 8 kujutatakse kolme kasutajat, kelle kliendisertifikaadid pärinevad erinevatest vahesertifikaatidest, kuid samast juursertifikaadist. Kõikide juur- ja vahesertifikaatide kohta peab teenusepakkuja pidama tühistusnimekirja ning kliendisertifikaadiga autentimisel vajab HAProxy neid kõiki. Loogiline järjekord tähendab siinkohal, et kehtivuse kontroll toimub järjest faili läbi lugedes, kliendi sertifikaadile lähemal olevad tühistusnimekirjad peavad olema eespool juursertifikaadi omast. Näiteks kasutaja 3 sertifitseerimisjada on järgnev: hüpoteetilise vahesertifikaadi nr. 3 tühistusnimekiri -> vahesertifikaadi nr 2 tühistusnimekiri -> juursertifikaadi tühistusnimekiri. Juhul, kui näiteks vahesertifikaadi nr 2 CRL on hüpoteetilise vahesertifikaadi nr 3 CRL taga, siis sertifikaadi kehtivuse kontroll ebaõnnestub. Juursertifikaadi CRL peab olema faili lõpus. Paraku antud muudatus tulemust ei andnud. Põgusal uurimisel selgus, et erinevad kasutajad on põrkunud samasuguste probleemide otsa ja sellest interneti foorumites teada andnud. Lahendusena soovitatakse üle kontrollida kõikide vahesertifikaatide (*intermediate certificate*) olemasolu ja käesoleval juhul juba proovitud tühistusnimekirjade järjekord failis.

Suurt probleemi sertifikaatide kehtivuskinnituse kontrolli mittetoimimine endas ei sisalda, sest vajadusel saab antud tegevust teha tagasüsteemis, mille puhul on teada, et seal kontroll toimib.

Eksamite infosüsteemi korrektseks funktsioneerimiseks ja ID-kaardiga autentimise tagamiseks peab koormusjaotur edastama sertifikaadid tagasüsteemile. Selleks tarbeks tuleb konfiguratsioonis soovitud parameetreid töödelda sarnaselt:

```
http-request set-header SSL_CLIENT_S_DN_CN %{+Q}[ssl_c_s_dn(cn)]
```

Muutujate sisulise vastavustabeli leiate alljärgnevalt. Kahjuks EIS-i tarbeks vajalik vormiline kuju on valdaval enamusel juhtudest täiesti erinev. Reaalselt oleks võimalik teatud muutujaid kasutada kui neid rakendustarkvaras ümber konverteerida.

Tabel 4. HAProxy muutujate väärtustamine Apachele sobivasse formaati

Apache keskkonnamuutuja	**HAProxy vastav muutuja**
SSL_CLIENT_I_DN	%{+Q}[ssl_c_i_dn]
SSL_CLIENT_I_DN_CN	%{+Q}[ssl_c_i_dn(cn)]
SSL_CLIENT_VERIFY	* %[ssl_c_verify]
SSL_CLIENT_S_DN	%{+Q}[ssl_c_s_dn]
SSL_CLIENT_S_DN_CN	%{+Q}[ssl_c_s_dn(cn)]
SSL_CLIENT_S_DN_O	%{+Q}[ssl_c_s_dn(o)]
SSL_CLIENT_M_SERIAL	* %[ssl_c_serial]
SSL_CLIENT_CERT	* %[ssl_c_der]

* - mittesobivas formaadis parameetrid

HAProxy puhul näitab *ssl_c_verify* probleemi koodi, mille taha kasutaja tuvastamine takerdus. Probleemide puudumisel kuvatakse 0 kuid Apache nõuaks seal väärtust *SUCCESS*. Eriti keeruline on *SSL_CLIENT_M_SERIAL* ja *SSL_CLIENT_CERT* väärtustega, sest kui teiste väärtuseid esitatakse lahtise tekstina, siis mainitud väärtused on HAProxyle iseloomulikult binaarformaadis. Nende konverteerimist käesoleva töö raames ei proovitud.

HAProxy tarkvarasse on täiendava moodulina ehitatud veebipõhine koormusjaoturi haldusliides, mille kaudu saab administraator seirata süsteemi seisukorda ja vajadusel lülitada tagaservereid klastrist välja. Seda tuleb antud projekti raames tuleks käsitleda kasuliku lisaväärtusena.

Ajutine klastrist eemaldamine on hea infosüsteemi uuendamise kontekstis. Eksamite infosüsteemi rakenduse uuendamine võtab sõltuvalt muudatuste suurusest mõned minutid, mille jooksul rakendusserver ei saa kliente teenindada. Traditsioonilisse

süsteemi, näiteks EIS-i testkeskkonda, kus kliendid edastavad päringud otse rakendusserverile, tuleks ehitada süsteem, mis teavitaks kliente hooldustöödest. Koormusjaoturi ja mitme rakendusserveri puhul vajadus puudub, sest sel ajal, kui ühe klastri sõlmega hooldustöid teostatakse, lülitab administraator vastava sõlme klastrist välja, kuid teine jätkab klientide teenindamist ning see võimaldab suurema osa hooldustöid teha töö ajal kliendile märkamatult.

HAProxy näol on tegemist üldjuhul lihtsa ja hea tarkvaraga, mille kohta on koostatud korralik dokumentatsioon. Tarkvara sobib kindlasti teatud lahenduste puhul koormusjaoturiks, kuid sellisel juhul on mõistlik infosüsteem kohe alguses HAProxy võimalusi arvestades disainida. Eksamite infosüsteemi puhul eeldaks lahenduse kasutuselevõtt teatud määral ümberehitusi, mida asutus pigem teha ei soovi.

4.2.2 Nginx

Nginx on laialt kasutatav avatud lähtekoodiga veebiserveri tarkvara, mida on arendatud 2002. aastast alates. [7] Tarkvara arendamist koordineerib ettevõte Nginx, Inc., mis ühtlasi pakub tootest ka kommertsversiooni koos juurde kuuluva tugiteenusega. Veebiserver on laialdase populaarsuse võitnud eeskätt tänu väga heale jõudlusele ja lüüsi funktsionaalsus on üks, mida eraldi erinevates materjalides rõhutatakse.

Sarnaselt HAProxy tarkvarale, suudab Nginx mitmel erineval viisil kasutaja päringuid rakendusserveritele vahendada, kuid lisaks on tegemist laialdaselt levinud veebiserveriga, millel ühtlasi ka ID-kaardi infrastruktuuri tugi ja sellest tulenevalt võiks olla sobilik EIS-i rakendusserverite vahel koormust jaotama.

Nginxi rakendamiseks antud rollis tuleb ette valmistada eraldi server, millel toimivad võrguühendused ning võimekus tarkvara repositooriumist pakette laadida. Sisenevate päringute suunamine tagaserverile toimub lihtsasti direktiivi *proxy_pass* abil, sealjuures saab sihtkohaks olla HTTP veebiserver kui Unix sokkel (*proxy_pass* http://unix:/sNginx.sock:/uri/;) Mitme tagaserverina toimiva rakendusserveri defineerimiseks võimaldab Nginx defineerida spetsiaalse *upstream* ploki, millele suunatakse liiklus eeltooduga sarnaselt. [9]

Kliendiühenduste turvalisuse tagamiseks on tarvilik sisse lülitada ka SSL moodul, mis võimaldab hüperteksti edastada pealtkuulamist raskendaval meetodil, võimaldades lisada

veebiserveri konfiguratsiooni usaldusväärse kolmanda osapoole poolt väljastatud sertifikaadi (direktiivid *ssl_certificate* ja *ssl_certificate_key*), mille autentsust kasutaja veebilehitseja ühenduse loomisel kontrollib.

Kliendi sertifikaatide kehtivust võimaldab veebiserver kontrollida nii OCSP kui sertifitseerimisasutuse tühistusnimekirjade alusel. CRLi alusel kontrollimiseks lisatakse Nginx veebiserveri konfiguratsiooni ssl_crl direktiivi abil viide tühistusnimekirja failile. ID-kaardi sertifikaatide väljastaja AS Sertifitseerimiskeskus hoiab kõikide juursertifikaatide tühistusnimekirju eraldi ja see lahendus ei ole Nginxi poolt toetatud. Piirangust üle saamiseks kasutame eelmises peatükis kirjeldatud protsessi: konverteerime tühistusnimekirjad PEM formaati ning lisame kõik CRLid ühte faili selliselt, et vahesertifikaadid on ees ja juursertifikaadid lõpus. Seejärel võib testida kehtivuskontrolli toimivust. [11]

Kasutajatuvastuse testimisel ebaõnnestub Nginxis kliendisertifikaadi kehtivuse kontroll. Silumisrežiim väljastab veebiserver logisse järgnevad kirjed:

*[debug] 18322#0: *1 verify:0, error:44, depth:2, subject:"/C=EE/O=AS Sertifitseerimis-keskus /CN=EE Certification Centre Root CA/emailAddress=pki@sk.ee", issuer: "/C=EE/O=AS Sertifitseerimiskeskus/CN=EE Certification Centre Root CA/emailAddress=pki@sk.ee"*

