KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

KOMPIUTERIŲ KATEDRA

Jonas Matačiūnas

VARTOTOJO PRIEIGOS DUOMENŲ SAUGOJIMO LUSTINĖSE KORTELĖSE METODO SUKŪRIMAS IR TYRIMAS

Magistro darbas

Darbo vadovas

doc. dr. Algimantas Venčkauskas

Kaunas, 2011

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

KOMPIUTERIŲ KATEDRA

Jonas Matačiūnas

VARTOTOJO PRIEIGOS DUOMENŲ SAUGOJIMO LUSTINĖSE KORTELĖSE METODO SUKŪRIMAS IR TYRIMAS

Magistro darbas

Recenzentas

doc. dr. Pranas Kanapeckas

2011-05-31

Vadovas

doc. dr. Algimantas Venčkauskas

2011-05-31

Atliko

IFN-9/3 gr. stud.

Jonas Matačiūnas

2011-05-31

Kaunas, 2011

Turinys

ĮVADAS 8
1 AUTENTIFIKAVIMO METODŲ ANALIZĖ 11
1.1 Tikslas 11
1.2 Uždaviniai 11
1.3 Nagrinėjama problema 11
1.4 Mokslinio tyrimo tikslas ir uždaviniai 11
1.5 Autentifikavimas 12
1.6 Lustinių kortelių analizė 14
1.7 E. paslaugos 16
1.8 Calypso sistemos analizė 17
1.9 Autentifikavimo metodai 20
1.9.1 TLS protokolas 20
1.9.2 Diffie-Hellman raktų apsikeitimo protokolas 24
1.9.3 DH-EKE protokolas 25
1.9.4 SRP protokolas 26
1.9.5 REVE protokolas 27
1.10 Išvados 28
2 SIŪLOMAS AUTENTIFIKAVIMO PROTOKOLAS 28
2.1 Protokolo apžvalga 28
2.2 Pranešimų formatas 29
2.3 Pranešimo eigos schema 30
2.4 Pranešimų specifikacijos 32
2.4.1 DH-EKE protokolas 33
2.4.2 SRP protokolas 34
2.4.3 REVE protokolas 34
2.4.4 Patikrinimo pranešimas 35
2.5 Kortelėje saugomi duomenys 36
2.6 Atliekamų skaičiavimų pasiskirstymas 37
2.7 Išvados 38
3 PROTOKOLO EMULIATORIAUS PROJEKTAS 39
3.1 Projekto aprašymas 39

3.2 Panaudos atvejai 39
3.3 Reikalavimai projektui 40
3.3.1 Nefunkciniai reikalavimai 40
3.3.2 Funkciniai reikalavimai 40
3.4 Projekto architektūrinė apžvalga 41
3.5 Klasių diagramos 41
3.6 Grafinės sąsajos 44
3.7 Realizavimo detalės 45
3.8 Vartotojo dokumentacija 47
3.8.1 Sistemos reikalavimai 47
3.8.2 Sistemos naudojimosi gidas 47
3.9 Išvados 48
4 PROTOKOLO TYRIMAS 48
4.1 Sistemos veikimo tyrimas 48
4.2 Protokolo funkcinių pranašumų apibūdinimas 51
4.3 Protokolų saugumo savybių palyginimas 52
4.4 Protokolo greitaveikos tyrimas 53
4.5 Išvados 58
5 IŠVADOS 59
6 LITERATŪRA 60
7 SANTRUMPŲ ŽODYNAS 62

Paveiksliukų turinys

1 pav. Lustinės kortelės fiziniai kontaktai 14
2 pav. Lustinės kortelės elektroniniai komponentai 15
3 pav. Debetinė operacija 19
4 pav. Rankos paspaudimas TLS protokole 24
5 pav. Bendras pranešimų eiliškumas 31
6 pav. Projekto panaudos atvejai 39
7 pav. Projekto architektūros diagrama 41
8 pav. Serverio klasių diagrama 42
9 pav. Kliento klasių diagrama 43
10 pav. Kliento grafinė sąsaja 44
11 pav. Serverio grafinė sąsaja 45
12 pav. Kliento vartotojo sąsaja su paaiškinimais 48
13 pav. Serverio tyrimo rezultatai DH-EKE protokolui 49
14 pav. Kliento tyrimo rezultatai DH-EKE protokolui 49
15 pav. Serverio tyrimo rezultatai SRP protokolui 50
16 pav. Kliento tyrimo rezultatai SRP protokolui 50
17 pav. Serverio tyrimo rezultatai REVE protokolui 51
18 pav. Kliento tyrimo rezultatai REVE protokolui 51
19 pav. Programoms prieinamų procesoriaus branduolių nustatymas 54
20 pav. DH-EKE protokolo greitaveikos matavimo rezultatai 55
21 pav. SRP protokolo greitaveikos matavimo rezultatai 55
22 pav. REVE protokolo greitaveikos matavimo rezultatai 56
23 pav. Protokolo greitaveikos rezultatų grafikas 57

Lentelių turinys

Lentelė Nr. 1 Lauko-reikšmės poros formatas....29
Lentelė Nr. 2 Pasisveikinimo pranešimas....32
Lentelė Nr. 3 Palaikomi protokolai....32
Lentelė Nr. 4 Serverio pasisveikinimo pranešimas....33
Lentelė Nr. 5 Klaidos pranešimas....33
Lentelė Nr. 6 DH-EKE raktų apsikeitimo pranešimas....33
Lentelė Nr. 7 SRP kliento rakto apsikeitimo pranešimas....34
Lentelė Nr. 8 SRP serverio rakto apsikeitimo pranešimas....34
Lentelė Nr. 9 REVE protokolo pranešimas....34
Lentelė Nr. 10 Kliento patikrinimo pranešimas....35
Lentelė Nr. 11 Serverio patikrinimo pranešimas....35
Lentelė Nr. 12 Skaičiavimo pasiskirstymas DH-EKE protokolo atveju....37
Lentelė Nr. 13 Skaičiavimo pasiskirstymas SRP protokolo atveju....37
Lentelė Nr. 14 Skaičiavimo pasiskirstymas REVE protokolo atveju....38
Lentelė Nr. 15 Skaičiavimo pasiskirstymas patikrinimo proceso metu....38
Lentelė Nr. 16 Protokolo greitaveikos rezultatų lentelė....56
Lentelė Nr. 17 Santykinis protokolų vykdymo laikas....57
Lentelė Nr. 18 Santykinis protokolo vykdymo laikas....58

Development and research of method for storage user access data in smart card

Summary

Rapid development and spread of computer and internet technologies led to rich variety of electronic services available to consumers. Some services are public and require no authentication to make available to users. Some are tied directly to specific user identity and are available to user only upon successful authentication. For example e. banking services can be used after user has successfully authenticated to bank system. Because of many different electronic services user must remember its username and password. Note that passwords must be of specific length and complexity to be considered secure. Research has shown that in general case users are unable to remember long and complex passwords.

Our proposed solution is to store authentication data in smart card and use smart card technology to authenticate users to target systems. We propose an authentication protocol which was specifically designed to be used with smart cards.

ĮVADAS

Gyventojų kompiuterizacija ir naujausių internetinių technologijų sėkminga plėtra lėmė spartų elektroninių paslaugų paplitimą visame pasaulyje. Šiomis dienomis vienokiomis ar kitokiomis elektroninėmis paslaugomis naudojasi kiekvienas. Kai kurios elektroninės paslaugos yra viešosios, tai yra kad jos būtų suteiktos vartotojui nereikia autentifikuotis į sistemą. Jos yra prieinamos visiems. Kitos paslaugos yra glaudžiai susijusios su vartotojo tapatybe ir jos suteikiamas tik vartotojui sėkmingai prisijungus prie sistemos. Pavyzdžiui elektroninės bankininkystės atveju vartotojas prieš pradedantis naudotis banko teikiamomis paslaugomis privalo autentifikuotis į elektroninės bankininkystės sistemą. Kitas pavyzdys galėtų būti mobilaus telefono sąskaitos peržiūrėjimas. Vėlgi kiekvienu atveju kai elektroninė paslauga yra siejama su konkrečiu vartotoju iškyla vartotojo autentifikavimo problema. Dažniausiai šiai problemai spręsti naudojamos vartotojo vardo ir slaptažodžio sistemos. Prieiga prie paslaugos suteikiama tik tuomet, kai vartotojas pateikia teisingą vartotojo vardą ir slaptažodį. Jeigu reikalinga dar didesnė sauga naudojamos papildomos saugos priemonės, tokios kaip kodų kortelės, vienkartinių slaptažodžių generatoriai ir kiti įtaisai suteikiantys papildomą saugumą.

Deja sparčiai vystantis naujoms informacinėms technologijoms taip pat didėja našta ir patiems jų vartotojams. Pavyzdžiui, didėjant kompiuterių pajėgumams, ilgėja ir saugaus slaptažodžio ilgis. Per trumpas slaptažodis gali būti paprasčiausiai atspėjamas pilno perrinkimo būdu. Tai lemia, kad vartotojai turėtų įsiminti vis sudėtingesnius ir ilgesnius slaptažodžius, kuriuos naudoja autentifikuojantis į įvairias sistemas. Atlikti tyrimai rodo, kad bendru atveju vartotojai nesugeba įsiminti ilgų slaptažodžių [10], ir yra linkę naudoti tuos pačius slaptažodžius daugelyje sistemų kurias naudoja.

Kaip vienas iš tokios problemos sprendimo būdų yra naudoti slaptažodžių duomenų bazes, kuriose yra saugomi vartotojo slaptažodžiai. Kadangi patys slaptažodžiai yra saugomi, tai jų įsiminti vartotojui nereikia. Todėl kiekvienai sistemai gali būti generuojami ilgi ir skirtingi slaptažodžiai. Tokios programinės įrangos yra įvairios, vienos jų pačios duomenų bazes saugo tame pačiame kompiuteryje kaip ir pati programa. Kitos slaptažodžius saugo centralizuotoje duomenų bazėje pas paslaugos tiekėją. Pačios duomenų bazės apsaugomos pagrindiniu slaptažodžiu, kurį turi įsiminti vartotojas. Tokiu atveju vartotojui reikėtų įsiminti tik vieną slaptažodį. Tokios sistemos problema jog praradus pagrindinį slaptažodį būtų iš karto atskleisti ir visi kiti vartotojo naudojami slaptažodžiai.

Šiame darbe nagrinėjame galimybę spręsti vartotojo prieigos duomenų apsaugos problemą panaudojant lustinę kortelę. Pačios lustinės kortelės yra stipriai paplitusios. Jos panaudojimas labai platus:

- Mokėjimo kortelės
- Mobilaus telefono SIM kortelės
- Įvairios prieigos kontrolės sistemos
- Elektroninis bilietas
- Elektroninė sveikatos paciento kortelė
- Ir įvairios kitos sistemos

Didelis lustinių kortelių panaudojimas lėmė ir stiprų jų technologijų išsivystymo laipsnį. Pati lustinė kortelė yra gerai apsaugota netgi ir nuo fizinių atakų, tokių kaip kortelės išardymas ar elektromagnetinių bangų matavimas veikiant kortelei.

Taigi darbe nagrinėjame galimybė panaudojant lustinę kortelę saugoti vartotojo prieigos duomenis. Svarbu paminėti, jog duomenys turi būti saugomi prasmingai, jog butų galimybė juos panaudoti autentifikuojant vartotoją į sistemą. Taip pat svarbus momentas yra komunikacijos kanalo nesaugumas. Pats komunikacijos kanalas, kuriuo vyksta bendravimas tarp lustinės kortelės ir terminalo, yra laikomas nesaugiu. Todėl šiame darbe nagrinėjamos ne tik lustinės kortelės, bet ir saugūs raktų apsikeitimo protokolai panaudojant nesaugų komunikacijos kanalą.

Atlikę analizė pateiksime savo autentifikavimo protokolą, kuris būtų atsparus įvairioms atakoms kurios galimos naudojant nesaugų komunikacijos kanalą. Iš autentifikavimo protokolo taip pat matysis kaip kortelėje saugomi duomenys galima panaudoti atliekant autentifikavimą. Patį autentifikavimo protokolą realizuosime programine įranga ir tai bus šio darbo eksperimentinė dalis.

Pirmame šio darbo skyriuje yra analizuojamos lustinės kortelės, autentifikavimo metodai ir protokolai. Nuodugniai buvo išanalizuotas TLS protokolas pasinaudojant oficialu TLS RFC dokumentu [16]. Analizuojami ir raktų apsikeitimo ir autentifikavimo protokolai DH-EKE ir SRP. Taip pat buvo išanalizuota Calypso elektroninių bilietų sistema. Remiantis analize buvo pasiūlytas autentifikavimo protokolas. Protokolas nuodugniai aprašytas antrame skyriuje. Protokole panaudojamos idėjos iš TLS protokolo ir panaudoji išanalizuoti raktų apsikeitimo ir autentifikavimo protokolai.

Trečiame skyriuje pateikiamas eksperimento projekto aprašymas. Kadangi eksperimentas bus realizuojamas programine įranga, tai šiame skyriuje rasime architektūros aprašymą, klasės diagramas ir kitas realizacijos detales.

Ketvirtame skyriuje aprašytas protokolo tyrimas. Pirmiausia buvo tiriama ar pasiūlytas protokolas tikrai veikia. Po to aptartos protokolų saugumo savybės. Galiausiai ištirta protokolo greitaveika.

1 AUTENTIFIKAVIMO METODŲ ANALIZĖ

1.1 Tikslas

Šio darbo tikslas yra sukurti prieigos duomenų saugojimo lustinėse kortelėse metodą, kuris naudojantis viena lustine kortele atpažintų e. paslaugos autentifikavimo metodą ir automatiškai autentifikuotų vartotoją toje sistemoje.

1.2 Uždaviniai

- Išanalizuoti esamas e. paslaugų autentifikavimosi sistemas, jų veikimo principus
- Išanalizuoti autentifikavimosi protokolus
- Sukurti unifikuotą autentifikavimosi metodą pasinaudojant lustinėje kortelėje saugomais prieigos duomenimis
- Realizuoti ir ištirti sukurtą metodą

1.3 Nagrinėjama problema

Dėl didžiulės e. paslaugų įvairovės vartotojams reikia įsiminti daug įvairių slaptažodžių ir vartotojų vardų, PIN kodų, turėti įvairių vienkartinių kodų generatorių, lustinių kortelių.

1.4 Mokslinio tyrimo tikslas ir uždaviniai

Tyrimo tikslas yra išanalizuoti skirtingų e-paslaugų naudojamus autentifikavimo metodus, jų veikimo principus. Išanalizavus įvairius autentifikavimo metodus išanalizuoti galimybę autentifikavimui naudoti lustinę kortelę. Atsižvelgiant į įvairius ištirtus autentifikavimo metodus sukurti vieningą autentifikavimo metodą, kuris galėtų būti naudojamas daugeliui e-paslaugų vartotojo autentifikavimui atlikti.

1.5 Autentifikavimas

Autentifikavimas – tai procesas kurio metu patikrinama ar subjektas yra tikrai kuo dedasi. Autentifikavimo metodai skirstomi į tris pagrindines grupes:

- Kažkas, ką žinai (slaptažodis, kriptografinis raktas ir t.t.).
- Kažkas, ką turi (lustinė kortelė ir t.t.).
- Kažkas, kas esi (piršto anspaudas, balsas, akies rainelė ir kiti biometriniai duomenys).

Vartotojo privačių autentifikavimo duomenų apsaugos problema – kaip apsaugoti privačius vartotojų autentifikavimo duomenis nuo pavogimo, nepageidaujamo pakeitimo, sunaikinimo. Autentifikavimo duomenys gali būti nuo paprasčiausio slaptažodžio iki RSA privataus rakto ar kitokios informacijos. Slaptažodžio atveju yra žinoma, jog vartotojai bendru atveju nesugeba atsiminti ilgų slaptažodžių [10], kurie turi būti ilgi, kad būtų pakankamai saugūs naudoti autentifikavimui, todėl vartotojai dažnai juos užsirašinėja, kas yra skaitoma bloga praktika. Vienas iš šios problemų sprendimų būdų yra naudoti specialiai tam sukurtą programinę įrangą (pvz.: KeePass, RoboForm ir kt.), kuri saugo slaptažodžius. Tokia programinė įrangą saugo vartotojo suvestus slaptažodžius juos užšifruodama faile. Patys failai dėl papildomo saugumo gali būti USB flash atmintinėje. Failai užšifruojami pagrindiniu slaptažodžiu, taigi, net jeigu ir kenkėjui pavyktų pavogti vartotojo failus, jis negalėtu iš jų išgauti slaptažodžių, nes jie yra užšifruoti. Iš esmės toks slaptažodžių saugojimo būdas yra saugus, nes vartotojui tereikia atsiminti vieną slaptažodį ir dėl to didėja tikimybė kad vartotojas naudos ilgesnį ir sudėtingesnį slaptažodį [10]. Tokio metodo trukumas akivaizdus – jeigu vis dėlto kenkėjui kaip nors pavyksta sužinoti vartotojo pagrindinį slaptažodį, tai jis sužino visus vartotojo suvestus slaptažodžius. Taip pat kaip papildomą apsaugą galima pasirinkti ir slaptą failą, kurį programa naudoja skaičiuodama raktą. Tokiu būdu reikalingas ne tik slaptažodis, bet ir slaptas failas, norint išgauti slaptažodžius. Kita problema susijusi su saugumu yra tai, kad vartotoją vardą ir slaptažodį reikia įvesti į kažkokią tai prisijungimo formą. Kenkėjas tą žinodamas, gali įdiegti tekstą registruojančią programą ir taip gauti įvedinėjamą slaptažodį.

