T A R T U Ü L I K O O L
MATEMAATIKA-INFORMAATIKATEADUSKOND
Arvutiteaduse instituut
Tarkvarasüsteemide õppetool

Jüri Harju

Turvaline CORBA nimeteenus

Magistritöö (40 AP)

Juhendajad: Meelis Roos
Jüri Kiho

Tartu 2005

Sisukord

Sissejuhatus 4
1 Lühiülevaade CORBA tehnoloogiast 6
1.1 Lühiülevaade CORBA arhitektuurist 7
2 CORBA tehnoloogia turvaelemendid 9
2.1 Üldine koostöövõime turvalisus (CSI) 9
3. CORBA turbetehnoloogiad 13
3.1 CORBA turvalisuse teenus 13
3.1.1 CORBA turvalisuse teenuse arhitektuur 15
3.1.2 Rakenduse arendusliidesed 21
3.1.3 Administratiivsed liidesed 21
3.1.4 Realisatsiooni liidesed 22
3.1.5 Turvalisuse teenuse protokollid 23
3.1.6 Turvalisuse teenuse seosed teiste turvatehnoloogiatega 23
3.1.7 Turvalisuse teenuse lühikokkuvõte 23
3.2 ATLAS 25
3.3 CORBA ja tulemüürimine 27
4 CORBA nimeteenus 30
5 Turvaline nimeteenus 34
5.1 Turvalise nimeteenuse spetsifikatsioon 35
5.2 Turvalise nimeteenuse liidesed 37
5.2.1 Turvalise nimeteenuse andmestruktuurid 37
5.2.2 Turvalise nimeteenuse andmekogud 42
5.2.3 Turvaline nimekontekst 44
5.2.4 Turbekontekst 46
5.2.5 Administreerimise abiliides 49
5.2.6 Turvalise nimeteenuse erindid 50
5.2.7 Turvalise nimeteenuse mudel 50
5.2.8 CSI koostöövõime turvalises nimeteenuses 51
5.3 Turvalise nimeteenuse realisatsioon 52
5.3.1 Realisatsioonis kasutatavad tehnoloogiad 53
5.3.2 Realisatsiooni projekti struktuur 54

5.3.3 Realisatsiooni ehitus 55
5.3.4 Liideste realisatsioonid 56
5.3.5 Turvalise nimeteenuse realisatsiooni abiklassid 58
5.4 Turvalise nimeteenuse konfigureerimine ja kasutamine 60
5.5 Turvalisel nimeteenuse põhinev testrakendus 60
5.5.1 Testrakenduse liides 60
5.5.2 Testrakenduse realisatsioon 61
5.5.3 Testrakenduse kasutamine 61
5.6 Turvalise nimeteenuse edasiarendamise võimalused 63
5.6.1 Spetsifikatsiooni edasiarendus 63
5.6.2 Realisatsiooni edasiarendus 64
6 Kokkuvõte 65
Резюме 66
Abstract 67
Kasutatud kirjandus 68

Sissejuhatus

Enamik tänapäeva suurettevõtete rakendustest (*enterprise applications*) projekteeritakse ja teostatakse hajussüsteemidena. Hajussüsteemid omavad rea eeliseid monoliitsete tarkvarasüsteemide ees. Enamik hajussüsteemide arhitektuure lubavad ühendada süsteemi heterogeensetes keskkondades paiknevaid ressursse: alamrakendusi ja andmehoidlaid. Kõik need ressursid ühendatakse võrgu kaudu, kasutades vahetarkvara (*middelware*) võimalusi. Üheks võimalikest tehnoloogiatest hajussüsteemi ehitamiseks on CORBA. CORBA tehnoloogia on loodud kasutamiseks erinevatel operatsioonisüsteemidel, erinevates programmeerimiskeeltes rakenduste loomiseks. CORBA spetsifitseerib suhtlusprotokolli, mida on võimalik ühendada erinevate võrguprotokollide transpordikihtidega. Lisaks on CORBA tehnoloogias spetsifitseeritud võimalused teiste hajussüsteemide konstrueerimiseks mõeldud tehnoloogiatega ühildamiseks. Spetsifitseeritud on ühildamisvõimalused Sun J2EE (Java 2 Enterprise Edition) ja Microsoft COM tehnoloogiatega.

Rakenduse turvalisuse tagamine on kujunenud tarkvaraarenduses oluliseks teemaks. Eriti puudutab see erinevaid hajussüsteeme, mille alamosad võivad füüsiliselt asuda erinevates kohtades, ning mis on mõeldud kasutamiseks paljude kasutajate poolt. Rakenduse turvalisust pole võimalik vaadelda ühtse struktuurina. Tarkvara turvaarhitektuur moodustatakse erinevatest komponentidest, mis valitakse sõltuvalt antud tarkvara spetsiifikast. Tähtsamate turvaaspektide kategooriatena võib vaadelda keskkonna turvalisust, rakenduses endas realiseeritud turvaelemente ja rakendustevahelise suhtluse turvalisust. Hajussüsteemi arhitektuuris ei piisa ainult alamrakenduste turvamisest, vaid tuleb kokku puutuda kõigi kolme kategooriaga ka vahetarkvara tasemel. Valitud tehnoloogia peab arvestama kõigi süsteemis ühendatud keskkondade turvaelemente. Peab võimaldama alamrakenduste turvaelementide loomist. Kõige olulisemaks osaks on alamrakenduste turvalise suhtluse tagamine.

CORBA tehnoloogias on spetsifitseeritud turvalise koostöövõime arhitektuur CSI (*Common Secure Interoperability*). Sellele lisaks on spetsifitseeritud CORBA turvalisuse teenus (*Security Service*). Turvalisuse teenuse spetsifikatsioon on mõeldud nii turvalisuse teenuse loomiseks pakutud IDL liideste põhjal kui ka abstraktse mudelina turvaelementide loomiseks CORBA tehnoloogial põhinevates süsteemides.

Antud töö jätkab semestritöös ja bakalaureusetöös alustatud CORBA tehnoloogia temaatika käsitlemist. Semestritöö raames anti ülevaade CORBA tehnoloogiast. Bakalaureusetöö raames käsitleti CORBA rakenduse serveri ehitust ning tehti ülevaade CORBA COS teenustest. Antud töö üheks eesmärgiks on uurida CORBA tehnoloogia turvalisuse aspekte. Teiseks eesmärgiks on spetsifitseerida turvaline nimeteenus – CORBA INS nimeteenuse laiendatud variant. Kolmandaks eesmärgiks on luua katserealisatsioon spetsifitseeritud teenusele. CORBA turvalisuse aspekte on vaadeldud OMG dokumentatsiooni ning kahe CORBA realisatsiooni põhjal: OpenORB 1.3 (mõned aspektid ka 1.4 BETA versiooni põhjal) ja Borland Visibroker 4.5. Turvaline nimeteenus on spetsifitseeritud IDL liidesena. Katserealisatsioon on teostatud Java programmeerimiskeeles OpenORB CORBA realisatsioonil. Antud magistritöö jaguneb kolmele eesmärgile vastavateks osadeks. Esimene osa tööst on referatiivse iseloomuga ja kirjeldab OMG poolt pakutud lahendusi CORBA tehnoloogial põhinevates rakendustes esinevatele turvaprobleemidele. Teine osa kujutab endast töö autori poolt pakutud turvalise nimeteenuse spetsifikatsiooni. Kolmas osa on pühendatud spetsifitseeritud turvalise nimeteenuse realiseerimisele. Magistritööle on lisatud turvalise nimeteenuse katserealisatsioon.

1 Lühiülevaade CORBA tehnoloogiast

Antud töö ei sisalda detailset CORBA tehnoloogia üldise arhitektuuri kirjeldust (sellele teemale olid pühendatud antud tööle eelnevad semestritöö [JH-ST] ning bakalaureusetöö [JH-BT]) – alljärgnev peatükk on mõeldud sissejuhatava osana CORBA turvalisuse aspekte vaatlevasse töö ossa. Lühiülevaade CORBA tehnoloogiast põhineb materjalidel [CORBA].

CORBA (*Common Object Request Broker Architecture*) – üldine objektipäringute vahendaja arhitektuur – on objekt-orienteeritud komponentprogrammeerimise tehnoloogia klient-server ühendusel põhinevate hajussüsteemide konstrueerimiseks. OMG (*Object Managment Group*) poolt on välja töötatud CORBA abstraktne mudel, mis kujutab endast IDL (*Interface Definition Language*) keele formaadis liideste komplekti ning põhjalikku funktsionaalsuse kirjeldust. OMG poolt pakutud liidesed ja kirjeldused hõlmavad nii CORBA põhiarhitektuuri kui ka rea standardseid COS (*Common Object Service* – üldised objektide teenused) lisateenused. IDL (*Interface Definition Language*) – liideste defineerimise keel – on OMG poolt spetsifitseeritud keel, mis on mõeldud programmeerimisliideste kirjeldamiseks. IDL keele kirjelduse üheks osaks on spetsifikatsioonid liideste erinevatesse programmeerimiskeeltesse kujutamiseks (*mapping*). OMG poolt on pakutud IDL liideste kujutamise spetsifikatsioonid järgmistele keeletele: C/C++, Java, XML, Ada, COBOL, Lisp, PL/1, Python ja Smalltalk. Lisaks on spetsifitseeritud Java liideste kujutis IDL liidesteks. Kujutamise spetsifikatsioonid sisaldavad täielikku informatsiooni IDL liideste ja programmeerimiskeele vastavusest. (IDL keel on spetsifitseeritud [CORBA] peatükis 3.)

OMG enda poolt CORBA arhitektuuri ja lisateenuste realisatsioone pakutud pole. CORBA tehnoloogial põhineva hajussüsteemi loomiseks tuleb võtta kasutusele CORBA spetsifikatsioonide realisatsioon (edaspidi CORBA realisatsioon). CORBA realisatsioonid võivad erineda kasutatava CORBA spetsifikatsiooni versiooni poolest, põhispetsifikatsioonis kirjeldatud lisavõimaluste realiseeritud komplekti poolest, realiseeritud COS lisateenuste komplekti poolest ning realisatsioonipõhiste lisavõimaluste poolest (sõltumatute arendajate poolt spetsifitseeritud lisateenused, administreerimise ja konfigureerimise lisavõimalused). Lisaks võivad realisatsioonid erineda programmeerimiskeelte komplekti poolest, millele on realiseeritud IDL liideste kujutamine, ning selle poolest, millises programmeerimiskeeles on kirjutatud realisatsioon ise.

1.1 Lühiülevaade CORBA arhitektuurist

CORBA põhispetsifikatsiooni olulisemad moodulid moodustavad tuuma (CORBA Core). CORBA tuuma realisatsioon on minimaalne vajalik osa korrektselt toimivas CORBA realisatsioonis. ORB (*Object Request Broker*) – objektipäringu vahendaja – moodustab keskse osa CORBA tuumast. CORBA tuuma struktuuri kuuluvad suhtlemiseks vajalikud vaheliidesed ja objektide adapterid. CORBA arhitektuuris on klientideks need olemid (kas alamrakendused või sama rakenduse alamosad), mis suudavad teostada teenusepäringuid (*request service request*) serverilt. Serveriks CORBA arhitektuuris on teenuseid pakkuvad olemid (samuti kas eraldiseisvad alamrakendused või sama rakenduse alamosad). Iga teenuse funktsionaalsus asub IDL liidestega vastavuses loodud objektide realisatsioonides. Serverilt teenuse küsimine toimub objektiviite (*object reference*) abil. IOR (*Interoperable Object Reference*) koostööd võimaldav objektiviide sisaldab endas informatsiooni serveri leidmiseks ja objekti pärimiseks ORB vahendaja kaudu. (Objektipäringu vahendaja ORB on spetsifitseeritud [CORBA] peatükis 4.)

OMG poolt on spetsifitseeritud GIOP (*General Inter Orb Protocol*) protokoll klient-server suhtluseks CORBA arhitektuuril põhinevas hajussüsteemis. GIOP protokolli arhitektuur on mõeldud teiste võrgusuhtlusprotokollide transpordikihtide kasutamiseks. Näitena võib tuua IIOP (*Internet Inter Orb Protocol*) protokolli, kus transpordikihina kasutatakse TCP/IP protokolli, ning tulemusena on võimaldatud TCP/IP adresseerimise kasutamine. (CORBA tehnoloogias kasutatavad protokollid on spetsifitseeritud [CORBA] peatükis 12.)

Serverirakenduse poolel on oluline roll eraldatud objekti adapteritele (*Object Adapter*). Objekti adapteri abil toimub teenust pakkuvate objektide viidete vahendamine. Lisaks toimub objekti adapterite kaudu teenust pakkuvate objektide juhtimine ja haldamine. Ainukeseks kasutusel olevaks OMG poolt spetsifitseeritud objekti adapteriks on POA (*Portable Object Adapter*) – portatiivne objekti adapter. POA adapteri kasutamine võimaldab säilivate objektide loomist ning laiendab võimalusi objektide aktiveerimiseks. Iga POA adapteriga on võimalik siduda mitut teenustpakkuvat objekti. Teenust pakkuvate objektide tegevust juhitakse POA adapterile teguviiside (*policy*) määramise kaudu. POA adapteri ehitus lubab ühendada ühes serveris

asuvaid adaptereid puukujulisse andmestruktuuri. (Portatiivne objekti adapter on spetsifitseeritud [CORBA] peatükis 11.)

CORBA arhitektuuri põhjal on võimalik luua nii kiiresti realiseeritavaid lihtsaid rakendusi kui ka suuri ja keerukaid hajussüsteeme. Enamus piiranguid süsteemide projekteerimisel tulevad sisse kasutatava CORBA realisatsiooni spetsiifikast – millised osad CORBA spetsifikatsioonist on realiseeritud ning milliseid lisavõimalusi antud CORBA realisatsiooni raames pakutakse.

2 CORBA tehnoloogia turvaelemendid

CORBA tehnoloogias spetsifitseeritud turvalisuse aspektid jagunevad põhiliselt kahte ossa: CORBA põhispetsifikatsioonis kirjeldatud turvalisuse teemad ning eraldi spetsifitseeritud CORBA turvalisuse teenus. Lisaks nendele kahele on ka teisi OMG poolt pakutud spetsifikatsioone, kus käsitletakse CORBA turvalisuse aspekte. Üheks sellistest on tulemüüridega eraldatud CORBA rakenduse alamosade suhtlemise spetsifikatsioon.

Antud peatükis on vaadeldud CORBA põhispetsifikatsioonis [CORBA] kirjeldatud turvalisuse teemasid. Alates CORBA kolmanda versiooni (CORBA 3) esitamisest OMG poolt on CSI üldise koostöövõime turvalisus muutunud põhispetsifikatsiooni osaks.

2.1 Üldine koostöövõime turvalisus (CSI)

Koostöövõime on üks olulisemaid osi CORBA tehnoloogiast. Koostöömehhanism võimaldab CORBA olemitel omavahel heterogeensetes keskkondades suhelda. Koostööd võimaldav objektiviide (*IOR*) sisaldab endas viiteid serveri asukohale ja teenust pakkuvale objektile. Koostöövõime spetsifikatsiooni raames on spetsifitseeritud protokollid, mis võimaldavad suhelda IOR viidete edastamise kaudu. GIOP protokoll on koostöömehhanismi tuumaks. IIOP protokoll on samuti spetsifitseeritud koostöömehhanismi raames. Koostöövõime olulisuse tõttu on sellele mehhanismile turvalisuse lisamine tähtis CORBA arhitektuuri turvataseme tõstmisel.

Üldise koostöövõime turvalisuse (*CSI*) spetsifikatsioon laiendab koostöövõime mehhanismi, lisades sellele turvalisuse elemente. Lühend CSIv2 tähistab CSI spetsifikatsiooni teist versiooni, mis defineeriti CORBA 3 põhispetsifikatsiooni raames. Spetsifitseeritud on rida protokolle turvalise suhtlemise tagamiseks: SAS (*Secure Attribute Service*) protokoll, SSLIOP (*SSL Inter Orb Protocol*) protokoll, SECIOP (*Secure Inter Orb Protocol*) protokoll, CSIIOP (*Common Secure Interoperability Inter Orb Protocol*) protokoll. Koostöövõime spetsifikatsiooni liidesed koosnevad mitmest moodulist. CSI moodul kirjeldab koostöövõime keskseid andmestruktuure. Teised moodulid kirjeldavad konkreetseid protokolle. Alljärgnevates lõikudes vaatleme CSI raames spetsifitseeritud protokollide eesmärke ja ehitust. (Koostöövõime turvalisus on spetsifitseeritud [CORBA] peatükis 24.)

SAS on turvaliste atribuutide teenuse protokoll, mis annab võimaluse GIOP protokolli teadetele lisada rea turvaatribuute. Selliste turvaatribuutide alla kuuluvad kliendi autentimise parameetrid, õigusi ja privileege määravad parameetrid ning lisaparameetrid. SAS protokolli kasutus eeldab transpordikihi turvalisust (näiteks transpordikihi turvamist SSL abil). SAS protokolli ehitus koosneb kahest kihist. Esimene kiht – autentimise kiht (*Authentication Layer*) – vastutab klientide autentimise eest, teine kiht – atribuutide kiht (*Attribute Layer*) – vastutab privileegide ja teiste lisaatribuutide käsitlemise eest. SAS protokolli kohta eraldiseisvat IDL moodulit kirjeldatud pole. SAS protokoll on abstraktne reeglistik teiste protokollide spetsifitseerimiseks. SAS protokolli raames kirjeldatud andmestruktuurid on defineeritud CSI ja CSIIOP IDL moodulites. (SAS protokoll on spetsifitseeritud [CORBA] peatükis 24.)

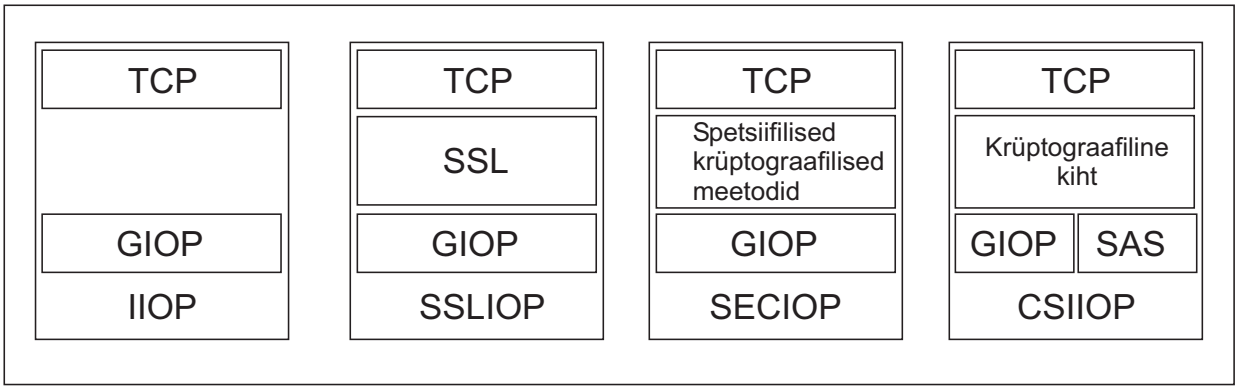
CSI koostöövõime spetsifikatsioonis on kirjeldatud rida andmetüüpe CSI protokollide kaudu toimiva suhtluse standardiseerimiseks. Autentimise ja autoriseerimise teostamiseks on spetsifitseeritud komplekt tõendeid (*tokens*). Kliendi autentimise tõend (*Client Autenthication Token*) koosneb kliendi tunnusest ja kliendi tunnust kinnitavast autentimise mehhanismi spetsiifilisest parameetrist. Identiteeditõend (*Identity Token*) koosneb binaarsel kujul olevast identifitseerivast objektist. Näiteks võib tuua X.509 sertifikaatide ahelal põhineva identiteeditõendi. Autoriseerimise tõendi (Authorization Token) moodustab autoriseerimise elementide (*AuthorizationElement*) järjend. Iga element koosneb binaarsel kujul olevatest autoriseerimise andmetest ja andmete tüübist.

SAS spetsifikatsioon laiendab GIOP protokolli. GIOP raames on defineeritud uus teenuse konteksti tüüp – turvaliste atribuutide teenuse (*SecurityAttributeService*) tüüp. Spetsifitseeritud kontekst võimaldab SAS spetsiifiliste teadete saatmist. Turvalise ühenduse käsitlemiseks on defineeritud neli uut teatetüüpi. Kaks nendest on mõeldud turvalise ühenduse konteksti tekitamiseks. Konteksti loomise (*EstablishContext*) teade saadetakse kliendi poolt turvalise ühenduse algatamiseks. Eduka loomise vastuseks saadetakse sihtobjekti poolt ühenduse konteksti loomise kinnitus (*CompleteEstablishContext*). Kinnituse käigus määratakse sihtobjekti poolt konteksti püsivus (*state*) – kas kontekst on taaskasutatav või mitte. Taaskasutatava konteksti puhul võib klient saata kontekstisiseseid teateid (*MessageInContext*). Konteksti loomisel või kontekstisisese teate saatmisel tekkinud vea puhul saadetakse kliendile konteksti veateade (*ContextError*).

CSI koostöövõime jaguneb kolmeks tasemeks. Baastase (*Level 0*) sisaldab autentimise tuge ja lubab SSL-kaitstud ühenduste kasutamist. Esimene tase (*Level 1*) laiendab baastaseme võimalusi, lisades autoriseerimise toetuse. Teine tase (*Level 2*) laiendab esimest taset, lisades autoriseerimistõendite delegeerimise võimalusi.

CSI koostöövõime spetsifikatsioonis on kirjeldatud kolm uut protokolli:

- SSLIOP – IIOP protokolli SSL laiendus. SSL krüptograafiline laiendus kehtib transpordikihi tasemel. Detailsemalt vaadeldes on näha, et SSLIOP protokollis asub SSL kiht GIOP ja TCP protokollide vahel. SSLIOP protokolli ehituse skeem on toodud joonisel 1.
- SECIOP – krüptograafilise laienduse kiht paikneb GIOP/IIOP ja ORB vahel. Antud protokoll võimaldab kasutada CORBA spetsiifilisi krüptograafia meetodeid. SECIOP protokolli ehituse skeem on toodud joonisel 1.
- CSIIOP – protokoll, mis kirjeldab CSIv2 teenuste konteksti ja IOR viite kodeerimisel kasutatavaid andmetüüpe. Antud protokollis kasutatavad turvalisuse spetsiifilised andmetüübid on spetsifitseeritud SAS teenuse raames. CSIIOP protokoll kujutab endast SAS teenuse poolt laiendatud IIOP protokolli. Erinevalt teistest CSI raames defineeritud protokollidest (SSLIOP, SECIOP) ei piirdu CSIIOP teadete krüptimisega, vaid lisab ka autentimise ja autoriseerimise mehhanismide kasutamise võimalusi. CSIIOP protokolli ehituse skeem on toodud joonisel 1.