*[debug] 18322#0: *1 verify:0, error:3, depth:2, subject:"/C=EE/O=AS Sertifitseerimis-keskus /CN=EE Certification Centre Root CA/emailAddress=pki@sk.ee", issuer: "/C=EE/O=AS Sertifitseerimiskeskus/CN=EE Certification Centre Root CA/emailAddress=pki@sk.ee"*

*[debug] 18322#0: *1 verify:1, error:3, depth:2, subject:"/C=EE/O=AS Sertifitseerimis-keskus/CN=EE Certification Centre Root CA/emailAddress=pki@sk.ee", issuer: "/C=EE/O=AS Sertifitseerimiskeskus/CN=EE Certification Centre Root CA/emailAddress=pki@sk.ee"*

*[debug] 18322#0: *1 verify:1, error:3, depth:1, subject:"/C=EE/O=AS Sertifitseerimis-keskus/CN=ESTEID-SK 2011/emailAddress=pki@sk.ee",issuer: "/C=EE/O=AS Sertifitseerimiskeskus/CN=EE Certification Centre Root CA/emailAddress=pki@sk.ee"*

*[debug] 18322#0: *1 verify:1, error:3, depth:0, subject: "/C=EE /O=ESTEID /OU=authentication/CN=LENDSAAR,MAREK,38305166518/SN=LENDSAAR/GN=MAREK/ serialNumber=38305166518",issuer: "/C=EE/O=AS Sertifitseerimiskeskus /CN=ESTEID-SK 2011/emailAddress=pki@sk.ee"*

Antud probleemi olemus ei lähe kokku dokumentatsiooniga, sest sertifitseerimisjada on täielik, vastavad paarid on värvidega tähistatud, kuid nagu HAProxy puhul, ei ole kliendi sertifikaadi kontrolli mitte-toimimine koormusjaoturis süsteemi üldist tööd silmas pidades halvav, sest äärmisel vajadusel on võimalik vastav funktsionaalsus realiseerida ka tagaserverites.

Küll on süsteemi tööks oluline ID-kaardiga logimine ja kohaliku serveri autentimine ja selle toimima saamiseks tuleb rakendusserveritesse kliendi sertifikaat edasi saata. Nginx veebiserveris kasutatakse selleks direktiivi *proxy_set_header*, mis võimaldab järgmisesse serverisse saadetava päringu päise ümber kirjutada või teatud info saatmata jätta määrates väärtuseks tühja *string*i. [9] Seadistame Nginxi saatma edasi kõiki kasutatavaid muutujaid ning väärtustama neid Apachele sobivasse formaati alljärgneval viisil:

```
proxy_set_header    SSL_CLIENT_CERT  $ssl_client_cert;
```

Muutujate vastavustabel on toodud tabelis nr. 6.

Tabel 5. Nginx muutujate väärtustamine Apachele sobivasse formaati

Apache keskkonnamuutuja	**Nginxi vastav muutuja**
SSL_CLIENT_I_DN	$ssl_client_i_dn
SSL_CLIENT_I_DN_CN	* $ssl_client_i_dn_cn
SSL_CLIENT_VERIFY	$ssl_client_verify
SSL_CLIENT_S_DN	$ssl_client_s_dn
SSL_CLIENT_S_DN_CN	* $ssl_client_s_dn_cn
SSL_CLIENT_S_DN_O	* $ssl_client_s_dn_o
SSL_CLIENT_M_SERIAL	* $ssl_client_serial
SSL_CLIENT_CERT	$ssl_client_cert

* - ise koostatud muutujad

Erinevalt HAProxyst, on Nginx oluliselt paremini Apachega kokku sobivam. Muutujad, nagu *SSL_CLIENT_VERIFY*, *SSL_CLIENT_CERT* on samas formaadis Apache poolt kasutatavatega. *SSL_CLIENT_I_DN ja SSL_CLIENT_S_DN* on sisaldavad sisu mõttes

vajalikke komponente, kuid edastatavad DN sisu erinevas järjestuses. EIS-i rakenduse tööd kasutaja tasemel see ei mõjuta, täiendada tuleb kohaliku serveriga suhtlemise liidest.

Esmasel seadistamisel tundus, et SSL_*CLIENT_I_DN_CN, SSL_CLIENT_S_DN_CN* ja *SSL_CLIENT_M_SERIAL* muutujaid väärtustada ei ole võimalik ning ka nende sobivaks töötlemine tuleks planeerida rakendusse, kuid ühel hetkel pakkus Nginxi tühistusnimekirjade probleemile lahenduse GitHubist Moloch paketifiltri Wiki lehekülg aadressilt https://github.com/aol/moloch/wiki/Client-Side-Certificate-Auth-in-Moloch. Seal kirjeldab autor toimivat võimalust, kuidas Nginxis *common name* ja *serial* regulaaravaldise abil *distinguished name*-ist välja filtreerida. Tulevikuperspektiivi silmas pidades on tegemist väga kasuliku leiuga, sest antud parameetrit kasutab päris mitu infosüsteemi ning vastava muutuja väärtustamine veebiserveri poolt võimaldaks infosüsteemi lihtsasti migreerida Nginx'i kasutama. Seepärast pean vajalikuks antud teisenduse ka käesolevas töös välja tuua.

```
# get CN
map $ssl_client_s_dn $ssl_client_s_dn_cn {
    default "should_not_happen";
    ~/CN=(?<CN>[^/]+) $CN;
}

# get serial
map $ssl_client_s_dn $ssl_client_s_dn_serial {
    default "should_not_happen";
    ~/serialNumber=(?<serialNumber>[^/]+) $serialNumber;
}
```

Sarnaselt koostatakse ka vajaminevad $ssl_client_s_dn_o ja $ssl_client_i_dn_cn muutujad, mis koos ülejäänutega tagasüsteemis asuvale Apachele edasi saadetakse.

Viimase muudatusega said koormusjaoturile püsitatud nõudeid täidetud ning veendumaks lahenduse toimivuses, proovime seda EIS-i testkeskkonnas kasutada. Eriti olulised läbi proovimiseks on vastavaid muutujaid kasutavad funktsioonid, nagu erinevate kaartidega autentimine ja kohaliku serveri tuvastamine.

Testimise tulemusena selgub, et kõik EIS-i poolt toetatavad autentimisviisid toimivad ka koormusjaoturiga. Lokaalserveri registreerimine ebaõnnestub, sest oodatavast

sertifikaadist välja loetav andmestik on formaadilt erinev, kuid vastav muudatus on rakendustarkvaras teostatav.

Eksperimentide tulemusest lähtuvalt sobib Nginx veebiserver mööndustega täitma Eksamite infosüsteemi koormusjaoturi ülesandeid. HAProxy kogemusest lähtudes võiks Nginx sisaldada ka veebipõhist *reverse proxy* haldusliidest, mis võimaldaks süsteemiadministraatoril efektiivselt süsteemi seirata, operatiivselt probleemidele reageerida ja tagasüsteemi komponente vajaduse tekkides suvalisel ajahetkel hooldada, kuid vastavat moodulit pole süsteemile veel loodud. Sellest hoolimata oleks Nginx tarkvara sobiv kandidaat koormusjaoturi funktsiooni täitma.

4.2.3 Apache

Apache on 20-aastase ajalooga praeguseks üks enimkasutatavaid veebiservereid, mis tänu oma paljudele erinevatele moodulitele on võimeline täitma mitmeid erinevaid funktsioone, olema vajadusel nii rakendusserveriks kui täitma ka koormusjaoturi ülesandeid. Modulaarus ja suur erinevate moodulite arv on Apache puhul ühelt poolt eeliseks ja teisalt puuduseks – tarkvara võib töötada mitmes erinevas multiprotsessi režiimis (Linuxis *prefork, worker, event*), kuid osade platvormide moodulid, nagu näiteks PHP (*mod_php5*), ei võimalda lõimesid toetava multiprotsessimooduli kasutamist. [6] Sellest tulenevalt ei kasutata riistvara kõige optimaalsemalt ning teenindatavate ühenduste arv jääb väikeseks. Käesolevas projektis antud piirang ei kehti, sest *proxy* moodul võimaldab kasutada kõiki multiprotsessi režiime.

Apachel põhineva koormusjaoturi eeltingimuseks on toimiv server koos operatsioonisüsteemiga, millel on häälestatud võrguühendus ja süsteem on võimeline ametlikust paketihoidlast tarkvara alla laadima. Füüsilisse serverisse peaks olema paigaldatud Apache2 tarkvarapakett koos sõltuvate lisadega.

Võrreldes eelmistes peatükkides käsitletud tarkvaradega, on Apache konfigureerimine keerukam. Ubuntu serverile saada olev Apache 2.4 pakett on üles ehitatud viisil, kus konfiguratsioonifailide ülesehitus on muutunud eelmise versiooniga võrreldes hoomatavamaks ning lisandunud on häid tööriistu, mida antud seadistuse loomisel saab kasutada.

Kliendile turvalise sidekanali loomiseks on Apache 2.4 versioonis vaikimisi kaasas iseallkirjastatud sertifikaat koos privaatvõtmega, loodud on SSL ühendust toetav *virtualhost*, mis pole küll vaikimisi sisse lülitatud. Esmasteks tegevusteks Apache rakendamisel oleks *mod_rewrite*, *mod_headers*, *mod_ssl* mooduli ja *default-ssl vhost*i sisse lülitamine käskudega a2enmod ja a2ensite, mille tulemusel tekivad *symlink* viited kaustadesse, kust Apache vastava konfiguratsiooni loeb ning usaldusväärse sertifitseerimisasutuse poolt väljastatud sertifikaadi paigaldamine. Kindlustamaks, et suurem osa veebilehitsejaid suudavad tekitada täieliku sertifitseerimisahela serveri sertifikaadi ning CA juursertifikaadi vahel, paigaldame *SSLCertificateChainFile* direktiivi abil ka kõik vahepealsed sertifikaadid kliendile edastamiseks.