Kita vertus, slaptažodžiai nėra vieninteliai prieigos duomenys. Kaip kitą prieigos duomenų tipą panagrinėkime RSA algoritmo privačius raktus. Prie šito tipo duomenų taip pat priskirsime ir sertifikatus (sertifikatus su privačiais raktais), nes iš esmės tai yra tie patys duomenys. Be abejo, kadangi tokie raktai yra ilgi, juos vartotojas atsiminti neturi jokios galimybės, dėl šios priežasties raktai yra paprastai saugojami faile. Failas dažnai būna apsaugotas papildomu slaptažodžiu. Slaptažodžio kaip apsaugos priemonės trūkumus jau

nagrinėjome. Pats privataus rakto naudojimas vietoj slaptažodžio yra tas, kad raktas yra ilgas ir šiuolaikiniai nėra tokie galingi, kad galėtų perrinkti visus įmanomus raktus, bandant įsibrauti į sistemą ar iššifruoti kokią nors žinutę užšifruotą privačiu RSA raktu.

Apie RSA privačių raktų (ar kitokią privačių autentifikavimo informacijos) įsiminimą galima pamiršti. Taigi, kadangi vartotojai yra nepajėgūs įsiminti savo autentifikavimo duomenų, jie turi būti saugiai saugomi kokioje nors laikmenoje. Taigi sekančiuose skyriuose analizuojama galimybė saugoti privačius autentifikavimo duomenis lustinėse kortelėse. Ten kur reikalingas ypač didelis saugumas, naudojami vadinamieji HSM (hardware security module) aparatai [11]. Didelis saugumas pasiekiamas, nes privatūs raktai egzistuoja tik pačiame HSM, kuris yra atskiras fizinis įrenginys. Jeigu reikia kad būtų pasirašomi duomenys privačiu raktu, tai duomenys siunčiami į HSM ir HSM pasirašęs duomenis gražina parašą. Tokių būdu privatūs raktai visada yra tik HSM įrenginyje, o tai suteikia didesnį saugumo lygį. HSM dažnai naudojami bankuose ir sertifikatų registracijos centruose, kur privataus rakto atskleidimas turėtų katastrofiškas pasekmes. Tarpinis variantas tarp HSM ir rakto saugojimo faile yra lustinės kortelės. Jos plačiau nagrinėjamos sekančiame skyriuje.

Taip pat kaip kitą prieigos duomenų tipą galime nagrinėti biometrinius autentifikavimo duomenis. Priešingai negu prieš tai nagrinėtų prieigos duomenų biometriniai duomenys yra susiję su vartotojo fizinėmis ar elgsenos savybėmis ir jis jų neturi įsiminti. Biometriniai duomenys yra skirstomi į dvi pagrindines grupes: fizinės savybės ir elgsenos savybės. Fizinės tai tokios kaip kraujagyslės skenavimas, piršto antspaudas, akies rainelė, veido atpažinimas, rankos ir delno geometrija ir kt. Elgsenos tai tokios kaip balso atpažinimas, rašysenos atpažinimas, eisenos atpažinimas ir t.t. Pirmą kartą vartotojas naudodamasis biometrine sistema „prisirašo“ save į sistemą. T.y. sistema išsisaugos biometrinio duomens šabloną, pagal kurį toliau bus tikrinama ar vartotojas yra tikrai tas kuo sako esąs. Sekančius kartus kai vartotojas nori autentifikuotis į sistemą, biometrinė sistema paėmusi iš vartotojo biometrinius duomenis ir palyginusi su turimu šablonu gali nuspręsti, ar vartotojas yra autentiškas. Visgi išlieka problema kaip saugiai saugoti patį šabloną, kuris reikalingas sulyginimui. Taip pat tokių biometrinių duomenų trūkumas yra tas, kad šie duomenys lyginant su paprastu slaptažodžiu nėra pakeičiami, t.y. jeigu biometrinių duomenų šablonas buvo sukompromituotas tai vartotojas negali tiesiog pasikeisti piršto antspaudo, kaip tą gali padaryti su slaptažodžiu. Ši problema sprendžiama prieš išsaugojant biometrinį šabloną, jį iškraipant kokiu nors dėsniu. Vėliau kai vyksta tikrinimas gauti biometriniai duomenys vėl iškraipomi tuo pačiu dėsniu, ir sulyginamas gautas rezultatas su iškraipytu rezultatu.

1.6 Lustinių kortelių analizė

Kadangi iš problemos analizės pamatėme, jog apsimoka autentifikavimo duomenis saugoti kokioje nors fizinėje laikmenoje, toliau nagrinėjame lustines korteles kaip autentifikavimo duomenų saugyklą. Lustinės kortelės analizei pagrinde naudosime knygą [14] kaip informacijos šaltinį.

Taigi lustinė kortelė yra įrenginys, kuris turi integruotą mikroprocesorių arba atminties mikroschemą. Kortelė su kortelės skaitytuvu susijungia tiesiogiai fiziniu būdu arba pasinaudojant RFID/NFC technologijomis kai kortelė yra netoli nuskaitomojo įrenginio. Jeigu kortelė turi mikroprocesorių, dažnai jinai gali atlikti ir kriptografines operacijas (TDES, RSA), ar net generuoti RSA raktų poras [13].

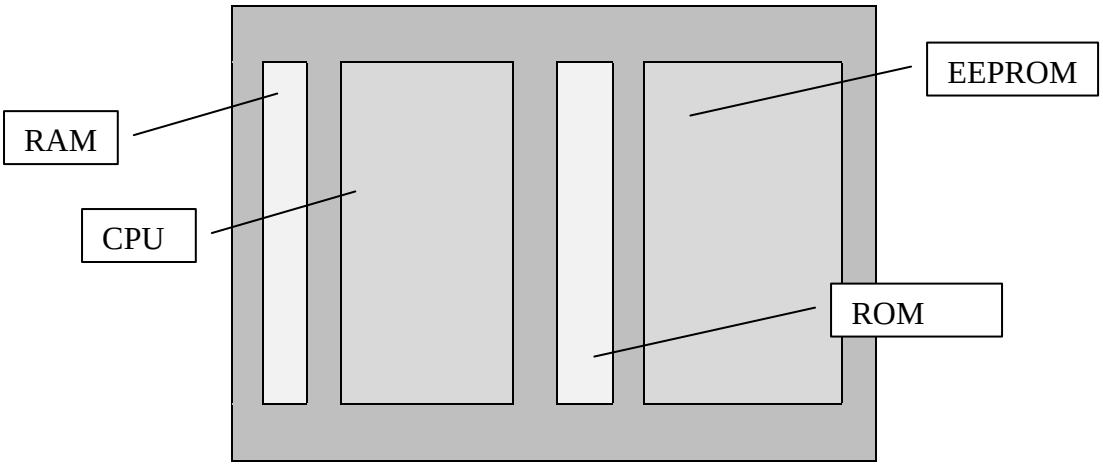
Tuo atveju kai naudojama fizinė sąsaja duomenų mainai vykdomi kortelė fiziškai prijungiant prie kortelių skaitytuvo, kuri turi aštuonias jungtis.

Vcc		GND
RST		Vpp
CLK		I/O
RFU		RFU

1 pav. Lustinės kortelės fiziniai kontaktai

Kadangi kortelės neturi savo maitinimo šaltinio tai jos yra užmaitinamos per Vcc (common collector voltage) ir GND (ground electricity) jungtis. I/O jungtis yra naudojama komunikacijai tarp kortelės ir skaitytuvo. VPP (voltage point to point) kontaktas senesnėse kortelėse buvo naudojamas EEPROM atminčiai valdyti, bet šiuo metu nėra naudojamas. Per CLK (clock) jungtį yra paduodamas sinchronizavimo signalas. Apatinės dvi RFU (reserved for future use) jungtys nebuvo naudojamos, tik yra planų jas naudoti realizuojant NFC ir didelės greitaveikos USB sąsają. Pati komunikacija vyksta pakaitinu dvipusiu (half duplex) režimu. T.y. vienu metu duomenys gali būti tik skaitomi arba tik siunčiami į kortelę. Yra du bendravimo protokolai informacijos apsikeitimui, žinomi kaip T=0 ir T=1. T=0 atveju informacija siunčiama po baitą, o T=1 informacija perduodama duomenų blokais.

Žiūrint fiziškai lustinė kortelė tai plastmasės karkasas, kurio viduje yra mikroprocesorius ir kiti elektroniniai komponentai reikalingi jos veikimui. Loginė elektroninių komponentų randamų kortelėje schema pavaizduota žemiau.



2 pav. Lustinės kortelės elektroniniai komponentai

Kaip matosi iš paveikslėlio yra vienas procesorius (CPU) ir trys skirtingos rūšies atmintys. ROM (read only memory) atmintis skirta tik skaitymui, jos turinys nustatomas gamybos metu ir informacija iš šios atminties gali būti tiktai nuskaitoma. RAM (random access memory) paskirtis yra tokia pati kaip ir personalinių kompiuterių (tik jos talpa yra daug kartų mažesnė), t.y. saugoti programos kintamiesiems ir steko informacijai. Kai atjungiamas išorinis maitinimo šaltinis visi duomenys buvę šioje atmintyje yra prarandami. EEPROM (electronically erasable programmable read only memory) atminties turinys gali būti perrašomas ir po gamybos etapo ir priešingai nuo RAM atminties nepraranda duomenų atjungus maitinimą. Taip pat nauja tendencija yra flash tipo atminties naudojimas lustinėse kortelėse. Tai plačiai paplitusi technologija tarp vadinamųjų USB atmintukų. Lyginant su EEPROM flash atmintis turi keletą pranašumų. Visu pirma EEPROM atmintis palyginus greitai susidėvi. Taip pat flash atminties duomenų rašymas vyksta sparčiau, pati atminties talpa yra didesnė negu EEPROM.

Reikėtų taip pat atkreipti dėmesį, kad paminėti elektroniniai komponentai yra papildomai fiziškai apsaugoti nuo atakų [1].

Yra du pagrindiniai tarptautiniai standartai susiję su lustinėmis kortelėmis: ISO/IEC 14443 ir ISO/IEC 7816. ISO/IEC 14443 standarte yra aprašoma nekontaktinių kortelių, kurios veikia 10 cm ir mažiau (nuo kortelių skaitytuvo antenos), fizinės charakteristikos,

radijo signalo prieigos aprašymas, susidūrimų išvengimo ir duomenų perdavimo protokolai. Tuo tarpu ISO/IEC 7816 standartas skirtas kontaktinėms lustinėms kortelėms.

Prieš įsivyraujant lustinėms mokėjimo kortelėms vyravo magnetinės mokėjimo kortelės. Jos nebuvo tokios saugios kaip lustinės kortelės, nes jas buvo galima palyginti lengvai padirbti. Tuo tarpu padirbti ar padaryti kokią nors kitą efektyvią ataką prieš lustinę kortelę yra daug sudėtingiau [1, 8].

Iš programinės pusės pirmosios lustinės kortelės nebuvo labai saugios, nes kortelės operacinė sistema ir taikomųjų programų funkcijos buvo labai susipynę [1]. Tuo tarpu šiuolaikiškos lustinės kortelės yra suprojektuotos taip, kad operacinės sistemos ir taikomųjų programų funkcijos būtų atskirtos. Populiari lustinių kortelių operacinė sistema: „Java Card“. Tai operacinė sistema pagrįsta Java technologija, bet pritaikyta lustinėms kortelėms. Ši operacinė sistema leidžia patogiai plėtoti taikomąsias programas, pvz.: taikomosios programos gali būti įkraunamos į kortelė po jos fizinio pagaminimo. Duomenys laikomi kortelėje yra organizuojami į hierarchinę failų sistemą. Kiekvienas failas gali turėti skirtingus priėjimo prie failo nustatymus. Galimi tokie priėjimo nustatymai kai priėmimas prie failo duodamas tik tada, kai įvedamas teisingas PIN kodas.

Lustinių kortelių panaudojimas itin platus: mokėjimo kortelės, tapatybės kortelės, sertifikatų saugyklos, prieigos kontrolė, elektroniniai bilietai, paciento medicininių įrašų saugojimas ir t.t.

Nors lustinės kortelės yra labai saugios, visgi jos turi silpnųjų vietų kuriomis pasinaudoja piktavaliai. Viena iš tokių silpnųjų vietų yra ryšys tarp kortelės ir kortelės nuskaitymo įrenginio. Kaip vieną iš pavyzdžių būtų galima paminėti sėkmingas atakas prieš EMV mokėjimo korteles kai pavyksta padaryti sėkmingą pirkimą, net ir nežinant teisingo PIN kodo[3]. Šiai atakai įgyvendinti buvo panaudojamas tas faktas, kad ryšys tarp kortelės ir terminalo nėra nei autentifikuojamas nei šifruojamas. Kitas pavyzdys būtų sėkminga ataka prieš balsavimo sistemą, kuri balsams saugoti naudoja lustines korteles [12]. Taigi svarbu atkreipti dėmesį ir į saugaus ryšio užmezgimą tarp kortelės ir kortelės skaitymo įrenginio.

1.7 E. paslaugos

Šiomis dienomis mus supa įvairios e. paslaugos, tokios kaip:

- e. bilietas
- e. bankininkystė

- e. parduotuvė

Minėtoms paslaugos paprastai vartotojui reikia įsiminti arba vartotojo vardą ir slaptažodį arba ir su savimi turėti kokią nors kortelę. Kaip jau minėjome, vartotojas nesugeba įsiminti ilgų slaptažodžių, be to jis gali naudoti tuos pačius slaptažodžius visoms ar kai kurioms e. paslaugų grupėms, o tai laikoma bloga praktika ir neigiamai įtakoja saugumą. Toliau atidžiau panagrinėsime Calypso e. bilietų sistemą iš saugumo pusės, atkreipdami dėmesį į išmaniosios kortelės panaudojimo saugumo klausimus.

1.8 Calypso sistemos analizė

Calypso analizei naudojome technines sistemos specifikacijas [3,4,5].

Taigi Calypso sistema susideda iš dviejų pagrindinių technologijų:

- Išmaniosios kortelės
- Bekontaktės sąsajos

Kadangi NFC technologija yra suderinama su bekontakte sąsaja naudojamu išmaniosiose kortelėse, tai Calypso kortelė gali būti ne tik išmanioji kortelė, bet ir laikrodis, mobilus telefonas ar kitoks įrenginys palaikantis NFC technologiją.

Žiūrint iš vartotojo pusės su kortele galima papildyti arba su ja atsiskaityti. Papildymas taip pat vyksta specialiuose, kur sumokėjus papildymo sumą, jinai yra papildoma į kortelę.

Calypso e-bilieto sistemos saugumas yra pagrįstas simetriniais raktais, kurie yra saugomi tiek kortelėje, tiek SAM. SAM – tai kortelės skaitytuve esantis specialus saugumo modulis, skirtas papildomai apsaugoti atliekamas kriptografines operacijas ir saugomus kriptografinius raktus. Toks modulis turi atitikti ISO/IEC 7816-3 T=0 standartą, palaikyti DES-X ir TDES šifravimo algoritmus. Visi terminalai (kortelių skaitytuvai), kurie yra naudojami su kortelėmis ir skaitytuvai turi turėti SAM modulį. SAM modulis vienu metu gali laikyti 126 kriptografinius raktus. Kiekvienas raktas yra susietas su parametrais, kurie nusako kokiems tikslams raktas gali būti naudojamas. Pavyzdžiui kai kurie raktai yra naudojami tik parašo patikrinimui, kiti turi būti periodiškai keičiami.

Pagrindinis mechanizmas kuris užtikrina kortelės komunikacijos saugumą yra saugi sesija. Saugi sesija prasideda pasiunčiant specialią komandą į kortelę ir baigiasi pasiunčiant saugios sesijos pabaigos komandą. Sesijos uždarymo pranešime kortelė ir SAM pasirašo visus duomenis kurie buvo siųsti sesijos metu. Tokiu būdų gaunamos tokios garantijos:

- Terminalas įrodo kortelei kad jis yra autentiškas
- Kortelė įrodo terminalui kad jinai yra autentiška
- Įrodoma kad duomenys kurie keliavo tarp terminalo ir kortelės yra autentiški ir nebuvo pakeisti

Sesijos metu galima tiek skaityti įrašus iš kortelės, tiek į ją rašyti (priėjimas prie kai kurių failų gali būti leistas tik įvedus PIN). Pagrindinės komandos, kurias supranta kortelė, yra tokios: atidaryti saugią sesiją, uždaryti saugią sesiją, pažymėti įrašą, perskaityti įrašą, papildyti įrašą, atnaujinti įrašą, įrašyti naują įrašą, padidinti skaitliuko reikšmę, sumažinti skaitliuko reikšmę ir t.t. Taigi pasinaudojant šiomis komandomis galima realizuoti įvairias schemas. Pavyzdžiui, pasinaudojant komandomis atnaujinti įrašą, galima būtų papildyti mėnesinį bilietą iki tam tikros dienos. Pasinaudojant įrašo nuskaitymo komanda galima patikrinti ar bilietas yra papildytas. Jeigu naudojami vienkartiniai bilietai, tai galima pasinaudojus skaitliukų mechanizmu realizuoti ir šitą schemą. Tuomet papildymo metu iškviečiama komanda padidinti skaitliuko vertę, o naudojant bilietą sumažinti. Atkreipi dėmesį į tai, kad visos šitos operacijos yra atliekamos sesijos ribose, ir skirtingiems veiksmams naudojami skirtingi raktai.

Kriptografiniai raktai yra skirstomi į tokią hierarchiją:

1. Leidėjo raktas. Raktas galintis įkrauti kitus raktus, autorizuoti duomenų modifikaciją ar patikrinti tam tikras vertes.
2. Užkrovimo raktas. Gali būti naudojamas autorizuoti duomenų pakeitimą, patikrinti vertes. Paprastai yra naudojamas papildant kortelę, pavyzdžiui pratęsiant mėnesinę kortelę.
3. Debeto raktas. Gali būti naudojamas autorizuoti duomenų pakeitimą arba patikrinti vertes. Paprastai yra naudojamas debeto operacijoms, pavyzdžiui vienkartinio bilieto panaudojimas.
4. Kiti raktai. Gali būti naudojami patikrinti tam vertes.