Joonus 1. CORBA protokollide ehituse skeem

Avatud koodiga CORBA realisatsioonidest on CSIv2 teostus olemas OpenORB 1.4 realisatsioonis. Kahjuks on antud realisatsioon praeguse seisuga alles beetaversioonis ning CSIv2 osa pole veel kasutatav. Samas on lähemas perspektiivis oodata OpenORB 1.4 CORBA realisatsiooni koos toimiva CSIv2 turvalise koostöövõimega. SSLIOP protokolli realisatsioon on olemas OpenORB 1.3 versioonis. Antud töös spetsifitseeritud turvaline nimeteenus kasutab SSLIOP protokolli. SSLIOP protokolli praktilist kasutust vaatleme detailsemalt katserealisatsiooni kirjeldavas peatükis. (Antud protokollid on spetsifitseeritud [SSS] peatükis 3 ning kirjeldatud [CORBA] peatükis 24.)

Kommertsrealisatsioonidest on CSIv2 teostus olemas Orbix E2A 6.0 CORBA realisatsioonis.

3. CORBA turbetehnoloogiad

CORBA arhitektuuril loodud rakendustele turvaelementide lisamiseks on OMG poolt spetsifitseeritud mitmed turbetehnoloogiad. Alljärgnevates jaotistes (3.1, 3.2 ja 3.3) vaatleme kolme CORBA turbetehnoloogiat: CORBA turvalisuse teenust, ATLAS autoriseerimise tõendite teenust ja tulemüüride kasutust CORBA arhitektuuris. CORBA turbetehnoloogiate ülevaade põhineb materjalidel [CORBA], [SSS] ja [JPC].

3.1 CORBA turvalisuse teenus

Turvalisuse teenuse spetsifikatsioon kirjeldab raamistiku CORBA turvalise arhitektuuri ehitamiseks. Antud spetsifikatsioon koosneb nii COS teenuse IDL liidestest kui ka CORBA tehnoloogial põhinevatesse hajussüsteemidesse turvalisuse kihtide lisamise võimaluste kirjeldustest. Lisaks on spetsifitseeritud seosed CSI koostöövõimega. Erinevalt teistest CORBA COS teenustest ei ole turvalisuse teenus disainitud eraldi kasutamiseks. Spetsifitseeritud liidesed kirjeldavad ainult turvalisuse funktsionaalsust pakkuvaid teenuseid. Nende teenuste abil on võimalik lisada turvaelemente CORBA tehnoloogial põhinevatesse rakendustesse. Turvalisust vajava süsteemi komponent tuleb eraldi siduda turvalisuse teenuse funktsionaalsusega. Eraldiseisva turvalisust pakkuva teenuse loomiseks on tarvis spetsifitseerida lisateenus, mis koos CORBA turvalisuse teenusega hakkab toimima vahendajana klientide ja teenust pakkuvate objektide vahel. Samas on turvalisuse teenuse spetsifikatsioonis kirjeldatud mudelit võimalik vaadelda juhendina oma turvalise arhitektuuri ehitamiseks. Spetsifikatsioon kirjeldab rea olemasolevate turvatehnoloogiate kasutuse CORBA tehnoloogial põhinevate rakenduste loomisel. Selliste tehnoloogiate alla kuuluvad SSL, Kerberos, SESAME, DCE turvalisus ja veel mõned teised. (Turvalisuse teenuse põhisätted on spetsifitseeritud [SSS] peatükis 1. Turvalisuse teenuse seosed CSI koostöövõimega on spetsifitseeritud [CORBA] peatükis 13. [JPC] peatükk 12 on pühendatud CORBA turvalisuse teemadele.)

Turvalisuse tagamiseks peavad süsteemis olema ära kirjeldatud ohud, mille vältimiseks antud turvalisuse teenus on disainitud. CORBA turvalisuse teenuse raames vaadeldakse kriitiliste ohtudena järgmisi probleeme:

- Andmete ja teenuste võltsimine või rikkumine – kliendile valede või vigastatud andmete edastamine; kliendile pakutud teenuse mittevastavus oma spetsifikatsioonile või ebakorrektne käitumine.
- Konfidentsiaalsete andmete lekkimine – konfidentsiaalsete andmete sattumine õigusi mitteomava kasutaja kätte liikluse pealtkuulamise või teenuse võltsimise kaudu.
- Teenusetõkestus (DoS – *Denial of Service*) – süsteemi ressursside hõivamine õigusi mitteomava kasutaja poolt.
- Autoriseeritud kasutajate mittetuvastamine – süsteemis registreeritud kasutaja mittetunnistamine; õigusi omavale kliendile juurdepääsu keelamine andmeteni või teenust pakkuva objektini; teenust pakkuva objekti keeldumine teenuse osutamisest.
- Autoriseerimata juurdepääs – süsteemis registreerimata kasutaja sisenemine; õigusi mitte omava kasutaja juurdepääs ressurssidele.

Lisaks otsestele turvaohtudele tuleb hajussüsteemi turvalisuse ülesehitamisel arvestada ka arhitektuuri omadusi. Ohtlikkuse taset tõstvate arhitektuuri omaduste all tuleb silmas pidada hajussüsteemide arhitektuuri olulisemaid põhimõtteid: süsteem koosneb reast olemitest, mis paiknevad heterogeensetes keskkondades, ja suurem osa süsteemi tööst tugineb olemite vahelisel suhtlusel. Turvaelemente tuleb rakendada süsteemi igale komponendile. Kogu süsteemi ohustamiseks piisab ühest ebaturvalisest komponendist. Turvaline suhtlus komponentide vahel on vajalik edastatavate andmete pealtkuulamise või võltsimise vältimiseks.

Vaadeldud probleemide lahendamiseks on spetsifikatsioonis kirjeldatud nii üldiste kui ka CORBA-spetsiifiliste turvalisuse aspektide grupid. Turvalisuse teenuse raames on spetsifitseeritud käsitlus järgmistele turvalisuse fundamentaalsetele aspektidele:

- Andmete ning teiste ressursside konfidentsiaalsuse tagamine – juurdepääs andmetele ning teenustele on lubatud vaid õigusi omavatele klientidele.
- Hajussüsteemi ning andmete säilivuse ja ühtsuse tagamine – süsteemis kasutatavad andmed peavad säilima muutusteta vaatamata rünnetele ning võimalikele süsteemi tõrgetele; andmeid tohivad muuta vaid vastavaid õigusi omavad kliendid; kõigis süsteemi komponentides peavad andmed omama sama seisu.

- Pakutavate teenuste usaldatavuse tagamine – kliendile pakutud teenus peab vastama antud teenuse kirjeldusele. Pakutavate teenuste kättesaadavuse tagamine – õigusi omav klient peab saama kätte talle pakutavaid teenuseid.

Loetletud neli turvalisuse aspekti omavad olulist rolli mitte ainult CORBA tehnoloogial põhinevates rakendustes, vaid ka teistes hajussüsteemides.

Lisaks on kirjeldatud mõned CORBA-spetsiifilised turvalisuse aspektid:

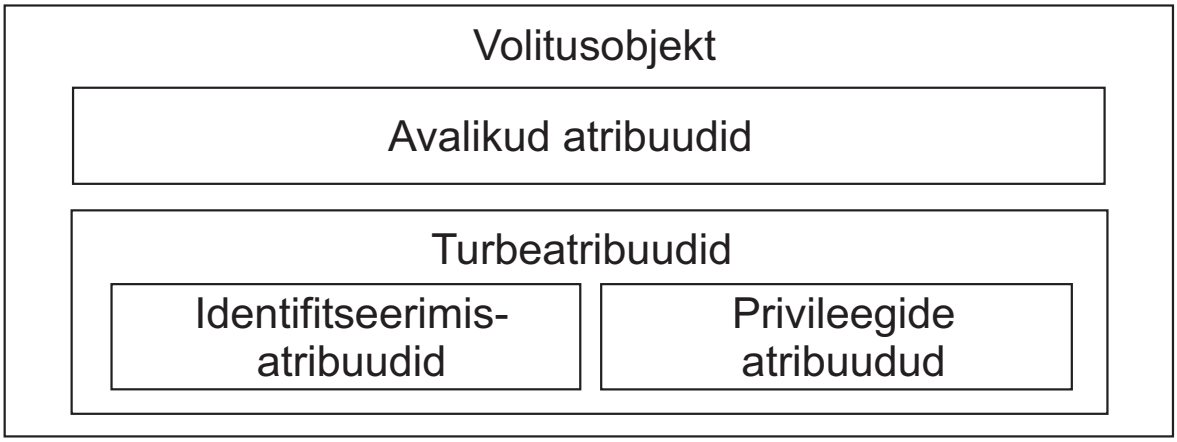
- Turvalise suhtluse tagamine heterogeensetes süsteemides, kus on kasutusel erinevate arendajate poolt pakutud CORBA realisatsioonide ORB vahendajad – turvalisus peab olema tagatud ka nendel juhtudel, kus süsteemi suhtlus on ehitatud mitme erineva ORB vahendaja peale.
- Turvateenuse objekt-orienteeritud liideste pakkumine – spetsifikatsioonis kirjeldatud IDL liideste kujul.
- Turvamehhanismide keerukuse peitmine lihtsate liideste taha – suhtlus turvalisuse teenusega nagu iga teise CORBA teenuse puhul toimub spetsifikatsioonis kirjeldatud IDL liidese põhjal, teenuse funktsionaalsus on kliendile nähtamatu.
- Objektide väljakutsete turvalisuse tagamine vastavuses määratud turvalisuse teguviisile – turvameetmete rakendamine toimub objekti väljakutsel, turvalisuse teenuse juhtimine toimub teguviiside määramise kaudu.
- Autentimise ning õiguste kontrolli tagamine objekti väljakutsel – kliendi autentimine ja õiguste kontroll turvalisuse teenuse poolt viiakse läbi teenust pakkuva objekti väljakutsel.

3.1.1 CORBA turvalisuse teenuse arhitektuur

Turvalisuse teenus on mõeldud kasutamiseks vahekihina erinevate süsteemi olemite vahel. Turvalisuse teenuse poolt pakutakse kahte põhiteenust: juurdepääsu juhtimine (*access control*) ja objekti turvaline väljakutse (s*ecure invocation*). (Turvalisuse teenuse ehitus ja liidesed on spetsifitseeritud [SSS] peatükis 2.)

Juurdepääsu juhtimine turvalisuse teenuses koosneb kliendi autentimisest antud süsteemis ning turvalisuse atribuutide sidumist kliendi tunnusega. Autentimine toimub kliendi tunnuse ning

autentimisatribuudi (selleks võib olla salasõna või sertifitseeritud võtmega tehtud digitaalne allkiri) vastavuse alusel. Autentimisele järgneb kliendi tunnuse seostamine volitusobjektiga (*credentials*). Volitusobjekt kujutab endast erinevate atribuutide komplekti: identifitseerimisatribuudid, privileegide atribuudid ja avalikud atribuudid. Identifitseerimisatribuudid määravad seose kliendiga, privileegide atribuudid kirjeldavad kliendi õigusi antud süsteemis ning avalike atribuutide kaudu võimaldatakse kliendiga siduda erinevaid lisaandmeid. Volitusobjekti ehitus on toodud joonisel 2.



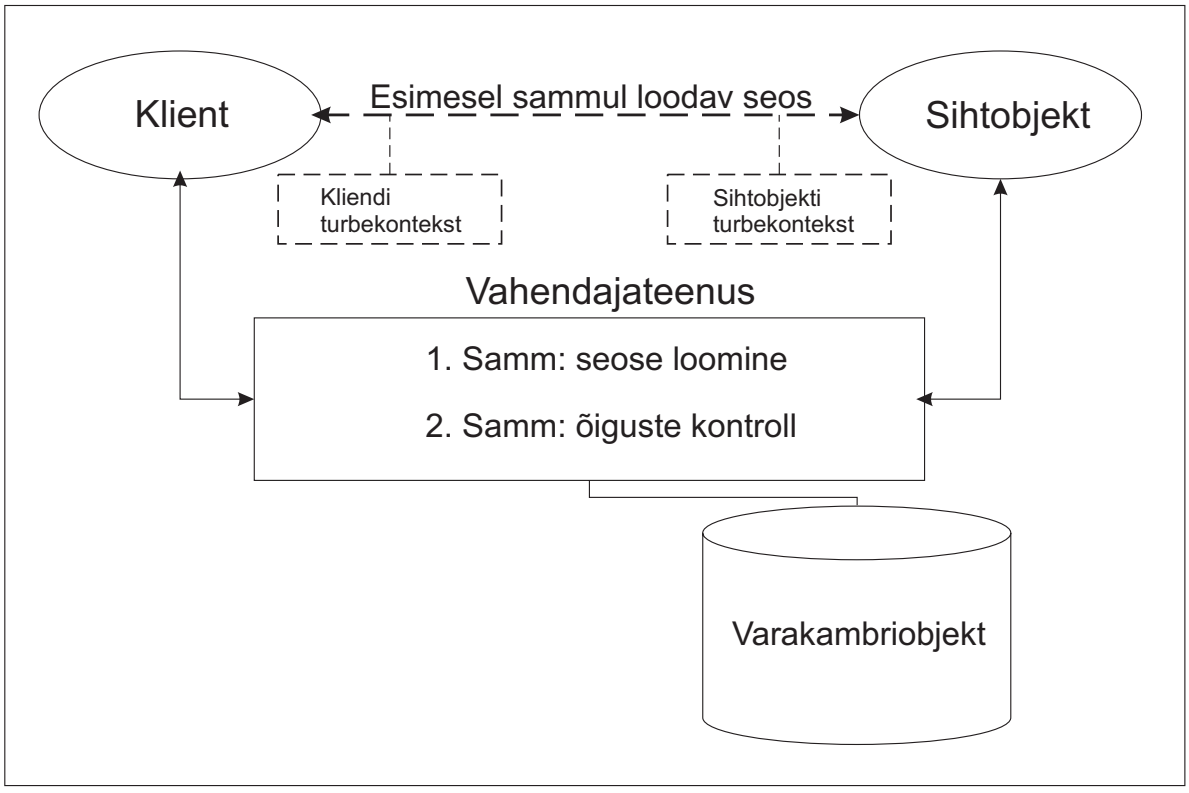
Joonis 2. Volitusobjekti skeem

Objekti turvalise väljakutse etapil toimub teenuseid pakkuvate objektide ning nende operatsioonide väljakutse juhtimine. Turvalise väljakutse tegemiseks peab teenust päriv klient olema läbinud juurdepääsukontrolli. Objekti turvalise väljakutse käigus kontrollitakse kliendi õigusi teenuseid pakkuvate objektide kasutamiseks ning operatsioonide väljakutseks. Kahe erineva juhuna tuleb vaadelda turvalisusega arvestavaid (*security-aware*) ja mittearvestavaid (*security-unaware*) teenuseid pakkuvaid objekte. Turvalisusega mittearvestavate objektide turvamine vajab vahendajat (lisateenust) turvalisuse teenusega sidumiseks. Teiseks lahenduseks on eraldiseisvalt toimimiseks täiendatud turvalisuse teenuse kasutamine.

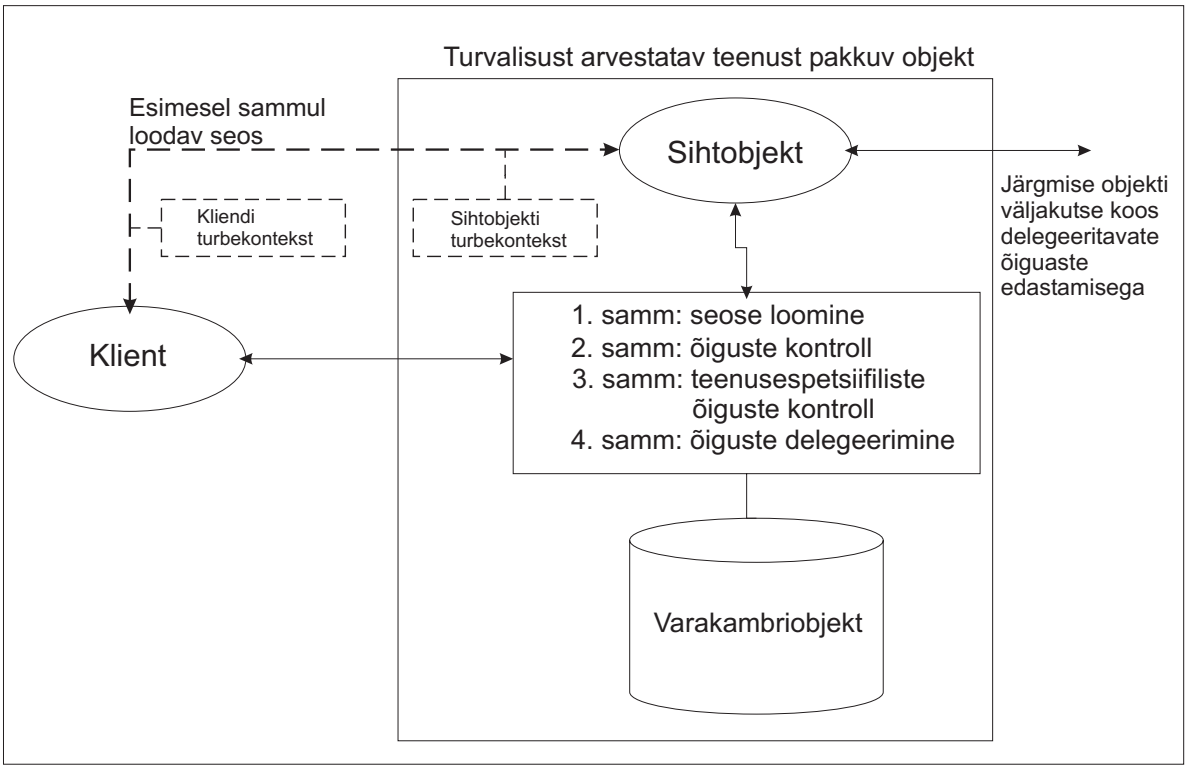
Esimesel juhul sisaldab teenust pakkuva objekti arhitektuur endas turvaatribuutide käsitlemise mehhanisme. Sellist liiki teenust pakkuv objekt on samaaegselt turvalisuse teenuse kliendiks. Turvalise väljakutse käigus on teenust pakkuvale objektile lubatud pärida teenuse päringu saatnud kliendi atribuute. Saadud atribuutide komplekti põhjal on teenust pakkuva objekti poolel võimalik läbi viia detailsemat õiguste kontrolli. Detailsem õiguste kontroll võimaldab

piirata teenuste kasutamist operatsiooni täpsusega. Lisaks on võimalik kasutada turvalisusega arvestatavat teenust pakkuvat objekti teiste teenuste turvaliseks väljakutseks. Antud käsitlus sobib paremini keerukamate süsteemide ehitamiseks, kus teenust pakkuvad objektid vajavad detailset õiguste kontrollimise võimalust. Turvalisust arvestavate objektide väljakutse skeem on toodud joonisel 4.

Teisel juhul toimub õiguste kontroll ainult turvalisuse teenuse tasemel ning piiramine toimub objekti väljakutse täpsusega. Teenust pakkuvatelt objektidelt ei nõuta turvamehhanismide toetuse omamist. Antud käsitlus sobib paremini lihtsamate süsteemide ehitamiseks, kus detailne õiguste kontroll pole vajalik. Lisaks võimaldab selline käsitlus kasutada muutmata kujul teenust pakkuvaid objekte, mille arhitektuuris pole arvestatud turvalisuse teenuse kasutamisega. Turvalisust mittearvestavate objektide käsitlemiseks tuleb kasutada turvalisuse teenuse funktsionaalsust pakkuvat vahendajateenust. Turvalisust mittearvestavate objektide väljakutse skeem on toodud joonisel 3.