ID-kaardi ja Digi-ID autentimise toetamiseks tuleb alla laadida SK (AS Sertifitseerimiskeskuse) juursertifikaadid ning need ühte faili ühildada. Apache kataloogi loome tühistusnimekirjade tarbeks uue alamkataloogi, kuhu laadime ka SK lehelt vastavad failid. Failid tuleb konverteerida PEM formaati ja luua CRL faili nimetuse räsi failinimena kasutavad sümboolsed viited. CRL-ide perioodiliseks uuendamiseks kasutame R. Küngase poolt loodud ja SK levitatavat *renew.sh* skripti. Määrata tuleb veel CA sertifikaadi tühistatuse kontrolli tüüp, kliendisertifikaadi kontrolli sügavus ja vajalikkus. [12].

Eksamite infosüsteemi veebiserveris on kasutuses kaks vhosti. Ühe puhul kliendi sertifikaati ei kontrollita, *SSLVerifyClient* väärtuseks on *none*, ning teine, mille puhul vastavaks väärtuseks on *require* ning sertifikaadi kontroll kohustuslik. Tehniliselt on tarvis ka erinevaid domeeninimesid ning ka sertifikaati, kus kõik nimed on *subject alternative name* direktiiviga viidatud.

Proxy funktsionaalsuse tarbeks tuleb täiendavalt paigaldada ja aktiivsesse konfiguratsiooni lülitada järgnevad moodulid:

1) mod_proxy – *proxy* ja *reverse proxy* (*gateway* e. lüüsi) funktsionaalsus;
2) mod_proxy_balancer – koormuse jaotamise funktsionaalsus ja aktiivsusmonitor, mis kontrollib tagaserverite (*backend*) toimimist vajadusel neid aktiivsest kasutusest välja lülitades.
3) mod_proxy_http – *proxy* http protokolli tugi;

4) lbmethod_byrequests (…bytraffic, …bybusiness) - koormuse jaotamise erinevad algoritmid

Sellele järgneb konfiguratsioonifaili täiendamine koormusjaoturi funktsioonile vastavalt. Kuna eesmärk on kasutada Apachet reverse *proxy*-na, tuleb välja lülitada standardne *forward proxy* funktsioon käsuga *ProxyRequests off*. Välja lülitamisest hoolimata jätkab Apache *ProxyPass* direktiivi täitmist. [13]

Järgnevalt tuleb sisse lülitada *ProxyVia* ja *ProxyPreserveHost* direktiividega. Esimene aitab vastavalt RFC 2616 standardile säilitada korrektset jada kliendi, lüüsi ja tagasüsteemi serveri vahel, teine säilitab paketi päises serveri nime, kui pakett läbi *proxy* ahela tagaserveri suunas liigub.

Backend serverite kasutamine defineeritakse Apache veebiserveris *reverse proxy* kontekstis kahe direktiiviga. Kui *ProxyPass* direktiiv täidab traditsioonilist rolli, aktsepteerides päringu ja edastades need tagasüsteemile täitmiseks, siis ProxyPassReverse tagastab päringu tulemuse kliendile, seejuures muudab päringu päises olevat *Location* kirjet selliselt, et välja väärtuseks oleks jätkuvalt infosüsteemi üldaadress, mitte mõne *backend* serveri oma. [14] Mitme tagasüsteemi rakendusserveri olemasolul saab need defineerida klastri sõlmedena direktiiviga BalancerMember. Sealjuures võib kasu olla erinevatest parameetritest, mille abil saab sõlmedele langevat koormust protsentuaalselt mõjutada, mõne sõlme vaikimis keelata (*status disabled*) või kuumvarusse määrata (*status standby*).

Koormusjaotuse sektsioonis on võimalik seadistada veel töö algoritmi (*lbmethod*), päringutevahelist ooteaega (*timeout*), ebaõnnestunud katsete kordamist (*disable failover*) ja kordamiste arvu (*failover attempts*) ning kindlate kliendisessioonide ja rakendusserveri vahelise seose loomist (*sticky sessioon*). Klastri sõlme puhul on võimalik häälestada koormusindeksi (*load factor*), klastrisse kuuluva sõlme nime (*route*) ja mitmeid teisi parameetreid. EIS-i puhul leiab kasutust veel aktiivsuse (*enabled/disabled*) ja kuumvaru (*hot standby*) parameetrid. Vastavaid seadeid on võimalik ka veebipõhise kasutajaliidesega jooksvalt muuta.

Kokkuvõtvalt oleks Apache veebiserver sobiv kandidaat koormusjaoturi funktsiooni täitma.

4.2.4 Lahenduste võrdlus

Kõik vaadeldud tarkvarad pakuvad funktsionaalsust erineval tasemel. Kui HAProxy on mõeldud eeskätt lihtsate veebisüsteemide teenindamiseks, siis Nginx ja Apache on võrdväärsed konkurendid ka keerukamate lahenduste puhul. Alljärgnev tabel kirjeldab kokkuvõtvalt eksperimentide tulemusi.

Tabel 6. Lahenduste funktsionaalsuse võrdlus

Funktsionaalsus	**HAProxy**	**Nginx**	**Apache**
Koormuse jaotamine	JAH	JAH	JAH
SSL/TLS kanali krüpteering	JAH	JAH	JAH
Sertifikaadi alusel kliendi tuvastamine	JAH	JAH	JAH
Sertifikaadi kehtivuse kontroll tühistusnimekirja alusel	JAH/EI	JAH/EI	JAH
Kliendi sertifikaadi edastamine tagaserverile	JAH/EI	JAH	JAH
Keskkonnamuutujate edastamine tagaserverile	EI/JAH	JAH	JAH
Lisaväärtus: veebipõhine koormusjaoturi haldusliides	JAH	EI	JAH

HAProxy ongi mõeldud eelkõige koormusjaoturi funktsiooni täitma. Välja on jäetud kõik funktsioonid, mis otseselt vajalikud ei ole. Tarkvara eesmärk on olla võimalikult vähese ressursinõudlikkusega ning teenindada maksimaalne kogus kliente. Sellest tulenevalt töötleb HAProxy ka keskkonnamuutujaid võimalikult efektiivselt, edastades kliendi sertifikaadi ainult DER formaadis binaarkujul. Eksamite infosüsteemis tähendaks see pigem suuremat arendustööd, kus täiendada tuleb nii ID-kaardiga autentimist kui ka andmevahetust kohaliku serveriga.

ID-kaardi ja Digi-ID sertifikaatide kehtivuse kontroll toimub dokumentatsiooni põhjal kõigis kolmes süsteemis PEM formaadis tühistusnimekirjade alusel. Kui HAProxy ja Nginx ootavad tühistusnimekirju ühte faili summeerituna, siis Apache loeb sümboolsete viidete põhjal ka eraldi faile. Teoreetilises plaanis ei tohiks tulemus rakendatavast meetodist erineda, kuid praktikas ei õnnestunud HAProxy ja Nginx tarkvaradega CRL-i alusel kliendisertifikaatide kehtivuse kontrolli tööle saada. Apache tarkvaraga lahendus töötab.

Hoolimata sarnasest käitumisest CRL-i töötlemisel, on Nginxil HAProxy ees ka eeliseid. Kuna tegemist on veebiserveriks mõeldud tarkvaraga, on administraatori kasutuses

olevate funktsioonide hulk suurem. Lisaks koormuse jaotamisele võib Nginx täita samal ajal ka vahemälu (cache) serveri rolli puhverdades klientidelt tulevaid päringuid ja säästes sellega tagasüsteemi ressurssi. Eksamite infosüsteemi puhul seda kasutada ei saa, sest valdav enamus informatsioonist on unikaalne ja kattuvad osad eelnevalt optimeeritud.

Küll aga kasutab EIS sertifikaatide abil klientide tuvastamist. Selles osas on Apache ja Nginx võrreldavad. Hoolimata sellest, et Nginx sisaldab vaikimisi ainult osa Apache poolt toetatud SSL keskkonnamuutujatest, on EISi jaoks vajalikud väärtused Nginx-is olemas või siis konfiguratsioonifaili abil tekitatavad ja väärtustatavad. Väikeste mööndusena oleks Nginx Eksamite infosüsteemi koormusjaoturi rollis kasutatav.

Apache tarkvaral baseeruv koormusjaotur ühildub väga hästi Eksamite infosüsteemi funktsionaalsusega, sest rakendusserverit käitab sama veebiserveri tarkvara. Veebiserver toimib lisamoodulite abil edukalt koormusjaoturina, mida saab hallata mod_proxy_-balancer moodulist pärineva veebipõhise haldusliidesega. Haldusliides on oma funktsionaalsuselt üsna sarnane eelnevates peatükkides kirjeldatud HAProxy omale. Jooksvalt on võimalik klastri sektsioonis muuta eelnevalt konfiguratsioonifailis seadistatud parameetreid, mis kokkvõttes teeb Apachest eelistatuima valiku Eksamite infosüsteemi koormusjaoturi rolli täitma.