Visi raktai yra 16 baitų ilgio, ir yra naudojami kaip TDES arba DESX algoritmų raktai. Raktai yra identifikuojami raktų identifikatoriumi, kuris susideda iš tokių laukų:

- Rakto tipas, nusako kokio tipo yra raktas (anksčiau minėti 4 hierarchijos lygiai).
- Rakto versija. Naudojamas nurodant rakto skirtingus raktus tame pačiame lygmenyje.

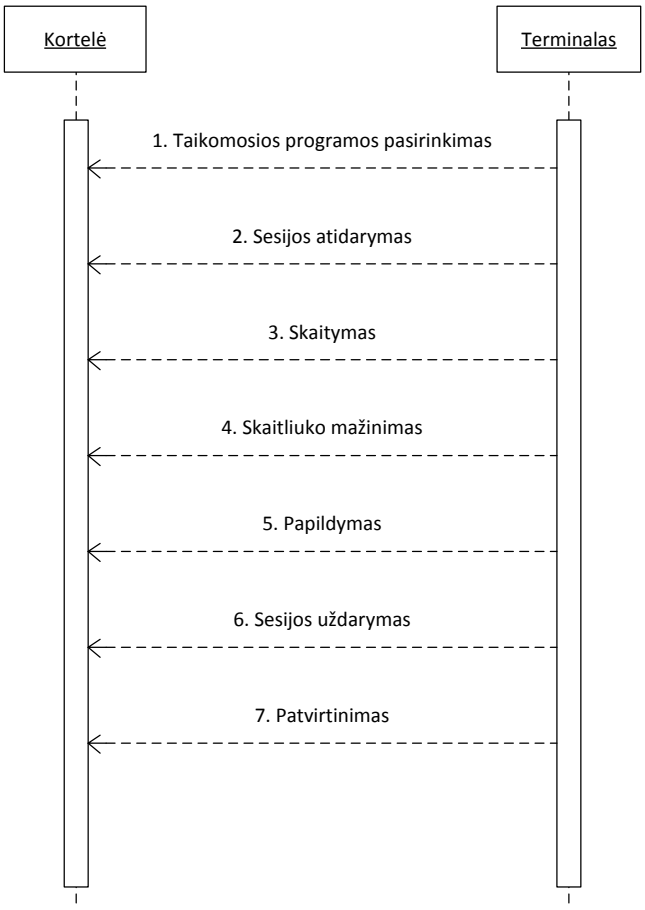
Raktai kurie yra gauti iš pagrindinio rakto turi turėti pagrindinio rakto tipą ir versijos numerį. Kad terminalas teisingai pasirinktų kokį raktą naudoti, jis turi sužinoti rakto

identifikatorių iš kortelės. Šį identifikatorių kortelės gauna kaip atsakymą į taikomosios programos pažymėjimo komandą. Taip pat kitas identifikatorius gali būti gražinamas sesijos inicijavimo pradžios komandoje.

Taip pat svarbus momentas yra tai, jog kortelės raktas yra unikalus tik tai vienai kortelei. Šis raktas yra suskaičiuojamas taip: pagrindiniu raktu yra užšifruojamas taikomosios programos serijinis numeris. Šis skaičiavimas atliekamas vieną kartą kortelės gamybos metu, ir kiekvieną kartą aptarnaujančiame terminale (nes kortelių yra labai daug). Terminale kriptografinius veiksmus, tame tarpe ir kortelės rakto gavimą atlieka SAM.

Dėl to kad kortelės raktai yra unikalūs per kortelę, tai atskleidus kortelės raktą būtų galima padirbti tik tą vieną kortelę.

Žemiau pateikiama debetinės operacijos pavyzdys, įvardijant kokios komandos yra siunčiamos tarp terminalo ir kortelės.



3 pav. Debetinė operacija

1. Siunčiama taikomosios programos pasirinkimo komanda kortelei. Kortelė gražina programos serijinį numerį, rakto identifikatorių. Terminalas pasinaudodamas informacija gali apskaičiuoti kortelės raktą.
2. Terminalas siunčia sesijos atidarymo komandą.
3. Nuskaitomi duomenys iš kortelės. Šiuo atveju tai gali būti skaitliuko parodymu nuskaitymas.
4. Jeigu skaitliuko parodymai yra pakankami, tada vykdoma skaitliukų sumažinimo operacija.
5. Vykdoma duomenų papildymo operacija. Šiuo atveju tai galį būti įrašas apie transakcijos duomenis: laikas, data, vieta ir t.t.
6. Sesijos uždarymo komanda.
7. Siunčiama patvirtinimo komanda, kad užtikrinti proceso įvykdymą.

1.9 Autentifikavimo metodai

Šiame skyriuje nagrinėsime autentifikavimo metodus. Panagrinėsime ir aukštesnio lygio protokolus tokius kaip TLS, ir arčiau kriptografijos lygio tokius šifravimo raktų apsikeitimo protokolus kaip SRP, kurie geba ne tik apsikeisti šifravimo raktais bet ir autentifikuoti vartotoją. Autentifikavimo protokolo įvykdymo rezultatas - vartotojas yra autentifikuotas sistemoje.

1.9.1 TLS protokolas

E. bankininkystės ir kitų e-paslaugų atveju vartotojas dažniausiai autentifikuojasi į internetinę svetainę. Dažniausiai naudojama technologija, siekiant apsaugoti perduodamus duomenis, tarp vartotojo interneto naršyklės ir paslaugos teikėjo svetainės – TLS.

Taigi panagrinėsime šiuo metu stipriai paplitusi SSL/TLS protokolą. Konkrečiai nagrinėsime TLS protokolo 1.2 versiją. Analizės šaltinis yra TLS RFC dokumentas [16]. Protokolas naudojamas užtikrinant saugų ryšį tarp kliento ir serverio, apsaugojant nuo tokių atakų kaip pasiklausymas, žinučių turinio keitimas ir klastojimas.

Protokolas susideda iš dviejų sluoksnių: TLS įrašo protokolo ir TLS rankos paspaudimo protokolu. Žemiausiame protokolo lygmenyje veikia TLS įrašo protokolas, po kuriuo veikia koks nors transporto protokolas (pvz.: TCP). Šiame protokolo lygyje taikomos tokios saugumo priemonės:

- Persiunčiami duomenys gali būti (ir dažniausiai yra, išskyrus tam tikrus atvejus) šifruojami pasinaudojant simetrine kriptografija, tokiais algoritmais kaip RC4, AES ir t.t. Šifravimo raktai yra generuojami kiekvienai sesijai ir jie yra unikalūs. Raktų susitarimui naudojamas kitas protokolas – toks kaip TLS rankos paspaudimo protokolas.
- Yra tikrinamas persiunčiamų žinučių integralumas. Tai yra pasiekiama naudojant MAC algoritmą. Kaip maišos funkcija gali būti naudojama SHA-1 algoritmas. Integralumas gali būti nenaudojamas tam tikrais atvejais, pavyzdžiui kai TLS įrašo protokolas yra naudojamas saugumo parametrų susitarimui, pavyzdžiui naudojamas TLS rankos paspaudimo protokolas.

Aukštesniame lygyje veikia TLS rankos paspaudimo protokolas, kurio metu klientas ir serveris vienas kitą autentifikuoja ir susitaria dėl šifravimo algoritmo, jo parametrų ir šifravimo raktų apsikeitimo būdo. Tiek kliento tiek serverio autentifikavimas gali būti atliekamas pasinaudojant simetrine arba asimetrine kriptografija. Dažnai autentifikuojasi tik serveris, klientas to nedaro (nors yra tam techninės galimybės). TLS rankos paspaudimo protokolas pasižymi trimis savybėmis:

- Subjekto tapatumas gali būti nustatomas pasinaudojant simetrine arba asimetrine kriptografija.
- Slaptos informacijos, iš kurios vėliau yra generuojami šifravimo raktai, susitarimas yra saugus. Net ir matant visą perduodamų duomenų srautą ar jį valdant (žmogaus viduryje ataka) neina sužinoti slaptos informacijos ir tuo labiau šifravimo raktų.
- Slaptos informacijos susitarimo procesas yra atsparus pakeitimams: jeigu yra modifikuojami duomenys kurie yra persiunčiami šio proceso metu, tai subjektai pastebės kad duomenys buvo modifikuoti.

TLS įrašo protokolas apgaubia kitus keturis TLS protokolus: rankos paspaudimo, pranešimo apie klaidą, šifravimo parametru pakeitimo ir taikomųjų duomenų. Jeigu duomenys netelpa į viena TLS įrašą, jie yra skaidomi į atskirus paketus ir gavėjo gale vėl sujungiami į vieną vientisą duomenų paketą. Taip pat kiekvienas TLS įrašas gali būti papildomai suspaudžiamas pasirinktu algoritmu.

TLS rankų paspaudimo algoritmas susideda iš kitų žemesnio lygio protokolų, kurie padeda saugiai susitarti dėl kriptografinių parametrų:

- Šifro keitimo (angl. change cipher spec) protokolas – naudojamas pranešti jog tolimesnis ryšio šifravimas turi būti atliekamas su naujai susitartu šifravimo metodu ir raktais.
- Įspėjimo (angl. alert) protokolas – naudojamas pranešti apie klaidą. Jeigu klaida yra kritinė (angl. fatal), po jos iškarto nutraukiamas ryšys su kita puse.

Bendru atveju rankos paspaudimas vyksta taip:

1. Apsikeičiama pasisveikinimo („hello“) pranešimais, kuriuose susitariama dėl algoritmų, perduodamos atsitiktinės reikšmės.
2. Apsikeičiama būtinais kriptografiniais parametrais kad tiek klientas tiek serveris galėtu susitarti dėl bendros slaptos informacijos, kuri bus naudojama raktams generuoti.
3. Apsikeičiama sertifikatais ir kita kriptografine informacija, kad klientas ir serveris galėtu autentifikuoti vienas kitą.
4. Sugeneruojama pagrindinė paslaptis (angl. master secret) pasinaudojant slapta informacija ir atsitiktinėmis reikšmėmis.
5. Klientui ir serveriui leidžiama įsitikinti, kad jų pagrindinė paslaptis yra vienoda ir kad rankos paspaudimų metu kenkėjas nepakeitė persiunčiamų duomenų.

Detaliau nagrinėjant vyksta tokie veiksmai: klientas siunčia ClientHello žinutę, kurioje klientas įvardina kokius šifravimo algoritmus palaiko, sesijos identifikatorių (jeigu norima užmegzti ne naują sesiją, o pratęsti prieš tai buvusią), atsitiktinę reikšmę, dabartinį kliento laiką, palaikomus suspaudimo algoritmus, protokolo versiją.

Gavęs tokią žinutę serveris išsirenka šifravimo metodą ir siunčia atgal ServerHello pranešimą. Jeigu serveris mato, kad nei vieno šifravimo metodo nepalaiko (iš kliento pateiktų) tuomet siunčia kad įvyko klaida ir nutraukia ryšį. Taigi kokį šifravimo metodą naudoti pasirenka serveris, klientas tik pateikia visus palaikomus šifravimo metodus tikėdamasis, kad serveris nors vieną iš jų irgi palaiko.

Jeigu serveris turi sertifikatą, ir sertifikato paskirtis atitinka pasirinktą šifravimo metodą, tuomet jis siunčia Certificate pranešimą. Šiame pranešime yra sertifikatų grandis, nuo serverio sertifikato iki aukščiausio lygio sertifikato (jo gali ir nebūti, nes klienas vis tiek jį turi turėti pas save). Tuomet klienas, patikrinęs visą sertifikatų grandį, gali patvirtinti, kad serverio sertifikatas yra autentiškas. Šis pranešimas gali būti ir nesiunčiamas tokiais atvejais kai serveris neturi sertifikato arba kai autentifikavimas nėra naudojamas.

Server Key Exchange pranešimas yra siunčiamas kai sertifikate esanti informacija yra nepakankama užmegzti saugų ryšį, konkrečiai tai priklauso nuo to koks šifravimo metodas buvo pasirinktas. Dažniausiai šita žinutė yra naudojama perduoti Diffie-Hellman viešąjį raktą klientui.

Toliau gali būti siunčiamas CertificateRequest pranešimas, kuriuo yra prašoma kliento pateikti savo sertifikatą. Dažniausiai šis metodas nėra naudojamas, nes klientai neturi savo privačių sertifikatų.

Paskutine žinute ServerHelloDone pranešama klientui, kad serveris baigė ir toliau laukia pranešimų iš kliento pusės.

Tada klientas, jeigu gavo CertificateRequest siunčia savo sertifikatų grandinę į serverį.

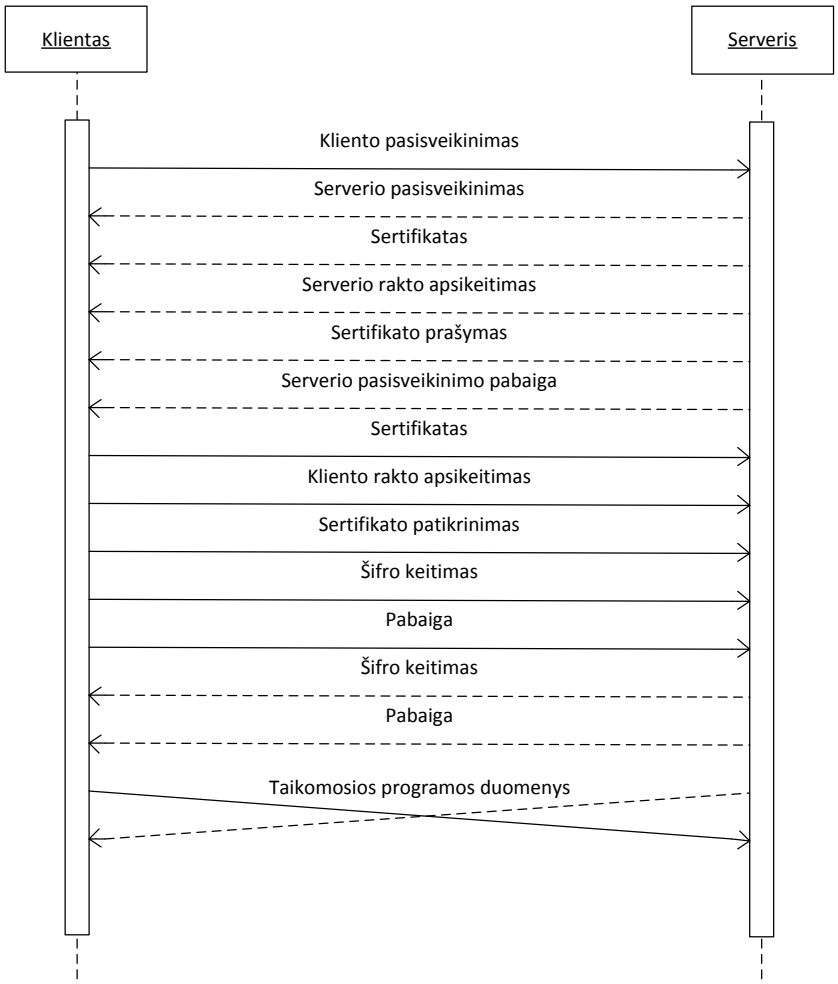
Toliau yra siunčiama ClientKeyExchange pranešimas. Šitame pranešime, jeigu buvo pasirinktas atitinkamas šifravimo metodas, siunčiama sugeneruota atsitiktinė slapta reikšmė, kuri yra užšifruojama serverio sertifikato viešuoju raktu.

Jeigu klientas siuntė Certificate pranešimą (su sertifikatu kuris gali pasirašinėti), tai jis siunčia ir CertificateVerify pranešimą, kuriame yra visų siųstų pranešimų parašas.

Po to yra siunčiamas ChangeCipherSpec pranešimas, kuris nurodo, kad susitarimas dėl raktų pradeda galioti. Šiame pranešime yra duomenys, kuriuos patikrinus galima įsitikinti kad tiek serveris tiek klientas siuntė ir gavo tuos pačius pranešimus (jie nebuvo pakeisti pakeliui) ir autentifikavimas sėkmingai pavyko. Po šio pranešimo kiti pranešimai jau yra šifruojami ir pasirašomi.

Galiausiai yra siunčiama Finished pranešimas, kuris sako, kad autentifikavimas ir raktų apsikeitimas pavyko sėkmingai. Gavę iš antros pusės Finished pranešimą jau galima siųsti pranešimus taikomojo lygio kurie yra apsaugoti.

Taigi bendras TLS rankos paspaudimo protokolas atrodo taip:



4 pav. Rankos paspaudimas TLS protokole

1.9.2 Diffie-Hellman raktų apsikeitimo protokolas

1976 metais Whitfield Diffie ir Martin Hellman paskelbė raktų apsikeitimo protokolą, kuris šiandien žinomas kaip Diffie-Hellman raktų apsikeitimo protokolas. Šį protokolą nagrinėjame todėl, kad norint suprasti DH-EKE ir SRP protokolus pirmiau būtina suprasti Diffie-Hellman protokolo veikimą. Kad išnagrinėti Diffie-Hellman protokolą įvedame pažymėjimus:

a, b – atsitiktinai sugeneruotos reikšmės

p – pirminis skaičius

g – grupės generatorius

Kad būtu paprasčiau nagrinėti protokolą bendraujančias puses toliau vadinsime A tašku ir B tašku. Taigi A taškas sugeneruoja slaptą reikšmę a. Suskaičiuoja reikšmę $A= g^a \bmod p$. Siunčia B taškui reikšmes g, p, A. B taškas susigeneruoja savo slaptą reikšmę b ir apskaičiuoja reikšmę $B = g^b \bmod p$ ir siunčia apskaičiuotą reikšmę A taškui. Taip pat B taškas suskaičiuoja sesijos rakta $K = A^b \bmod p$. A taškas gavęs reikšmę B suskaičiuoją sesijos raktą $K = B^a \bmod p$. Toliau pranešimai tarp taško A ir B gali būti šifruojami simetriniu algoritmu pasinaudojant suskaičiuotu raktu K. Reikšmės a ir b yra nereikalingos, ir suskaičiavus reikšmę K turėtu buti sunaikinamos. Protokolo saugumas remiasi į Diffie-Hellman matematinę problemą, t.y. turint g^x ir g^y rasti g^{xy}. Kol kas šiuo metu efektyviausias būdas išspręsti šia problemą yra išspręsti diskretaus logaritmo uždavinį, t.y. rasti x žinant g^x, bet pati diskretaus logaritmo problema yra gerai žinoma matematikoje problema, kuriai efektyvus sprendimo būdas dar nerastas.