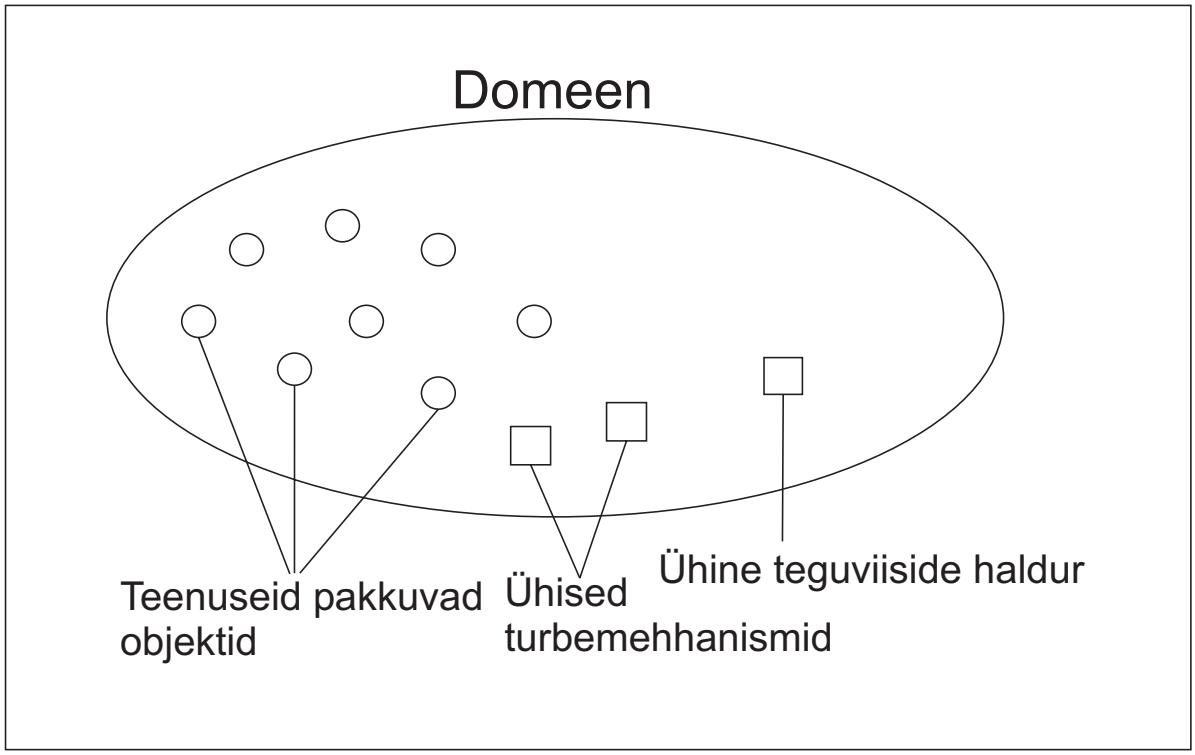
Joonis 3. Turvalise väljakutse skeem (security-unaware)



Joonis 4. Turvalise väljakutse skeem (security-aware)

Objekti turvaline väljakutse koosneb mitmest sammust. Esimesel sammul luuakse seos kliendi ja sihtobjekti vahel. Seose loomisel kinnitatakse turvalisuse teenuse poolt pakutava teenuse usaldatavust (objektiviide kuulub välja kutsutud teenust pakkuvale objektile). Kliendi ja sihtobjekti seose loomise eest vastutab varakambriobjekt (v*ault*). Autentimata kliendi puhul suudab varakambriobjekt automaatselt käivitada kliendi juurdepääsukontrolli. Varakambriobjekti poolt luuakse nii kliendi kui ka sihtobjekti kohta turbekontekstid (s*ecurity context*). Turbekontekstide paar moodustab vajaliku seose kliendi ja sihtobjekti vahel. Teisel sammul kontrollitakse kliendi õigusi antud objekti väljakutseks. Turvalisust mittearvestavate objektide puhul koosneb väljakutse vaid esimesest kahest sammust. Turvalisuse teenuse funktsionaalsust pakkuva vahendajateenuse tasemel teostatakse mõlemad sammud. Turvalisust arvestava teenuse puhul pole eraldiseisva vahendajateenuse kasutamine kohustuslik – turvalisuse teenuse funktsionaalsuse võib täielikult üle kanda teenust pakkuva rakenduse kihti. Turvalisust arvestava objekti väljakutsel võib samme olla rohkem. Esimesed kaks sammu on samad, mis turvalisust mittearvestatavate objektide puhul. Kolmanda sammuna kontrollitakse

kliendi õigusi teenust pakkuva objekti poolel. Teenust pakkuv objekt võib omakorda olla kliendiks mingi teise teenuse suhtes. Neljanda sammu käigus on võimalik kliendi õiguste delegeerimine teenust pakkuvale objektile järgmise väljakutse teostamiseks. Turvalised väljakutsed võivad moodustada piiramata pikkusega ahela, kus igaks lüliks on neljasammuline väljakutse. Turvalise väljakutse või väljakutsete ahela eduka lõpetamise peale saab klient päritud teenuse tulemuse.



Joonis 5. Turvalise domeenide üldine skeem

CORBA koostöövõime spetsifikatsiooni raames on defineeritud domeeni (*Domain*) mõiste. Domeeniks nimetatakse omavahel assotsieeritud teenust pakkuvate objektide gruppi. Grupi objekte siduv teenust pakkuv objekt moodustab domeeni. Domeeni moodustav objekt võib omakorda olla mingi teise domeeni liikmeks. Turvalisuse teenuse raames on defineeritud kolm domeeniliiki. Turvalisuse teguviiside domeen (*Security Policy Domain*) seob objekte ühise teguviiside haldamise mehhanismiga. Turvalise keskkonna domeeni (*Security Environment Domain*) liikmed peavad paiknema samas keskkonnas. Keskkonna mõiste alla kuuluvad nii füüsiliselt määratud keskkonnad (erinevad arvutid, riistvaraliselt eraldatud võrgusegmendid) kui ka tarkvaraliselt määratud keskkonnad (virtuaalsed masinad, tarkvaraliselt seotud

alamvõrgud). Turvalisust pakkuva domeeni moodustaja peab toetama keskkonnaspetsiifilisi turvalisuse aspekte. Turvalisuse tehnoloogia domeeni (*Security Technology Domain*) kõiki liikmeid turvatakse sama funktsionaalsust omavate teenuste poolt. Turvalisuse teenuse spetsifikatsioonis domeenide arhitektuuri kirjeldavaid liideseid defineeritud pole. Liideste puudumine annab võimaluse paindlikumalt arvestada loodava süsteemi spetsiifilisi vajadusi. Turvaliste domeenide kasutamine süsteemi arhitektuuris annab võimaluse turvalisust mittearvestavate objektide turvamiseks. Turvalisust arvestavate objektide koondamine ühisesse domeeni annab võimaluse parema haldamise korraldamiseks. Joonisel 5 on toodud turvalise domeeni variant, milles teenuseid pakkuvad objektid on ühendatud nii teguviiside halduse kui ühiste turbemehhanismide kasutuse poolelt. Lisaks võib selline domeen paikneda eraldatud keskkonnas.

Süsteemis toimuvate sündmuste jälgimine võimaldab avastada turvareeglite rikkumisi või rikkumiskatseid. Turvalisuse teenuse raames on võimalik teostada auditeerimist. Auditeerimine võimaldab jälgida kahte liiki sündmusi: turvalisuse teenuse sündmusi ja teostatavate teenuste sündmusi. Teiste teenuste sündmuste jälgimiseks peab antud teenust pakkuma turvalisust arvestav objekt ise. Jälgimisele kuuluvaid sündmusi ning käitumist erinevate sündmuste põhjal määratakse auditeerimise teguviiside komplektiga. Iga jälgitava sündmuse kohta on võimalik salvestada logikirje (sündmuse nimi koos olulisemate detailidega) või käivitada mingi tegevus (näiteks juurdepääsu keelamine).

Turvalisuse teenuse funktsionaalsust on võimalik vaadelda mitme kihina. Esimese kihi funktsionaalsus on suunatud turvalisust mittearvestavate olemite turvamiseks. Teise kihi funktsionaalsus laiendab esimest kihti, lisades olemite poolt kasutatavat funktsionaalsust, mis võimaldab detailsemat lähenemist turvalise arhitektuuri ehitamiseks. Turvalisuse teise kihi kasutus võimaldab turvalisust arvestavate olemite käsitlust. Esimesed kaks kihti on kaetud turvalisuse teenuse funktsionaalsusega. Turvalisuse teenuse funktsionaalsuse liidesed on jaotatud kolme gruppi: rakenduse taseme liidesed, administratiivse taseme liidesed ja turvalisuse taseme liidesed. Liidesegruppide detailsemad kirjeldused tulevad alljärgnevates punktides. Kolmanda kihi moodustab spetsiifilise turvatehnoloogia funktsionaalsuse kasutamine. Turvalisuse kolmas kiht tuleb detailsema vaatluse alla turvalisuse teenuse protokolle kirjeldavas peatükis.

3.1.2 Rakenduse arendusliidesed

Rakenduse arendusliideste paketid on mõeldud kasutamiseks süsteemi äriloogika realiseerimisel. Arendusliidesed on jaotatud kolme paketi vahel. Esimene pakett koosneb turvalisuse esimese kihi funktsionaalsust kirjeldavatest liidestest. Teine pakett koosneb turvalisuse teise kihi funktsionaalsust kirjeldatavatest liidestest. Kolmanda paketi moodustavad abifunktsionaalsust kirjeldavad liidesed.

3.1.2.1 Turvalisuse esimese kihi liidesed

Turvalisust mittearvestatavates süsteemi olemites turvalisuse teenuse poole pöördumisi ei toimu. Esimese kihi liideste pakett on mõeldud turvalisuse teenuse enda poolt kasutamiseks ning teise kihi liideste poolt laiendamiseks. Rakenduse arendusliideste esimene pakett koosneb vaid ühest liidesest: aktiivse kliendi liides (*Current*), mis kirjeldab operatsiooni kliendi atribuutide tagastamiseks.

3.1.2.2 Turvalisuse teise kihi liidesed

Süsteemi olemite poolt turvalise väljakutse tegemisel kasutatakse teises kihis kirjeldatud liideseid. Selles punktis vaatleme olulisemaid teise kihti kuuluvaid arendusliideseid. Teises kihis laiendatakse esimeses kihis defineeritud praeguse kliendi liidest. Lisaks kliendi atribuute tagastavale operatsioonile on kirjeldatud kliendi volitusobjekti käsitlevad operatsioonid. Kliendi tunnuse autentimise liides (*PrincipalAuthenticator*) defineerib operatsioone autentimise läbiviimiseks. Eduka autentimise tulemusena tagastatakse kliendi volitusobjekt. Lisaks volitusobjektile võib autentimise operatsioon tagastada spetsiifilisi lisaandmeid. Volitusobjekti liides (*Credentials*) kirjeldab andmestruktuuri, mis koosneb kliendi atribuutidest ning operatsioone nende käsitlemiseks. Auditeerimise otsustamise liides (*AuditDecision*) pakub funktsionaalsust rakenduses toimuvate sündmuste jälgimise korraldamiseks. Antud liideses defineeritud operatsiooni abil on võimalik seadistada sündmustele vastava auditeerimise tegevuse. Auditeerimise kirje salvestuse teostamiseks on defineeritud auditeerimise kanali liides (*AuditChannel*). Juurdepääsu otsuse liides (*AccessDecision*) on samuti defineeritud teise kihi liideste paketis. Antud liideses defineeritud operatsioon võimaldab kontrollida juurdepääsu õigusi sihtobjekti konkreetsele operatsioonile. Juurdepääsu otsus tehakse sisendina antud volitusobjektide komplekti põhjal.

3.1.3 Administratiivsed liidesed

Administratiivsed liidesed kirjeldavad turvalisuse teenuse juhtimise funktsionaalsust. Turvalisuse teenuse töö juhtimine toimub erinevate teguviiside määramise kaudu. Teenusele

seadistatud teguviiside põhjal langetatakse otsuseid erinevate tegevuste läbiviimiseks. Administratiivsetes liidestes kirjeldatakse kasutatavate teguviiside gruppe ja nende määramiseks mõeldud operatsioone. Juurdepääsu teguviiside määramiseks on defineeritud õiguste (*Rights*) ja õiguste perekondade (*Rights Family*) objektid. Administratiivsete liideste paketis on iga tegevuse liigi kohta defineeritud teguviiside käsitlemise liides:

- Juurdepääsukontrolli administreerimise liidesed – sihtobjekti operatsioonide väljakutseks käsitletakse vajalikke õigusi kohustuslike õiguste liidese (*RequiredRights*) kaudu; klientide kehtivate õiguste haldamine toimub juurdepääsu teguviisi liidese (*AccessPolicy*) kaudu; domeeni juurdepääsu liidese (*DomainAccessPolicy*) operatsioonide abil toimub õiguste haldamine turvalisuse teenuse domeenis.
- Auditeerimise juhtimise administreerimise liides – jälgimisele kuuluvate sündmuste liigid määratakse auditeerimise teguviisi liidese (*AuditPolicy*) kaudu; lisaks seadistatakse selle liidese operatsiooni abil auditeerimise kanali objekt.
- Turvalise väljakutse administreerimise liides – kliendi ja sihtobjekti sidumise parameetrite määramiseks kasutatakse turvalise väljakutse teguviisi liidese (*SecureInvocationPolicy*) operatsioone. Sidumise olulisemate parameetrite alla kuuluvad ebaturvalise sidumise võimaldamine (*NoProtection*), sidumise konfidentsiaalsuse nõue (*Confidentiality*), saadetavate teadete terviklikkuse nõue (*Integrity*), sihtobjekti usaldatavuse kontrolli nõudmine (*EstablishTrustInTarget*) ja kliendi usaldatavuse kontrolli nõudmine (*EstablishTrustInClient*).
- Õiguste delegeerimise administreerimise liides – turvaliste väljakutsete ahelas volitusobjektide delegeerimist lubatakse delegeerimise teguviisi liidese (*DelegationPolicy*) operatsioonidega.

3.1.4 Realisatsiooni liidesed

Realisatsiooni liidesed kirjeldavad turvalisuse teenuse struktuuri põhiobjekte. Põhiobjektide hulka kuuluvad turvalisuse teenuse funktsionaalsuses kasutatavad andmetüübid ning teenuse sisemise struktuuri olemid. Selliste liideste alla kuuluvad:

- Varakambriobjekti liides (*Vault*) – defineerib operatsioone volitusobjektide loomiseks, turbekontekstide loomiseks ja sidumiseks.
- Turbekonteksti liides (*SecurityContext*) – defineerib operatsioone kliendi ja sihtobjekti turbekontekstide vahel oleva seose loomiseks ning teadete vahetamiseks.

3.1.5 Turvalisuse teenuse protokollid

Turvalisuse teenuse spetsifikatsioonis on kirjeldatud CSI koostöövõime funktsionaalsuse kasutuse võimalused. Turvalise suhtluse saavutamiseks on kirjeldatud CSI protokollide kasutust. Erinevalt CSI koostöövõime spetsifikatsioonist vaadeldakse turvalisuse teenuse raames eelkõige protokollide kasutamise aspekte. Uusi protokolle turvalisuse teenuse spetsifikatsioonis kirjeldatud pole.

3.1.6 Turvalisuse teenuse seosed teiste turvatehnoloogiatega

CORBA turvalisuse teenuse spetsifikatsioonis on kirjeldatud võimalused teiste turvatehnoloogiate kasutamiseks. Lisaks on nendes spetsifikatsioonides defineeritud rida IDL liideseid turvalisuse teenuse funktsionaalsuse täiendamiseks. Põhjalikumalt vaatleme kahte nendest tehnoloogiatest: SSL integreerimist CORBA turvalisuse teenusega ja Kerberose tehnoloogia kasutust CORBA turvalisuse teenuse raames.

3.1.6.1 SSL integreerimine CORBA turvalisuse teenusega

SSL (*Secure Socket Layers*) pakub TCP/IP protokolli baasil turvalist transpordikihti. Kuna IIOP protokoll kasutab transpordikihina TCP/IP protokolli, siis SSL võimaldab turvata ka IIOP protokolli transpordikihi tasemel. IIOP protokolli SSL laiendus on defineeritud SSLIOP protokolli kujul CSI koostöövõime spetsifikatsioonis.

3.1.6.2 Kerberos

Kerberose tehnoloogia võimaldab luua ühise autentimismehhanismiga hajussüsteeme. Peale edukat autentimist väljastatakse kliendile pilet (*ticket*), millega ta saab pöörduda süsteemi komponentide poole. Kerberose pileti standardiseeritud kuju annab võimaluse kasutada ühist autentimist CORBA tehnoloogial põhinevates rakendustes ja teistes rakendustes. Turvalisuse teenuse spetsifikatsioonis on defineeritud rida IDL liideseid Kerberose kasutamiseks.

3.1.7 Turvalisuse teenuse lühikokkuvõte

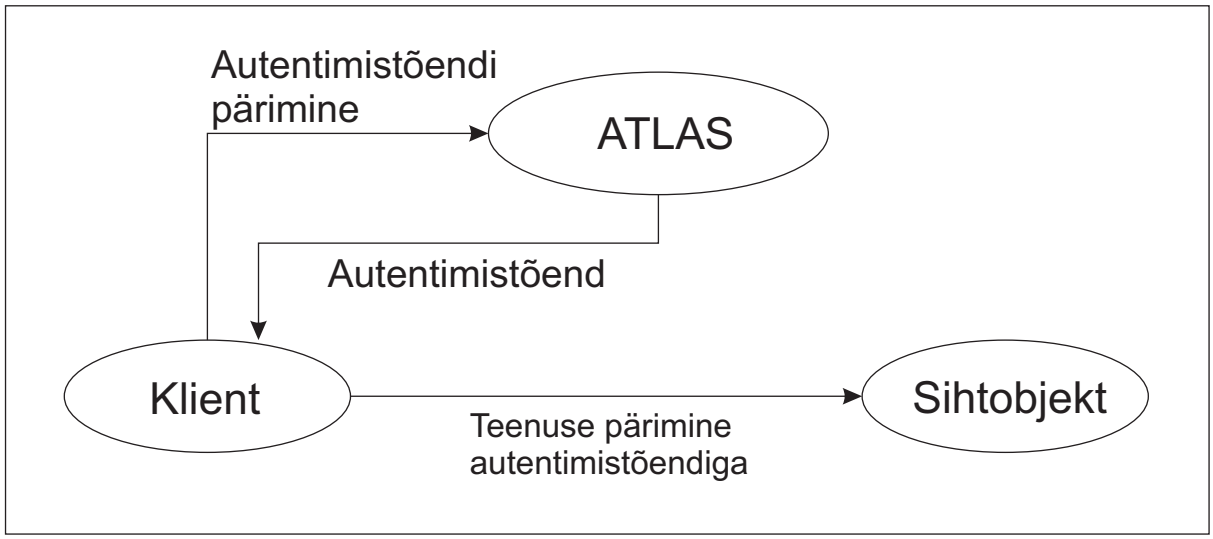
CORBA turvalisuse teenuse üheks suuremaks puuduseks on eraldiseisva teenusena realiseerimise võimaluse puudumine. Turvalisuse teenus võimaldab vaid laiendada teisi teenuseid. Ainult turvalisuse eest vastutava teenuse loomiseks tuleb turvalisuse teenusele lisada vahendaja, mis käsitleks teiste teenuste väljakutset. Antud puudus on üheks võimalikuks põhjuseks, miks turvalisuse teenus pole realiseeritud paljudes levinumates CORBA realisatsioonides (semestri-, bakalaureuse- ja magistritöö raames vaadeldud CORBA

realisatsioonides ta puudub). Osaline turvalisuse teenuse realisatsioon on olemas Orbix E2A 6.0 CORBA realisatsioonis (kommertsrealisatsioonina Orbix antud töö raames detailse vaatluse all pole, kõik märkused selle realisatsiooni kohta on tehtud tasuta jagatava dokumentatsiooni [Orbix] põhjal).

3.2 ATLAS

Erinevate turvalisust pakkuvate teenuste illustreerimiseks on antud töösse lisatud ATLAS-teenuse lühikirjeldus. Autoriseerimistõendi hankimise teenus (*ATLAS – Authorization Token Layer Acquisition Service*) on OMG poolt spetsifitseeritud lisateenus. Antud teenuse põhieesmärk on kajastatud selle nimes – kliendi autoriseerimistõendi koostamine identiteeditõendi alusel. ATLAS-teenus on mõeldud kasutamiseks autoriseerimise kihina CORBA tehnoloogial põhinevates rakendustes. Spetsifikatsioonis on kirjeldatud vajalikud IDL liidesed teenuse loomiseks. ATLAS-teenuse raames on kasutusel rida objekte teistest spetsifikatsioonidest. CSI koostöövõime spetsifikatsioonist on kasutusel identiteedi- ja autoriseerimistõendid, millest on koostatud ATLAS-teenuse autoriseerimistõend (*AuthTokenData*). CSI tõendite kasutus lisab ATLAS-teenusele CSIIOP protokolli kasutuse eelduse. ATLAS autoriseerimistõend väljastatakse kliendile identiteeditõendi põhjal. Autoriseerimistõendite väljastamise eest vastutab autoriseerimise juht (*AuthTokenDispenser*). Autoriseerimise juhi liides on defineeritud ATLAS-teenuse spetsifikatsioonis. Autoriseerimise juhi objektide struktureerimiseks kasutatakse nimeteenust. Struktureerimine on vajalik erinevaid privileege käsitlevate autoriseerimise juhtide mugavaks hoidmiseks ühe ATLAS-teenuse raames. Nimeteenuse struktuuris õige autoriseerimise juhi leidmiseks on defineeritud ATLAS-lokaator (ATLASLocator). ATLAS-lokaator on oma olemuselt autoriseerimise juhtide aadressiraamatuks.

ATLAS-teenus omab lihtsat kasutusskeemi. Klient pöördub ATLAS-teenuse poole autoriseerimistõendi saamiseks. Vajalik autoriseerimise juht leitakse ATLAS-lokaatori abil nimeteenuse struktuurist. Autoriseerimise juhi poolt väljastatakse kliendile autoriseerimistõend, millega klient pöördub järgmisel sammul sihtobjekti poole. ATLAS-teenuse kasutusskeem on toodud joonisel 6.

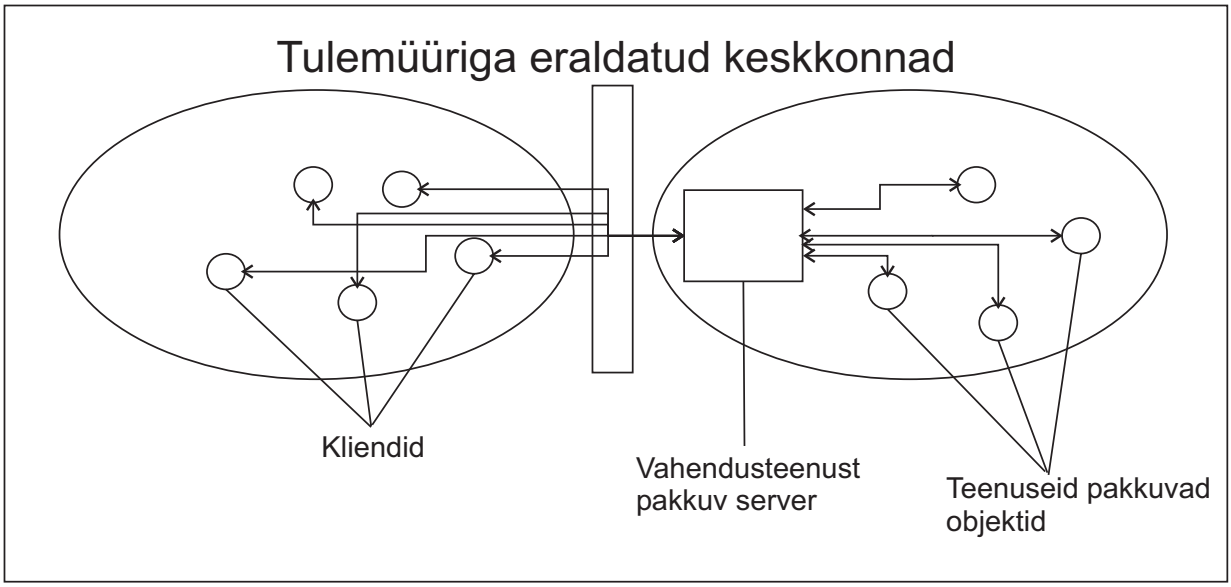


Joonis 6. ATLAS-teenuse kasutusskeem

ATLAS-teenuse spetsifikatsioon näitab turvaelementide loomise võimalusi väikeste lisateenuste kujul. ATLAS-teenuse realisatsiooni puudumine (semestri-, bakalaureuse- ja magistritöö raames vaadeldud CORBA realisatsioonides) muudab praktilise kasutuse raskeks – rakenduse arendaja peab realiseerima ka ATLAS-teenuse. ATLAS-teenuse tihe seos CSIv2 koostöövõime tehnoloogiaga on realisatsiooni puudumise üheks põhjuseks. Vaadeldud realisatsioonides puudub ka CSIv2 koostöövõime lõplik realisatsioon (OpenOrb 1.4 on beeta seisus). (ATLAS-teenus on spetsifitseeritud [ATLAS] dokumendis.)