4.2.5 Muudatused EIS-i rakendusserveri veebiserveris

Lähteülesande üks tingimus oli, et Eksamite infosüsteemi rakendustarkvara ei ole lubatud muuta. Sellest tulenevalt saab rakendusservereit ümber ehitada ainult operatsiooni-süsteemi ja muude rakendust käitavate tarkvarade tasemel. Kasutatavast lahendusest sõltuvalt on rakendusserveri komponentide teatud täiendamine vajalik rakenduse toimimiseks kui ka ressursside optimaalsemaks kasutamiseks.

Süsteemi ressursside säästmiseks võib loobuda SSL moodulist, sest koormusjaotur on ühenduslüliks kliendi ja rakendusserveri vahel. Ebaturvalises kanalis ehk avalikus internetis liikuvad paketid on kaitstud koormusjaoturi ja kliendi vahel loodava SSL/TLS sokli abil. Koormusjaoturi ja rakendusserveri vahel toimub andmevahetus suletud sisevõrgus. Vajadusel on võimalik ka sisevõrguühendused täiendavalt turvata, kuid praegusel hetkel on infosüsteem majutatud usaldusväärses keskkonnas ning lisanduva krüptograafia järgi puudub vajadus.

Koormusjaotur täidab ainult päringute vahendaja rolli, reaalne kliendi päringule vastuse moodustamine, info tervikluse ja konfidentsiaalsuse tagamine toimub tagasüsteemis (*backend*), samuti säilitatakse seal ka seireinfot. Oluliseks kasutajaid kirjeldavaks parameetriks on IP aadress, millega klient süsteemi poole pöördub. Olukorras, kus koormusjaotur vahendab kliendilt tulevad päringud rakendusserverile, jõuab kliendi reaalne aadress ainult koormusjaoturini. Nii HAProxy, Nginx kui Apache võimaldavad kliendi reaalse IP aadressi lisada päringu päisesse *X-Forwarded-For* välja, kuid vaikeseades ei ole rakendusserveri komponendid võimelised kirjet töötlema. EIS-i tarkvara võib programmeerida antud välja lugema ja selle kaudu edastatavat IP aadressi kasutama, kuid probleemide diagnoosimisel sellest ei piisa, sest rakendustarkvara käivitav veebiserver peab kliendi IP-ks jätkuvalt koormusjaoturi sisevõrguliidese aadressi. Sellest tulenevalt on probleemsete klientide info kokku viimine rakenduse ja veebiserveri logides jätkuvalt raskendatud.

Reaalse kliendi IP viimiseks kõikide rakendusserveri süsteemi komponentideni on Apache tarbeks loodud kaks moodulit. Alates Apache 1.3 versioonist on võimalik kasutada kolmanda osapoole loodud *mod_rpaf* nimelist komponenti, mille konfiguratsioonis on võimalik defineerida kliendi päringuid edastava proxy serveri IP aadress, kliendi reaalset IP aadressi kandva päise välja nimetus. Antud moodul asendab töödeldavas andmestikus *proxy* serveri aadressi *X-Forwarded-For* väljal asuvaga ja tänu sellele on kõikides süsteemi komponentides, nii rakendustarkvaras kui ka näiteks logimise moodulis kirjas kliendi reaalne IP aadress. Alates Apache versioonist 2.4 on Ubuntu tarkvarapaketti lisatud *mod_remoteip*, mis ühelt poolt teeb samu asju, mida *rpaf*, kuid pakub ka täiendavaid võimalusi, mida võib inseneridel teatud olukordades tarvis minna.

ID-kaardi parameetrite tagasüsteemi veebiserveri keskkonnamuutujatena kasutamiseks tuleb Apaches vastavad väljad HTTP päisest välja lugeda ning SSL keskkonnamuutujatena väärtustada. Sealjuures pole vahet, millist koormusjaoturit kasutatakse – konfiguratsioon jääb täpselt samaks, kui koormusjaotur väärtustab väljad selliselt, mida Apache on seadistatud lugema.

```
SetEnvIf SSL_PROTOCOL "(..*)" SSL_PROTOCOL=$1 [13]
```

Sarnast seadistust on mõistlik kõikide *mod_ssl* dokumentatsioonis toodud muutujatega rakendada kui koormusjaoturina on kasutusel Apache. See välistab hilisemad probleemid,

kui arendaja võtab kasutusele väärtused uuest muutujast, kuid toodang-süsteemis lahendus mingil põhjusel ei tööta. Nginxi ja HAProxy kasutamisel võib välja filtreerida need, mida vastavad tarkvarad tagasüsteemile saata ei suuda.

Pärast SSL_* parameetrite töötlemist, tuleks vastavad väljad päisest eemaldada, et ei tekiks HTTP_SSL_* muutujaid. Kõikide töödeldavate muutujate kohta tuleks konfiguratsioonis defineerida järgnev rida:

```
RequestHeader unset SSL_PROTOCOL [13]
```

Nende muudatustega peaks tagasüsteemi veebiserver olema valmis päringuid vastu võtma ja neid korrektselt teenindama, kus vajadusel on võimalik ka operatiivselt erinevatest elementidest vigu tuvastada ja vajalikke logikirjeid kõrvutada.

Olukorras, kus infosüsteemi sisemine liiklus toimub mitme serveri vahel, tuleb süsteemi turvalisuseks käitlemiseks luua privaatne võrgusegment, kuhu pääsevad ligi ainult koormusjaotur ja rakendusserverid. Info, mida kliendi ja koormusjaoturi vahel läbi interneti krüpteerituna edastatakse, liigub koormusjaoturi ja rakendusserveri vahel krüpteerimata. Seetõttu on mõistlik ligipääsu andmetele võimaluste piires maksimaalselt piirata.

5 Muudatuste mõju mõõtmine ja analüüs

Tehniline valmisolek infosüsteemi on loodud. Efektiivseima konfiguratsiooni välja selgitamiseks testitakse infosüsteemi jõudlusnäitajaid erinevate parameetrite muutumisel. Saadud tulemused esitatakse tabelite ja graafikutena.

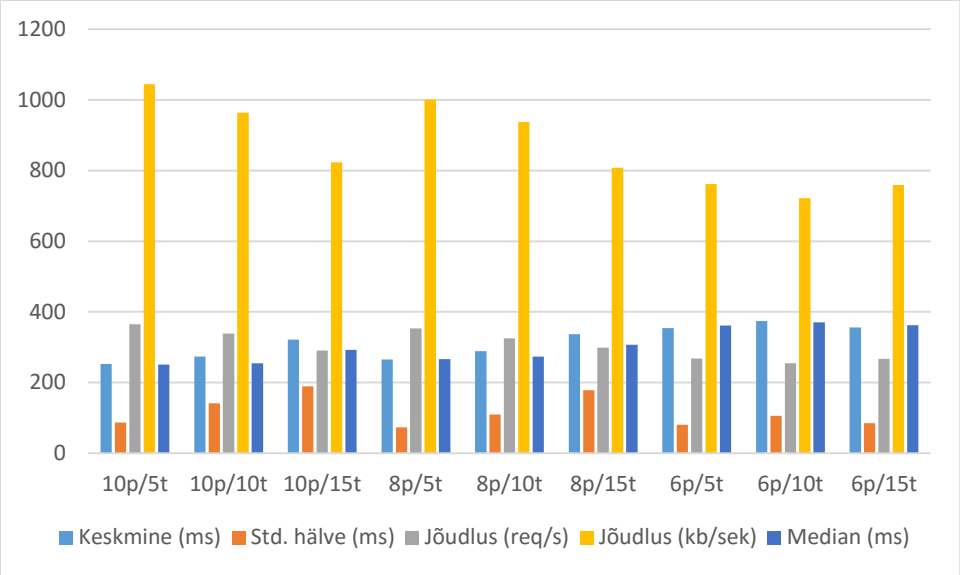
Esimeses etapis on vajalik välja selgitada maksimaalne jõudlus, mida on võimalik saavutada ühest rakendusserverist koosneva konfiguratsiooni abil. Eksamite infosüsteemi töötab *python*i WSGI rakendustena, mida konfigureerides on võimalik seadistada rakenduse protsesside arvu serveris ning lõimede arvu protsessis. EIS-is töötab 5 erinevat rakendust (eis, ekk, adapter, kysi, plank), kuid enamuse kasutusmaht on väike ning valdav koormus koondub infosüsteemi avalikku vaatesse ehk „eis“ rakendusse.

Parima jõudluse saavutamiseks ühe rakendusserveri kohta tuleb esmalt leida sobivaim rakenduse konfiguratsioon. Kõikide EISi rakenduste puhul saab konfigureerida käivitatavate protsesside ja lõimede arvu. Avaliku vaate rakenduses protsesside ja lõimede muutmine ning serveri Apache JMeteriga koormamine annab tulemuse, mida kajastatakse tabelis nr. 8. Vaikimis töötas avalik vaade 6 protsessi ja 10 lõimega.