Be abejo, minėtosios saugumo savybės galioja tik teisingai parinkus parametrus a, b, p, g. Parametrai a ir b turi būti pakankamai ilgi, p turi būti netik pakankamai didelis, bet ir tenkinti kitas matematines savybes. Parametro g reikšmė dažniausiai naudojama 2 arba 5.

Tai pat, atkreipkime dėmėsi, jog protokolas nėra saugus prieš žmogaus viduryje ataką (angl. man in the middle attack). Jeigu piktavalis valdo kanalą kuriuo komunikuoja taškai A ir B, tai jis apsimesdamas taškui A tašku B (ir atvirkščiai) gali susitarti tiek su tašku A ir B dėl skirtingų raktų K. Tuomet gavusi pranešima gali atšifruoti pranešimą, jį pakeisti arba perskaityti, ir užšifravus reikiamu raktu nusiųsti kitai pusei. Taškas A ir B gali net nežinoti kad jų pranešimai yra perskaitomi ar net keičiami.

1.9.3 DH-EKE protokolas

DH-EKE tai Diffie-Hellman patobulintas protokolas. Jis susideda iš dviejų etapų. Pirmas etapas skiriasi nuo DH protokolu tik tuo, kad A ir B reikšmės prieš siunčiant yra užšifruojamos raktu $F_s=F(s)$, kur F – vienkryptė funkcija. Nors daug kur literatūroje minima, jog siunčiant į vieną pusę A arba B galima nešifruoti [7], visgi rekomenduojama šifruoti tiek A tiek B norint išvengti tam tikrų atakų prieš protokolą [1].

Antrame protokolo etape taškas A turi įrodyti taškui B (ir atvirkščiai), kad jie tikrai žino slaptažodį s. Tai gali būti atliekama tokiu būdu: A taškas sugeneruota atsitiktinį dydį C_A, užšifravusi suskaičiuotu raktu K nusiunčia šį pranešimą taškui B. B sugeneruoja atsitiktinį dydį C_B, ir siunčia užšifravęs raktu K dydžius C_B ir C_A Alisai. A taškas gavęs atšifruoja C_A, ir patikrina ar tai ta pati reikšmė kurią išsiuntė. Tuomet išsiunčia B taškui užšifruotą C_B

reikšmę. B taškas patikrina ar C_B reikšmė sutampa. Po šių veiksmų taškai A ir B gali būti tikri, jog jie turi tą patį raktą K.

Atkreipti dėmesį į tai, jog šis protokolas yra atsparus žmogaus viduryje atakai, nes pirmame etape piktavalis negali pakeisti A ir B, nes jie yra šifruoti, o norint juos dešifruoti reikalingas slaptažodis s, kurio piktavalis nežino.

1.9.4 SRP protokolas

Konkrečiai nagrinėjame SRP v6 protokolo versiją. Protokolą nagrinėsime pasiremdami RFC dokumentu [17]. Autentifikavimas vyksta klientui pateikiant vartotojo vardą ir slaptažodį. Iš saugumo pusės protokolas yra netik saugus nuo pasyvių pilno perrinkimo atakų, bet ir nuo žodyno pagrįstų atakų. Kita protokolo savybė yra ta kad klientui nereikia saugoti jokių raktų ar kitokių autentifikavimo duomenų kad serveris butų autentifikuotas. Tai reiškia kad nereikia jokios viešo rakto infrastruktūros.

Tiek klientas tiek serveris pirmiausia turi būti susitarę dėl pirminio skaičiaus N. Pirminis skaičius turi turėti tokia savybę, kad q butų pirminis ir N = 2q + 1 irgi būtų pirminis. Kitaip tariant pirminio skaičiaus N paieškos algoritmas turi būti toks: sugeneruojamas pirminis skaičius q, padauginamas iš 2, pridedama 1 ir tikrinama ar gautas skaičius vėl yra pirminis. Gautas pirminis skaičius N yra vadinamas saugiu pirminiu skaičių nes turi tam tikrų savybių kurios reikalingos kad protokolas veiktų saugiai. Atkreipti dėmesį į tai kad skaičius N generuojamas vieną kartą ir po to naudojamas tas pats. Kaip ir DH algoritmo atveju N skaičiaus atskleidimas neigiamos įtakos saugumui neturi. Toliau visi aritmetiniai veiksmai bus atliekami skaičių ciklinėje aibėje N.

Toliau turėtu būti pasirenkamas ciklinės aibės generatorius g. Dažniausiai vartojamos reikšmės yra 2 ir 5. Kaip ir N atveju g reikšmė yra vieša, jos atskleidimas neigiamos įtakos saugumui nedaro.

Tiek klientas tiek serveris visur kur reikia turėtu naudoti vienodą maišos funkciją. Gali būti naudojamos SHA šeimos funkcijos, MD5 ar kokia kita maišos funkcija. Toliau maišos funkciją žymėsime raide H.

Toliau protokolo eiga vyksta taip: klientas ir serveris susiskaičiuoja reikšmę k = H(N,g).

Serverio pusėje kriptografinė druska žymima raide s. Jos paskirtis yra kovoti prieš žodyno atakas.

Raide I žymėsime vartotojo identifikatorių. Paprasčiausiu atveju tai gali būti vartotojo vardas. Raide p žymėsime vartotojo slaptažodį.

Kitas svarbus kintamasis yra serverio slaptažodžio teisingumo įrodymas $v = g^x$, $x = H(s,p)$. v turėtų būti suskaičiuojamas prieš naudojantis sistema, p reikšmė serverio pusėja niekada neturi būti saugoma. Saugoma tik v reikšmė kurios pakanka patikrinti ar klientas pateikė teisingą slaptažodį. Reikėtų atkreipti dėmesį į tai jog jeigu v yra atskleidžiamas tai iš jo išgauti slaptažodį p yra sudėtinga.

Skaičiai a ir b yra atsitiktiniai skaičiai, atitinkantys DH algoritmo privačias reikšmes. Simboliu „|“ žymėsime baitų masyvo apjungimo operaciją.

Bendravimas prasideda klientui siunčiant užklausą į serverį: I, A, kur $A = g^a$. Serveris pasitikrinęs ar identifikatorius I egzistuoja atsako į klientui tokiu pranešimu: s, B, kur $B = kv + g^b$.

Tuomet abi pusės susiskaičiuoja reikšmę $u = H(A,B)$.

Kliento slapta reikšmė yra suskaičiuojama pagal formulę $S_k = (B - kg^x)^{(a+ux)}$

Kadangi suskaičiuotas skaičiaus ilgis bitais priklauso nuo N, tai galutinis raktas gaunamas panaudojant maišos funkciją. T.y. Slaptas raktas gaunamas atlikus veiksmą $K = H(S_k)$.

Tuo tarpu serverio slapta reikšmė yra suskaičiuojama pagal formulę $S_s = (Av^u)^b$. Ir slaptas raktas $K = H(S_{serverio})$.

Gautas slaptas raktas yra vienodas kliento ir serverio pusėje ir yra saugus naudoti šifravimo ar kitoms reikmėms.

Papildomai gali būti atliekami papildomi veiksmai norint įsitikinti jog klientas ir serveris suskaičiavo ta patį raktą K.

Klientas siunčia $M1 = H(H(N) \text{ XOR } H(g) \mid H(I) \mid s \mid A \mid B \mid K)$. Serveris taip pat suskaičiuoja šia reikšmę ir patikrina ar jinai yra tokia pati kaip gauta iš kliento. Jeigu reikšmės sutampa tai siunčiama reikšmė $M2 = H(A \mid M_1 \mid K)$ klientui. Klientas taip pat savo pusėje suskaičiuoja šią reikšmę ir patikrina ar jinai sutampa su gauta reikšme iš serverio.

Jeigu reikšmės sutampa tai reiškias raktai yra vienodi kliento ir serverio pusėje.

1.9.5 REVE protokolas

Darant darbą iškilo poreikis trečiam raktų apsikeitimo protokolui, kuris butų iš visų paprasčiausias, greičiausias ir butų naudojamas kaip atskaitos taškas lyginti kitiems autentifikavimo protokolams. Remdamiesi žiniomis kaip veikia Calypso bilietų sistema aprašome protokolą, kuris yra kiek galima daugiau priartintas Calypso veikimo principui.

Patogumo dėlei toliau protokolą visur vadinsime REVE (angl. random encrypted value exchange) protokolu.

Protokolo veikimo principas: abi pusės autentifikavimui naudoja simetrinį raktą. Sesijos rakto susitarimas vyksta taip: taškas A sugeneruoja atsitiktinę reikšmę a, ir užšifravęs siunčia taškui A. Taškas B sugeneruoja atsitiktinę reikšmę b ir užšifravęs siunčia taškui A. Kadangi abi pusės naudoja tą patį raktą tai gali atšifruoti gautas reikšmes. Tuomet sesijos raktas gaunamas suskaičiavus maišos funkciją nuo a ir b reikšmių. Kadangi reikšmės a ir b yra siunčiamos šifruotos, o raktą žino tik autentifikuojamos pusės, tai ryšio kanalo klausytojas negali suskaičiuoti ar žinoti koks bus sesijos raktas.

1.10 Išvados

1) Išnagrinėję Calypso e. bilietų sistemą matome, kad sistemos saugumas remiasi simetrine kriptografija. Matome kad sistema yra labai saugi, yra apsauga netik nuo kortelių padirbinėjimo, bet ir nuo kortelės siunčiamų pranešimų pakeitimo ar suklastojimo.
2) E. bankininkystės ir e. paslaugų atveju saugų komunikacijos kanalą sukuria TLS technologija. Išanalizavime ir šį protokolą, gautą informacija panaudosime kuriant savo autentifikavimo protokolą.
3) Išnagrinėjome keletą raktų apsikeitimo protokolų, kuriuos panaudosime kurdami savo siūlomą autentifikavimo protokolą.
4) Kilus poreikiui įvedėme REVE protokolą, kurį naudosime kaip atskaitos tašką lyginant kitus raktų apsikeitimo protokolus.

2 SIŪLOMAS AUTENTIFIKAVIMO PROTOKOLAS

2.1 Protokolo apžvalga

Atsižvelgdami į informaciją kurią gavome analizės etape siūlome autentifikavimo protokolą kuris tiktų tiek e. bilietų autentifikavimui, tiek e. bankininkystės vartotojų autentifikavimui. Žiūrint iš protokolo pusės e. bilieto paslaugos autentifikavimui bus naudojamas simetrinis raktas, kuris saugomas kortelėje ir suskaičiuojamas terminale. Saugiam ryšio užmezgimui (ir autentifikavimui) bus naudojamas DH-EKE arba REVE protokolas. E. bankininkystės ar kitos e. paslaugos atveju kur naudojamas vartotojo vardas ir

slaptažodis panaudosime SRP protokolą. Vartotojo vardas ir slaptažodis taip pat bus saugomi lustinėje kortelėje. Kadangi vartotojui šio slaptažodžio įsiminti nereikės jis gali būti labai ilgas.

Ryšio užmezgimo pradžioje serveris (terminalas) siųs pranešimą kortelei kuriame pasirenkama autentifikavimo programa. Kortelė savo ruožtu siųs pranešimą, kuriame išvardijami visi autentifikavimo protokolai kuriuos jinai palaiko. Serveris, išsirinkęs protokolą kokį palaiko, siųs pranešimą kuris protokolas bus naudojamas atliekant autentifikavimą. Po to priklausomai nuo pasirinkto protokolo vyksta informacijos mainai, po kurių vartotojas autentifikuojamas į sistemą ir susitariama dėl bendro slapto rakto. Šis raktas gali būti tiesiogiai naudojamas pasirašyti ar užšifruoti siunčiamus pranešimus, arba gali būti naudojamas suskaičiuoti atskirus raktus šifravimo ir parašo formavimo funkcijoms. Šio slapto rakto panaudojimo protokolas neapibrėžia, tai nulemia konkrečios taikomosios programos sukurtos konkrečios paslaugoms.

2.2 Pranešimų formatas

Bendravimas vyksta siunčiant pranešimus iš siuntėjo į gavėją. Kiekvienas pranešimas prasideda 2 baitais, kurie nusako pranešimo ilgį (atmetus ilgio baitus). Pirmiausia rašomas vyresnysis baitas, toliau jaunesnysis. Pranešimo turinys yra laukų reikšmių poros. Patogumui laukų vardai bus koduojami ASCII formatu. Taigi lauko-reikšmės užkodavimas atrodo taip:

Lentelė Nr. 1 Lauko-reikšmės poros formatas

Baito numeris	Paaiškinimas
1	Nusako lauko pavadinimo ilgį n.
2 + n	Lauko pavadinimo reikšmė užkoduota ASCII eilute.
2 + n + 1	Du baitai nusakantys reikšmės ilgį m. Pirma eina vyriausias baitas, toliau jaunesnysis.
2 + n + 2 + m	Lauko reikšmė.

Po kiekvienos lauko-reikšmės poros gali eiti sekanti lauko-reikšmės pora. Jų gali būti tiek kiek reikia perduoti informacijai.

Atkreipti dėmesį jog nėra koduojama informacija apie lauko tipą. Galimi laukų tipai:

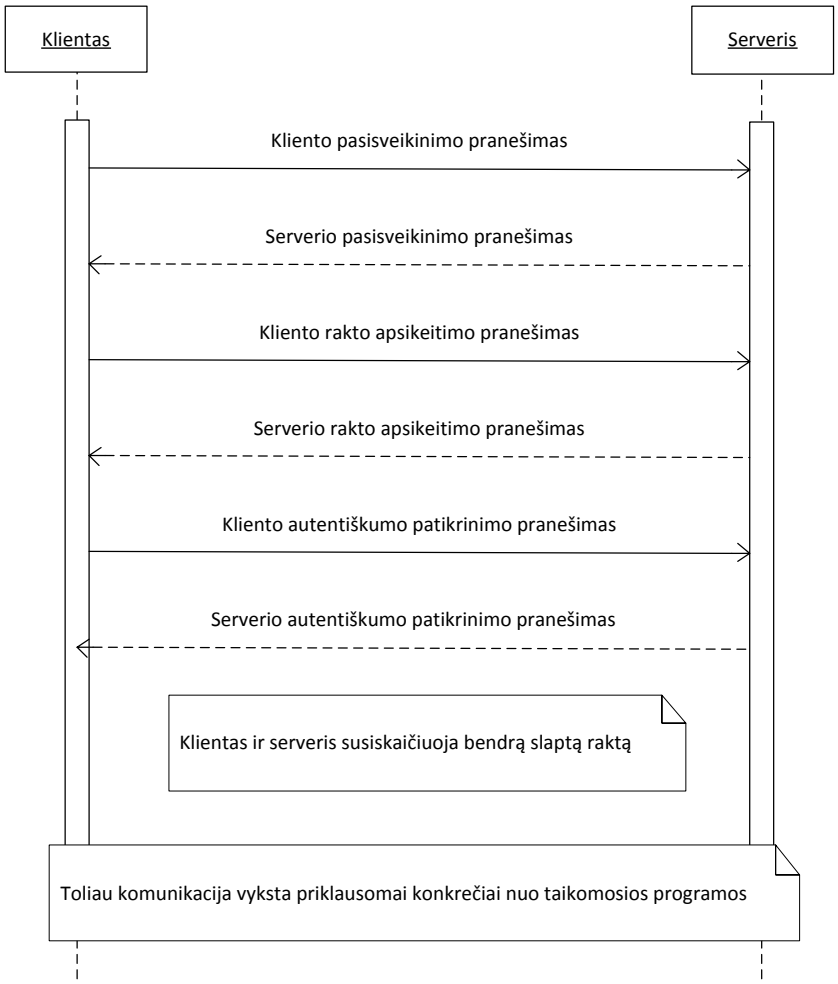
- Baitų masyvas. Tokiu atveju lauko reikšmė yra baitų masyvas. Dažniausiai naudojamas perduoti raktų apsikeitimo protokolui reikalingas reikšmes.
- ASCII eilutė. Tokiu atveju lauko reikšmė yra ASCII eilutė, be terminuojančio ‚\0‘ simbolio. Naudojamas identifikuoti protokolą ir t.t.

Iš esmės protokolą būtų galima realizuoti pasitelkus tik ASCII eilutės lauko tipą ir baitų masyvus perduoti konvertavus baitų masyvą i šešioliktainės sistemos ASCII eilutę. Tokiu atveju perduodamų duomenų kiekis smarkiai išaugtu todėl buvo nuspręsta įvesti baitų masyvo lauko tipą.

Koks tipas kokiu atveju naudojamas bus aprašoma pranešimų specifikacijose ir yra žinomas iš anksto taikomajai programai.

2.3 Pranešimo eigos schema

Žemiau pavaizduota bendra pranešimų eiga, kurioje matosi kada koks pranešimas siunčiamas.



5 pav. Bendras pranešimų eiliškumas

Protokolas prasideda klientui siunčiant pasisveikinimo pranešimą. Serveris savo ruožtu siunčia atsaką į pasisveikinimo pranešimą. Bendru atveju po to siunčiami raktų apsikeitimo pranešimai po kurių klientas ir serveris susitaria dėl bendro slapto rakto. Poto siunčiami patikrinimo pranešimai kuriais įsitikinama jog kliento ir serverio suskaičiuoti slapti raktai sutampa. Jeigu raktas gaunamas vienodas abejuose taškuose reiškias autentifikavimas pavyko. Raktų apsikeitimo protokolai užtikrina kad raktai abejuose taškuose gausis vienodi tik tuo atveju jeigu abu taškai turi reikiamą slaptą informaciją kad autentifikavimas pavyktu. Priklausomai nuo raktų apsikeitimo protokolo tai gali būti arba simetrinis raktas arba vartotojo ir slaptažodžio pora. Slaptą raktą rekomenduojama panaudoti kaip pradinį šaltinį, iš kurio būtų paveldimi pasirašymo ir šifravimo atvejai. Tai galėtų būti padaroma pasinaudojant maišos funkcijomis. Toliau priklausomai nuo konkrečios taikomosios programos siunčiami reikalingi pranešimai kad galėtu būti suteikta reikiama paslauga.