3.3 CORBA ja tulemüürimine

CORBA tehnoloogial põhineva rakenduse loomisel tuleb tulemüüridega kokku puutuda kahel juhul. Esimesel juhul paiknevad klient ja server tulemüüriga eraldatud keskkondades. Kliendi ja serveri vaheline suhtlus toimub IIOP protokolli kaudu ja tulemüüriga TCP pakettide filtreerimise seadistamisel tuleb seda arvestada. Üheks võimalikuks lahenduseks on serverile fikseeritud pordi määramine koos tulemüüri poolt selle pordi lubamisega. Teine juht laiendab esimest – süsteemi ehituses turvalisuse taseme tõstmiseks kasutatakse ära tulemüüri võimalusi. Tulemüüriga eraldatud keskkonnast lubatakse suhtlust vaid vahendusteenust pakkuvale serverile. Vahendusteenust pakkuval serveril põhinev suhtlusskeem on toodud joonisel 7. Kõik teised serverid on väljastpoolt tulemüüri kättesaamatud. Teisi teenuseid vahendava serveri kohustuste hulka tulevad klientide autentimine ja autoriseerimine. Sageli kasutatakse tulemüüre erinevate võrkude vahel suhtlemise filtreerimiseks. Keerulisem juht esineb siis, kui klient ja server paiknevad erinevates võrkudes ning pole sama võrgu raames teineteisele nähtavad. Seda juhtu vaatleme detailsemalt allpool.



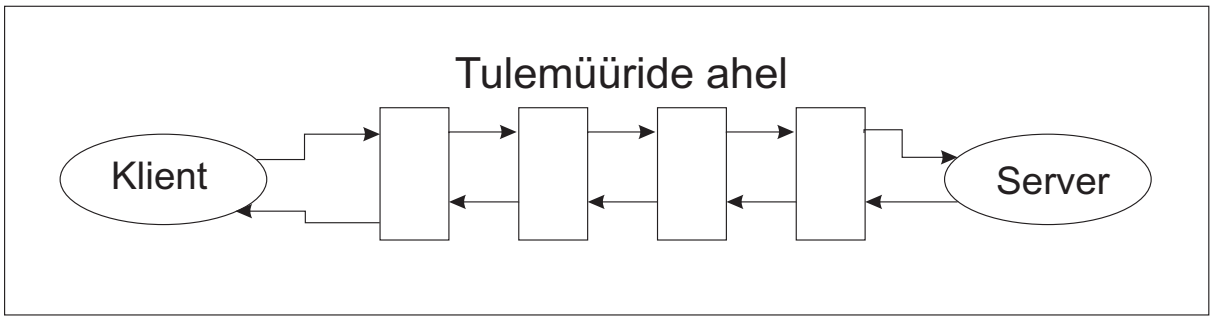
Joonis 7. Tulemüürimise kasutamise skeem

Tavaliste tulemüürirakenduste kasutamine võimaldab filtreerida CORBA rakenduste suhtlust vaid transpordikihi tasemel. OMG poolt on pakutud tulemüüri läbimise (*Firewall Traversal*) spetsifikatsioon, mis kirjeldab laiemaid võimalusi tööks tulemüüriga eraldatud võrgusegmentides. Antud spetsifikatsiooni raames on kirjeldatud GIOP protokolli tasemel

filtreerimise võimalused ja tulemüüride ahela läbimise mehhanismid. Sihtobjekti kliendist eraldavad mitme kihina paiknevad tulemüürid moodustavad tulemüüride ahela (*Firewall Path*).

Tulemüüri läbimise spetsifikatsiooni raames on defineeritud GIOP protokolli ja IOR objektiviidet täiendavad IDL liidesed. Filtreerimise võimaluste laiendamiseks on defineeritud identsusliittõendi (*CompoundIdentityToken*) liides. IOR objektiliides sisaldab endas piisava informatsiooni teenust pakkuva objekti asukoha kohta. Tavaline IOR objektiviide sisaldab serveri hosti nime, porti ja objektivõtit. Tulemüüride ahela läbimiseks on IOR objektiviidet täiendatud. IOR objektiviite struktuuri on lisatud järjend, mille elementideks on läbitavate tulemüüride aadressid. GIOP protokolli on defineeritud uus teate tüüp – kokkuleppeteade (*NegotiateSession*). Enne teenusepäringu saatmist saadab klient kokkuleppeteate ühenduse algatamiseks. Ühenduse algatamise käigus koostatakse tulemüüride ahela aadresside järjend, mille põhjal toimub järgnev suhtlus. Identsusliittõendi objekt sisaldab informatsiooni kliendi identiteedi ja läbitud tulemüüride ahela sammude kohta. Spetsifikatsioonis defineeritud identiteeditõendit (*IdentityToken*) kasutatakse nii kliendi kui ka läbitud tulemüüride identifitseerimiseks. Klienti tähistava identiteeditõedi põhjal toimub põhiline päringu filtreerimine. Edukalt läbitud tulemüüride identiteeditõendid pakuvad lisainformatsiooni, mille põhjal on võimalik kitsendada filtreerimise tingimusi.

Tulemüüride läbimise tehnoloogia pakub CORBA rakenduste suhtluse laiendatud filtreerimist ja mitme tulemüüriga eraldatud rakenduste suhtlemist. Tulemüüride läbimise teenuse skeem on toodud joonisel 8. Antud tehnoloogia kasutusele võtmine nõuab CORBA tulemüürimist teostava vahendaja realiseerimist ning täiendatud protokollide olemasolu kasutatavas CORBA realisatsioonis. Antud töös vaadeldavates CORBA realisatsioonides tulemüüride läbimise



Joonis 8. Tulemüüri läbimise teenuse skeem

vahendite teostused puuduvad. (Tulemüüride läbimine on spetsifitseeritud [CFTR] dokumendis. [JPC] peatükk 12 on pühendatud CORBA turvalisuse teemadele ning kirjeldab ka CORBA rakenduste tulemüürimise võimalusi.)

Kliendi ja serveri paiknemine erinevates võrkudes nõuab erilist käsitlust. Antud probleemi lahenduseks on mõlemast võrgust nähtavate vahendajate (*proxy*) kasutamine. Lisaks lahendab vahendajate kasutamine probleemi, kus kliendi ja serveri suhtlemiseks vajaliku pordi avamine pole võimalik. Näiteks võib tuua juhu, kus kliendile on lubatud vaid HTTP ühendused. Vahendaja ja serveri suhtlus toimub IIOP ühenduse kaudu ja samal ajal kliendi ja vahendaja suhtlus toimub HTTP ühenduse kaudu. Kliendi rakendus peab samuti omama vahekihti HTTP päringu tulemuste teisendamiseks. Heaks näiteks on Visibroker CORBA realisatsioonis pakutud vahendaja Väravavaht (*Gatekeeper*). Väravavahiteenus pakub vahendust nii tulemüüridega suhtlemisel kui ka HTTP tunnelite ehitamisel. (Visibroker CORBA realisatsiooni kohta käivad materjalid: [VBDG], [VBGK].)

4 CORBA nimeteenus

CORBA põhiarhitektuuri laiendamiseks on OMG poolt spetsifitseeritud komplekt üldiseid objektide teenuseid (COS – *Common Object Services*). Üldised teenused on mõeldud abiteenustena CORBA tehnoloogial põhinevatesse rakendustesse. Nende teenuste alla kuuluvad teenust pakkuvate objektide struktureerimist, pärimist, haldamist ja turvamist abistavad teenused. Üldiste teenuste alla kuulub ka ülalkirjeldatud turvalisuse teenus. Üldiste teenuste standardiseeritud kuju võimaldab sama teenuste samaaegset kasutamist erinevate rakenduste poolt. Erinevates CORBA realisatsioonides pakutavad üldiste teenuste komplektid võivad erineda. Teenuse olemasolu ühes või teises realisatsioonis sõltub sellest, kui oluliseks seda teenust peetakse. Erinevalt paljudest teistest teenustest on nimeteenus realiseeritud enamikus CORBA realisatsioonidest. (CORBA nimeteenus on spetsifitseeritud [INS] dokumendis.)

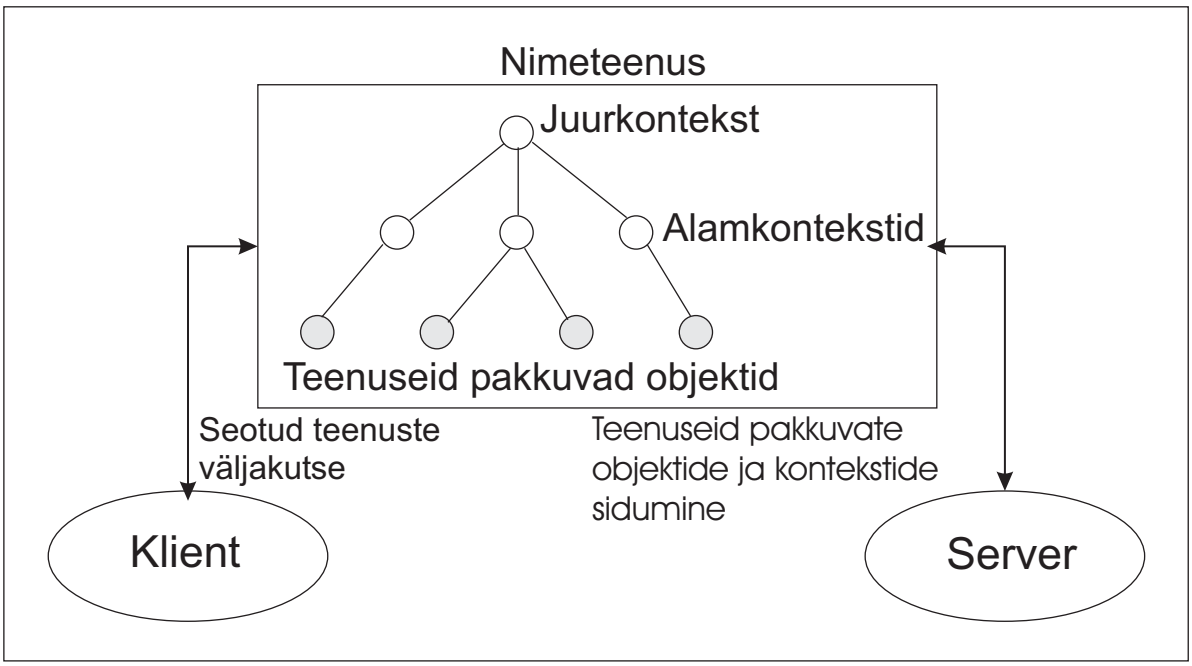
CORBA arhitektuuris toimub teenust pakkuva objekti pärimine objektiviite kaudu. Koostööd võimaldav objektiviide sisaldab endas piisava informatsiooni teenust pakkuva objekti asukoha kohta. Objektiviite puuduseks on selle kuju, millest pole võimalik otse välja lugeda informatsiooni objekti kohta. Nimeteenuse põhieesmärgiks on teenust pakkuvate objektide sidumine loetavate nimedega. Nimeteenuse teiseks eesmärgiks on teenust pakkuvate objektide struktureerimine puukujuliseks andmestruktuuriks. Nimeteenuse arhitektuur lubab koostada ka graafikujulisi struktuure, kuid neid ei soovitata kasutada. Nimeteenuse laiendatud versioon – koostööd võimaldav nimeteenus ehk INS (*Interoperable Naming Service*) spetsifitseerib URL kujulisi objektiviidete kujutisi. INS raames spetsifitseeritud ‘corbaname’ URL standard võimaldab kujutada objektiviidet nimede järjendina, mis kirjeldab tee päritava objektini nimeteenuse struktuuris. Nimeteenus sarnaneb paljude omaduste poolest erinevate kataloogiteenuste ja failisüsteemidega. Nimeteenuse üheks suuremaks erinevuseks enamikust kataloogiteenustest on turvalisuse mehhanismide puudumine. Antud töö üheks eesmärgiks ongi nimeteenuse ehituse täiendamine, lisades sellele turvalisuse elemente.

OMG poolt spetsifitseeritud nimeteenuse ehitust vaadeldi autori bakalaureusetöö raames. Antud töö raames kirjeldame enne turvalise nimeteenuse spetsifitseerimist lühidalt nimeteenuse arhitektuuri olulisemaid aspekte.

Nimeteenuse arhitektuur põhineb nimekonteksti (*naming context*) objektil. Nimekonteksti raames luuakse seoseid objektide ja neile antud nimede vahel. Ühe konteksti raames objektidele antud nimed on unikaalsed. Üks objekt võib olla seotud mitme nimega ühe või mitme konteksti raames. Nimekontekstid seotakse nimedega nagu kõik teised objektid. Nimekonteksti struktuuri ülesehitus toimub nimekontekstide omavahelise sidumise kaudu. Ühe nimekonteksti mitme nimega sidumise puhul on võimalik luua ka graafikujulisi struktuure (alamkontekstis on võimalik siduda ülemkontekstid mingi teise nimega). Sellise võimaluse tõttu nimetatakse nimekontekstide struktuuri nimede graafiks (*naming graph*). Nimeteenust kasutatavate rakenduste poolel soovitatakse võimalike segaduste vältimiseks (näiteks juhul kui nimedegraafis tekivad tsüklid) piirduda puukujuliste struktuuride loomisega. Joonisel 9 on toodud nimeteenuses loodava struktuuri näide.

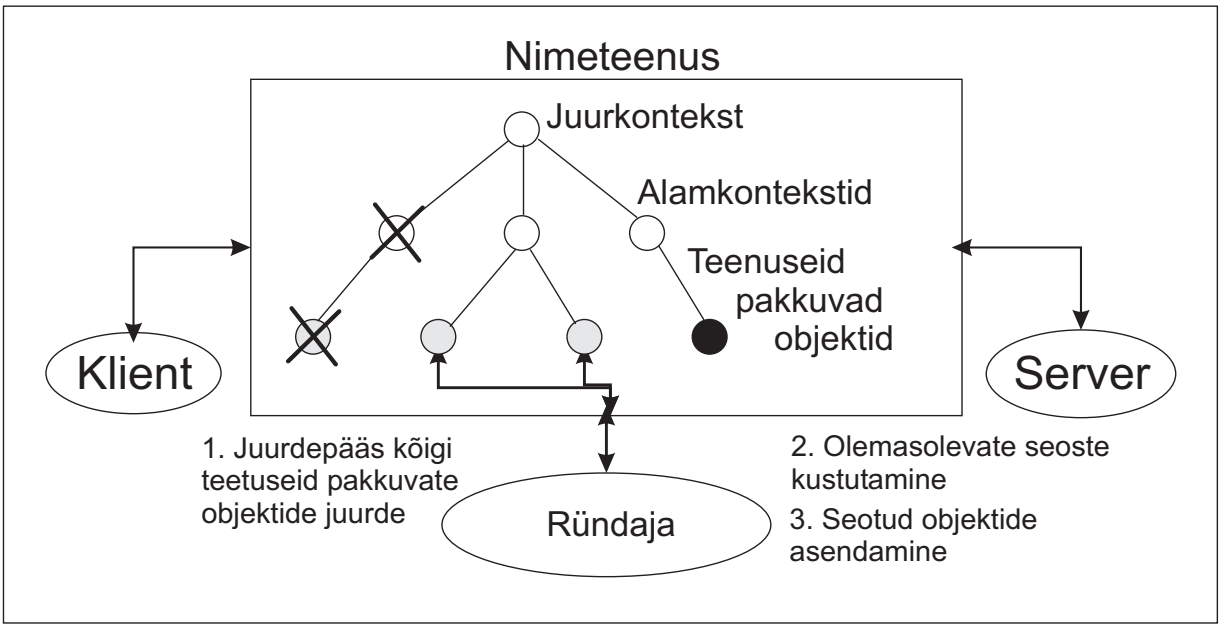
Nimeteenuse spetsifikatsioonis on defineeritud kõik vajalikud IDL liidesed teenuse loomiseks. Nimeteenuse liideste paketis on defineeritud nimekonteksti liides, laiendatud nimekonteksti liides (INS nimekontekst) ja nimejärjendi (*Binding Iterator*) liides. Nimekonteksti liideses on kirjeldatud operatsioonid nimede seoste käsitlemiseks. Kasutusmugavuse huvides on nimekontekstide sidumise operatsioonid defineeritud eraldi objektide sidumise operatsioonidest. Nimejärjendi liides kirjeldab nimeseoste listide läbimist abistavat objekti.

Nimeteenuse suhtes on klientideks nii teenust pakkuvaid objekte serveeriv rakendus kui ka teenust päriv klient. Klient- ja serverrakenduse nimeteenusega suhtlemise skeem on toodud joonisel 9. Objekti nimega sidumiseks pöördub serveri rakendus nimeteenuse poole. Teenuse pärimisel pöördub omakorda kliendi rakendus nimeteenuse poole. Nimeteenuse poolel lahendatakse päringus etteantud nimi ja vastusena saadetakse kliendile viide teenust pakkuva objektile. Sellise suhtlusskeemi puhul ei pea klient omama informatsiooni serverirakenduste asukohtadest. Nimeteenuse kaudu saab klient pärida teenuseid erinevatelt rakendustelt teadmata, kuhu täpselt päring edasi suunatakse. Nimeseose loomiseks ei pea rakendus serveerima seotavat teenust, piisab kui on teada viide teenust pakkuvale objektile. Seda omadust kasutades on võimalik luua nimeteenuse administreerimise eest vastutavaid abirakendusi. Nimeteenuse omadused võimaldavad omavahel mugavalt siduda ühte hajussüsteemi erinevaid CORBA tehnoloogial põhinevaid rakendusi.



Joonis 9. Nimeteenuse struktuuri skeem

Nimeteenuse struktuuriga on seotud rida ohte. Nimeteenusele juurdepääsu omavad kliendid tohivad sooritada kõiki nimeteenuse operatsioone. Mingit võimalust operatsioonide väljakutsete piiramiseks pole. Piirangute puudumise tõttu võib iga klient muuta ja kustutada olemasolevaid seoseid. Tulemusena võib teenust päriv klient saada ebakorrektse vastuse. Veelgi ohtlikum olukord võib tekkida teenuse väljakutsel konfidentsiaalsete andmete edastamiseks. Samuti puudub võimalus nimeseoste lahendamise piiramiseks. Nimeteenuse poole pöörduv klient võib pärida kõiki nimedega seotuid teenuseid. Nimeteenuse arhitektuuris puuduvad võimalused nende ohtude vältimiseks. On küll võimalik piirata klientide juurdepääsu nimeteenusele väliste meetoditega (näiteks paigutada nimeteenus turvalisse keskkonda). Nimeteenuse sisemisele struktuurile turvaelementide lisamine nõuab siiski arhitektuuri täiendamist. Kokkuvõtteks võib tuua välja nimeteenusega seotud kolm põhilist turvaprobleemi: ei kontrollita nimeteenuse administreerimisega tegelevaid kliente, ei kontrollita nimeteenuses seotuid teenust pakkuvaid objekte ega kontrollita nimeteenuse kaudu teenuseid pärivaid kliente. Ründaja rakenduse ja nimeteenuse vahel suhtlemise skeem on toodud joonisel 10. Nimeteenuse ründaja rakenduse näide on toodud punktis 5.5.2 (testrakendust kirjeldav punkt).



Joonis 10. Nimeteenuse ohtude skeem

5 Turvaline nimeteenus

CORBA nimeteenus pakub mugavaid lahendusi suhtluse korraldamisel erinevate rakenduste vahel. Samas on nimeteenusega seotud rida turvaprobleeme. Neid turvaprobleeme ja nende põhjuseid sai eelnevates peatükkides põhjalikult vaadeldud. Nimeteenuse omadused teevad selle kasutuse oluliseks CORBA tehnoloogial põhinevate rakenduste loomisel. Hajussüsteemides on turvalisuse tagamine üks olulisemaid aspekte. Turvalisuse taseme tõstmiseks peavad kõigis süsteemi komponentides olema turvalisuse eest vastutavad mehhanismid. Paljude komponentide puhul piisab välise juurdepääsu piiramise realiseerimisest. Eelmises peatükis vaadeldud turvaprobleemide lahendamiseks sellest siiski ei piisa. Nimeteenus vajab turvalisuse eest vastutavaid mehhanisme ka oma sisemise struktuuri turvamiseks. Antud töö üheks eesmärgiks ongi spetsifitseerida nimeteenuse täiendatud variant, kus oleks pakutud nimeteenusega seotud turvalisuse probleemide lahendused. Nimeteenuse turvamine võimaldab lisaks ülalkirjeldatud probleemide lahendamisele saavutada ka rea teisi eesmärke. Turvalise nimeteenuse kaudu on võimalik pakkuda turvaelemente teenustele, mille ehitamisel ei ole arvestatud turvalisusega. Turvalisust arvestatavaid rakendusi on võimalik tihedamalt siduda turvalise nimeteenuse arhitektuuriga. Autentimise ja autoriseerimise andmete hoidmine ühises struktuuris aitab paremini jagada ja delegeerida klientide õigusi süsteemis. Turvalise nimeteenuse raames defineeritud turbetõendite formaat laseb neid kasutada ka teistes rakendustes. Turvalise nimeteenuse kaudu on võimalik tsentraliseerida turvamehhanismide administreerimist. Turvalise nimeteenuse avatud arhitektuur võimaldab täiendada selle struktuuri vastavalt loodavate rakenduste vajadustele.