Tabel 7. Rakendusserveri jõudlusnäitajate võrdlus

Prot./lõim	**Keskm.(ms)**	**St.hälve.(ms)**	**Vigu (%)**	**Jõudl. (pär./s)**	**Mediaan (ms)**
10p/05t	252,2	86,79	2,11	365,08	251,0
10p/10t/	273,8	140,94	3,43	338,72	254,2
10p/15t	321,8	189,02	5,782	290,80	292,8
08p/05t	265,0	73,342	1,76	352,90	266,0
08p/10t	288,6	109,53	1,52	324,76	273,4
08p/15t	337,0	178,18	4,09	299,20	306,6
06p/05t	353,8	80,40	1,73	268,32	361,4
06p/10t	374,4	105,42	1,64	254,28	370,6
06p/15t	355,6	85,09	1,90	267,50	362,6

Tabelis nr. 8 välja toodud testide tulemuste saavutamiseks testiti kõiki välja toodud erinevaid konfiguratsioone vähemalt 5 korda. Saadud tulemuste põhjal arvutati välja keskmised väärtused, mida kujutab visuaalselt joonis nr 10. Arvutatud tulemusi kasutati parima seadistuse välja valimiseks.



Joonis 10. Jõudlusnäitajate graafik

Mõõtmistest selgub, et pigem on efektiivsemad väikeste lõimede arvuga protsessid ning parimat jõudlust pakub selline konfiguratsioon, kus on rohkem protsesse. 10 protsessiga konfiguratsioon kasutab täielikult ära kõik serveri 8 tuuma, protsesside arvu tõstmine kõrgemaks hakkas jõudlusele kahjulikult mõjuma.

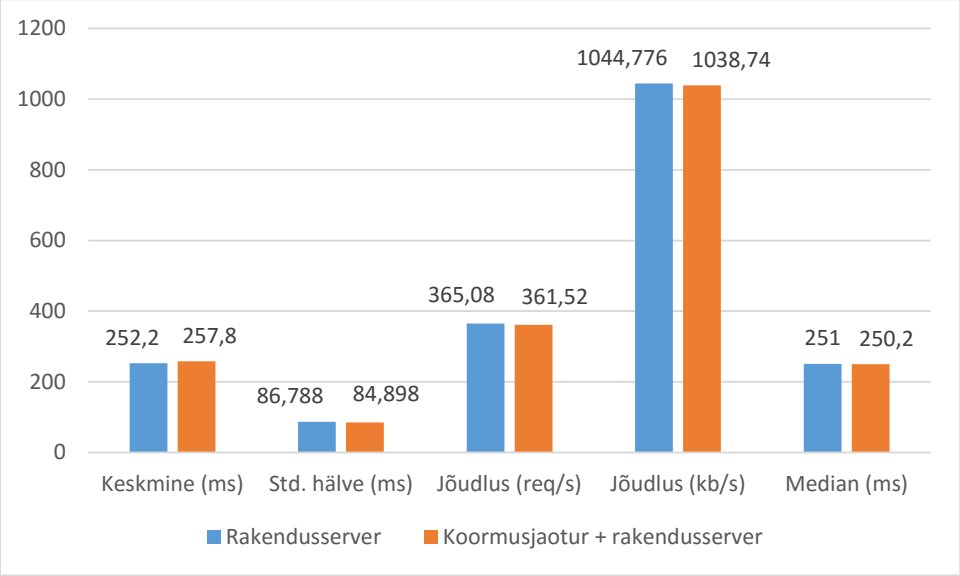
5.1 Komponendi edastamise jõudluse mõõtmine ja analüüs

Kui parim rakendusserveri konfiguratsioon on välja selgitatud, tuleb hinnata koormusjaoturi lisamise mõju. Selleks koormatakse sama JMeter testiga esmalt optimeeritud seadistuses rakendusserveri veebiliidest, seejärel koormusjaoturit olukorras, kus klastris on ainult üks sõlm ning võrreldakse saadud tulemusi.

Tabel 8. Koormusjaoturi mõju mõõtmine

	Keskm.(ms)	**St.hälve.(ms)**	**Vigu (%)**	**Jõudl. (pär/s)**	**Mediaan (ms)**
Rak. srv	255	88,22	2,09	357,1	250
Rak. srv	251	85,79	1,78	369,2	254
Rak. srv	245	87,09	2,48	376,2	251
Rak. srv	259	84,97	1,73	352,1	249
Rak. srv	251	87,87	2,47	370,8	251
LB/1x CLU	266	85,52	0,82	350,8	258
LB/1x CLU	252	86,12	1,24	358,4	254
LB/1x CLU	257	84,92	1,00	360,9	249
LB/1x CLU	248	82,34	1,50	372,7	242
LB/1x CLU	256	85,59	1,20	364,8	248

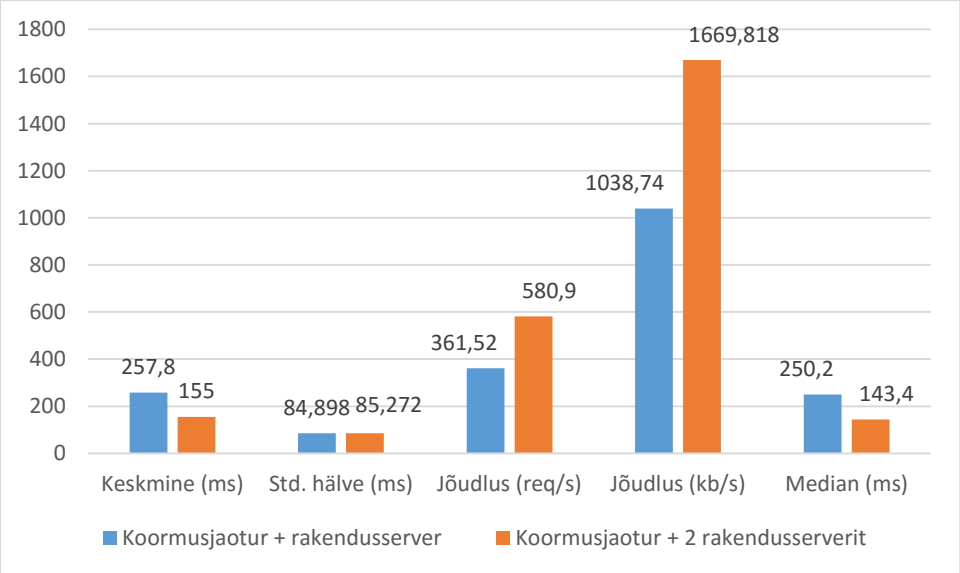
Tabelis nr 8 kajastatakse rakendusserveri ja koormusjaoturi keskmiseid tulemusi. Neid võrreldes võib öelda, et koormusjaoturi lisandumine infosüsteemi ei ole jõudlusele negatiivselt mõjunud ja teenindusvõimekust kärpinud.



Joonis 11. Rakendusserveri mõju visualiseering

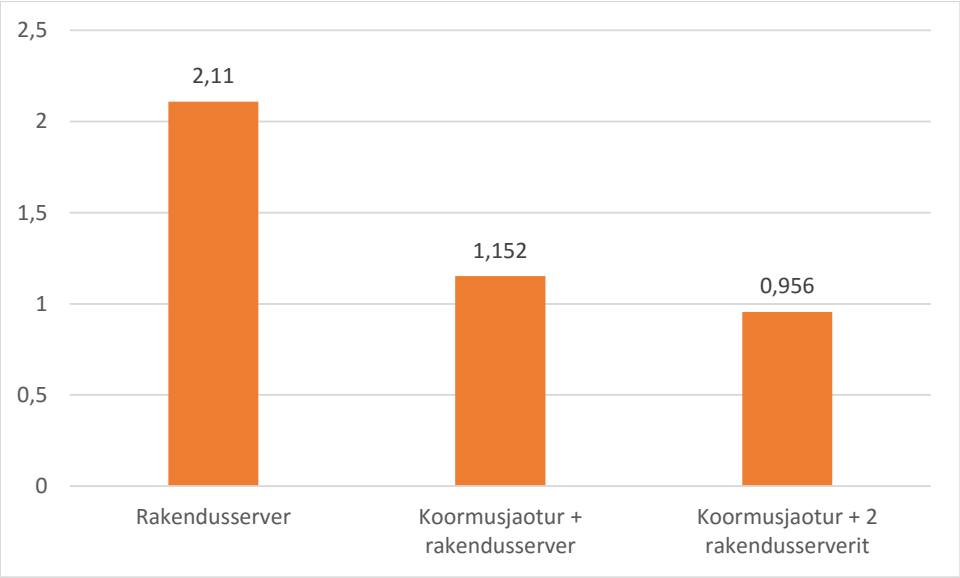
Joonise nr. 11 põhjal selgub, et koormusjaoturi lisamisel on jõudlusparameetrid sisuliselt paika jäänud. Veebirakendusi teenindav veebiserver ei hoia piiramatult protsesse mälus, vaid tekitab neid vajadusel juurde. Kui testi sooritav tarkvara käivitub hetkel, millal veebiserver on mälukasutust optimeerinud ning jõude seisvad protsessid lõpetanud, kulub uute protsesside ja lõimede tekitamiseks aega. Aeglaselt reageeriva veebiserveri tõttu lõppeb osade päringute ooteaeg ning koormust mõõtev tarkvara märgib need vigade hulka.