2.4 Pranešimų specifikacijos

Pats pirmas pranešimas kurį turėtu siųsti kortelė yra pasisveikinimo pranešimas. Jame turi būti protokolai kuriuos palaiko kortelė taip pat kortelės serijinis numeris arba koks nors unikalus ID. Žemiau pateikiama lentelė su konkrečiais laukų vardais ir reikšmių tipais.

Lentelė Nr. 2 Pasisveikinimo pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Pasisveikinimo pranešimo atveju turi būti „HELLO“.	ASCII eilutė
SERIAL	Kortelės serijinis numeris arba koks kitas unikalus ID. Siunčia tik klientas. Serveris šitos reikšmės nesiunčia.	ASCII eilutė
CIPHERSUITE	Nusako kokius protokolus kortelė palaiko. Jeigu kortelė palaiko daugiau negu vieną protokolą, tuomet protokolai turi būti atskirti kabliataškio simboliu „;“. Galimos reikšmės pateikiamos protokolų lentelėje.	ASCII eilutė

Galimi protokolai išvardijami lentelėje žemiau.

Lentelė Nr. 3 Palaikomi protokolai

Reikšmė	Paaiškinimas
DHEKE_SHA1	Raktų apsikeitimui naudojamas DH-EKE protokolas. SHA-1 maišos funkcija.
SRP_SHA1	Raktų susitarimui naudojamas SRP protokolas. SHA-1 maišos funkcija.
REVE_SHA1	Raktų susitarimui naudojamos atsitiktinės reikšmės, šifruotos kortelės raktu. SHA-1 maišos funkcija.

Serveris, gavęs pranešimą ir patikrinęs ar palaiko nors vieną protokolą atsako pasisveikinimo pranešimu. Po šio pranešimo vyksta raktų susitarimo etapas, kuriame klientas ir serveris siunčia pranešimus kad susitarti dėl bendro slapto rakto.

Serverio pasisveikinimo pranešimas atrodo taip:

Lentelė Nr. 4 Serverio pasisveikinimo pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Turi būti „HELLO“.	ASCII eilutė
CIPHERSUITE	Nusako koks protokolas buvo pasirinktas.	ASCII eilutė

Jeigu serveris nepalaiko nei vieno protokolo iš gautų tuomet privalo išsiusti klaidos pranešimą ir nutraukti ryšį su klientu.
Bet kurioje vietoje įvykus klaidai turi būti siunčiamas klaidos pranešimas ir nutraukiamas ryšys su kita puse. Klaidos pranešimas atrodo taip:

Lentelė Nr. 5 Klaidos pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Turi būti „ERROR“.	ASCII eilutė
TEXT	Klaidos pranešimo tekstas.	ASCII eilutė

Toliau siunčiamų pranešimų seka priklauso nuo to koks protokolas buvo pasirinktas.

2.4.1 DH-EKE protokolas

Toliau detalizuosime atvejį kai buvo pasirinktas DK-EKE raktų apsikeitimo protokolas.
Po serverio pasisveikinimo klientas siunčia raktų apsikeitimo pranešimą, kurio turinys toks:

Lentelė Nr. 6 DH-EKE raktų apsikeitimo pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Turi būti „KE“.	ASCII eilutė
KE	Perduodama Diffie-Hellman užšifruota reikšmė.	Baitų masyvas

Serveris gavęs šia žinutę turi išsiųsti analogišką pranešimą serveriui. Tiek klientas tiek serveris įvykdę šiuos veiksmus turi susiskaičiavę slaptą raktą.

2.4.2 SRP protokolas

Toliau detalizuosime atvejį kai buvo pasirinktas SRP raktų apsikeitimo protokolas. Po serverio pasisveikinimo klientas turi siųsti tokį pranešimą:

Lentelė Nr. 7 SRP kliento rakto apsikeitimo pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Turi būti „SKE“.	ASCII eilutė
I	Kliento identifikatorius arba vartotojo vardas.	ASCII eilutė
KE	SRP protokolo A reikšmė.	Baitų masyvas

Tuo tarpu jeigu serveris randa vartotoją I atsako tokiu pranešimu:

Lentelė Nr. 8 SRP serverio rakto apsikeitimo pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Turi būti „SKE“.	ASCII eilutė
S	SRP protokolo s reikšmė.	Baitų masyvas
KE	SRP protokolo B reikšmė.	Baitų masyvas

Po šių pranešimų klientas ir serveris susiskaičiuoja slaptą raktą.

2.4.3 REVE protokolas

Klientas susigeneruoja atsitiktinę reikšmę a, ją užšifruoja kortelės raktu ir siunčia ją serveriui. Pranešimo turinys:

Lentelė Nr. 9 REVE protokolo pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Turi būti „RVE“.	ASCII eilutė
R	Užšifruota atsitiktinė reikšmė.	Baitų masyvas

Serveris gavęs pranešimą sugeneruoją atsitiktinę reikšmę b, ją užšifruoja kortelės raktu ir siunčia klientui.

Tiek klientas tiek serveris suskaičiuoja slapta raktą pagal formulę H(a,b).

2.4.4 Patikrinimo pranešimas

Autentifikavimas yra baigtas kai įsitikinama jog kliento ir serverio suskaičiuoti raktai sutampa. Tam siunčiami patikrinimo pranešimai. Patikrinimo pranešimai nepriklauso nuo naudojamo protokolo. Pranešimai yra siunčiami po to kai suskaičiuojamas slaptas raktas.

Klientas siunčia tokį pranešimą serveriui:

Lentelė Nr. 10 Kliento patikrinimo pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Turi būti „CVERIFY“.	ASCII eilutė
R	Sugeneruota atsitiktinė reikšmė.	Baitų masyvas
KH	Maišos funkcijos rezultatas nuo slapto rakto.	Baitų masyvas

Serveris gavęs šį pranešimą turi susiskaičiuoti KH reikšmę ir patikrinti ją su gautąja reikšme. Jeigu reikšmės sutampa tada turi išsiusti tokį pranešimą:

Lentelė Nr. 11 Serverio patikrinimo pranešimas

Lauko pavadinimas	Paaiškinimas	Lauko tipas
MT	Nusako pranešimo tipą. Turi būti „CVERIFY“.	ASCII eilutė
KRH	Maišos funkcijos rezultatas nuo slapto rakto ir gautos R reikšmės.	Baitų masyvas

Klientas gavęs šį pranešima turi susiskaičiuoti KRH reikšmę ir pasitikrinti ar jinai atitinka gautą reikšmę. Jeigu reikšmės sutampa reiškia tiek klientas tiek serveris turi vienodą slaptą raktą ir autentifikavimas pavyko sėkmingai.

2.5 Kortelėje saugomi duomenys

Reiktu išskirti visgi kokie duomenys turėtų būti saugomi kortelėje kad minėtasis protokolas veiktu. Taigi DH-EKE protokolo atveju kortelėje saugomi tokie duomenys:

- g, N – generatorius ir pirminis skaičius. Iš esmės šitos reikšmės gali būti saugomos ne kortelės failų sistemoje o įrašyti pačioje kortelės taikomojoje programoje nes pačios reikšmės niekada nesikeičia.
- Kortelės autentifikavimo raktas kuriuo užšifruojamos ir atšifruojamos A, B reikšmės.

SRP protokolo atveju kortelėje turi būti saugomi tokie duomenys:

- Kaip ir DH-EKE atveju g, N reikšmės. Gali būti saugomos pačioje taikomojoje programoje.
- I, p - vartotojo vardas ir slaptažodis.

Paminėti duomenys gali būti papildomai apsaugomi PIN kodu kad priėjimas prie jų būti suteiktas tik pateikus teisingą PIN kodą. Ar papildomai duomenis apsaugoti PIN kodu priklauso nuo pačios teikiamos paslaugos. Pavyzdžiui elektroninio bilieto atveju PIN kodo vedimas sukeltu papildomų nepatogumų, tuo tarpu elektroninės bankininkystės atveju papildoma apsauga butų naudinga. Taip pat galimas ir ribotų PIN kodų spėjimų skaičiaus mechanizmas. Tai yra suvedus kažkiek kartų neteisingą pin kodą kortelė ištrintu saugomus duomenis ir užsiblokuotų.

2.6 Atliekamų skaičiavimų pasiskirstymas

Šiame skyriuje yra aprašoma kuriame komponente (kortelėje ar terminale) kokie skaičiavimai turi būti atliekami. Bus paminimi tik patys svarbiausi skaičiavimai tam protokolui, tokie skaičiavimai kaip žinutės surinkimas, skaitymas nebus minimi. Kokie konkrečiai atliekami skaičiavimai priklauso nuo pasirinkto autentifikavimo metodo, todėl atliekamus skaičiavimus detalizuosime kiekvienam protokolui.

DH-EKE protokolu atveju skaičiavimo veiksmai pasiskirsto taip:

Lentelė Nr. 12 Skaičiavimo pasiskirstymas DH-EKE protokolo atveju

Klientas	Serveris
• Generuojama atsitiktinė reikšmė a • $A = g^a \bmod p$. • A užšifruojamas AES algoritmu. • B atšifruojamas AES algoritmu. • $K = B^a \bmod p$. • Slaptas raktas = SHA1(K)	• Generuojama atsitiktinė reikšmė b • $B = g^b \bmod p$. • B šifruojamas AES algoritmu. • A atšifruojamas AES algoritmu. • $K = A^b \bmod p$. • Slaptas raktas = SHA1(K)

Kaip matome tiek kliento ir serverio skaičiavimai yra vienodai pasiskirstę.

SRP protokolo atveju skaičiavimai yra pasiskirstę taip:

Lentelė Nr. 13 Skaičiavimo pasiskirstymas SRP protokolo atveju

Klientas	Serveris
• Generuojama atsitiktinė reikšmė a. • $A = g^a$. • $S_k = (B - kg^x)^{(a+ux)}$ • Slaptas raktas K = SHA1(S_k)	• Generuojama atsitiktinė reikšmė b. • $B = kv + g^b$. • $S_s = (Av^u)^b$ • Slaptas raktas K = SHA1(S_s)

Matome kad skaičiavimų veiksmų pasiskirstymas nėra toks tolygus kaip DH-EKE protokolo atveju.

REVE atveju atliekami tokie skaičiavimai:

Lentelė Nr. 14 Skaičiavimo pasiskirstymas REVE protokolo atveju

Klientas	Serveris
• Generuojama atsitiktinė reikšmė a. • Šifruojama reikšmė a AES algoritmu. • Atšifruojama reikšmė b. • Slaptas raktas K = SHA1(a,b)	• Generuojama atsitiktinė reikšmė b. • Šifruojama reikšmė b AES algoritmu. • Atšifruojama reikšmė a. • Slaptas raktas K = SHA1(a,b)

Taigi kaip matome skaičiavimo pasiskirstymas yra labai vienodas, kaip ir DH-EKE atveju.

Toliau yra atliekamas patikrinimo procesas, kuris vyksta visuomet nepriklausomai nuo protokolo.

Lentelė Nr. 15 Skaičiavimo pasiskirstymas patikrinimo proceso metu

Klientas	Serveris
• Generuojama atsitiktinė reikšmė r. • Skaičiuojama patikrinimo reikšmė KH = SHA1(K) • Skaičiuojama patikrinimo reikšmė KH = SHA1(K, r)	• Skaičiuojama patikrinimo reikšmė KH = SHA1(K) • Skaičiuojama patikrinimo reikšmė KRH = SHA1(K, r)

2.7 Išvados

1) Sukūrėme autentifikavimo protokolą skirtą autentifikuoti kortelę (vartotoją) į tam tikrą sistemą.
2) Protokolas palaiko 3 raktų apsikeitimo (ir tuo pačiu autentifikavimo) mechanizmus.
3) DH-EKE, SRP ir REVE mechanizmus. DH-EKE ir REVE mechanizmas skirtas kliento autentifikavimui tokioms paslaugoms kaip e. bilietas ar kitoms paslaugoms kur autentifikavimo aplinka veikia be ryšio su centriniu serveriu.

4) SRP mechanizmas skirtas paslaugoms kur paprastai autentifikavimui naudojamas vartotojo vardas ir slaptažodis. Tai gali būti e. bankininkystės ar e. parduotuvės ar kokios kitos elektroninės paslaugos.

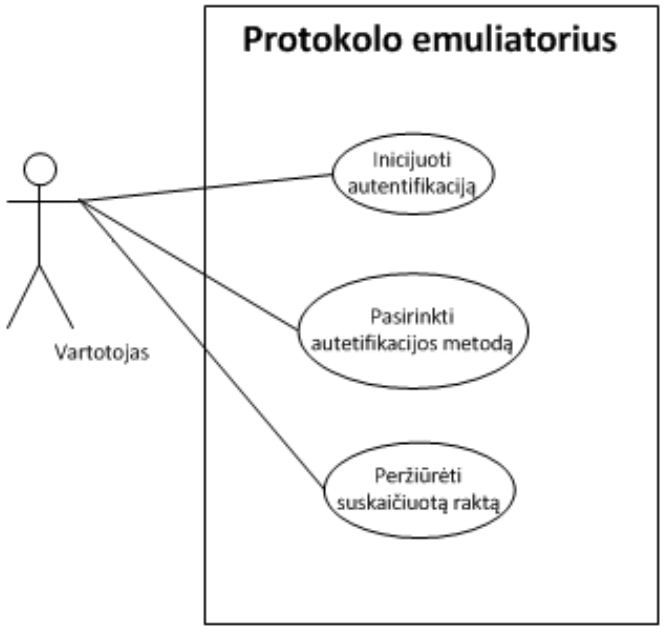
3 PROTOKOLO EMULIATORIAUS PROJEKTAS

3.1 Projekto aprašymas

Šiame skyriuje aprašomas programinė įranga kuri bus naudojama eksperimentui. Eksperimentas yra realizuojamas programine įranga, realios lustinės kortelės ir realūs terminalai nebus naudojami. Architektūros prasme eksperimentas yra kliento-serverio architektūros programinė įranga, kur klientas atitinka lustinę kortelę, o serveris atitinka terminalą ar kitą galinį autentifikavimo tašką. Programinė įranga bus programuojama C# programavimo kalba. Naudosime .NET Framework 4.0 versiją. Naudojamas programavimo įrankis: Microsoft Visual Studio 2010.

3.2 Panaudos atvejai

Iš vartotojo pusės programa nėra labai interaktyvi. Kaip ir realiu atveju, vartotojas gali atlikti tik vieną veiksmą – inicijuoti autentifikavimą. Realiu atveju tai būtų tiesiog įkišimas kortelės į skaitytuvą. Patogumo dėlei vartotojui bus papildomai leidžiama pasirinkti koks autentifikavimo protokolas bus naudojamas.



6 pav. Projekto panaudos atvejai

Taigi iš vartotojo pusės žiūrint programa turi būti labai paprasta. Programos sudėtingumas atsiskleidžia protokolo realizavimo vietose.

3.3 Reikalavimai projektui

Šiame skyriuje išdėstomi reikalavimai kuriamai programinei įrangai. Reikalavimus suskirstyme į dvi grupės: funkcinius ir nefunkcinius reikalavimus.

3.3.1 Nefunkciniai reikalavimai

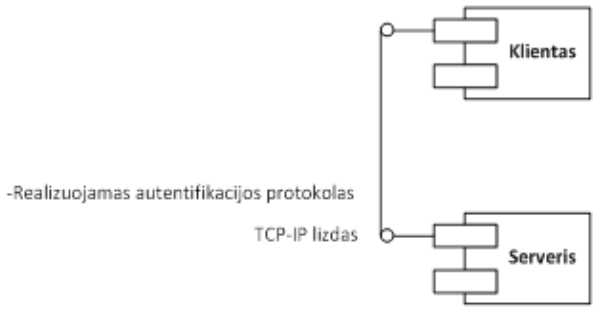
- Programa turi turėti paprastą vartotojo sąsają
- Konfigūruojamų parametrų skaičius turi būti minimalus kad kuo labiau supaprastinti programos naudojimąsi
- Programa neturi pakibti kol vyksta autentifikavimo procesas

3.3.2 Funkciniai reikalavimai

- Vartotojas prieš pradėdamas autentifikavimo procesą turi turėti galimybė pasirinkti koks autentifikavimo protokolas bus naudojamas
- Vartotojas turi turėti galimybę bet kada uždaryti programa net ir nesibaigus autentifikavimo procesui
- Po sėkmingo autentifikavimo ekrane turi matytis suskaičiuoti slapti raktai
- Programa turi realizuoti siūloma protokolą
- Klientui pasirinkus vykdyti autentifikavimą kliento programa turi automatiškai prisijungti prie serverio ir įvykdyti autentifikavimą be vartotojo įsikišimo
- Turi būti skaičiuojamas laikas kiek trunka skaičiavimai ir rezultatas parodomas ekrane

3.4 Projekto architektūrinė apžvalga

Kaip jau buvo minėta programa bus realizuojama kliento-serverio architektūros principu. Klientas mūsų atveju atitiks lustinė kortelę, o serveris – terminalą į kurį autentifikuojamasi. Kadangi klientas turi siųsti ir gauti pranešimus serveriui reikalinga komunikacijos terpė. Kaip komunikacijos terpę panaudosime TCP/IP lizdus (angl. sockets).