5.1 Turvalise nimeteenuse spetsifikatsioon

Turvaline nimeteenus (*SNS – Secure Naming Service*) laiendab tavalist COS nimeteenust, lisades sellele turvalisuse eest vastutavaid mehhanisme. Turvalise nimeteenuse spetsifikatsioon koosneb teenuse IDL liidestest, liideste kirjeldustest ja abstraktsest mudeli kirjeldusest. Turvalise nimeteenuse arhitektuur käsitleb nii teenuse sisemise struktuuri turvamist kui ka teenusele juurdepääsu väljastpoolt. Antud töö eelmistes peatükkides vaatlesime CORBA arhitektuuri erinevad turvamisvõimalusi. Kahjuks pole paljud OMG poolt spetsifitseeritud turvalahendused realiseeritud (või on realiseeritud ainult CORBA kommertsrealisatsioonide raames). Realiseerimata tehnoloogiate alla kuuluvad nii CORBA turvalisuse teenus kui ka enamus CSI koostöövõime protokollidest. Realiseerimata omadustele orienteerumine seaks lisanõudeid teenuse realiseerimisele. Lahenduseks oleks realiseerida puuduv tehnoloogia lisaks loodavale turvalisele nimeteenusele või oodata täiendusi kasutatavas CORBA realisatsioonis. Puuduvate tehnoloogiate realiseerimise probleemiks on CORBA turvatehnoloogiate tihe seos CORBA tuumaga. Efektiivse realisatsiooni loomiseks tuleks valida avatud koodiga CORBA realisatsioon, milles oleks detailselt dokumenteeritud koodi sisemine ehitus. Sellist realisatsiooni vaadeldute hulgas ei leidunud. Oodata täiendusi tasub vaid siis, kui nende teostamisest on ametlikult teatatud. Turvalise nimeteenuse spetsifikatsiooni raames on arvestatud realiseerimise võimalusi praeguste CORBA realisatsioonide põhjal. Samas on arvestatud turvalise nimeteenuse võimalusi edaspidiseks edasiarenguks koos CORBA turvalisuse teenuse ja teiste CSI protokollide kasutusele võtmisega. Sobivaks CORBA realisatsiooniks on valitud OpenOrb 1.3. Valiku põhjusteks on avatud lähtekood, SSL laienduse olemasolu praeguses stabiilses versioonis, teiste CSI protokollide olemasolu tulevases versioonis (praeguses 1.4 beetaversioonis) ja enamiku COS teenuste realisatsioonide olemasolu. OpenOrb CORBA realisatsiooni kasutuse kirjeldus turvalise nimeteenuse teostamise raames on esitatud peatükis 5.3.

Turvaline nimeteenus laiendab COS nimeteenust turvaelementide lisamisega. COS nimeteenuses defineeritud eesmärgid on toodud turvalise nimeteenuse spetsifikatsioonis. Turvaline nimeteenus vastutab teenust pakkuvate objektide nimedega sidumise ning struktureerimise eest. Turvalise nimeteenuse struktuur põhineb turvalisel nimekontekstil. Erinevalt COS nimeteenuse nimekontekstist on turvalises nimekontekstis võimalik määrata vanemkontekst. Turvalisuse operatsioonide läbiviimiseks on defineeritud turbekonteksti objekt.

Iga turvaline nimekontekst on seotud turbekontekstiga. Kliendi autentimine süsteemis toimub parooli või kliendi avaliku võtme signatuuri põhjal. Eduka autentimise tulemusena väljastatakse kliendile turbetõend. Tõend sisaldab kliendi autentsuse kinnitust ja privileegide komplekti. Tõend on mõeldud turvalise nimeteenuse operatsioonide välja kutsumiseks ning teistele teenustele edastamiseks. Kliendile väljastatud tõend on signeeritud väljaandja konteksti poolt. Kliendi autoriseerimine toimub tõendi esitamise põhjal. Kliendi autentimise ja autoriseerimise andmed paiknevad nimekontekstidega seotud andmekogumites. Nimekontekstides kehtivad õigused ning operatsioonide teostamiseks vajalikud õigused paiknevad samuti seotud andmekogumites. Nimekontekstid tohivad erinevaid andmekogumeid omavahel jagada. Nimekontekstid tohivad pöörduda ülemkonteksti poole kliendi autentimise läbi viimiseks. Turvalise nimeteenuse süsteemis on defineeritud juurkasutaja (*root*), kellele on lubatud kõigi operatsioonide läbiviimine. Juurkasutaja olemasolu võimaldab paremini käsitleda turvalise nimeteenuse administreerimist. Juurkasutaja määratakse juurkasutaja privileegi väljastamise kaudu. Juurkasutaja õiguste piiramiseks võivad privileegide komplekti olla lisatud negatiivsed privileegid – nende kaudu võetakse kasutajalt ära vastavad õigused. Turvalises nimeteenuses on defineeritud rida nimekonteksti struktuuri käsitlevaid abioperatsioone, mis ei ole lubatud enamikule kasutajatele. Need operatsioonid on mõeldud teiste nimekontekstide poolt kasutamiseks. Nimekontekstile väljastatavasse tõendisse lisatakse konteksti privileeg, mis annab õiguse spetsiifiliste abioperatsioonide teostamiseks. Turvalise nimeteenuse objektides on defineeritud rida avalikke operatsioone. Avalikeks on operatsioonid, mille käivitamiseks klient autoriseerimist ei vaja. Sellise operatsiooni näitena võib tuua avaliku informatsiooni tagastavaid operatsioone või sisselogimise operatsiooni, mille käigus klienti autenditakse parooli põhjal ning väljastatakse talle turbetõend autoriseerimise atribuutidega (privileegide komplekti kujul). Operatsioonide sooritamiseks (ka avalike operatsioonide puhul) vajab klient ligipääsu turvalisele nimeteenusele. Lisaks turvalise nimeteenuse sisemiste autentimisele ja autoriseerimisele piiratakse ligipääsu suhtluskihi tasemel. Turvalise suhtluse tagamiseks eeldab SNS nimeteenuse ehitus SSLIOP protokolli kasutamist. Turvalise nimeteenuse ehituse ja suhtluse detailsem kirjeldus tuleb peale IDL liideseid kirjeldavaid punkte (5.2.1 – 5.2.5).

Turvalise nimeteenuse raames kasutatavad turvalahendused sarnanevad mitme teise turvatehnoloogiaga. Turvalises nimeteenuses teenustest ja nimekontekstidest loodav

andmestruktuur koos privileegidepõhise juurdepääsukontrolliga sarnaneb LDAP (*Lightweight Directory Access Protocol*) kataloogiteenuse ehitusega. Turbetõendite jagamine klientidele sarnaneb Kerberose tehnoloogias piletite jagamisega. Turvalise nimeteenuse raames spetsifitseeritud turbetõendid sarnanevad ehituse poolest atribuutsertifikaatidega. Nende tehnoloogiate kooskasutust turvalise nimeteenusega praeguses spetsifikatsiooni versioonis kirjeldatud pole. COS nimeteenusele turvaelementide lisamine LDAP kataloogiteenusega ühildamise kaudu nõuaks struktuuride dubleerimist. Privileegide määramiseks tuleks nimekontekstidest koostatud struktuuri dubleerida LDAP kataloogide struktuuriga. Kahe struktuuri sünkroniseerimise vajadus muudab antud lähenemise ebaefektiivseks. Kerberose kasutus turvalise nimeteenuse arhitektuuris seaks rea piiranguid turbetõendi ehitusele. Turvalise nimeteenuse edasiarendustes on võimalik ühilduvus nii LDAP kataloogiteenuse kui ka Kerberose tehnoloogiaga. LDAP kataloogiteenuse kooskasutust turvalise nimeteenusega on võimalik teostada, spetsifitseerides vahendajateenuse, mis vastutaks nende kahe teenuse suhtluse eest. Turbetõendite ja Kerberose piletite teisendamise abiteenus teeks võimalikuks turvalise nimeteenuse ja Kerberose kooskasutuse.

5.2 Turvalise nimeteenuse liidesed

CORBA teenusena baseerub turvalise nimeteenuse spetsifikatsioon IDL liidestel. Turvalise nimeteenuse andmetüübid ja liidesed on kirjeldatud 'SecNaming' IDL moodulis. Antud moodul kasutab COS nimeteenuse IDL moodulit (*CosNaming*) ja ajateenuse IDL baasmoodulit (*TimeBase*). SNS nimeteenuse poolt laiendatav nimekontekst võetakse COS nimeteenuse IDL moodulist. Lisaks kasutatakse COS nimeteenuse IDL moodulis kirjeldatud andmetüüpe. Ajateenuse IDL baasmoodulist kasutatakse ajatüüpi 'UtcT' (*Universal timezone coordinated Time*) aja määramise standardiseerimiseks. SNS nimeteenuse spetsifikatsioonis on defineeritud neli andmekogumi liidest ning kolm põhiliidest: turvalise nimekonteksti liides (*NamingContextSec*), turbekonteksti liides (*SecurityContext*) ja administreerimise abiliides (*AdministrationHelper*). IDL liideste fail 'SecNaming.idl' asub magistritööle lisatud CD kettal.

5.2.1 Turvalise nimeteenuse andmestruktuurid

Lisaks COS nimeteenuse IDL moodulis kirjeldatud andmetüüpidele on SNS nimeteenuse raames kirjeldatud komplekt turvamehhanismides kasutatavaid andmetüüpe. Andmetüübid on defineeritud CORBA väärtusetüüpide (*valuetype*) kujul. Väärtusetüübi arhitektuur võimaldab kirjeldada CORBA arhitektuuris edastatavaid lihtobjekte. Erinevalt liidestega kirjeldatavatest

teenustpakkuvatest objektidest edastatakse kliendile objekti koopia, mitte aga viide objektile. Erinevalt lihttüüpidest võimaldab väärtusetüübi ehitus kirjeldada objekti andmeid käsitlevaid operatsioone. Turvalises nimeteenuses kirjeldatud väärtuseobjektid koosnevad privaatsetest atribuutidest. Juurdepääs andmetele toimub vastavate operatsioonide kaudu. Selline ehitus võimaldab vältida andmete muutmist valmisehitatud objektides. Lisaks väärtusetüüpidele on kirjeldatud komplekt lihttüüpe. Defineeritud on järgmised lihttüübid: baidimassiivi tüüpidena on defineeritud serialiseeritud objekti tüüp (*BinaryObject*), signatuuri tüüp (*Signature*), võtme tüüp (*Key*), parooli räsi tüüp (*PassHash*) ja võtmekogumi objekti tüüp (*KeyStoreStoredObject*), objekti unikaalse identifikaatori tüüp tähistab IDL stringi, objekti identifikaatori tüüp tähistab IDL täisarvuliste arvude tüüpi (*long*) ja konteksti aja tüüp tähistab ajateenuse ajatüüpi. Baidimassiivina defineeritud tüübid on mõeldud erinevate objektide kahendandmetena edastamiseks. IDL liidese struktuuri loetavuse eesmärgil on samasuguste tüüpidele määratud erinevad sünonüümid. Väärtusetüüpidena on defineeritud kolm andmetüüpi: privileegi andmetüüp, turbetõendi andmetüüp ja kliendi registratsiooni objekti andmetüüp. Defineeritud väärtusetüüpide ehitus arvestab OpenOrb realisatsiooni omadusi. Andmete hoidmiseks kõigis väärtusetüüpides on defineeritud baidimasiivi atribuut. Baidimasiivi kujul hoitakse väärtusetüübi sisemist struktureeritud objekti. Sisemise objekti atribuute käsitletakse abioperatsioonie kaudu. Väärtusetüübi sisemise objekti iga atribuudi kohta on defineeritud väärtuse tagastamise operatsioon.

Privileegi (*PrivilegeObject*) andmetüüp kirjeldab klientidele jagatavaid õigusi. Privileegi andmetüübi sisemine objekt koosneb järgmistest atribuutidest:

- privileegi identifitseeriv number (*privilegeId*) – privileegi unikaalne identifikaator; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getPrivilegeId*;
- privileegi nimi (*privilegeName*) – määrab privileegi olemust, seost operatsiooniga, mille sooritamist antud privileeg õigustab; privileegi nimi ei ole alati unikaalne – ühe nime kohta võivad olla määratud erinevad kehtivusajad, erinevad omanikud ja erinevad negatiivsuse ja delegeeritavuse atribuudid; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getPrivilegeName*;
- ülemprivileegi identifitseeriv number (*parentPrivilegeId*) – privileeg võib omada ülemprivileegi, alamprivileegi poolt laiendatakse ülemprivileegi poolt antavaid õigusi

(või vastupidi kitsendatakse); atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getParentPrivilegeId*;

- privileegi omaniku identifikaator (*ownerUid*) – erijuhtude käsitlemiseks on privileegidele võimalik määrata omanikku (omanikuks on klient); atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getOwnerUid*;
- privileegi omava objekti identifikaator (*ownerObjectUid*) – konkreetse objekti privileegi määramiseks vajalik atribuut; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getOwnerObjectUid*;
- privileegi negatiivsuse atribuut (*negPrivilege*) – antud atribuuti kasutatakse kahel juhul: kliendi privileegide komplektis õiguste keelamise atribuudina ja nimekontekstis privileegi valideerimise keelamise atribuudina; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getNegPrivilege*;
- privileegi delegeerimise atribuut (*delegateable*) – määrab kliendile antud privileegi delegeerimise õiguse; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getDelegateable*;
- privileegi kehtivuse perioodi atribuudid (*startDate, endDate*) – kliendile väljastatud privileegi kehtivuse periood; atribuudi väärtuse tagastamiseks on defineeritud operatsioonid *getStartDate* ja *getEndDate*.

Parameetri signatuuri objekti (*ParameterSignature*) andmetüüp kirjeldab kliendi poolt edastatava parameetri omadusi koos kliendikinnitusega digitaalse signatuuri kujul (parameetri väärtust edastatakse signatuuri objektist eraldi). Signatuuri koostamiseks ning valideerimiseks on defineeritud abiandmetüüp (*ParameterSignatureBase*). Parameetri signatuuri objekt koosneb järgmistest atribuutidest:

- parameetri nimi (*parameterName*); atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getParameterName*;
- parameetri sihtmärk (*parameterTarget*) – objekti identifikaator, millele antud parameeter oli suunatud (turvalise nimeteenuse raames antud identifikaatori kuju rangelt määratud pole); atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getParameterTarget*;
- parameetri signatuur (*signature*); atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getSignature*.

Parameetri signatuuri abiandmetüüp koosneb järgmistest atribuutidest:

- parameetri nimi (*parameterName*); atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getParameterName*;
- parameetri sihtmärk (*parameterTarget*) - objekti identifikaator – millele antud parameeter oli suunatud (turvalise nimeteenuse raames antud identifikaatori kuju rangelt määratud pole; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getParameterTarget*;
- parameetri väärtus (*parameterValue*) – signatuuri koostamiseks vajalik parameetri väärtus; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getParameterValue*.

Turbetõendi (*SecurityToken*) andmetüüp kirjeldab kliendile väljastatava objekti koos lubatud privileegide komplektiga ja väljastaja nimekonteksti kinnitust digitaalse signatuuri kujul. Turbetõendi andmetüübi sisemine objekt (*SecurityTokenObject*) koosneb järgmistest atribuutidest:

- kliendi identifikaator (*clientUid*) – tõendi omaniku unikaalne identifikaator, mis määrab, kellele antud tõend on väljastatud; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getClientUid*;
- tõendi väljastaja identifikaator (*issuerUid*) – tõendi väljastajaunikaalne identifikaator; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getIssuerUid*;
- kliendi avalik võti (*clientPublicKey*) – tõendit omava kliendi avalik võti. Antud võti on kinnitatud tõendi väljastaja poolt; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getClientPublicKey*;
- väljastaja avalik võti (*issuerPublicKey*) – tõendi väljastaja avalik võti on vajalik tõendi mingile muule teenusele edastamisel; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getIssuerPublicKey*;
- privileegide komplekt (*privilegeCollection*) – kliendile lubatud õigusi kirjeldav privileegide komplekt; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getPrivilegeCollection*;
- tõendi kehtivuse perioodi atribuudid (*startDate, endDate*) – määravad ajavahemiku, millal antud tõend on kehtiv (kui perioodi määratud pole, siis on tõendi kehtivusaeg

piiramatu); atribuudi väärtuse tagastamiseks on defineeritud operatsioonid *getStartDate* ja *getEndDate*;

- kontekstiks olemise atribuut (*clientIsContext*) – tähistab nimekontekstidele väljastatud tõendeid; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getClientIsContext*.

Turbetõendi tüüpi objekt koosneb järgmistest atribuutidest:

- turbetõendi sisemine objekt (*securityTokenObject*) – turvalise nimeteenuse poolt väljastatud tõendi sisu;
- digitaalne signatuur (*signature*) - turvalise nimeteenuse poolt väljastatud tõendi signatuur;
- lisaparameetrite signatuuride kogum (*parameterSignatureCollection*) – kliendi poolt lisatud parameetrite signatuuride kogum; võimaldab tõendi esitamisel edastada teenust pakkuvale objektile suvalise parameetri signatuuri; kogumi seadmiseks on defineeritud operatsioon *setParameterSignatureCollection* ja tagastamiseks on defineeritud operatsioon *getParameterSignatureCollection*.

Lisaks on defineeritud ka üks abioperatsioon:

- tõendi enesevalideerimise operatsioon (*selfValidating*) – kontrollib tõendis olevat signatuuri tõendiga kaasas oleva väljastaja avaliku võtme suhtes.

Kliendi registratsiooni objekti (*ClientRegObject*) andmetüüp kirjeldab turvalises nimeteenuses hoitavaid kliendi andmeid. Kliendi registratsiooni andmetüübi sisemine objekt koosneb järgmistest atribuutidest:

- kliendi identifikaator (*clientUid*) – turvalises nimeteenuses klienti tähistav identifikaator. Antud identifikaator peab olema unikaalne ühe registreeritud klientide andmekogumi raames; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getClientUid*;
- parooli räsi (*passwordHash*) – kliendi identifikaatori ja parooli alusel genereeritud räsi, mida kasutatakse kliendi paroolipõhisel autentimisel; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getPasswordHash*;
- privileegide komplekt (*privilegeCollection*) – kliendiga seotud privileegide komplekt. Antud komplekt lisatakse kliendile väljastatavasse turbetõendisse; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getPrivilegeCollection*;

- kliendi avalik võti (*clientPublicKey*) – turvalise nimeteenuse poolt usaldatud kliendi avalik võti; atribuudi väärtuse tagastamiseks on defineeritud operatsioon *getClientPublicKey*;
- kliendi registratsiooni kehtivuse perioodi atribuudid (*startDate, endDate*) – määravad, millises ajavahemikus on antud kliendi andmed kehtivad; atribuudi väärtuse tagastamiseks on defineeritud operatsioonid *getStartDate* ja *getEndDate*.

5.2.2 Turvalise nimeteenuse andmekogud

Turvalise nimeteenuse funktsionaalsuse alla kuulub klientide autentimise ja autoriseerimise andmete haldamine. SNS nimeteenuse andmekäsitluse teostamiseks on spetsifitseeritud komplekt andmekogumite struktuure. SNS IDL moodulis on defineeritud kolme liiki andmekogumeid: registreeritud klientide andmekogum (*ClientRegStore*), privileegide andmekogum (*PrivilegeObjectStore*) ja turbetõendite andmekogum (*SecurityTokenStore*). Kõiki kogumeid kasutatakse turbekonteksti andmete haldamiseks. Andmekogumid võivad olla seotud ühe kontekstiga või jagatud mitme konteksti vahel.

Andmekogumite liideste üldiste operatsioonide defineerimiseks on kirjeldatud baaskogumi liides (*BaseStore*). Baaskogumis kirjeldatud operatsioonid käsitlevad jagatud kogumite kasutust. Defineeritud on järgmised operatsioonid:

- jagatavuse suurendamise operatsioon (*increaseSharedStore*) – suurendab antud kogumit kasutatavate kontekstide kogust tähistavat arvu.
- jagatavuse vähendamise operatsioon (*decreaseSharedStore*) – vähendab antud kogumit kasutatavate kontekstide kogust tähistavat arvu.

Nende kahe operatsiooniga reguleeritava arvu põhjal otsustatakse, kas säilitada jagatud andmekogumit peale seotud nimekonteksti likvideerimist. Kui andmekogum on seotud vaid kustutatava nimekontekstiga, siis seda kogumit ei säilitata.

Klientide andmekogumis hoitakse kõigi registreeritud klientide andmeid. Klientide andmekogum koosneb klientide registratsiooni objektide komplektist ning operatsioonidest klientide haldamiseks. Klientide registratsiooni objektide komplekt kogumis on kirjutuskaitsega atribuudina, andmekäsitlus toimub andmekogumi operatsioonide kaudu. Klientide andmekogumis on defineeritud järgmised operatsioonid:

- kliendi lisamise operatsioon (*addClientRegObject*) – lisab antud kogumisse uue kliendi;
- kliendi eemaldamise operatsioon (*removeClientRegObject*) – kustutab antud kogumist etteantud identifikaatoriga kliendi;
- kogumi likvideerimise operatsioon (*destroy_s*) – kustutab antud kogumi seose ORB vahendajaga;
- identifikaatori järgi kliendi leidmise operatsioon (*getClientRegObject*) – leiab kliendi etteantud unikaalse identifikaatori põhjal.

Privileegide andmekogumit kasutatakse kahel eesmärgil: kehtivate privileegide komplekti hoidmiseks ja kontrolli vajavate privileegide komplekti hoidmiseks. Lisaks privileegide komplektile sisaldab privileegide andmekogum operatsioone privileegide haldamiseks. Privileegide komplekt on kogumis kirjutuskaitsega atribuudina, andmekäsitlus toimub andmekogumi operatsioonide kaudu. Privileegide andmekogumis on defineeritud järgmised operatsioonid:

- privileegi lisamise operatsioon (*addPrivilegeObject*) – lisab antud kogumisse uue privileegi;
- privileegi eemaldamise operatsioon (*removePrivilegeObject*) – kustutab antud kogumist etteantud identifikaatoriga privileegi;
- kogumi likvideerimise operatsioon (*destroy_s*) – kustutab antud kogumi seose ORB vahendajaga;
- identifikaatori järgi privileegi leidmise operatsioon (*getPrivilegeObjectById*) – tagastab etteantud unikaalsele identifikaatorile vastava privileegi;
- nime järgi privileegide leidmise operatsioon (*getPrivilegeObjectByName*) – tagastab etteantud nimele vastavate privileegide komplekti;
- omaniku järgi privileegide leidmise operatsioon (*getPrivilegesByOwner*) – tagastab etteantud omaniku identifikaatorile vastavate privileegide komplekti;
- kõigi privileegide tagastamise operatsioon (*getAllPrivileges*) – tagastab komplekti kõigi kogumis olevate privileegidega.