Koormusjaoturi lisamise põhimõtteks on algusest peale olnud infosüsteemi teenindusvõimekuse tõstmine täiendavate rakendusserverite lisamise võimaluse kontekstis. Täiendava jõudluse saamiseks lisatakse klastrisse ka teine rakendusserver. Infosüsteemi koormatakse taas testiga ning tulemused on toodud joonisel nr 12. Joonis kujutab visuaalselt mõlema konfiguratsiooni läbilaskevõimet mõõdetuna päringutes ja kilobaitides ühe sekundi jooksul. Infot saab keskmise päringute teenindamise aja, päringute teenindamiskiiruse mediaani ning standardhälbe.



Joonis 12. Klastri parameetrid ühe ja mitme rakendusserveriga

Jooniselt nr. 12 selgub, et infosüsteemi teenindusvõime on kahe rakendusserveri paigaldamisel oluliselt kasvanud nii jõudlusparameetrite kui ka päringu teenindamise kiiruse kontekstis. Keskmine päringu teenindamise aeg on langenud ligikaudu 100 millisekundit, päringukiiruste kõikumine on jäänud sisuliselt samaks ning koguseliselt teenindatakse päringuid kiiremini.



Joonis 13. Vigade arvu muutus

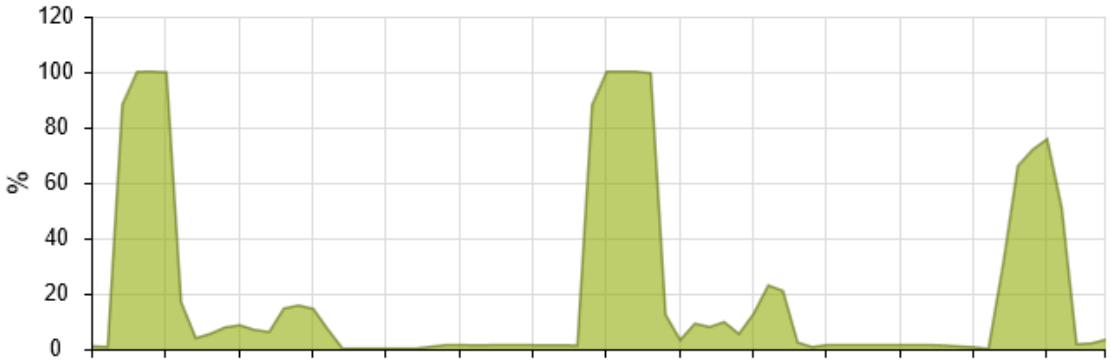
Joonise nr. 13 kirjeldatav Apache JMeteri poolt registreeritud vigade arv vähenes täiendava rakendusserveri lisamisel kokkuvõttes ligikaudu kaks korda.

5.2 Eksamitöö edastamise jõudluse mõõtmine analüüs

Eksamitöö edastamise jõudluse mõõtmiseks koormatakse infosüsteemi Apache JMeter jõudlustestiga. Antud test erineb eelmises peatükis käsitletust selle poolest, et simuleerib ühelt poolt kasutaja käitumist tehes erinevate klikkide vahel pause ning teiselt poolt sisaldades kõiki kujunduselemente, mida tavajuhul kasutajale edastatakse. Antud test on infosüsteemile oluliselt koormavam, sest edastatavate failide hulka kuulub muuhulgas ka suuremahulisi pildi ja helifaile, mille pärimine andmebaasist on nii aja- kui ka ressursi-mahukas.

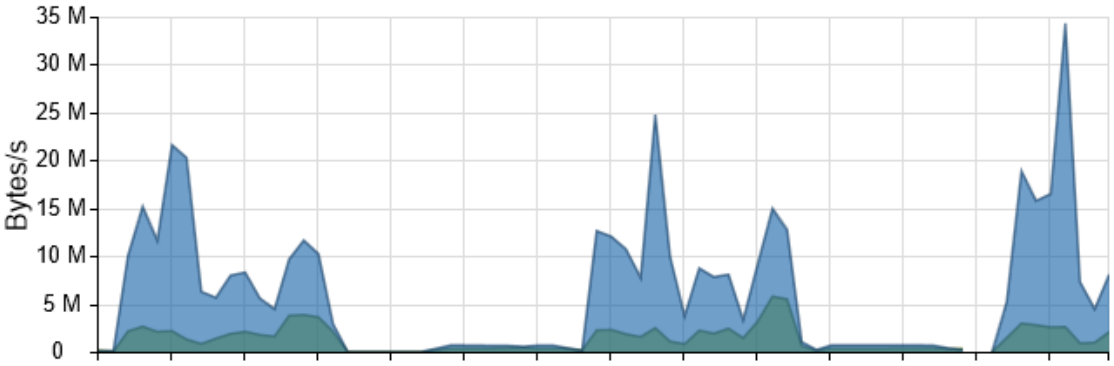
Tulemuste võrdlemiseks viidi läbi kolm erinevat testi. Esimeses testis teenindas kliente ainult rakendusserver, teises testis oli töös kaks rakendusserverit ning kolmanda puhul kaks rakendusserverit ja suurema jõudlusega andmebaasi masin. Seadmete jõudlusparameetrid asuvad lisas nr 4 ning joonistel toodud koormuste tipud näitavadki testide ressursikasutust.

Kohe alguses osutub süsteemi järgmiseks kitsaskohaks andmebaasiserveri läbilaskevõimekus päringute kiiruse töötlemise näol.



Joonis 14. Andmebaasiserveri protsessori koormamise graafik

Joonise nr. 14 põhjal saab väita, et üks kiiresti päringuid teenindav rakendusserver suudab testi alguses süsteemi üle koormata. Sealjuures jääb joonise nr 15. alusel andmeedastusmaht ja kiirus tipphetkedel väiksemaks kui mitme rakendusserveri kasutamisel.



Joonis 15. Andmebaasiserveri võrguühenduse graafik

Koormusjaoturi protsessori ja võrguühenduste koormus jääb andmebaasiga võrreldes väikeseks, antud graafikutega on võimalik tutvuda lisas nr 4. Mainitud lisasse on pandud ka rakendusserverite jõudluse graafikud. Nende põhjal saab öelda, et käesolevas töös rakendatud koormusjaotur töötab hästi ning jagab klientidelt tulevad päringud ühtlaselt tagasüsteemi vahel laiali. Süsteemist võib saada palju kasu, kui andmebaasi teenindusvõimekus tõsta paremale tasemele.

6 Kokkuvõte

Käesoleva töö põhieesmärk on Eksamite infosüsteemi tarbeks sobiva arhitektuuri ja selle komponentide leidmine ja rakendamine. Töö käigus tuli leida parimad lahendused alameesmärkidele, mis muudaks võimalikuks süsteemi töö mõõtmise ja muudatuste mõju analüüsimise, võimaldaks klientide poolt loodavaid päringuid jagada mitme teenindava serveri vahel ning vähendada süsteemi hooldustöödest tingitud mõju lõppkasutajale.

Infosüsteemi töö mõõtmiseks valiti Apache JMeter nimeline tarkvara ja koostati testid, mida on pärast muudatuse tegemist võimalik piiramatul arvul kordi käivitada ja seeläbi koguda infot muudatuse mõju kohta infosüsteemi toimivusele ning jõudlusele.

Infosüsteemi tasemel koormuse jaotamise funktsionaalsuse loomiseks vaadeldi kolme tarkvara: HAProxy, Nginx ja Apache. Esimene nimetatutest ongi spetsiaalselt mõeldud vastava ülesande täitmiseks, ülejäänud kaks on eelkõige veebiserverid, mida on võimalik asetada vastavasse rolli nii integreeritud funktsionaalsuse kui täiendavate moodulite abil. Kõik mainitud lahendused põhinevad avatud lähtekoodiga tarkvaral, ning nende rakendamine otseseid kulusid rakendajale kaasa ei too.

Kõik kolm mainitud tarkvara sobiksid valdava enamuse veebilahenduste koormusjaoturiks. Kõik võimaldavad vastu võtta nii HTTP kui HTTPS ühendusi ning päringud saata edasi tagaserveritele. Suures osas toimib kõigi kolmega ka Eksamite infosüsteem, peamine probleemikoht on Eestile omane ID-kaardi infrastruktuur ning infosüsteemi ehitatud kohaliku serveri kasutamist võimaldav funktsionaalsus.

Parameetreid, mida tagasüsteem vajab korrektseks toimimiseks, saab küll sinna saata ning ka ID-kaardiga töö tarbeks on suurem osa süsteemseid muutujaid olemas, kuid erinevalt Nginxist ja Apachest eelistatakse HAProxys muutujate väärtuseid edastada binaarkujul.

Sellest lähtuvalt pole HAProxy EIS-i koormusjaoturi rollis eelistatud. Küll annab HAProxy tarkvaras rakendatud haldusliides idee hooldustöödest tekitatud katkestuste vähendamiseks. Vastav liides võimaldab hooldustööde tarbeks koormusjaoturi teenust käivitamata vajaliku sõlme klastrist välja võtta. Seda võimalust hakati otsima ka teistest lahendustest.