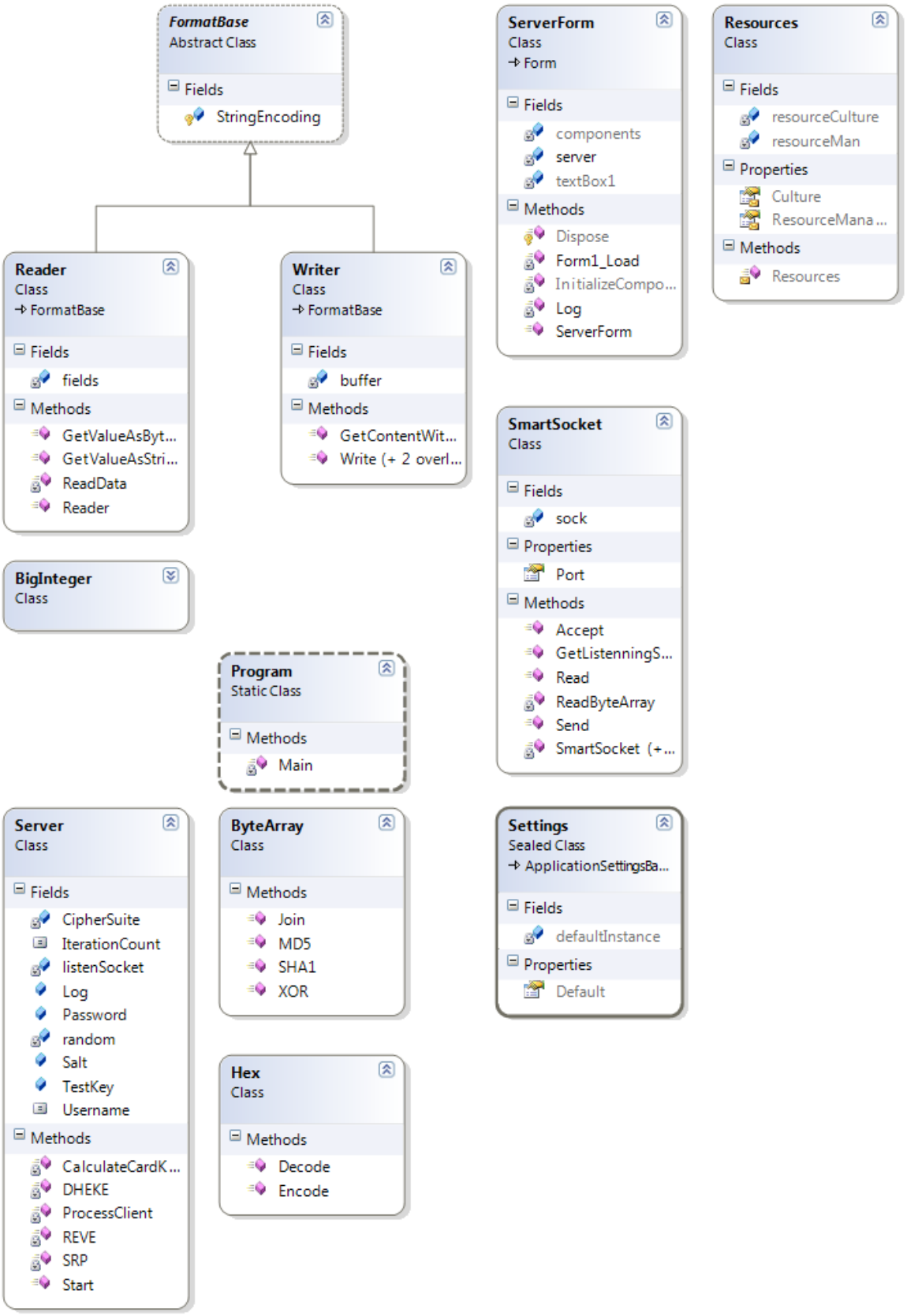
7 pav. Projekto architektūros diagrama

Tiek klientas tiek serveris bus realizuojami kaip grafinės sąsajos programos. Tai leis patogiai peržiūrėti gautus rezultatus (slaptus raktus). Kliento atveju tai pat bus patogu realizuoti protokolo pasirinkimo galimybę.

3.5 Klasių diagramos

Žemiau pateikiamos serverio ir kliento projektų klasių diagramos ir trumpi svarbiausių klasių apibūdinimai.

Serverio klasių diagrama:

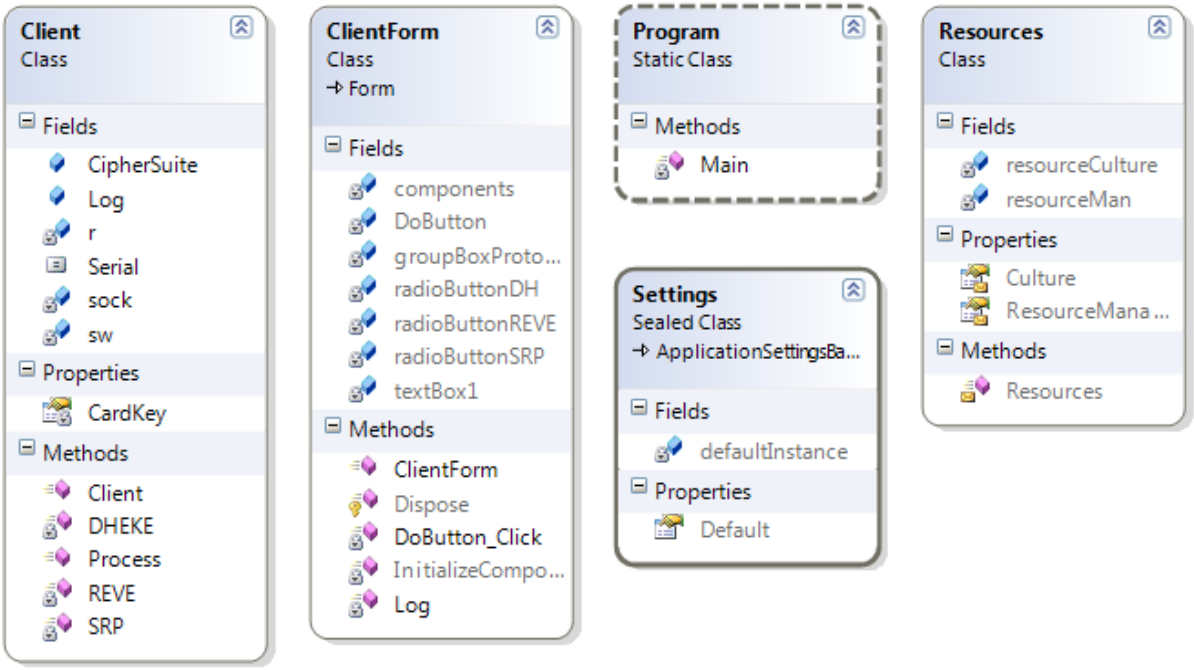


8 pav. Serverio klasių diagrama

Svarbiausių serverio klasių trumpas apibūdinimas:

- FormatBase – bazinė klasė pranešimų fiziniam formavimui.
- Reader – pranešimų laukų skaitymo klasė. Skirta skaityti pranešimų laukų reikšmes.
- Writer – pranešimų formavimo klasė. Skirta formuoti pranešimus atitinkančius protokolą aprašytą protokolą.
- ServerForm – grafinės sąsajos klasė.
- SmartSocet – klasė naudojama komunikacijai realizuoti. Viduje naudoja TCP/IP lizdus.
- Server – serverio logiką realizuojanti klasė. Priima ir apdoroja kliento pranešimus.
- ByteArray – pagalbinė klasė skirta veiksmams su baitų masyvais palengvinti.
- Hex – pagalbinė klasė skirta šešioliktainės sistemos eilutėms formuoti.
- BigInteger – klasė realizuojanti veiksmus su dideliais sveikaisiais skaičiais. Realizacija buvo paimti iš [2] šaltinio. Nors .NET 4.0 versijoje turi analogišką klasę buvo panaudota trečio asmens klasė dėl papildomų patogių metodų kuriuos turi ši klasė.

Kliento klasių diagrama:



9 pav. Kliento klasių diagrama

Kadangi kliento projektas panaudoja ir tas pačias klases kaip ir serverio projektas tai diagramoje matomos tik klientui unikalios klasės. Tai yra klasės kurios rodomos serverio klasių diagramoje čia nėra rodomos.

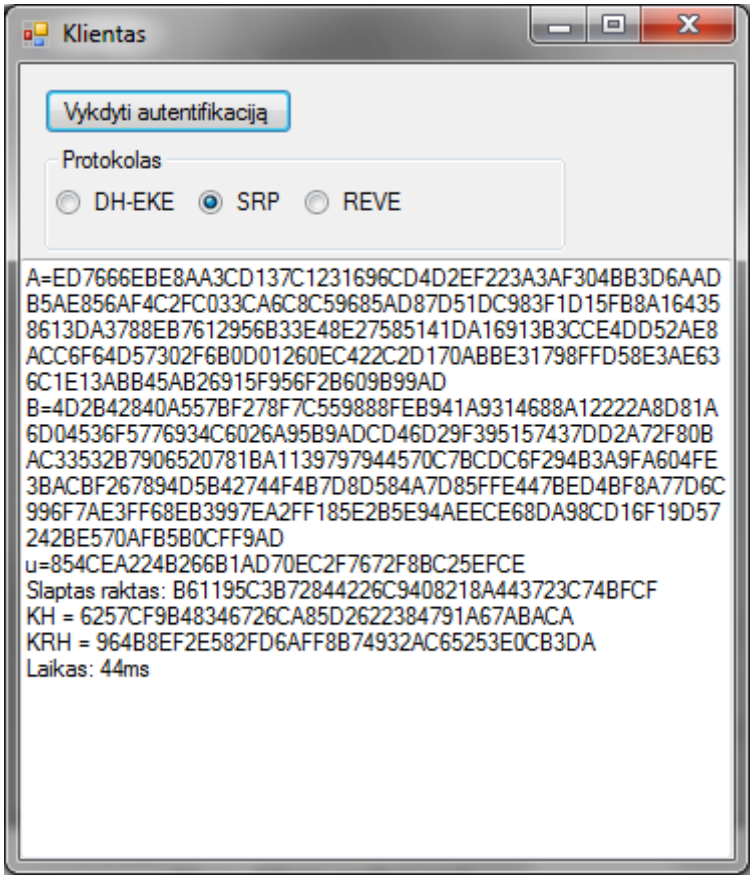
Trumpas svarbiausių klasių apibūdinimas:

- Client – kliento logiką realizuojanti klasė.
- ClientForm – kliento grafinę sąsają realizuojanti klasė.

3.6 Grafinės sąsajos

Kaip jau buvo minėta tiek kliento tiek serverio programos buvo realizuojamas kaip grafinės sąsajos. Toks sprendimas buvo pasirinktas todėl kad būtų galima paprastai ir patogiai atvaizduoti rezultatus ekrane (slaptus raktus). Taip pat panaudojant grafines sąsajas lengvai realizuojamas protokolo pasirinkimas klientui.

Žemiau pateikiama kliento grafinė sąsaja ir trumpi paaiškinimai:

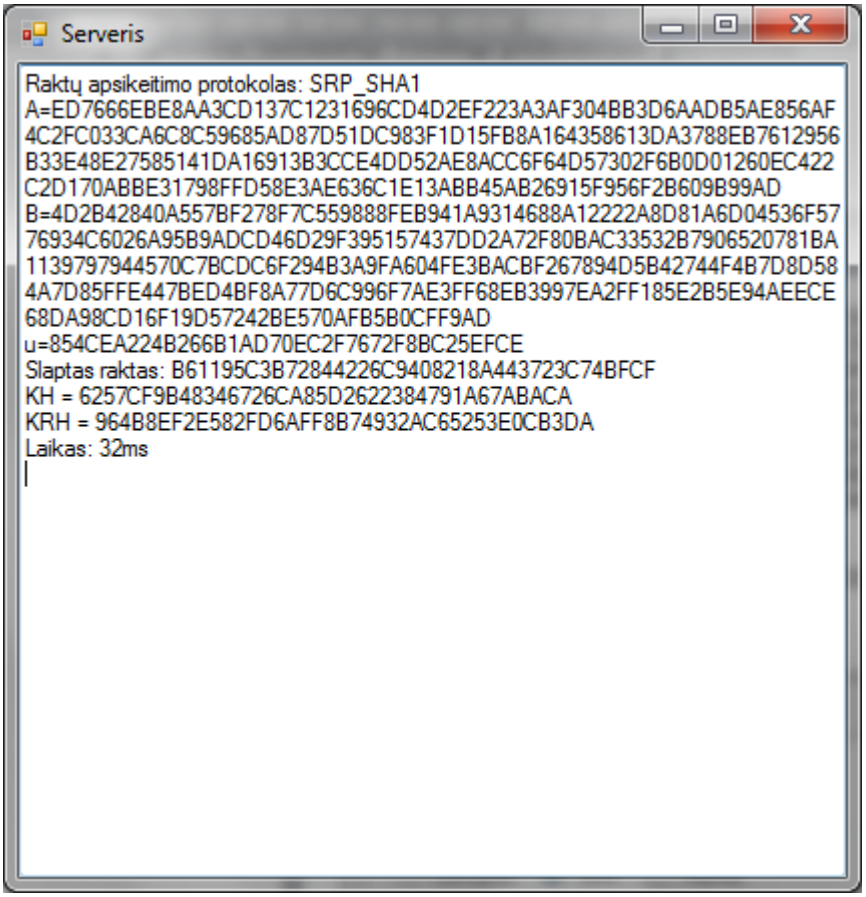


10 pav. Kliento grafinė sąsaja

Taigi kaip matome kliento vartotojo sąsaja pagrinde sudaro trys komponentai: autentifikavimo iniciavimo mygtukas, protokolo pasirinkimo ir vieta teksto išvedimui. Teksto

išvedimo vietoje parodomas slaptas raktas po sėkmingo autentifikavimo ir kiti tarpiniai kriptografiniai kintamieji būdingi pasirinktam protokolui.

Serverio grafinė sąsaja ir trumpi paaiškinimai:



11 pav. Serverio grafinė sąsaja

Serverio grafinė sąsaja susideda tik iš teksto erdvės kurioje rašoma koks protokolas bus naudojamas raktų apsikeitimui, kintamieji konkrečiam protokolui ir slaptas raktas.

Smulkesnė naudojimo instrukcija pateikiama vartotojo dokumentacijos skyriuje.

3.7 Realizavimo detalės

Šiame skyriuje pateikiamos realizavimo detalės kurios nebuvo paminėtos praeituose skyriuose.

- Tik paleidus serverį jis pereina į klausimosi režimą ir klausosi 3333 TCP/IP prievado. Jokio mygtuko specialiai spausti nereikia jog tai būtų daroma. Jeigu paleidimo metu prievadas yra užimtas kitos programos yra metamas klaidos

pranešimas ir serveris nepereina i klausymosi režimą. Tuomet reikia uždaryti programą kuri užėmė prievadą ir serverį paleisti iš naujo.

- Paspaudus autentifikavimo iniciacijos mygtuką programa automatiškai jungiasi į vietinio kompiuterio 3333 TCP/IP prievadą. Taigi kad sėkmingai įvyktu autentifikavimas turi būti paleista serverio programa. Jeigu mygtuko paspaudimo metu nebus paleista serverio programa bus metamas klaidos pranešimas apie nepavykusį ryšio užmezgimą.
- Serveris naudoja po vieną giją kiekvienai kliento užmegztai sesijai apdoroti. Tai leidžia serveriui vienu metu aptarnauti daugiau negu viena klientą.
- Raktai ir tarpinės reikšmės kurios matomos sąsajose yra pateikiamos šešioliktainėje sistemoje.
- Visiems kriptografiniams veiksmams atlikti buvo naudotas generatorius g=2.
- Visiems kriptografiniams veiksmams atlikti buvo naudotas 1024 bitų ilgio pirminis skaičius N = 00C037C37588B4329887E61C2DA3324B1BA4B81A63F9748FED2D8A410C2FC21B1232F0D3BFA024276CFD88448197AAE486A63BFCA7B8BF7754DFB327C7201F6FD17FD7FD74158BD31CE772C9F5F8AB584548A99A759B5A2C0532162B7B6218E8F142BCE2C30D7784689A483E095E701618437913A8C39C3DD0D4CA3C500B885FE3. Skaičius pateikiamas šešioliktainėje formoje.
- DH-EKE atveju buvo naudojamas 128 bitų testinis raktas K = { 0, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115 }. Kortelės serijinis numeris „123“. Kortelės testinis raktas buvo gautas sujungus testinį raktą su serijos numeriu ir paimta MD5 maišos funkcijos rezultatas.
- SRP atveju buvo naudotas vartotojas vardas „TestUsername“ ir slaptažodis „Slaptazodis“. Kaip kriptografinė druska buvo naudojamas baitų masyvas S = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 }.
- Visi minėti raktai, vartotojų vardai ir slaptažodžiai yra įrašyti į pačia programą, jų nereikia niekur konfigūruoti.

3.8 Vartotojo dokumentacija

3.8.1 Sistemos reikalavimai

Pagrindinis reikalavimas yra sistema kuri turi suinstaliuotą ir veikiantį .NET Framework 4.0 paketą.