Turbetõendite andmekogumit kasutatakse valmisgenereeritud tõendite hoidmiseks. Sellise kogumi abil kiirendatakse süsteemi taassisenevatele klientidele tõendite väljastamist. Turbetõendite andmekogum koosneb tõendite komplektist ning operatsioonidest tõendite

haldamiseks. Turbetõendite komplekt on kogumis kirjutuskaitsega atribuudina, andmekäsitlus toimub andmekogumi operatsioonide kaudu. Turbetõendite andmekogumis on defineeritud järgmised operatsioonid:

- turbetõendi lisamise operatsioon (*addSecurityToken*) – lisab antud kogumisse uue turbetõendi;
- turbetõendi eemaldamise operatsioon (*removeSecurityTokensByClient*) – kustutab antud kogumist kõik tõendid etteantud kliendi identifikaatoriga;
- kogumi likvideerimise operatsioon (*destroy_s*) – kustutab antud kogumi seose ORB vahendajaga;
- kliendi identifikaatori järgi tõendi leidmise operatsioon (*getSecurityTokenCollectionByClient*) – tagastab etteantud omaniku identifikaatorile vastavate tõendite komplekti;
- väljastaja identifikaatori järgi tõendi leidmise operatsioon (*getSecurityTokenCollectionByIssuer*) – tagastab etteantud väljastaja identifikaatorile vastavate tõendite komplekti.

5.2.3 Turvaline nimekontekst

Turvaline nimekontekst laiendab COS nimeteenuse nimekonteksti, lisades sellele turvaelementide kasutamise võimalusi. COS nimeteenuses defineeritud operatsioonidele on lisatud autoriseerimine turbetõendi abil. Lisaks laiendatud operatsioonidele on defineeritud komplekt järgmisi abioperatsioone: Uue nimekonteksti sidumise operatsioon koos jagatud andmekogumite edastamisega (*bind_new_context_wshare_s*) – antud operatsiooni käigus luuakse uus turvaline nimekontekst, seotakse ta jooksva konteksti alamaks ning loomisel seadistatakse jagatud andmekogumid. Antud operatsioon on sarnane COS nimeteenuse ‘bind_new_context’ operatsiooniga, erinevus seisneb andmekogumite edastamises ning tegevuse autoriseerimises.

- Nimekonteksti seose uuendamise operatsioon koos jagatud andmekogumite edastamisega (*rebind_context_wshare_s*) – antud operatsiooni käigus uuendatakse nimekonteksti seotust nimega, uuendatavale kontekstile edastatakse uus komplekt jagatud andmekogumeid. Antud operatsioon on sarnane COS nimeteenuse

‘rebind_context’ operatsiooniga, erinevus seisneb andmekogumite edastamises ning tegevuse autoriseerimises.

- Nimekonteksti sidumise operatsioon koos jagatud andmekogumite edastamisega (*bind_context_wshare_s*) – antud operatsiooni käigus luuakse nimeseos varem loodud nimekontekstiga, seotavale kontekstile edastatakse komplekt jagatud andmekogumeid. Antud operatsioon on sarnane COS nimeteenuse ‘bind_context’ operatsiooniga, erinevus seisneb andmekogumite edastamises ning tegevuse autoriseerimises.
- Nimekonteksti ülema lahendamise operatsioon (*resolve_parent_context_s*) – tagastab viite jooksva nimekonteksti ülemale. Antud operatsioon on mõeldud kliendi autentimise ja autoriseerimise käigus ülemkonteksti poole pöördumiseks. Ülemkontekstilt päritakse andmeid nende klientide kohta, kelle andmed puuduvad jooksva nimekontekstiga seotud andmekogumis.
- Nimekonteksti ülema seadmise operatsioon (*set_parent_context_s*) – seadistab ülema jooksvale nimekontekstile. Analoogselt COS nimeteenusele võib nimekontekst olla nimeseose kaudu seotud mitme ülemkontekstiga. Antud operatsioonis üheselt seadistatav ülem on vajalik autentimise ja autoriseerimise käigus kliendiinformatsiooni pärimiseks. Nimekontekst ei pea omama seost ülemaga, sellise seose puudumisel peab antud nimekontekst piirduma oma andmekogumites oleva informatsiooniga.
- Nimekonteksti sisese nime seadmise operatsioon (*set_current_name_s*) – seadistab nimekonteksti sisemise nime. Antud nimi on nimeteenuse raames unikaalne. Nimi koostatakse ülemkonteksti nimest, millele lisatakse jooksva konteksti nimi. Antud operatsiooni sooritus nõuab konteksti privileegi.
- Nimekontekstile turbetõendi seadmise operatsioon (*giveSecurityToken*) – võimaldab saada ülemkontekstilt turbetõendit nimekontekstile. Tõendit läheb vaja ülemkonteksti operatsioonide välja kutsumiseks. Antud operatsioon nõuab konteksti privileegi.
- Nimekontekstis sisemise nime tagastamise operatsioon (*get_local_name*) - tagastab antud nimekontekstiga lokaalselt seotud täisnime.

Kõik ülejäänud turvalise nimekonteksti operatsioonid on COS nimekonteksti operatsioonide laiendused. Enamus turvalise nimeteenuse operatsioone võtavad sisendi lisaparameetrina turbetõendi ning toetavad kahte lisaerindit (autoriseerimata päringu erind ja privileegi puudumise erind). Lisaks on olemas ka avaliku juurdepääsuga operatsioonid. IDL liideste keele

standardid ei luba defineerida ühe liidese raames samanimelisi operatsioone. Operatsiooni signatuur ei võta arvesse parameetrite komplekti erinevusi. Selle kitsenduse tõttu on kõigile laiendatud operatsioonide nimedele lisatud suffiks '_s'. Alljärgnevas tabelis 1 on toodud COS nimeteenuse ja turvalise nimeteenuse operatsioonide vastavused koos lühikirjeldustega.

COS nimekonteksti operatsioon	**Turvalise nimekonteksti operatsioon**	**kirjeldus**
bind	bind_s	Teenust pakkuva objekti nimega seose loomise operatsioon
rebind	rebind_s	Teenust pakkuva objekti nimega seose uuendamise operatsioon
bind_context	bind_context_s	Nimekonteksti nimega seose loomise operatsioon
rebind_context	rebind_context_s	Nimekonteksti nimega seose uuendamise operatsioon
resolve	resolve_s	Nimelahendamise operatsioon – tulemusena tagastatakse nimega seotud nimekontekst või teenust pakkuv objekt
unbind	unbind_s	Nimeseose kustutamise operatsioon – likvideeritakse nimekonteksti või teenust pakkuva objekti seos etteantud nimega
new_context	new_context_s	Uue nimekonteksti loomise operatsioon
bind_new_context	bind_new_context_s	Uue nimekonteksti koos nimeseosega loomise operatsioon
destroy	destroy_s	Nimekonteksti likvideerimise operatsioon
list	list_s	Antud nimekontekstis olevate nimeseoste järjendi tagastamise operatsioon
to_url	to_url_s	Etteantud nime 'corbaname' URL kujul tagastav operatsioon
resolve_str	resolve_str_s	Stringikujulise nime lahendamise operatsioon – tulemusena tagastatakse nimega seotud nimekontekst või teenust pakkuv objekt
to_string	to_string	Nimeobjekti stringiks teisendamise operatsioon. Turvalises nimeteenuses on antud operatsioon jäetud avalikuks
to_name	to_name	Stringikujulise nime nimeobjektiks teisendamise operatsioon. Turvalises nimeteenuses on antud operatsioon jäetud avalikuks

Tabel 1. COS nimekonteksti ja turvalise nimekonteksti sarnased operatsioonid

5.2.4 Turbekontekst

Turbekontekst luuakse koos turvalise nimekontekstiga. Antud kontekst hoiab endas turbespetsiifilist informatsiooni ning vastutab SNS nimeteenuse turvalisuse funktsionaalsuse

eest. Turbekonteksti liideses on defineeritud turvalise nimeteenuse privileegide identifikaatorite ja nimede vastavus. Turbekontekstis hoitakse turvalise nimekonteksti avalikku ja salajast võtit. Kirjeldatud privileegide identifikaatorid vastavad privileegide üldkujule – lisaatribuute liidese tasemel ei määrata. Turbekonteksti raames on defineeritud järgmised operatsioonid:

- Nimeteenusest jooksva aja tagastamise operatsioon (*getContextCurrentTime*) – turbekonteksti jooksvat aega tagastav avalik operatsioon. Turbekonteksti jooksvat aega kasutatakse erinevate objektide (tõendid, privileegid, kliendi registratsiooni objektid) kehtivusperioodide kontrollimisel.
- Kliendi sisselogimise operatsioon (*clientLogin*) – autendib klienti sisestatud parooli abil. Operatsiooni tulemusena väljastatakse kliendile turbetõend.
- Kliendi signatuuripõhise sisselogimise operatsioon (*clientLoginBySignedKey*) – autendib klienti avaliku võtme signatuuri põhjal. Signatuur peab olema genereeritud salajase võtmega võtmepaarist, mille avalik võti on nimeteenuses kinnitatud. Kehtivust kontrollitakse kliendi registratsiooniobjektis oleva avaliku võtme abil.
- Andmekogumite sisselugemise operatsioon (*reloadStores*) – taastab andmekogumite salvestatud seisu. Operatsioon on mõeldud süsteemiadministraatori poolt kasutamiseks.
- Jooksvate andmekogumite mahasalvestuse operatsioon (*saveStores*) – salvestab maha andmekogumite jooksva seisu. Operatsioon on mõeldud süsteemiadministraatori poolt kasutamiseks.
- Objekti signeerimise operatsioon (*signObject*) – signeerib etteantud objekti antud turbekontekstis oleva salajase võtmega. Signeerimise operatsioon on ennekõike mõeldud väljastatavate tõendite signeerimiseks. Tõendi signeerimiseks peab operatsiooni käivitaja omama konteksti privileegi. Omanikuga määratud privileegide kasutamisel võib signeerimise õigus olla antud nii konteksti loonud kliendile kui ka teistele usaldatud klientidele.
- Turbetõendi seadistamise operatsioon (*setSecurityToken*) – ülemkonteksti poolt väljastatud turbetõendi turbekontekstisse seadmine.
- Turbetõendi valideerimise operatsioon (*validateSecurityToken*) – turbetõendi kehtivust kontrolliv avalik operatsioon. Tõendi valideerimise operatsioon ei nõua autoriseerimist. Iga klient tohib kontrollida talle väljastatud tõendeid.

- Privileegide komplekti kontrollimise operatsioon (*isPrivilegedToDo*) – etteantud tõendis privileegide komplekti olemasolu kontroll. Lisaks privileegide kontrollile valideeritakse ka tõendi kehtivust. Antud operatsioon on samuti avalik.
- Üksiku privileegi kontrollimise operatsioon (*hasPrivilegeToDo*) – erinavalt eelmisest kontrollib üksiku privileegi olemasolu turbetõendis. Lisaks privileegi kontrollile valideeritakse ka tõendi kehtivust. Antud operatsioon on samuti avalik.
- Klientide registratsiooni objektide jagatud andmekogumi seadmise operatsioon (*setSharedClientRegStore*) – võimaldab edastada antud turbekontekstile klientide registratsiooni kogumi. Antud operatsioon on ennekõike mõeldud teise konteksti poolt jagatud andmekogumi seadistamiseks. Süsteemiadministraator saab kasutada antud meetodit uue andmekogumi seadistamiseks.
- Kehtivate privileegide jagatud andmekogumi seadmise operatsioon (*setSharedExistingPrivilegeObjectStore*) – võimaldab edastada antud turbekontekstile kehtivate privileegide kogumi. Antud operatsioon on ennekõike mõeldud teise konteksti poolt jagatud andmekogumi seadistamiseks. Süsteemiadministraator saab kasutada antud meetodit uue andmekogumi seadistamiseks.
- Kontrollimist vajavate privileegide jagatud andmekogumi seadmise operatsioon (*setSharedRequiredPrivilegeObjectStore*) – võimaldab edastada antud turbekontekstile kontrollimist vajavate privileegide kogumi. Antud operatsioon on ennekõike mõeldud teise konteksti poolt jagatud andmekogumi seadistamiseks. Süsteemiadministraator saab kasutada antud meetodit uue andmekogumi seadistamiseks.
- Turbetõendite jagatud andmekogumi seadmise operatsioon (*setSharedSecurityTokenStore*) – võimaldab edastada antud turbekontekstile valmisgenereeritud turbetõendite kogumi. Antud operatsioon on ennekõike mõeldud teise konteksti poolt jagatud andmekogumi seadistamiseks. Süsteemiadministraator saab kasutada antud meetodit uue andmekogumi seadistamiseks.
- Turbekonteksti likvideerimise operatsioon (*destroy_s*) – kustutab antud objekti seose ORB vahendajaga. Turbekonteksti tohib likvideerida vaid koos sellega seotud nimekontekstiga. Antud operatsiooni läbi viimiseks on vajalik konteksti privileeg või juurkasutaja privileeg.

- Turbekontekstiga kaasnevate parameetrite signatuuride valideerimise operatsioon (*validate_parameter*) – koostab parameetri signatuuri abiobjekti ning kontrollib selle signatuuri kliendi avaliku võtme suhtes.

5.2.5 Administreerimise abiliides

Administreerimise abiliideses kirjeldatud operatsioonid on mõeldud turvalise nimeteenuse konfiguratsiooni haldamiseks ning SSL kihil kasutatavate võtmete kogumi ja usaldatud sertifikaatide kogumi haldamiseks. Suures osas on konfiguratsiooni parameetrid mõeldud kasutamiseks juurnimekontekstis ja selle kaudu nimeteenuse töö seadmiseks. SSL laienduse kasutamise jaoks vajalikud võtmekogumid on samuti mõeldud juurkonteksti tasemel kasutamiseks. Administreerimise abiliidese operatsioonid nõuavad kliendilt süsteemiadministraatori õigusi. Defineeritud on järgmised operatsioonid:

- Uue konfiguratsiooni sisse lugemise ja seadistamise operatsioon (*setAndReloadConfiguration*) – loeb konfiguratsiooni parameetrid etteantud failist ja seadistab nimeteenuse konfiguratsiooni.
- Olemasoleva konfiguratsiooni uuendamise operatsioon (*reloadConfiguration*) – loeb endised konfiguratsiooni parameetrid failist ja seadistab nimeteenuse konfiguratsiooni.
- SSL võtmekogumi tagastamise operatsioon (*getKeyStoreStoredObject*) – tagastab SSL ühendustes kasutatava võtmekogumi nimeteenuse salajase ja avaliku võtme paariga.
- SSL usaldatud sertifikaatide kogumi tagastamise operatsioon (*getTrustedStoreStoredObject*) – tagastab SSL ühendustes kasutatava võtmekogumi turvalise nimeteenuse poolt usaldatud klientide sertifikaatidega.
- SSL võtmekogumi seadmise operatsioon (*setKeyStoreStoredObject*) – seadistab SSL ühendustes kasutamiseks mõeldud võtmekogumi nimeteenuse salajase ja avaliku võtme paariga.
- SSL usaldatud sertifikaatide kogumi seadmise operatsioon (*setTrustedStoreStoredObject*) – seadistab SSL ühendustes kasutamiseks mõeldud võtmekogumi turvalise nimeteenuse poolt usaldatud klientide sertifikaatidega.
- administreerimise abiobjekti likvideerimise operatsioon (*destroy_s*) – kustutab antud objekti seose ORB vahendajaga. Administreerimise abiobjekti tohib likvideerida vaid

koos sellega seotud nimekontekstiga. Antud operatsiooni kasutamiseks on vajalik konteksti privileeg või juurkasutaja privileeg.

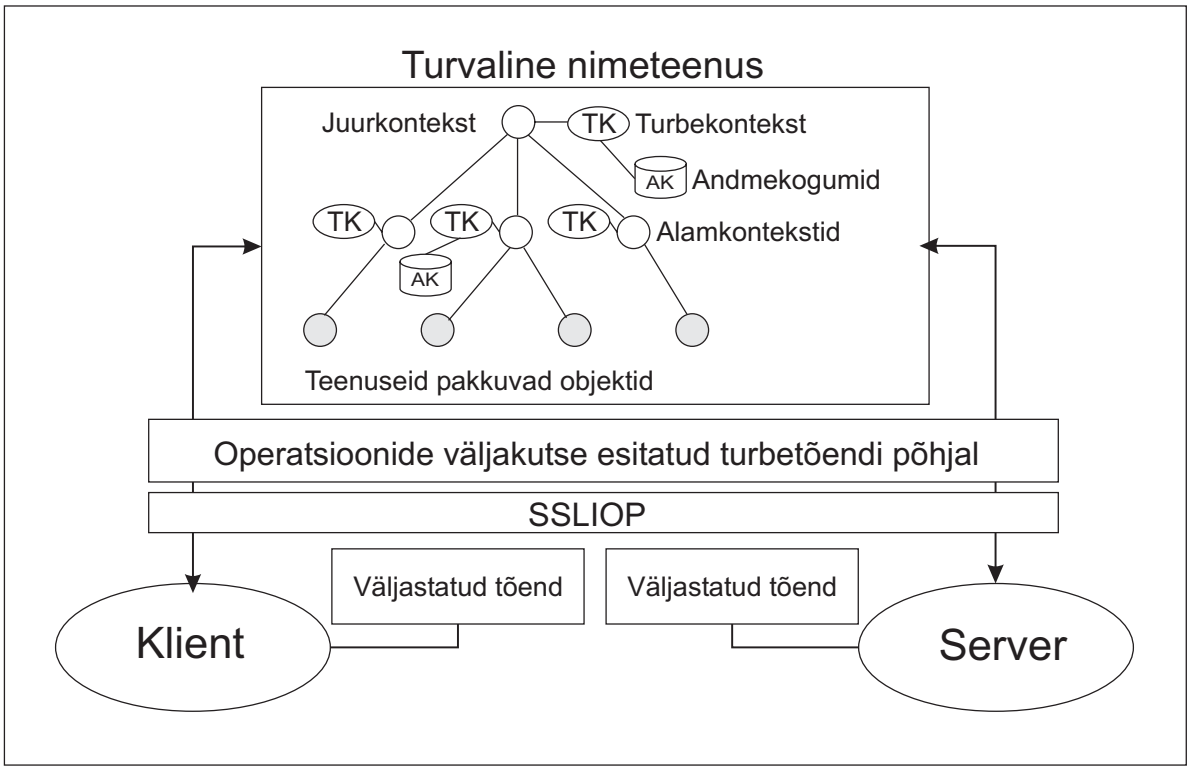
5.2.6 Turvalise nimeteenuse erindid

Turvalises nimeteenuses kasutatakse veaolukordade käsitlemiseks COS nimeteenuses kirjeldatud erindeid. Seoste loomisel ja lahendamisel tekkivate vigade puhul ning ebakorrektsete nimede edastamisel visatavad erindid on defineeritud COS nimeteenuse IDL liideste moodulis. Lisaks nendele on defineeritud turbespetsiifiliste erindite komplekt. Autoriseerimata kliendi poolt operatsioonide välja kutsumisel visatakse 'NoAuthorizedAccess' erind. Juhul kui autoriseeritud kliendil puudub operatsiooni sooritamiseks vajalik privileeg, visatakse 'NoRequiredPrivilege' erind.