Nginx veebiserver on kiire, väikese ressursinõudlusega ja lihtsasti konfigureeritav tarkvara. Vaikimisi kõiki EIS-i lähtekoodis kasutatavaid muutujaid Nginx-is ei eksisteeri, kuid veebiserveri sisemise seadistuse tulemusena on need võimalik luua ja teistest muutujatest regulaaravaldise abil selekteeritava info alusel väärtustada.

HAProxy tarkvarast tuttavad koormusjaoturi haldusliidest Nginx ei sisalda, kuid muu funktsionaalsus on kaetud ning Nginx tarkvara võiks parema alternatiivi puudumisel EIS-koormusjaoturina kasutada.

Apache veebiserver on ülejäänud kahega võrreldes oluliselt funktsionaalsem tänu rohketele moodulitele. Kuna eksamite infosüsteem on algselt loodud töötama eraldiseisvana (*standalone*) Apache serveri peal, on ka ühilduvus vaikimisi parim. Lisaväärtusena pakub Apache veebipõhist koormusjaoturi haldamise liidest ja sellest tulenevalt on tarkvara eelistatuim valik EIS-i koormusjaoturi rolli.

Koormusjaoturi rakendamine võimaldas infosüsteemis kasutusele võtta üheaegselt mitu rakendusserverit. Komponendi edastamise koormustesti tulemustest selgub, et muudatus mõjutas positiivselt infosüsteemi jõudlust. Kliendi päringud jagunevad kahele rakendusserverile, väheneb päringute keskmine teenindusaeg ja vigade arv ning suureneb läbilaskevõime nii päringute kui andmemahu osas.

Eksamitöö soorituse töökindlusele aitas koormusjaoturi rakendamine kaasa, kuid jõudluse tõstmisele pigem vähesel määral. Suurem väärtus lisandus rakendusserveri protsesside ja lõimede optimeerimisest.

Kokkuvõtteks saab öelda, et töös püsitatud eesmärgid said küll täidetud, kuid suurt soovitud kasu sellest ei kaasnenud ning koheselt tekkis järgmine oluline probleemikoht andmebaasi teenindusvõimekuse näol.

Võimaliku edasiarendusena tulevikus saab ja tuleb vaadelda andmebaasi osa infosüsteemis. Kuna suurenev üheaegsete kasutajate arv tõstab nii üheaegsete ühenduste arvu kui ka andmebaasi üldist koormust, tuleks leida lahendus, mis võimaldaks probleemi tükeldada ja süsteemselt lahendama hakata. Senise põgusa uurimistöö tulemusena võiks suurim potentsiaal olla pgpool tarkvaral, sest ülesehituse poolest moodustataks PostgreSQL andmebaasi mootori ette käesolevas töös käsitletuga sarnanev koormusjaotur. Pgpool koormusjaotur võimaldab moodustada mitmest

andmebaasiserverist koosneva klastri, töödelda serveriga ühendust loovaid andmebaasiühendusi, olla teatud päringute vahemäluks ning vajadusel seadistada ka ainult lugemiseks (read-only) ja ainult kirjutamiseks (write-only) andmebaasiservereid.

Mahult oleks kirjeldatud uurimisülesanne võrreldav käesolevas töös käsitletuga, kuid kindlasti mõistlik rakendada käesolevas töös käsitletud koormusjaoturi täieliku potentsiaali ära kasutamiseks.

Kasutatud kirjandus

[1] Haridus- ja Teadusministeerium, „Digipööre,“ 8 aprill 2015. [Võrgumaterjal]. Available: https://www.hm.ee/et/tegevused/digipoore. [Kasutatud 31 jaanuar 2016].

[2] Eesti Koostöö Kogu, Haridus- ja Teadusministeerium ja Eesti Haridusfoorum, „Eesti elukestva õppe strateegia 2020,“ 13 veebruar 2014. [Võrgumaterjal]. Available: https://www.hm.ee/et/elukestva-oppe-strateegia-2020. [Kasutatud 31 jaanuar 2016].

[3] Haridus- ja Teadusministeerium, „Õppe kvaliteedi parendamine õppeasutuste sise- ja õpitulemuste välishindamissüsteemi arendamise kaudu,“ 14 august 2012. [Võrgumaterjal]. Available: http://www.innove.ee/UserFiles/%C3%9Cldharidus/%C3%95kva/okva_lisa1_uus.pdf. [Kasutatud veebruar 2016].

[4] A. Parman, *Eksamite infosüsteemi hanke läbiviimiseks vajalik tehniline kirjeldus,* Tallinn, 2009.

[5] „Apache JMeter User Manual,“ Apache Software Foundation, 2016. [Võrgumaterjal]. Available: http://jmeter.apache.org/usermanual/index.html. [Kasutatud 2016].

[6] Apache Software Foundation, „Apache HTTP Server Documentation,“ Apache Software Foundation, [Võrgumaterjal]. Available: http://httpd.apache.org/docs/.

[7] HAProxy kommuun, „HA Proxy, The Reliable, High Performance TCP/HTTP Load Balancer Documentation,“ HAProxy kommuun, 2016. [Võrgumaterjal]. Available: http://www.haproxy.org/#docs. [Kasutatud 2016].

[8] I. Nginx, „Nginx+ and Nginx, Inc. homepage,“ Nginx, Inc., 2016. [Võrgumaterjal]. Available: https://www.nginx.com/company/. [Kasutatud 2016].

[9] C. Nedelcu, Nginx HTTP Server (3 edition), Birmingham: Packt Publishing Ltd., 2015.

[10] Nginx, Inc., „nginx,“ Nginx, Inc, 2016. [Võrgumaterjal]. Available: http://nginx.org/en/. [Kasutatud 2016].

[11] D. Aivaliotis, Mastering NGINX, Birmingham: Packt Publishing Ltd., 2013.

[12] A. Sertifitseerimiskeskus, „Configuring Apache web server to support ID," 2016. [Võrgumaterjal]. Available: http://www.id.ee/public/Configuring_Apache_web_server_to_support_ID.pdf. [Kasutatud 02 2016].

[13] B. Laurie ja P. Laurie, Apache: The Definitive Guide, Second Edition, Sebastopol: O'Reilly & Associates, Inc., 1999.

[14] I. Oolberg ja J. Jans, „Kuutõrvaja wiki," Eesti Hariduse ja Teaduse Andmesidevõrk, [Võrgumaterjal]. Available: http://kuutorvaja.eenet.ee/.

Lisa 1 – HAProxy konfiguratsioon

```
frontend http
  mode  http
  bind  193.40.101.24:443  ssl  crt  /etc/haproxy/eis_2015_tera.pem  ca-file
/etc/haproxy/id.crt verify optional
  #crl-file       /etc/haproxy/sk1.pem
  Option          log-separate-errors
  http-request    set-header   SSL_CLIENT_I_DN      %{+Q}[ssl_c_i_dn]
  http-request    set-header   SSL_CLIENT_I_DN_CN   %{+Q}[ssl_c_i_dn(cn)]
  http-request    set-header   SSL_CLIENT_VERIFY    %[ssl_c_verify]
  http-request    set-header   SSL_CLIENT_S_DN      %{+Q}[ssl_c_s_dn]
  http-request    set-header   SSL_CLIENT_S_DN_CN   %{+Q}[ssl_c_s_dn(cn)]
  http-request    set-header   SSL_CLIENT_S_DN_O    %{+Q}[ssl_c_s_dn(o)]
  http-request    set-header   SSL_CLIENT_M_SERIAL  %[ssl_c_serial]
  http-request    set-header   SSL_CLIENT_CERT      %{+Q}[ssl_c_der]
  default_backend nodes

backend nodes
  mode            http
  balance         roundrobin
  option          forwardfor
  option          http-server-close
  http-request    set-header      X-Forwarded-Port %[dst_port]
  http-request    add-header      X-Forwarded-Proto https if { ssl_fc }
  option          httpchk         HEAD / HTTP/1.1\r\nHost:localhost
  server          node1           10.101.120.131:80 check
  server          node2           10.101.120.132:80 check

  stats uri       /server-status
  stats realm     Haproxy\ Statistics
  stats auth      kasutajanimi:parool
  stats admin     if TRUE
```

Lisa 2 – Nginx konfiguratsioon

```
server {
  listen                        443;
  server_name             eis.innove.ee;

  ssl  on;
  ssl_certificate         ssl/eis_2015_tera.crt;
  ssl_certificate_key     ssl/eis_2015_tera.key;
  ssl_client_certificate  ssl/id.crt;
  #ssl_crl                crl/sk.crl; # Probleemne
  ssl_verify_client       optional;
  ssl_verify_depth        2;

  location / {
    proxy_pass            http://10.101.120.101:80;
    proxy_set_header      Host               $host;
    proxy_set_header      X-Real-IP          $remote_addr;
    proxy_set_header      X-Forwarded-For    $proxy_add_x_forwarded_for;
    proxy_set_header      X-Forwarded-Proto  $scheme;
    proxy_set_header      SSL_CLIENT_VERIFY  $ssl_client_verify;
    proxy_set_header      SSL_CLIENT_CERT    $ssl_client_cert;
    proxy_set_header      SSL_CIPHER         $ssl_cipher;
    proxy_set_header      SSL_CLIENT_SERIAL  $ssl_client_serial;
    proxy_set_header      SSL_CLIENT_S_DN    $ssl_client_s_dn;
    proxy_set_header      SSL_CLIENT_I_DN    $ssl_client_i_dn;
    proxy_set_header      SSL_PROTOCOL       $ssl_protocol;
    proxy_set_header      SSL_SESSION_ID     $ssl_session_id;
  }
}
```