Šiuo metu ant šių operacinių sistemų veikia .NET Framework 4.0:

- Windows 7
- Windows Server 2003
- Windows Server 2008
- Windows Vista
- Windows XP

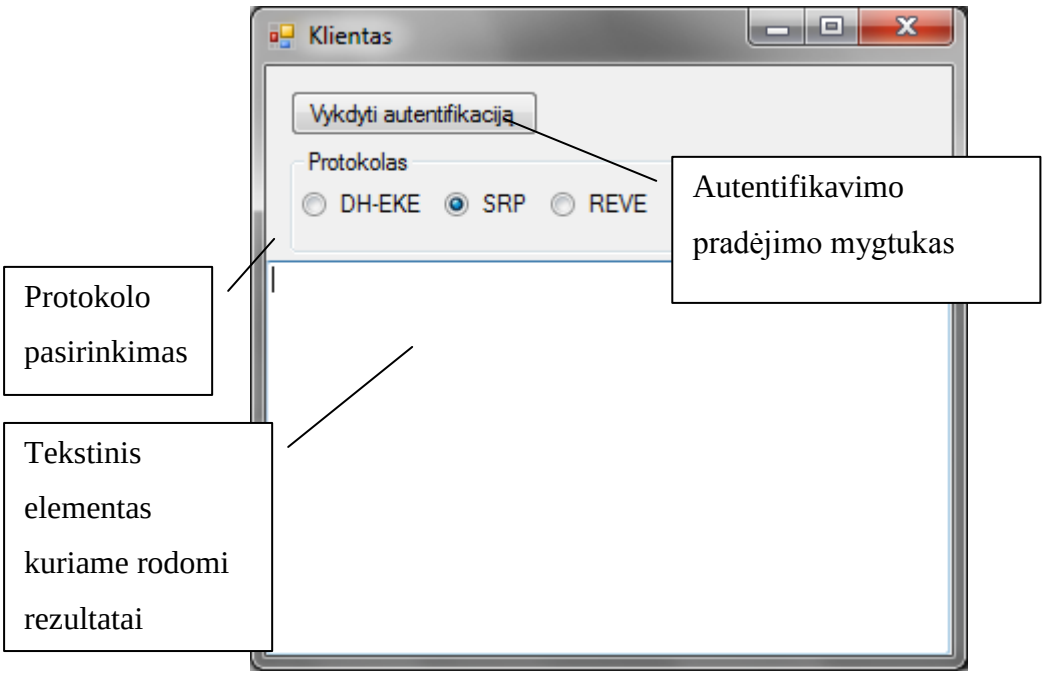
Kadangi aparatūros pajėgumai neturi įtakos programinės įrangos rezultatams, o tik jų vykdymo laikui tai specialių reikalavimų aparatūrinei įrangai nėra. Visgi kad veiktu pats .NET Framework kompiuteris turi turėti tokius arba geresnius parametrus:

- 1 GHz Pentium arba greitesnis
- 512 MB RAM atmintis arba daugiau

3.8.2 Sistemos naudojimosi gidas

Norint naudotis sistema pirmiausia reikia paleisti serveri. Serveris paleidžiamus server.exe failą. Paleidus serveri matomas serverio langas, kuriame bus rodomi visi pranešimai vartotojui. Tame pačiame lange ir bus rodomi rezultatai – suskaičiuotas slaptas raktas serverio pusėje.

Paleidus serverį toliau turėtų būti paleidžiama kliento programa. Tai galima padaryti paleidus client.exe failą. Paleidus klientą pirmiausia reikėtų pasirinkti kokį autentifikavimo protokolą naudosime. Po to spausti autentifikavimo pradėjimo mygtuką.



12 pav. Kliento vartotojo sąsaja su paaiškinimais

Po sėkmingo autentifikavimo lange matomas slaptas raktas kuris buvo suskaičiuotas kliento pusėje. Jeigu autentifikavimas sėkminga tai serverio ir kliento slapti raktai turi sutapti.

3.9 Išvados

1) Protokolo emuliatoriaus projektui buvo suformuluoti reikalavimai, ir suprojektuota programinė įranga atitinkanti keliamus reikalavimus.
2) Projektas realizuojamas kliento ir serverio architektūros programine įranga. Klientas atitinka lustinę kortelę, serveris – terminalą.
3) Pateikta klasių diagrama, grafinės sąsajos vaizdai su paaiškinimais.
4) Parašytas vartotojo gidas, kuriame smulkiai išdėstoma kaip naudotis sukurta programine įranga.

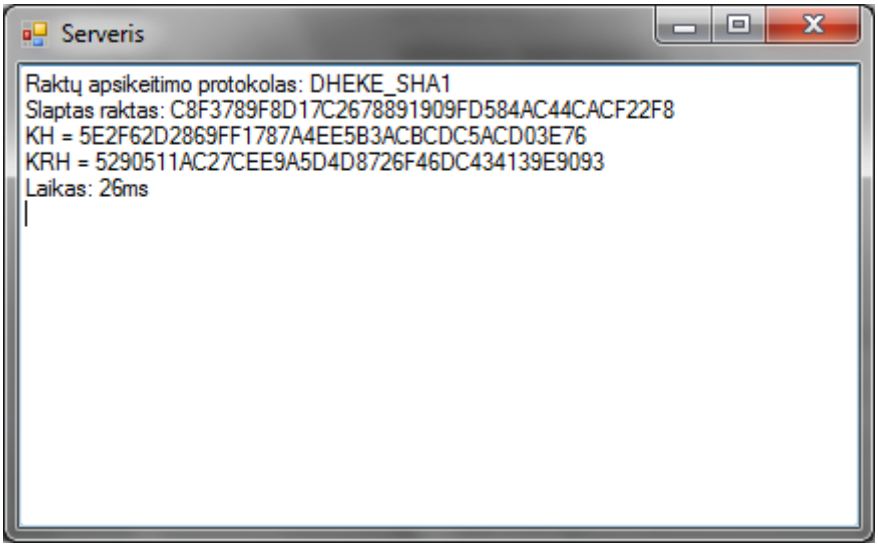
4 PROTOKOLO TYRIMAS

4.1 Sistemos veikimo tyrimas

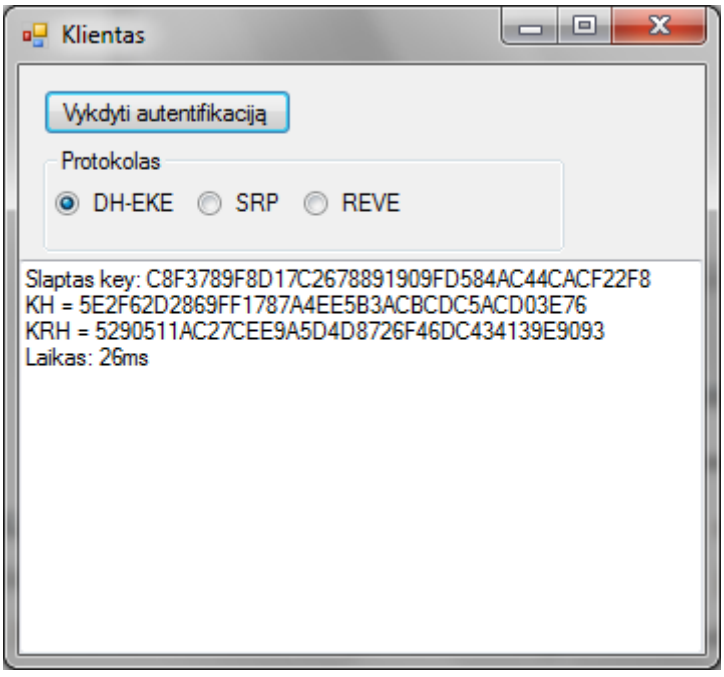
Šio eksperimento esmė yra patikrinti ar pasiūlyto protokolas tikrai veikia. Tuo pačiu įsitikinti ar realizacija yra tikrai teisinga. Eksperimentas bus vykdomas taip: paleidus serverį ir klientą bus bandoma autentifikuoti klientą ir po to bus lyginamos slaptų raktų reikšmės abejuose galuose. Jeigu raktai visais atvejais gaunami vienodi reiškias protokolas ir jo

realizacija yra teisingai veikianti. Kadangi raktai yra išvedami į ekraną tai visur kur reikia pateiksime paveiksliukus kur bus matomi slapti raktai. Tuomet vizualiai sutikrinę raktų reikšmes matysime ar raktai sutampa ar ne. Taip pat turi sutapti KH ir KRH reikšmės abejuose galuose.

Pirmiausia tirsime DH-EKE protokolo veikimą. Žemiau pateikiami rezultatai iš tyrimo.

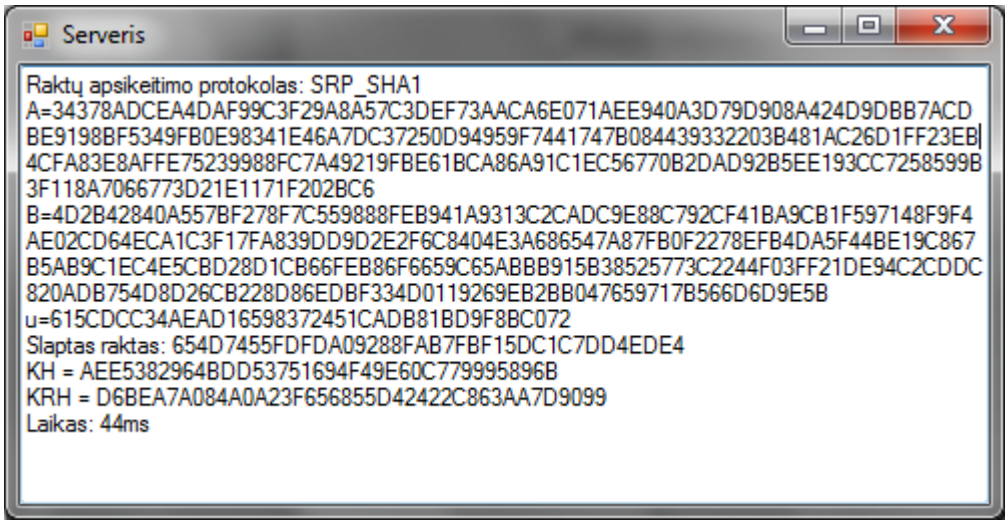


13 pav. Serverio tyrimo rezultatai DH-EKE protokolui

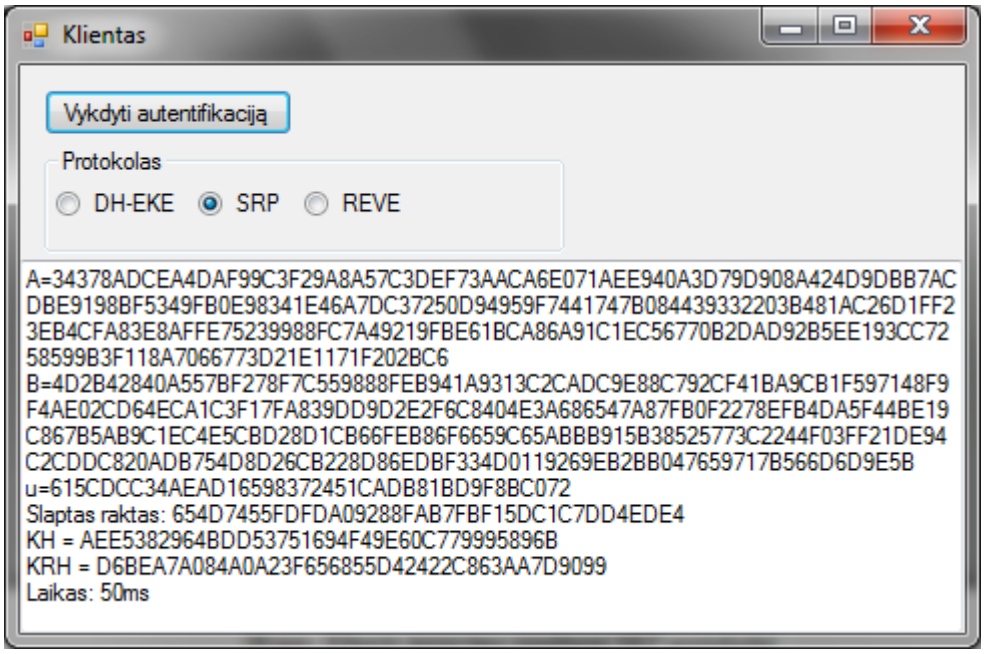


14 pav. Kliento tyrimo rezultatai DH-EKE protokolui

Kaip matome iš paveikslėlių tiek kliento tiek serverio slaptas raktas yra „99AB348D6BA2E3935B337C361A65B23B769D36E3“ ir jis yra vienodas abiejuose galuose. KH ir KRH reikšmės taip pat sutampa abejuose galuose. Protokolas DH-EKE atveju veikia teisingai. Toliau tokiu pačiu principu tirsime SRP protokolo veikimą.



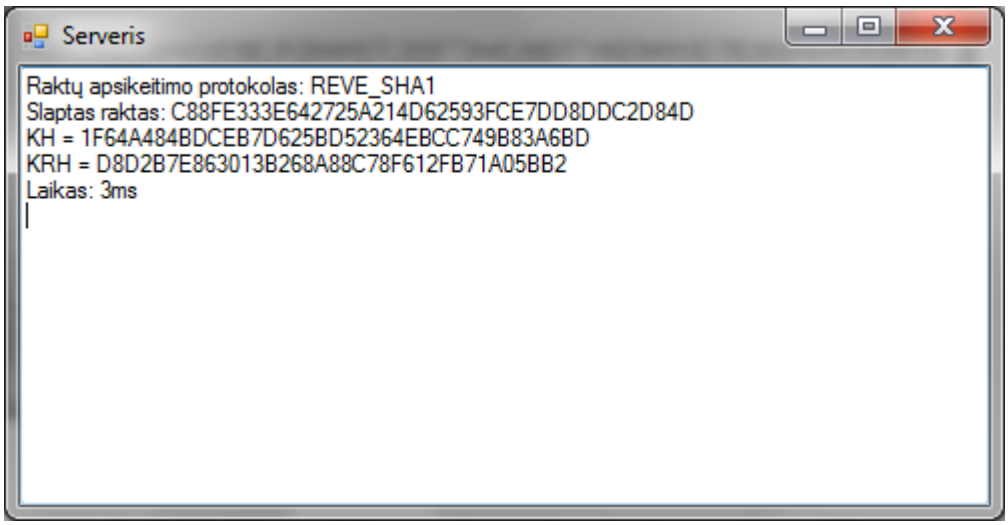
15 pav. Serverio tyrimo rezultatai SRP protokolui



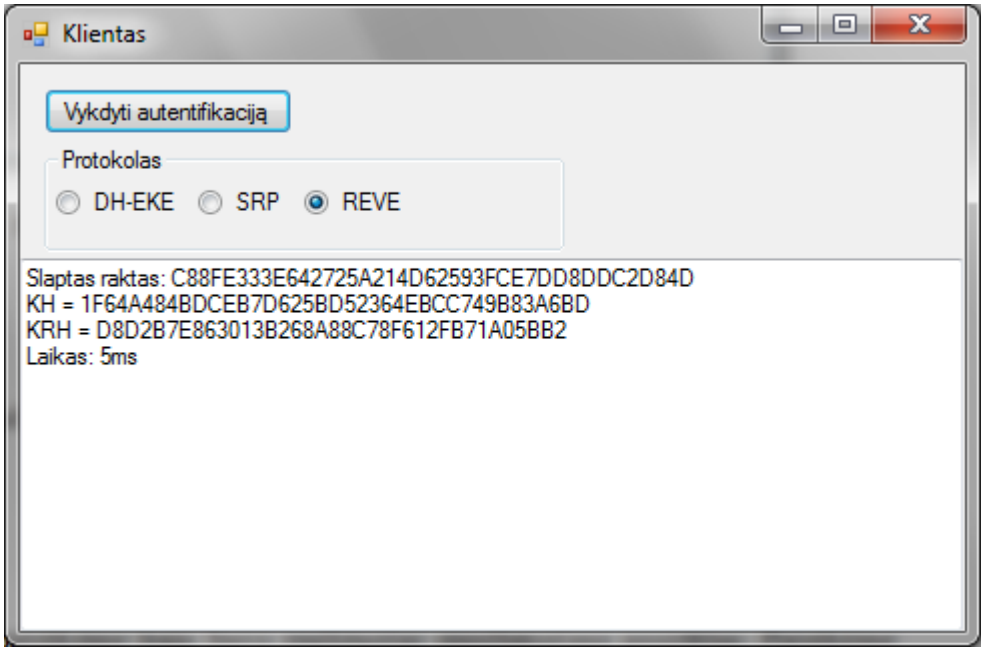
16 pav. Kliento tyrimo rezultatai SRP protokolui

Matome kad tiek kliento tiek serverio tarpinės ir slapto rakto reikšmės sutampa ir yra vienodos. Šio tyrimo metu buvo gautas slaptas raktas „51AEA095DA64801BBEB6C4AA88531B368B28D3“.

Toliau tirsime REVE protokolo darbą.



17 pav. Serverio tyrimo rezultatai REVE protokolui



18 pav. Kliento tyrimo rezultatai REVE protokolui

Gautas slaptas raktas „C88FE333E642725A214D62593FCE7DD8DDC2D84D“ abejose pusėse sutampa. KH ir KRH reikšmės serverio ir kliento pusėse vėl sutampa. Matome kad protokolas dirba teisingai.

Taigi visais atvejais protokolas ir realizacija veikia teisingai.

4.2 Protokolo funkcinių pranašumų apibūdinimas

Šiame skyriuje aptarsime kokius funkcinius pranašumus suteikia protokolo naudojimas.

Pagrindinis protokolo pranašumas yra tai kad klientas ir serveris dinamiškai susitaria dėl raktų apsikeitimo protokolo. Galimi įvairūs scenarijai kada toks funkcionalumas gali būti labai naudingas, keletą jų aptarsime smulkiau.

Tarkime protokolas naudojamas ilgą laiką. Natūralu jog einant laikui senuose protokoluose randama saugumo spragų, o nauji sukurti protokolai šioms spragoms būna atsparūs. Protokolas yra sukurtas taip, jog kortelė pateikia kokiais protokolais moka susitarti dėl sesijos raktų o serveris priima sprendimą koki protokolą naudoti. Jeigu kortelė moka kelis protokolus tai serveris gali tiesiog teikti pirmenybę saugesniam protokolui, kad butų visada naudojamas saugesnis protokolas. Taip pat svarbu atkreipti dėmesį į tai jog protokolas sukurtas taip, kad įdiegti naują raktų apsikeitimo protokolą nesugriaunant jau veikiančių protokolų yra gan paprasta. Paprasčiausiai naujam protokolui reikia sutekti naują protokolo identifikatorių ir jį realizuoti kliento ir serverio pusėje.