5.2.7 Turvalise nimeteenuse mudel

Turvalise nimeteenuse struktuur põhineb turvalisel nimekontekstil. Iga nimekontekstiga on seotud turbekontekst. Turbekonteksti kaudu toimub nimekonteksti ja andmekogumite sidumine. Turbekontekst hoiab endas avaliku võtme ja salajase võtme paari. Salajast võtit kasutatakse väljastatavate tõendite signeerimiseks. Avalikku võtit kasutatakse autoriseerimise käigus kliendi tõendi kontrollimiseks. Lisavõimalusena on olemas abimeetod suvalise objekti signeerimiseks nimekonteksti poolt. Alamkontekstide tagasiside ülemkontekstidega on vajalik jagatud kogumite kasutamiseks ning ülemkontekstide poolt klientide autentimise võimaldamiseks. Kliendi autentimise ja autoriseerimise operatsioonid viiakse läbi turbekontekstis. Piisavate andmete puudumisel saab turbekontekst pärida vajalikke andmeid seotud nimekonteksti ülemalt. Teiste kontekstide autoriseerimine operatsioonide väljakutsete käigus toimub samuti turbekontekstis. Turbetõendite väljastamise eest vastutab turbekontekst. Eduka autentimise järel väljastatakse kliendile tõend koos registratsiooniobjektis oleva privileegide komplektiga. Alamnimekontekstide loomisel edastatakse turbetõend koos nimekontekstile spetsiifilise privileegide komplektiga. Turvalise nimeteenuse autoriseerimist vajavates operatsioonidesse on sisse ehitatud pöördumine turbekonteksti poole. Lisaks on defineeritud operatsioonid, mille kaudu teised teenused saavad kontrollida turbetõendis olevate privileegide komplekti. Turvalise nimeteenuse ehituse skeem on toodud joonisel 11. Samal joonisel on toodud ka kliendi- ja serverrakenduse nimeteenusega turvaliselt suhtlemise skeem. Joonisel 12 on toodud kliendi sisselogimise skeem (eduka sisselogimise peale väljastatakse

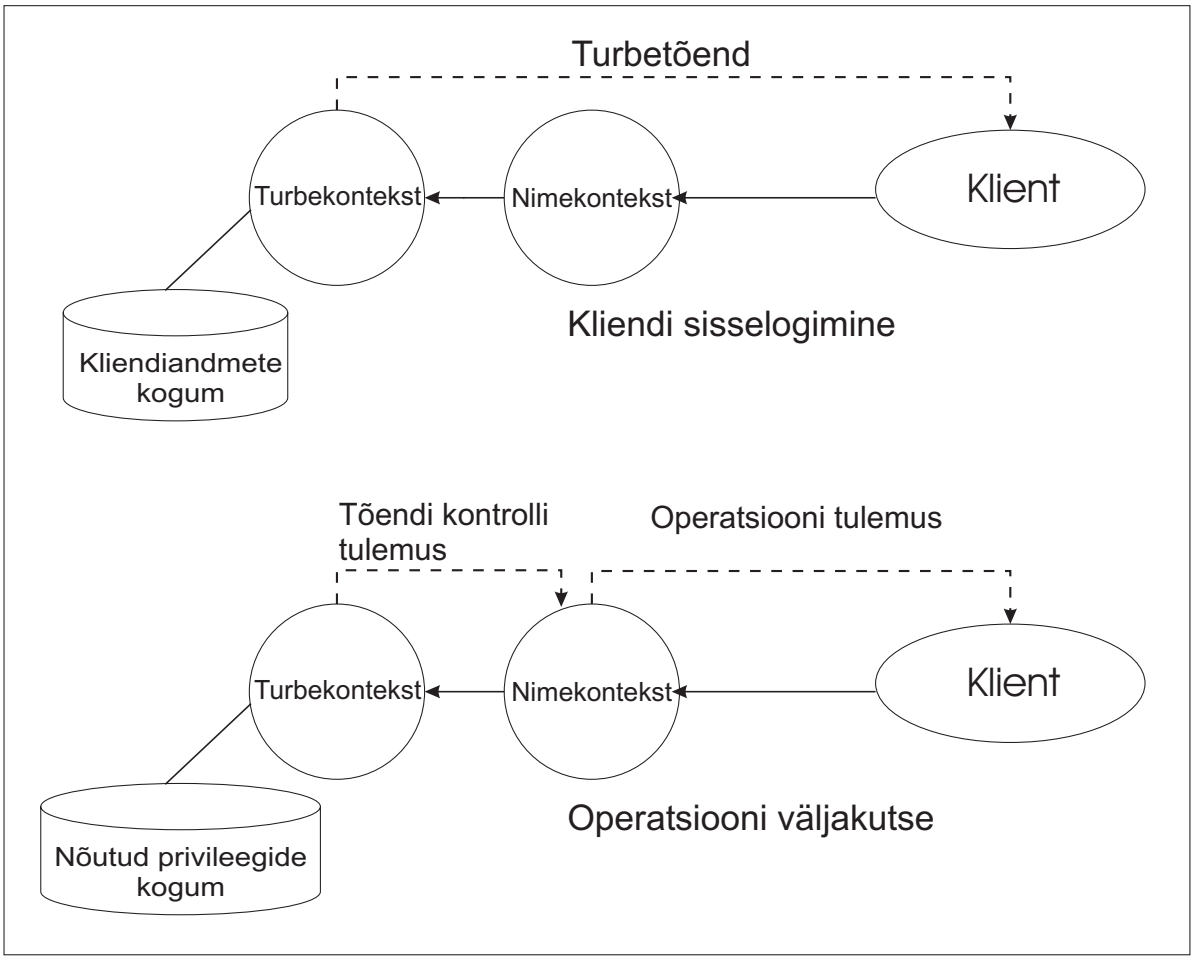
kliendile turbetõend; tõend väljastatakse kliendiandmete kogumis olevate andmete põhjal) ja nimeteenuse operatsioonide väljakutse skeem (saadud turbetõendiga kutsub klient välja nimekonteksti operatsiooni; kliendi õigusi operatsiooni väljakutseks kontrollitakse nõutud privileegide kogumi põhjal).



Joonis 11. Turvalise nimeteenuse skeem

5.2.8 CSI koostöövõime turvalises nimeteenuses

Turvalise nimeteenuse arhitektuur eeldab CSI koostöövõime mehhanismide kasutust. Turvalise nimeteenuse suhtlemine klientidega toimub SSLIOP protokolli kaudu. Antud protokoll võimaldab transpordikihi turvamist SSL abil. Nimeteenusele juurdepääsu lubamine toimub usaldatud klientide sertifikaatide kogumi (*trusted store*) alusel. Turvalise nimeteenuse kaudu teenust päriv klient saab viite teenust pakkuvale objektile. Viite väljastamise hetkest teenust pakkuvale objektile toimub suhtlus kahe kliendi vahel otse saadud viite kaudu. Kahe kliendi vahelise suhtluse turvamiseks on samuti võimalik kasutada SSLIOP protokolli. Autoriseeritud klientidele jagab turvaline nimeteenus usaldatud klientide sertifikaatide kogumit. Usaldades turvalist nimeteenust, saavad kliendid täiendada oma sertifikaatide kogumeid SSLIOP protokolli kasutamiseks.



Joonis 12. Turvalise nimeteenuse kliendisuhtluse skeem

5.3 Turvalise nimeteenuse realisatsioon

Spetsifitseeritud turvalise nimeteenuse praktiliseks kasutamiseks on vajalik realisatsiooni loomine. Antud töö raames loodud turvalise nimeteenuse realisatsioon on teostatud katserealisatsioonina. Katserealisatsiooni mõiste rõhutab eesmärke, mida on mõeldud antud realisatsiooniga saavutada. Esimeseks eesmärgiks on spetsifitseeritud teenuse korrektse toimimise katsetamine. Teiseks eesmärgiks on loodud kontseptsiooni eeliste ja puuduste väljaselgitamine. Kolmandaks eesmärgiks on turvalise nimeteenuse edasiarendamise võimaluste uurimine. Antud realisatsioon ei ole loodud lõpliku rakendusena, kuigi teostatud funktsionaalsus võimaldab selle kasutamist teiste rakenduste loomisel. Turvalise nimeteenuse realisatsioon on teostatud Java (Sun Java JDK 1.4.2) programmeerimiskeeles. Realisatsiooni loomiseks kasutatud CORBA realisatsioon on Exolab OpenOrb 1.3. (OpenOrb dokumentatsioon - [OpenOrb].)

5.3.1 Realisatsioonis kasutatavad tehnoloogiad

Turvalise nimeteenuse funktsionaalsuse realiseerimiseks kasutatakse mitmeid olemasolevaid turbetehnoloogiaid. Esimeseks on OpenOrb raames realiseeritud IIOP protokolli SSL laiendus. Turvalise nimeteenuse suhtlus klientidega on realiseeritud SSLIOP protokolli kasutades. Teiseks on Java krüptograafia laiendus JCE (*Java Cryptography Extension*), mis kuulub Java (Sun Java JDK1.4.2) standardpakettidesse. Java krüptograafia laienduse klasse kasutatakse avalike ja salajaste võtmepaaride genereerimiseks, võtmekogumite haldamiseks, turbetõendite signeerimiseks ja parooliräsi genereerimiseks. (Realisatsiooni loomisel kasutati materjale: [JSDK], [JSEC], [JCRY], [JPC] ja [CPU])

SSLIOP protokolli kasutamiseks konfigureeritakse ORB vahendaja parameetreid. ORB vahendajale määratakse IIOP protokolli SSL toega transpordikihi klassi kasutus (org.openorb.ssliop.SSLTransportServerInitializer). Määratakse võtmetekogum salajase võtmega ning kogum usaldatud klientide sertifikaatidega. Lisaks seadistatakse rida SSL ühenduse atribuute. Detailsemat informatsiooni SSLIOP protokolli seadistamisest saab OpenOrb SSL laienduse dokumentatsioonist. [OOSSL]

Võtmepaari genereerimiseks kasutatakse ‘java.security.KeyPairGenerator’ klassi. Teenuse konfiguratsiooni kaudu on võimalik seadistada genereeritavate võtmete pikkust ning genereerimise algoritmi. Võtmepaarid genereeritakse kõigi nimeteenuse struktuuris olevate nimekontekstide jaoks. Erandiks on juurnimekontekst, mille võtmepaar loetakse sisse võtmekogumist. Sama võtmekogumit kasutatakse SSL ühenduse loomiseks. Võtmekogumi objekti kirjeldab ‘java.security.KeyStore’ klass. Võtmekogumis on võimalik hoida nii salajasi võtmeid kui ka sertifikaate avalike võtmetega. Salajasi võtmeid hoitakse parooliga kaitstud kujul. Salajase võtme abil on võimalik signeerida suvalist serialiseeritavat objekti. Signatuuri kontrollimiseks on vajalik avalik võti samast võtmepaarist. Signatuuride loomist ja valideerimist võimaldab ‘java.security.Signature’ klass. Objekti signeerimist kasutatakse väljastatud turbetõendi allkirjastamiseks. Turvalises nimeteenuses toimub kliendi autentimine edastatud parooli põhjal. Kuna parooli hoidmine lahtise tekstina on ebaturvaline, siis parooli verifitseerimiseks nimeteenuses registratsiooni objektis hoitakse ainult parooli räsi. Räsi genereerimiseks kasutatakse ‘MessageDigest’ klassi. Teenuse konfiguratsiooni kaudu on võimalik seadistata räsi genereerimise algoritmi.

5.3.2 Realisatsiooni projekti struktuur

Projekti failide haldamiseks on kasutusel Apache Ant projektiehituse vahend. Ant skriptis on defineeritud ülesanded (*target*) IDL liideste Java koodiks kujutamiseks, Java klasside kompileerimiseks, konfiguratsioonifailide kopeerimiseks, genereeritud failide kustutamiseks ja teenuse ning turvalise nimeteenuse käivitamiseks. Kui ülesanne jäetakse ette andmata, trükitakse ekraanile ülesandeid kirjeldav lühijuhend. Tabelis 2 on toodud turvalise nimeteenuse Ant skripti ülesanded. (Ant projektiehituse vahendi kasutamise ja konfigureerimise kohta saab põhjalikumalt lugeda [ANT] dokumendis.)

Ülesande nimi	**Ülesande kirjeldus**
help	trükib ekraanile ülesannete lühijuhendi
all	käivitab järgmised ülesanded: clean, idl, compile, config
idl	genereeritakse Java klassid turvalise nimeteenuse IDL liidese põhjal
compile	kogu projekti kompileerimise ülesanne; käivitab järgmised ülesanded: compile_gen, compile_src
compile_gen	kompileerib IDL liideste põhjal genereeritud Java klassid
compile_src	kompileerib turvalise nimeteenuse lähtekoodi
clean	täispuhastuse ülesanne; käivitab järgmised ülesanded: clean_gen, clean_classes
clean_gen	kustutab IDL liideste põhjal genereeritud klassid
clean_classes	kustutab kõik kompileeritud klassid
config	konfiguratsiooni failide kopeerimine
jar	koostab turvalise nimeteenuse teegifaili - sns.jar
javadoc	genereerib valmis turvalise nimeteenuse JavaDoc dokumentatsiooni
start_srv	käivitab turvalise nimeteenuse serveri

Tabel 2. Turvalise nimeteenuse Ant skripti sihid

Turvalise nimeteenuse realisatsiooni projekti juurkataloogis olevad alamkataloogid jagunevad järgmiselt:s

- idl – IDL liideste kataloog, kus asuvad nii turvalise nimeteenuse IDL liidesed kui ka kasutusel olevad OMG CORBA liidesed (COS nimeteenuse ja ajabaasi IDL liidesed)
- gensrc – IDL liideste põhjal genereeritud Java klasside hoidmise kataloog

- src – lähtekoodi juurkataloog, selle alamkataloogid moodustavad turvalise nimeteenuse pakettide struktuuri
    - sec/service/sns – turvalise nimeteenuse IDL liideste teostuste pakett
    - sec/service/sns/config – konfigureerimise abiklasside pakett
    - sec/service/sns/util – turvalise nimeteenuse abiklasside pakett
- lib – IDL liideste põhjal Java koodi genereerimiseks, lähtekoodi kompileerimiseks ja teenuse käivitamiseks vajalikud OpenOrb Java-teegid
- config – konfiguratsioonifailide hoidmise kataloog
- classes – valmiskompileeritud klasside hoidmise kataloog
- doc – projekti dokumentatsioonifailide hoidmise kataloog
- test – testrakenduse hoidmise kataloog
    - idl – testrakenduse IDL liidese hoidmise kataloog
    - gensrc – testrakenduse IDL liideste põhjal genereeritud Java klasside hoidmise kataloog
    - src – lähtekoodi juurkataloog, selle alamkataloogid moodustavad testrakenduse pakettide struktuuri
    - lib – turvalise nimeteenuse teegifaili paiknemise kataloog
    - config – testrakenduse konfiguratsiooni hoidmise kataloog
    - classes – valmiskompileeritud klasside hoidmise kataloog
- projekti juurkataloogis asuvad failid: Ant script (*build.xml*), lühijuhend (*readme*), katserealisatsiooni litsensifail (*License.txt*) ja viited kasutatud teegide litsensidele (*Used_Api_Licenses.txt*).

5.3.3 Realisatsiooni ehitus

Katserealisatsioon on teostatud eraldiseisva CORBA teenusena. Realisatsioon koosneb serverrakendusest ja teenuseid pakkuvatest objektidest. Teenuseid pakkuvateks objektideks on turvalise nimeteenuse IDL liideste moodulis defineeritud komponendid. Realisatsiooni vaatleme kolmes osas: serverrakenduse ehituse osa, turvalise nimeteenuse realiseeritud IDL liideste osa ja abiklasside komplekti osa.

CORBA teenuse realisatsiooni tähtsaks komponendiks on serverrakendus, kus konfigureeritakse ORB vahendajat ja POA adaptereid ning ühendatakse teenuseid pakkuvad

objektid ORB vahendajaga. Peale selle võib serverrakendus olla samaaegselt teiste CORBA teenuste kliendiks. Serverirakenduste ehituse põhimõtted on kirjeldatud antud töö autori bakalaureusetöös [JH-BT], ning siin vaatleme vaid turvalise nimeteenuse serverrakendusele spetsiifilisi asju. Serverrakendus on realiseeritud teenust käivitava peaklassina. Serverrakenduse tasemel seadistatakse ORB vahendaja parameetreid: edastatakse käsurealt sisestatud ja loetakse sisse konfiguratsioonifailis olevad parameetrid. Osa parameetritest kirjeldab SSLIOP protokolli seadistust. POA adapteri töö juhtimine toimub teguviiside määramise abil. Turvalises nimeteenuses kasutatakse kahte POA adapterit erinevate teguviisidega. Nende kahe adapteri vahe seisneb objekti identifikaatori määramise teguviisi poolest. Esimeses adapteris objekti identifikaator genereeritakse automaatselt, teises võimaldatakse seadistada objekti aktiveeriva klassi poolt. Serverrakenduses luuakse ja aktiveeritakse turvalise nimeteenuse juurnimekontekst. Loomise käigus edastatakse juurnimekontekstile sisseloetud konfiguratsioon. Turvalise nimeteenuse kasutamise lihtsustamiseks seotakse juurnimekontekst nimega ‘corbaloc’ URL formaadis. Viide ‘corbaloc’ formaadis võimaldab klientidel lahendada turvalist nimeteenust algviitena (*initial references*). Sidumine ‘corbaloc’ viitega on teostatud OpenOrb CORBA lokaatori teenuse abil (org.openorb.corbaloc.CorbalocService).

5.3.4 Liideste realisatsioonid

Katserealisatsioonis on teostatud enamus turvalise nimeteenuse IDL liidestes kirjeldatud operatsioonidest. Realiseerimata on mõned vähemtähtsad abioperatsioonid. Nende realiseerimine kuulub turvalise nimeteenuse edasiarendamise plaanide alla. Alljärgnevates osades vaatleme IDL liideste põhjal realiseeritud klasse.

IDL liideste kujutamine Java programmeerimiskeelde on tehtud OpenOrb vahendi abil. Kujutamise tulemuseks on genereeritud Java liidesed ning abstraktsed klassid, mille põhjal luuakse teenuse realisatsioon. Serverrakenduses kasutatakse POA objekti adapterit. ‘POA’ suffiksiga abstraktsed klassid kirjeldavad teenust pakkuvat objekti POA adapteriga seotava teenrina ning on mõeldud IDL liidese kirjeldatud funktsionaalsuse poolt laiendamiseks. Teenrite klassid teostavad abstraktsete meetoditena IDL liidestes kirjeldatud operatsioone.

Turvalise nimekonteksti teostus (sec.service.sns.NamingContextSecImpl) – laiendab IDL liidese põhjal 'NamingContextSec' genereeritud abstraktset klassi (sec.service.sns.SecNaming.NamingContextSecPOA). Lisaks IDL liideses defineeritud operatsioonidele sisaldab nimekontekst rea privaatse juurdepääsuga abimeetodeid. Teostuse klassis defineeritud konstruktor lubab nimekonteksti loomisel seadistada olemasolevat ORB vahendajat, POA adapterit, konfiguratsiooni klassi ja ülemkonteksti. Lisaks nendele parameetritele saab konstruktori kaudu seadistada juurkonteksti ja kohese aktiveerimise lipud. Turvalise nimekonteksti loomise käigus luuakse automaatselt ka vastav turbekontekst. Nimekontekstiga seotavate objektide ja nimekontekstide objektiviiteid hoitakse paisktabelis. Paisktabeli võtmetena kasutatakse objekti või nimekontekstiga seotud nime.
Turbekonteksti teostus (sec.service.sns.SecurityContextImpl) – laiendab IDL liidese põhjal 'SecurityContext' genereeritud abstraktset klassi (sec.service.sns.SecNaming. SecurityContextPOA). Lisaks IDL liideses defineeritud operatsioonidele sisaldab nimekontekst rea privaatse juurdepääsuga abimeetodeid.

Administreerimise abiliidese teostus (sec.service.sns.AdministrationHelperImpl) – laiendab IDL liidese põhjal 'AdministrationHelper' genereeritud abstraktset klassi (sec.service.sns.SecNaming.AdministrationHelperPOA).

Klientide andmekogumi teostus (sec.service.sns.ClientRegStoreImpl) – laiendab IDL liidese põhjal 'ClientRegStore' genereeritud abstraktset klassi (sec.service.sns.SecNaming.ClientRegStorePOA).

Privileegide andmekogumi teostus (sec.service.sns.PrivilegeObjectStoreImpl) – laiendab IDL liidese põhjal 'PrivilegeObjectStore' genereeritud abstraktset klassi (sec.service.sns.SecNaming.PrivilegeObjectStore POA).

Turbetõendite andmekogumi teostus (sec.service.sns. SecurityTokenStoreImpl) – laiendab IDL liidese põhjal 'SecurityTokenStore' genereeritud abstraktset klassi (sec.service.sns.SecNaming.SecurityTokenStorePOA).

Turvalises nimeteenuses defineeritud lihttüübid teostamist ei vaja. IDL liideste põhjal Java koodi genereerimisel tõlgitakse lihttüübid standardseteks Java tüüpideks. Väärtusetüüpidena defineeritud andmetüüpide teostus on samuti tehtud OpenOrb vahendi abil genereeritud Java liideste ja abstraktsete klasside põhjal.

5.3.5 Turvalise nimeteenuse realisatsiooni abiklassid

Lisaks spetsifikatsioonis kirjeldatud IDL liideste põhjal loodud klassidele sisaldab realisatsioon rea abiklassise, mida spetsifikatsioonis otseselt kirjeldatud pole. Turvalise nimeteenuse arhitektuur sisaldab järgmisi abiklasse:

- Konfiguratsiooni abiklass (sec.service.sns.config.Configuration) – sisseloetud konfiguratsiooni hoidmiseks ja komponentide vahel edastamiseks mõeldud klass. Katserealisatsiooni testimise lihtsustamiseks sisaldab konfiguratsiooni abiklass vaikimisi seadistatud parameetreid. Konfiguratsioonifailist sisselugemise ebaõnnestumisel kasutatakse vaikimisi määratud parameetreid.
- Baasabiklass (sec.service.sns.util.Util) – teiste abiklasside poolt laiendamiseks mõeldud baasklass. Antud klassis on defineeritud kaks meetodit: serialiseeritava objekti baidimasiiviks teisendamise abimeetod ja baidimasiivi objektiks teisendamise meetod.
- Administreerimise abiklass (sec.service.sns.util. AdministrationUtil) – koosneb võtmekogumite käsitlemise abimeetoditest. Võtmekogumite käsitlemiseks kasutatakse ‘java.security.KeyStore’ klassi Java (Sun JDK 1.4.2) standardpaketist.
- Autentimise abiklass (sec.service.sns.util.AuthUtil) – sisaldab paroolipõhise autentimise abimeetodeid: parooli räsi genereerimise meetodit, kliendinime ja parooli põhjal räsi genereerimise meetodit ning etteantud kliendinime ja parooli kliendi registratsioonis oleva räsi põhjal valideerimise meetodit. Räsi genereerimiseks ja valideerimiseks kasutatakse ‘java.security.MessageDigest’ klassi Java (Sun JDK 1.4.2) standartpaketist. Räsi genereerimiseks on vaikimisi määratud ‘MD5’ algoritm.
- Klientide haldamise abiklass (sec.service.sns.util.ClientRegUtil) – kirjeldab klientide käsitlemist abistavaid meetodeid.
- Võtmete genereerimise abiklass (sec.service.sns.util.KeyGeneratorUtil) – koosneb võtmepaari genereerimise abimeetoditest. Lisaks pakub meetodeid avaliku ja salajase võtme baidimasiivi kujule teisendamiseks. Võtmepaari genereerimiseks kasutatakse

klassi ‘java.security.KeyPairGenerator’. Võtmepaari genereerimisks on vaikimisi määratud ‘RSA’ algoritm ning võtme pikkuseks on määratud 2048 bitti. Avaliku võtme käsitlemiseks kasutatakse ‘java.security.PublicKey’ klassi ja salajase võtme käsitlemiseks kasutatakse ‘java.security.PrivateKey’ klassi. Baidimassiivina salvestatud võtmete võtmeobjektiks teisendamiseks kasutatakse ‘java.security.KeyFactory’ klassi. Kõik ülalmainitud klassid ‘java.security’ paketist võetakse Java (Sun JDK 1.4.2) standardpaketist.