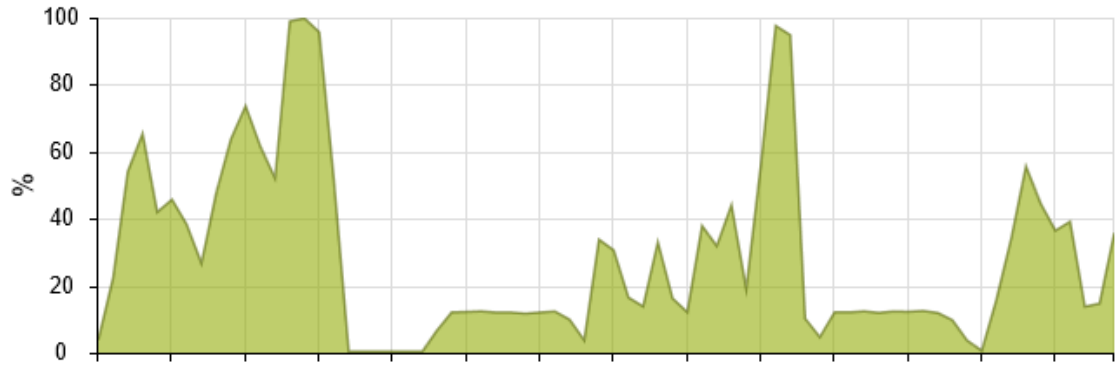
Lisa 3 – Apache2 konfiguratsioon

```
<VirtualHost 193.40.101.22:443>
  ServerAdmin             webmaster@ekk.edu.ee
  ServerName              eis-id.innove.ee
  DocumentRoot            /var/www

  CustomLog ${APACHE_LOG_DIR}/ssl_access.log combined
  ErrorLog ${APACHE_LOG_DIR}/error.log
  LogLevel warn

  SSLEngine               on
  SSLCertificateFile      /etc/apache2/ssl.crt/eis.innove.ee_tera.crt
  SSLCertificateKeyFile   /etc/apache2/ssl.key/eis.innove.ee_tera.key

  SSLCertificateChainFile /etc/apache2/ssl.crt/chain_2015_tera.crt
  SSLCACertificateFile    /etc/apache2/ssl.crt/id.crt
  SSLCARevocationPath     /etc/apache2/ssl.crl/
  SSLVerifyClient         optional
  SSLVerifyDepth          2
  SSLOptions              +ExportCertData +StdEnvVars

 <IfModule mod_proxy.c>
   ProxyRequests          off
   <Proxy *>
     AddDefaultCharset    off
     Order                deny,allow
     Allow                from all
   </Proxy>
   ProxyVia                            On
   ProxyPreserveHost                   On
   ProxyPass        /balancer-manager  !
   ProxyPass        /server-status     !
   ProxyPass        /server-info       !
   ProxyPass        /                  balancer://mycluster/
```

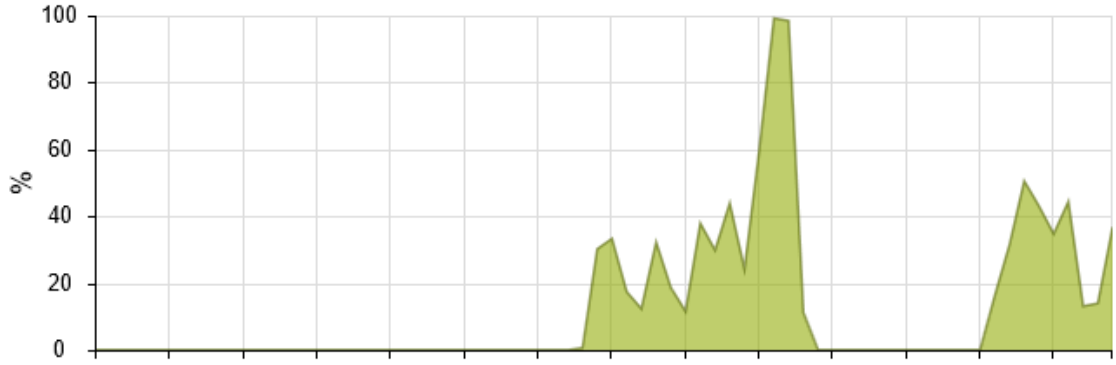
```
    SetEnv                proxy-nokeepalive   1
    ProxyPassReverse      /     http://10.101.120.131:80
    ProxyPassReverse      /     http://10.101.120.132:80
    ProxyPassReverse      /     http://10.101.120.131:800

    <Proxy balancer://mycluster>
      BalancerMember      http://10.101.120.131:80 route=clu1
      BalancerMember      http://10.101.120.132:80 route=clu2
      BalancerMember      http://10.101.120.131:800 route=clu-error status=H
    </Proxy>
  </IfModule>
</VirtualHost>
```

Lisa 4 – Jõudluse graafikud

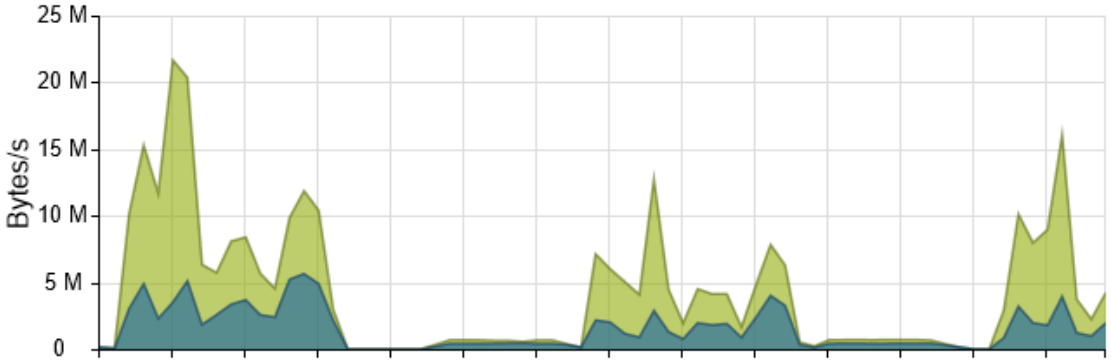
Esimese rakendusserveri protsessori kasutamise graafik



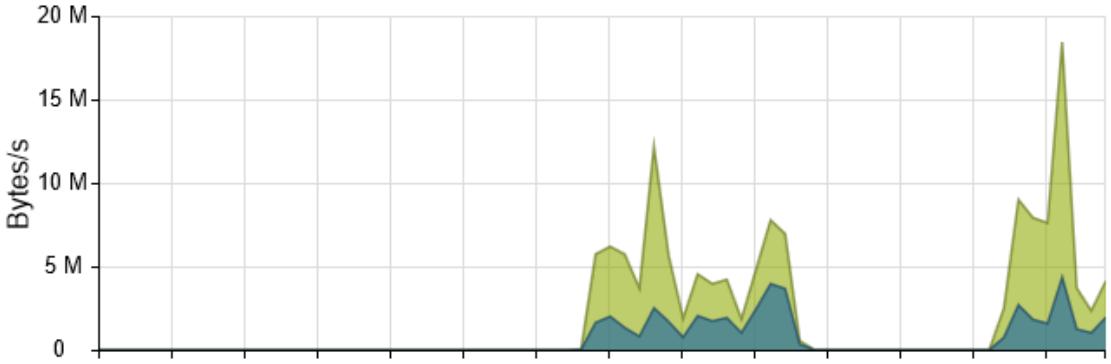
Teise rakendusserveri protsessori kasutamise graafik



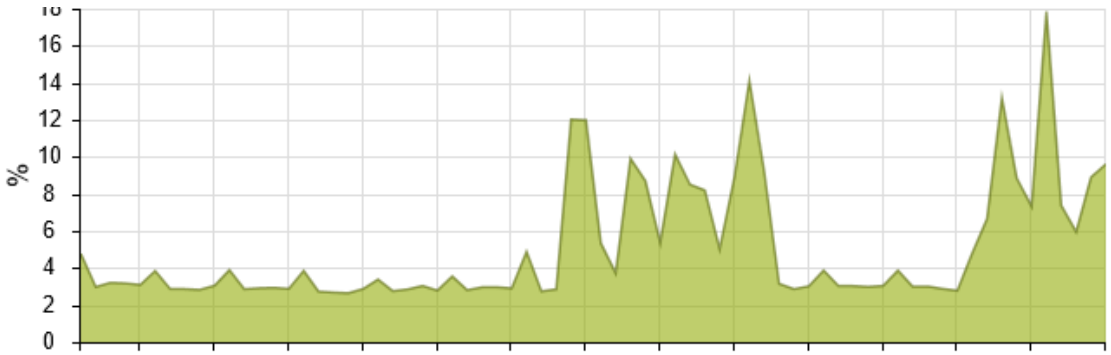
Esimese rakendusserveri võrguühenduse graafik



Teise rakendusserveri võrguühenduse graafik



Koormusjaoturi protsessori kasutamise graafik



Koormusjaoturi võrguühenduse kasutamise graafik