- Sündmuste logi abiklass (sec.service.sns.util.LogUtil) – on mõeldud rakenduse sündmuste kohta käivate teadete suunamiseks standardväljundisse. Sündmuste jälgimise lihtsustamiseks lisab logiklass automaatselt teadetele päise, milles on märgitud sündmuse kategooria ja kellaaeg. Antud klassi planeeritud edasiarendusse kuulub logikirjete suunamise võimaluse (faili, andmebaasi või monitoorimise vahendisse). Katserealisatsiooni edasiarenduses on logiklass ühendatav sündmuste üldise jälgimise süsteemiga.
- Privileegide käsitlemise abiklass (sec.service.sns.util.PrivilegeUtil) – koosneb privileegide loomise abimeetoditest. Nende meetodite hulka kuuluvad: vaikimisi kehtivate privileegide komplekti loomise meetod, nimekontekstis vaikimisi nõutavate privileegide komplekti loomise meetod ja privileegi objekti korrektsust kontrolliv meetod.
- Turbetõendite käsitlemise abiklass (sec.service.sns.util.SecureTokenUtil) – koosneb turbetõendite käsitlemise abimeetoditest: tõendi signeerimise ja signatuuri valideerimise meetoditest, tõendi koostamise abimeetoditest ja tõendi kehtivuse valideerimise meetoditest. Tõendi signeerimiseks kasutatakse signeerimise abiklassi.
- Signeerimise abiklass (sec.service.sns.util.SignatureUtil) – koosneb serialiseeritavate objektide signeerimise meetoditest. Lisaks on olemas ka signatuuri valideerimise meetodid. Signatuuri käsitlemiseks kasutatakse ‘java.security.Signature’ klassi Java (Sun JDK 1.4.2) standartpaketist. Signatuuri genereerimiseks on vaikimisi määratud ‘MD5withRSA’ algoritm.

5.4 Turvalise nimeteenuse konfigureerimine ja kasutamine

Turvalise nimeteenuse seadistamine toimub konfiguratsioonifailis parameetrite seadistamise kaudu. Konfiguratsiooni fail omab 'java.util.Properties' atribuudifaili struktuuri. Konfigureerimiseks vajalikud parameetrid on kirjeldatud alljärgnevas tabelis:

Parameeter	**Kirjeldus**	**Väärtuse näidis**
javax.net.ssl.keyStore	võtmetekogumi faili asukoht	basic_ks
javax.net.ssl.trustStore	usaldatud võtmete kogumi faili asukoht	basic_ts
javax.net.ssl.keyStorePassword	võtmetekogumi parool	pwd
javax.net.ssl.trustStorePassword	usaldatud võtmete kogumi parool	pwd
rootContextName	juurnimekonteksti nimi	NameServiceSec
rootContextNameAlias	juurnimekonteksti sünonüüm	NameServiceSec
ssliop.port	SSLIOP protokollile seadistatav kuulamise port	684

Tabel 3. Konfigureerimise parameetrid

5.5 Turvalisel nimeteenuse põhinev testrakendus

Turvalise nimeteenuse toimimise demonstreerimiseks on loodud testrakendus. Testrakenduse komponendid kasutavad turvalise nimeteenuse operatsioone omavahelise suhtluse korraldamiseks. Testrakendus kujutab endast lihtsat CORBA rakendust, mis pakub failide haldamise teenust. Turvalises nimeteenuses seotud testrakenduse teenust pakkuvatest objektidest koostatud struktuur moodustab lihtsa hajusfailisüsteemi.

5.5.1 Testrakenduse liides

Testrakenduse eesmärgiks on turvalise nimeteenuse töö näitlikustamine. Testrakenduse IDL liideste moodul (*SecNamingFileAccess*) koosneb ühest liidesest (*FileAccess*). Faililigipääsuliides kirjeldab vaid ühte operatsiooni (*accessFile*) faili käsitlemiseks. Liideses on defineeritud ka rida ligipääsutüüpe, mille abil määratakse faili käsitlemise liiki.

5.5.2 Testrakenduse realisatsioon

Testrakendus on realiseeritud Java programmeerimiskeeles (Sun Java JDK 1.4.2) OpenOrb CORBA realisatsiooni põhjal. Sarnaselt turvalisele nimeteenusele koosneb testrakenduse realisatsioon kolmest osast: serverrakendus, IDL liideste realisatsioonid ja abiklassid.

Testrakendus on jaotatud kaheks alamrakenduseks. Testrakenduse server on üheaegselt serveriks jagatavate failide objektidele, turvalise nimeteenuse kliendiks ning nimeteenusega seotuse failiobjektide kliendiks. Testrakenduse klient suhtleb testrakenduse serveriga turvalise nimeteenuse kaudu.

Võrdluseks on lisatud tavalisel nimeteenusel toimiv testrakendus ning SSL toega nimeteenusel toimiv testrakendus. (SSL tuge kasutamiseks tavalist nimeteenust tuleb konfigureerida vastavuses [OOSSL] dokumendiga.) Nendele kahele testrakendusele on lisatud ka ründaja rakendus. Ründaja rakenduse eesmärgiks on eemaldada nimeseosed juurkonteksti tasemel. Tavalise nimeteenusega seotud ohud on kirjeldatud peatükis 4. Tavalise nimeteenuse puhul ründajale piisab asukoha teadmisest. SSL tuge kasutades (eeldame varianti, kus kliendi autentimine on nõutud) ilma juurdepääsu loata rünne ebaõnnestub. Kui juurdepääs nimeteenusele on antud, siis SSL toe kasutamine ei kaitse sisemist struktuuri (pole võimalik piirata tegevust – näiteks keelata nimeseoste muutmist). Võrdluseks toodud rakenduste põhjal saab näha ka arendusmahu suurenemist iga järgneva turbetaseme saavutamiseks. SSL toe kasutamiseks piisab konfiguratsioonide täiendamisest. Turvalise nimeteenuse kasutamisel tuleb lisada sisselogimise osa ning edaspidine turbetõendi edastamine.

Faililigipääsu objekti teostus (test.sns.service.sec.FileAccessImpl) – laiendab IDL liidese ‘FileAccess’ põhjal genereeritud abstraktset klassi (test.sns.service.sec.FileAccessPOA).

5.5.3 Testrakenduse kasutamine

Testrakenduse kasutamine põhineb turvalise nimeteenuse kasutamisel. Testrakenduse failide haldamiseks on samuti kasutusel Apache Ant projektiehituse vahend. Ant skriptis on defineeritud ülesanded (*target*) IDL liideste Java koodiks kujutamiseks, Java klasside kompileerimiseks, konfiguratsioonifailide kopeerimiseks, genereeritud failide kustutamiseks ja teenuse ning testrakenduse käivitamiseks. Kui ülesanne jäetakse ette andmata, trükitakse

ekraanile ülesandeid kirjeldav lühijuhend. Tabelis 4 on toodud testrakenduse Ant skripti ülesanded.

Ülesande nimi	**Ülesande kirjeldus**
help	trükib ekraanile ülesannete lühijuhendi
all	käivitab järgmised ülesanded: clean, idl, compile, config
idl	genereeritakse Java klassid turvalise nimeteenuse IDL liidese põhjal
compile	kogu projekti kompileerimise ülesanne; käivitab järgmised ülesanded: compile_gen, compile_src
compile_gen	kompileerib IDL liideste põhjal genereeritud Java klassid
compile_src	kompileerib turvalise nimeteenuse lähtekoodi
clean	täispuhastuse ülesanne; käivitab järgmised ülesanded: clean_gen, clean_classes
clean_gen	kustutab IDL liideste põhjal genereeritud klassid
clean_classes	kustutab kõik kompileeritud klassid
config	konfiguratsiooni failide kopeerimine
javadoc	genereerib valmis turvalise nimeteenuse JavaDoc dokumentatsiooni
start-test-server	käivitab testrakenduse serveri
start-test-client	käivitab testrakenduse kliendi
start-test-server2	käivitab testrakenduse serveri (tavaline nimeteenus)
start-test-client2	käivitab testrakenduse kliendi (tavaline nimeteenus)
start-test-bad-client2	käivitab ründaja rakenduse (tavaline nimeteenus)
start-test-server3	käivitab testrakenduse serveri (tavaline nimeteenus SSL toega)
start-test-client3	käivitab testrakenduse kliendi (tavaline nimeteenus SSL toega)
start-test-bad-client3	käivitab ründaja rakenduse (tavaline nimeteenus SSL toega)
start-ns	tavalise nimeteenuse käivitamine
start-mb	OpenOrb haldusvahendi käivitamine (tavalise nimeteenuse jälgimise abiks)

Tabel 4. Testrakenduse Ant skripti sihid

5.6 Turvalise nimeteenuse edasiarendamise võimalused

Turvalise teenuse spetsifikatsioonis oli tehtud rida kitsendusi kohese realiseerimise lihtsustamiseks. Osa CORBA turvatehnoloogiatest jäeti välja seoses nende realisatsioonide puudumisega. Arvestades puuduvate realisatsioonide loomise võimalikkust tulevikus, võib avaneda võimalus nende kasutamisks turvalise nimeteenuse tulevases spetsifikatsioonis. Loodud katserealisatsioonis jäeti mõned vähemtähtsad operatsioonid välja.

Turvalise nimeteenuse katserealisatsiooni loomise käigus selgus rida edasiarendamise võimalusi. Edasiarendamise võimalused jagunevad kahte gruppi: olemasoleva spetsifikatsiooni põhjal realisatsiooni edasi arendamine ning spetsifikatsiooni enda edasi arendamine. Realisatsiooni edasiarenduse alla kuuluvad katserealisatsiooni täiendamine stabiilseks teenuseks, teenuse jõudluse parandamine ja spetsifikatsioonis otseselt mitte kirjeldatud omaduste lisamine. Spetsifikatsiooni edasiarendus on suunatud uute võimaluste sissetoomiseks.

On olemas ka selliseid edasiarenemise võimalusi, mille sisseviimine on võimalik olemasoleva spetsifikatsiooni põhjal, kuid parema tulemuse saavutamiseks on vajalikud ka täiendused spetsifikatsiooni tasemel. Selliseks on säiliva (*persistent*) turvalise nimeteenuse loomise võimalus. Säilivuse all on mõeldud võimalust jätkata tööd sama andmestruktuuriga peale teenuse katkestamist. Katserealisatsioonis säilitatakse andmeid vaid osaliselt ja nimekontekstidest moodustatud struktuur ei säili üldse. Nimeteenuse struktuuri säilitamise teostamine on võimalik olemasoleva realisatsiooni täiendamise kaudu. Spetsifikatsiooni täiendamise kaudu on võimalik laiendada teenuse säilivuse võimalusi: säilitamise operatsioonide lisamine administreerimise abiliidese tasemele, nimeteenuse struktuuri alamosade säilivuse võimaldamine ja säilitamise operatsioonidele autoriseerimise lisamine.

5.6.1 Spetsifikatsiooni edasiarendus

Spetsifikatsiooni edasiarendust vajab nii CORBA turvalisuse teenusega integreerimine kui ka CSIIOP protokolli kasutusele võtmine. Turvalise nimeteenuse edasiarenduses on üks oluline suund: tegevuste jälgimise monitoorimise spetsifitseerimine. Jälgimist vajavate sündmuste alla kuuluvad nii turvalise nimeteenuse operatsioonide sündmused kui ka teenust pakkuvate objektide väljakutse sündmused. Teiste turbetehnoloogiate kasutusele võtmine nõuab samuti

spetsifikatsiooni täiendamist. Võimalike turvalise nimeteenuste täienduste hulka kuuluvad ühendused LDAP kataloogiteenusega ja Kerberose tehnoloogiaga. Turvalise nimeteenuse sarnasusi nende kahe tehnoloogiaga on kirjeldatud ka olemasolevas spetsifikatsioonis. Turvalise nimeteenuse ja LDAP kataloogiteenuse struktuuride kooskasutamist on võimalik realiseerida vahendajateenuse kaudu. Turbetõendi ja LDAP privileegide kooskõlastamise kaudu on võimalik ühendada ka autoriseerimise mehhanisme. Turbetõendi standardi kooskõlastamine Kerberose pileti formaadiga annab võimaluse ühendada turvalise teenuse autoriseerimist Kerberosel põhinevate süsteemidega. (Spetsifikatsiooni edasiarenduse analüüs sai tehtud [CORBA], [SSS], [LPWJ] ja [KUUG] materjalide põhjal.)

5.6.2 Realisatsiooni edasiarendus

Realisatsiooni edasiarenduse põhieesmärgiks on katserealisatsiooni täiendamine täisväärtuslikuks teenuseks. Selle alla kuuluvad katserealisatsioonist välja jäänud operatsioonide teostamine ning defineeritud privileegide võimaluste teostamine täies mahus (privileegi omaniku atribuutidega seotud funktsionaalsus). Teiste lisavõimaluste alla kuuluvad teenuse konfigureerimise võimaluste täiendamine ning realisatsioonipõhise säilivuse teostamine. Java programmeerimiskeeles kirjutatud realisatsioonis on võimalik võtta kasutusele Java-spetsiifilisi tehnoloogiaid. Turvalise nimeteenuse autentimise ja autoriseerimise funktsionaalsust on võimalik ühendada JAAS (*Java Autentication and Authorization Service*) raamistikuga. JAAS raamistiku eesmärkideks on kasutajate autentimise ja autoriseerimise Java rakendustes. JAAS raamistiku kasutamine annab võimaluse ühendada turvalist nimeteenust teiste JAAS raamistikku kasutavate rakendustega. (Realisatsiooni edasiarenduse analüüs sai tehtud [JSDK], [JSEC] ja [J2NS] materjalide põhjal.)

6 Kokkuvõte

CORBA arhitektuur aitab luua heterogeensetes keskkondades toimivaid hajussüsteeme. Suhtlus hajussüsteemides toimub nii süsteemi komponentide endi vahel kui ka väliste kliendirakendustega. Sellise süsteemi ehitamisel on oluline tagada nii suhtluse kui ka komponentide sisemiste protsesside turvalisus. Komponentide vaheline suhtlus peab olema stabiilne ning raskesti pealtkuulatav. Komponentidele juurdepääsu saamiseks peavad kliendid läbima autoriseerimise protsessi. Turvaline nimeteenus ühendab endas mitut olemasolevat tehnoloogiat. COS nimeteenusest on pärandatud komponentide sidumine nimedega ning struktureerimise võimalused. SSLIOP protokolli tugi võimaldab tagada turvalise suhtluse klientidega. Lisaks nendele on spetsifitseeritud autentimise ja autoriseerimise mehhanismid.

Töö raames oli püstitatud kolm eesmärki: uurida CORBA arhitektuuri turvatehnoloogiaid, spetsifitseerida turvaline nimeteenus ja selle spetsifikatsiooni põhjal luua katserealisatsioon. Töö esimene osa on referatiivne ja selles on vaadeldud järgmisi CORBA turbetehnoloogiad: CSI koostöövõime, CORBA turvalisuse teenus ja ATLAS autoriseerimise teenus. Turvatehnoloogiate teoreetilise ülevaate raames on kirjeldatud erinevaid turvaprobleeme ning võimalusi nende lahendamiseks. Töö esimeses osas vaadeldud turvamehhanismide abstraktsed mudelid on aluseks teises osas kirjeldatud spetsifikatsioonile. Lisaks on esimeses osas tehtud lühiülevaade COS nimeteenusest ning analüüsitud sellega seotud turvaprobleeme. Töö teises osas on spetsifitseeritud turvaline nimeteenus – COS nimeteenuse laiendus, millele on lisatud turvalisust tagavad mehhanismid. Nimeteenuse arhitektuuri lisatud turvaelemendid pakuvad lahendust nii esimeses osas kirjeldatud COS nimeteenuse probleemidele kui ka seotud teenuste turvamiseks. Spetsifikatsioon põhineb tööle lisatud turvalise nimeteenuse IDL liideste moodulil. Antud moodul sisaldab kõiki vajalikke liideseid eraldiseisva teenuse loomiseks. Töö kolmanda osa moodustab turvalise nimeteenuse katserealisatsiooni projekti kirjeldus. Katserealisatsioon on tööle lisatud CD kettal. Töö lõpus on vaadeldud turvalise nimeteenuse edasiarendamise võimalusi. Kirjeldatud on võimalikke lisalahendusi realisatsiooni tasemel ja spetsifikatsiooni edasiarendamise võimalusi.

Magistritöö käigus loodud katserealisatsioon võimaldab ehitada turvalisel nimeteenusel põhinevaid rakendusi. Turvalise nimeteenuse planeeritud tulevane edasiarendus on suunatud pakutud võimaluste laiendamiseks.

Безопасная Услуга Наименований для CORBA архитектуры

Магистрская работа

Юрий Харью

Резюме

В данной работе рассмотрена тема безопасности в распределенных приложениях основанных на технологии CORBA (Обобщенная Архитектура построения Брокеров Объектных Запросов). Работа разделена на три части: 1) теоретическая - посвящённая обзору имеющихся технологий безопасности архитектуры CORBA; 2) спецификация Безопасной Услуги Наименований; 3) эксперементальная реализация Безопасной Услуги Наименований.

В обзор имеющихся технологий безопасности архитектуры CORBA включены темы: CSI (Общая Безопасная Интероперабельность), Услуга Безопасности (из набора стандартных дополнительных CORBA-услуг) и услуга авторизации ATLAS. Также рассмотрены проблемы безопасности, связанные с использованием Услуги Наименований (из набора стандартных дополнительных CORBA-услуг).

Безопасная Услуга Наименований специфирована на основе Услуги Наименований (из набора стандартных дополнительных CORBA-услуг). Спецификация содержит описание расширения возможностей базовой Услуги Наименований путем добавления технологий безопасноти. Специфицированная услуга направлена на решение проблем, описанных в первой части, а также на расширение применения технологий безопасности в других услугах, связываемых между собой Услугой Наименований. Модуль IDL-интерфейсов, описанный в спецификации, приложен к работе на CD диске.

Третья часть посвящена описанию экспериментальной реализации Безопасной Услуги Наименований. Также в данной части описаны пути дальнейшего развития рассматриваемой услуги. Экспериментальная реализация приложена к работе на CD диске. Для написания реализации был использован язык Java (версии 1.4.1), используемая реализация CORBA - OpenOrb 1.3.1.

Secure Naming Service for CORBA

Master Thesis

Jüri Harju

Abstract

This current research focuses on security problems in applications based on CORBA architecture. It is divided into three main parts: 1) overview of existing security solutions in CORBA architecture; 2) Secure Naming Service (SNS) specification; 3) experimental implementation of SNS.

CORBA architecture security solutions overview contains the following topics: CSI, COS Security Service and ATLAS authorization service. Additionally, the current research discusses the issue of security problems in the applications based on INS Naming Service.

SNS specification is based on INS Naming Service. Specification contains a detailed description of extending base naming service with added security elements. The aim of the specified service is to solve security problems with INS Naming Service (which are described in first part) and provide security for the services connected by naming service. Use of authentication, authorization and cryptographic extensions are main security features added to naming service. SNS IDL interfaces module is located on the added CD.

Third part of current research describes experimental implementation of SNS. Ways of future development and improvements of SNS specification and implementation are also touched upon. Implementations source code is located on the added CD.

Kasutatud kirjandus

Kasutatud kirjandus on järjestatud viidete tähestikulises järjekorras.

[ANT] "Apache Ant 1.6.1 Manual", The Apache Software Foundation, 2004, http://ant.apache.org/manual/index.html

[ATLAS] "Authorization Token Layer Acquisition Service (ATLAS) Specification", Version 1.0 Object Management Group, October 2002, ftp://www.omg.org/pub/docs/formal/02-10-01.pdf

[CFTR] "CORBA Firewall Traversal Specification", Object Management Group, March 2003, ftp://www.omg.org/pub/docs/ptc/03-01-13.pdf

[CORBA] "CommonObject Request Broker Architecture: Core Specification", Version 3.0 Object Management Group, December 2002, ftp://www.omg.org/pub/docs/formal/02-12-06.pdf

[CNJ] George M. Doss "CORBA Networking with Java", Wordware Publishing, Inc., 1998

[CPU] Suhail Ahmed "CORBA ProgrammingUnleashed", Macmillan Computer Publishing, 1998

[INS] "Naming Service Specification", Version 1.2 Object Management Group, September 2002, ftp://www.omg.org/pub/docs/formal/02-09-02.pdf

[J2NS] Marco Pistoia, Duane F. RellerDeepak Gupta, Milind Nagnur, Ashok K. Ramani "Java 2 Network Security" Second Edition, IBM International Technical Support Organization, 1999

[JCRY] Jonathan B. Knudsen "Java Cryptography", O'REILLY, 1998

[JH-BT] Jüri Harju, "Ülevaade CORBA serverist ja lisateenustest, Bakalaureusetöö", Tartu 2002

[JH-ST] Jüri Harju, "Semestritöö, teemal:Ülevaade CORBA'st", Tartu 2001

[JPC] Gerald Brose, Andreas Vogel, Keith Duddy "Java Programming with CORBA Advanced Techniques for Building Distributed Applications", Third Edition, Wiley Computer Publishing, 2001

[JSDK] "Java™ 2 SDK, Standard Edition Documentation" Version 1.4.2, Sun Microsystems, 2003, http://java.sun.com/j2se/1.4.2/docs/index.html

[JSEC] Scott Oacs "Java Security" Second Edition, O'REILLY, 2001

[KUUG] "Kerberos V5 User's Guide" Release 1.3, Massachusetts Institute of Tehnology, 2004, http://web.mit.edu/kerberos/www/

[LPWJ] Rob Weltman, Tony Dahbura "LDAP Programming with Java", Addison-Wesley, 2002

[OOSSL] Jerome Daniel, Chris Wood, Michael Rumpf "The Community OpenORB - SSL", Version 1.3.1, 2003, http://openorb.sourceforge.org

[OpenOrb] Jerome Daniel, Chris Wood, Michael Rumpf "The Community OpenORB", Version 1.3.1, 2003, http://openorb.sourceforge.org

[Orbix] "Orbix E2A Application Server Platform 6.0 Documentation", IONA Technologies, 2004, http://www.iona.com/support/docs/e2a/asp/6.0/index.xml

[SSS] "Security Service Specification", Version 1.8 Object Management Group, March 2002, ftp://www.omg.org/pub/docs/formal/02-03-11.pdf

[VBDG] "VisiBroker Developer's Guide" version 5.1, Borland Software Corporation, 2002, http://www.borland.com/

[VBGK] "VisiBroker GateKeeper Guide" version 5.1, Borland Software Corporation, 2002, http://www.borland.com/