Kitas scenarijus galėtų būti toks: įsivaizduokime paslaugą, į kurios centrinį autentifikavimo serverį vienu metu autentifikuojasi labai daug klientų. Kadangi serveris valdo koks protokolas bus naudojamas autentifikavimui, jis gali didesnės apkrovos metu naudoti autentifikavimo protokolą kuris mažiau apkrauna sistemą bet nėra toks saugus. Kai serverio apkrova sumažėja serveris gali nuspręsti naudoti saugesnį protokolą. Kai kuriais atvejais toks sprendimo būdas gali būti geresnis negu atsisakyti aptarnauti klientus dėl per didelės serverio apkrovos.

Dar kitas scenarijus: išrandamas naujas algoritmas, kuris yra ne tik saugesnis už prieš tai buvusius, bet jo vykdymo laikas yra trumpesnis. Tokiu atveju serveryje realizavus šį algoritmą galima palaipsniui pereiti prie naujo algoritmo naudojimo nenutraukiant palaikymo senoms kortelėms kurios neturi naujo algoritmo realizacijos.

Taigi kaip matome dinamiškas protokolo pasirinkimas gali turėti daug teigiamų savybių, bet konkretus jų taikymas priklauso nuo konkrečios paslaugos tipo, aplinkos ar kitų specifinių su ta e. paslauga susijusių veiksnių.

4.3 Protokolų saugumo savybių palyginimas

Šiame skyriuje trumpai aptarsime kokius saugumo pranašumus turi kiekvienas protokolas. Pradėsime nuo mažiausiai saugaus iš visų protokolų – REVE. REVE protokolo saugumas pagrįstas slaptu simetriniu raktu, kuris saugomas lustinėje kortelėje. Terminalo pusėje kortelės raktas suskaičiuojamas iš pagrindinio rakto ir kortelės serijinio numerio. Tokio protokolo trūkumas yra tas, kad jeigu terminalo raktas butų atskleistas kenkėjas galėtų lengvai padirbti daug kortelių. Kitas protokolo saugumo trūkumas yra tas kad atskleidus

kortelės arba terminalo raktą visi buvę ir būsimi sesijos raktai nėra saugūs, t.y. juos galima suskaičiuoti. Pavyzdžiui jeigu kenkėjas įrašinėjo komunikaciją tarp kortelės ir terminalo, ir po kiek laiko sužinojo kortelės arba terminalo raktą, jis gali be didelio vargo ir suskaičiuoti slaptą kiekvienos sesijos raktą ir tuo pačiu atšifruoti ir pamatyti visą komunikaciją.

Kiek saugesnis yra DH-EKE protokolas. Nors jis ir turi tą patį trūkumą kad atskleidus kortelės ar terminalo raktą galima padirbinėti korteles, bet neturi to trūkumo kad galima pamatyti buvusios komunikacijos turinį. Ši savybė kriptografijoje vadinama išankstine paslaptimi (angl. forward secrecy). Taip yra todėl kad norint sužinoti konkrečios sesijos raktą reikia žinoti ir privačias reikšmes a ir b, kurios niekur nėra saugomos.

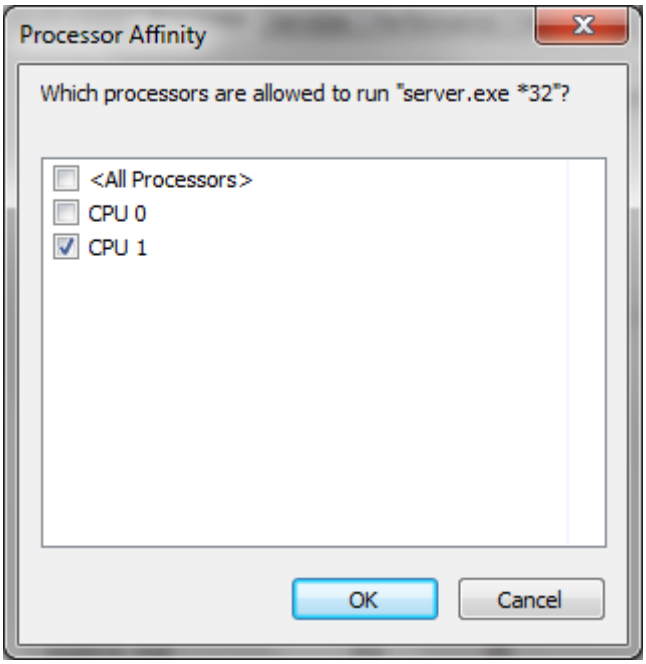
Iš visų trijų protokolų yra saugiausias SRP protokolas. Jis turi netik minėtą DH-EKE išankstinės paslapties savybę, bet ir papildomą saugumo savybę. Net ir pavogus terminalo slaptažodžio patikrinimo reikšmę v kenkėjui vis tiek nepavyks ištraukti iš šios reikšmės slaptažodžio kad kortelė galėtų būti padirbta. Netgi ir pats serveris protokolo veikimo metu nežino ir negali sužinoti koks yra vartotojo slaptažodis, jis gali tik patikrinti ar vartotojas tikrai žino tikrą slaptažodį. Tokia protokolo savybė gali praversti tokiais atvejais kai serveris yra užkrėstas piktavališka programine įranga kuri skanuoja kompiuterio darbinę atmintį ir perduoda jos kopiją kenkėjui. Jeigu butų naudojamas DH-EKE ar REVE protokolas tokiu atveju kenkėjas galėtu surasti koks raktas yra naudojamas kortelei autentifikuoti ir sėkmingai padirbti kortelę. Bet naudojant SRP protokolą tokia ataka nepavyktu.

4.4 Protokolo greitaveikos tyrimas

Įsitikinę jog protokolas veikia teisingai toliau darysime protokolo greitaveikos tyrimą. Šiame tyrime matuosime protokolo vykdymo laiką kiekvienam raktų apsikeitimo metodui. Norint tiksliau išmatuoti vykdymo laikymą padarysime tokius pakeitimus programinėje įrangoje:

- Tarpinių reikšmių ir slapto rakto vaizdavimas į ekraną bus išjungtas. Taip nusprendėme padaryti todėl jog buvo pastebėta jog išvedimas į ekraną užima daug laiko ir gali pastebimai iškraipyti matavimo rezultatus.
- Paspaudus autentifikavimo pradėjimo mygtuką autentifikavimas bus atliekamas ne vieną kartą, bet 1000 kartų. Kai autentifikavimas pasibaigs bus parodoma kiek laiko buvo truko autentifikavimo procesas. Laikas matuojamas milisekundžių tikslumu.

Padarę šiuos pakeitimus galėsime išmatuoti vidutinį autentifikavimo greitį kiekvienam protokolui. Visus matavimus atliksime du kartus, nes pirmą kartą matuojant laiką suveikia .NET Framework realaus laiko kompiliavimo iš IL į mašininį kodą procesas, kuris savaime užima laiko ir iškraipo rezultatus. Norint to išvengti imsime tik antro matavimo rezultatus. Kadangi tiek serverio, tiek kliento matavimo laikas yra praktiškai identiški (taip ir tikėtasi), tai imsime kliento matavimo laiką. Matavimas yra atliekamas ant vienos fizinės mašinos, kuri turi 2 procesoriaus branduolius. Dėl šios priežasties kiekvienai programai uždėsime apribojimą kad jinai naudotų tik vieną ir skirtingus branduolius. Taip darome norint priartinti tyrimo sąlygas prie lustinės kortelės aplinkos (lustinės kortelės turi tik vieną procesorių) ir norint kad serverio ir kliento veikimas neturėtu vienas kitam įtakos. Šį apribojimą galima nustatyti per užduočių menedžerio (task manager). Žemiau pateikiamas paveikslas kuriame matome kad serverio procesui nustatomas priėjimas tik prie vieno procesoriaus branduolio.



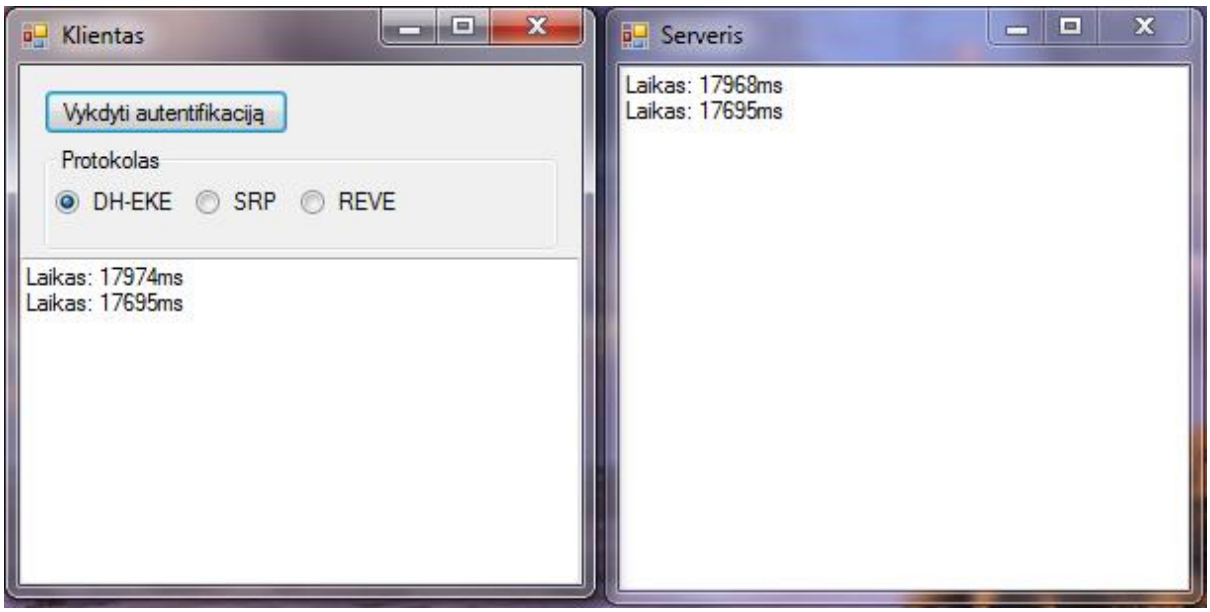
19 pav. Programoms prieinamų procesoriaus branduolių nustatymas

Kompiuterio, ant kurio buvo atliekamas matavimas, parametrai:

- Darbinė atmintis: 2x2 GB DDR2 800 MHz RAM
- Procesorius: Intel e6300 1.8 GHz, 2 fiziniai branduoliai
- Operacinė sistema: Windows 7 64-bit

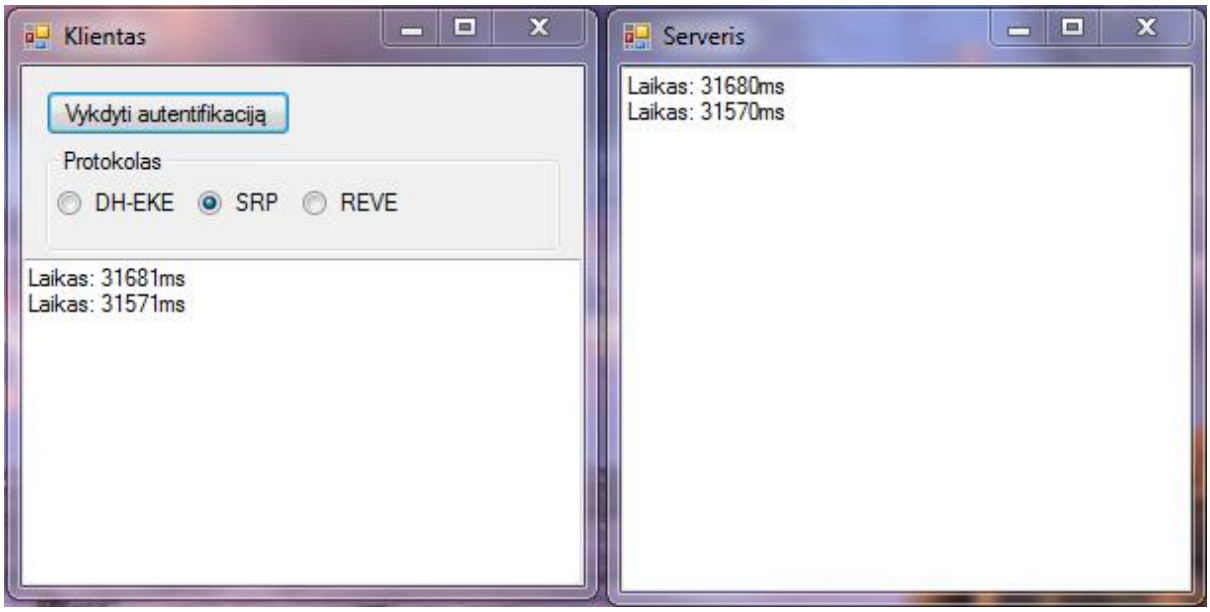
Taigi, atliktų matavimų ekrano vaizdai pateikiami žemiau.

DH-EKE protokolo matavimo rezultatai:



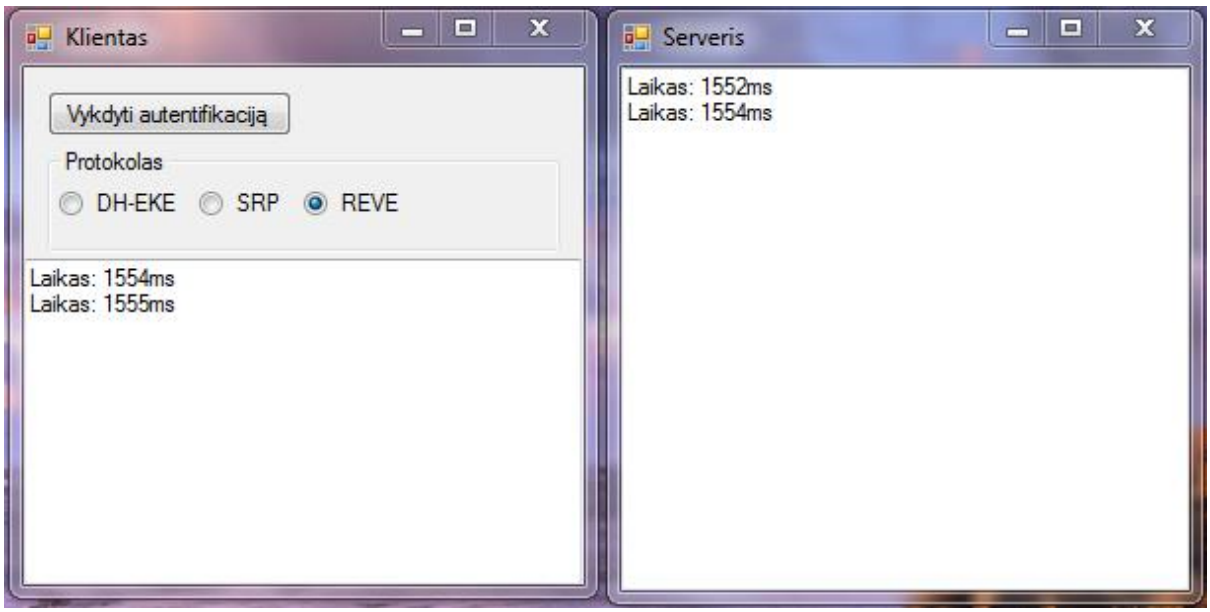
20 pav. DH-EKE protokolo greitaveikos matavimo rezultatai

SRP protokolo matavimo rezultatai:



21 pav. SRP protokolo greitaveikos matavimo rezultatai

REVE protokolo matavimo rezultatai:



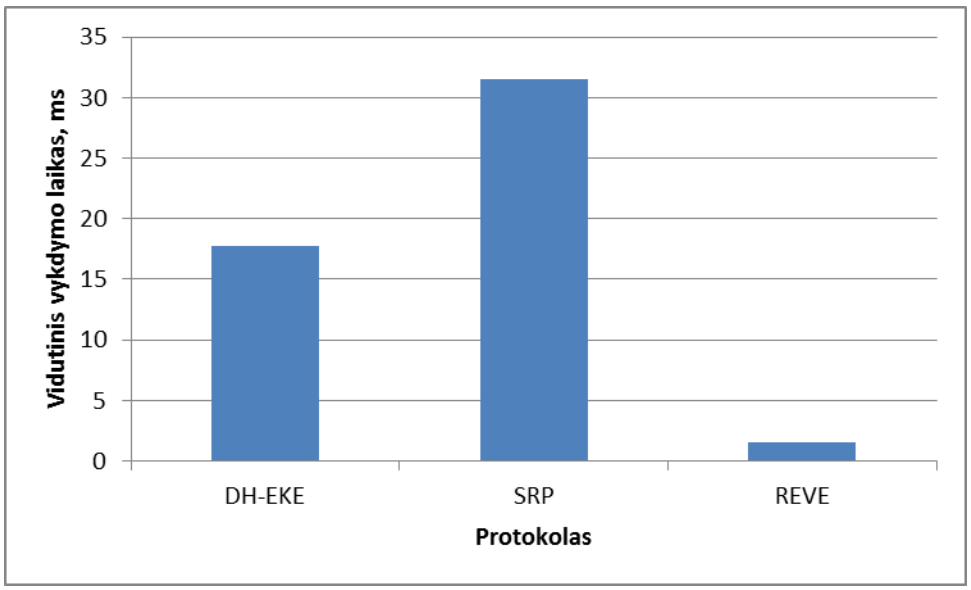
22 pav. REVE protokolo greitaveikos matavimo rezultatai

Kaip ir tikėjomės serverio ir kliento laikas beveik identiškas. Kaip minėjome imsime tik kliento laiką. Žemiau pateikiama rezultatų lentelė.

Lentelė Nr. 16 Protokolo greitaveikos rezultatų lentelė

Protokolas	Bendras vykdymo laikas, ms	Matavimų skaičius	Vidutinis vienos operacijos laikas, ms
DH-EKE	17695	1000	17,70
SRP	31571	1000	31,57
REVE	1555	1000	1,56

Žemiau pateikiama rezultatų atvaizdavimas grafike:



23 pav. Protokolo greitaveikos rezultatų grafikas

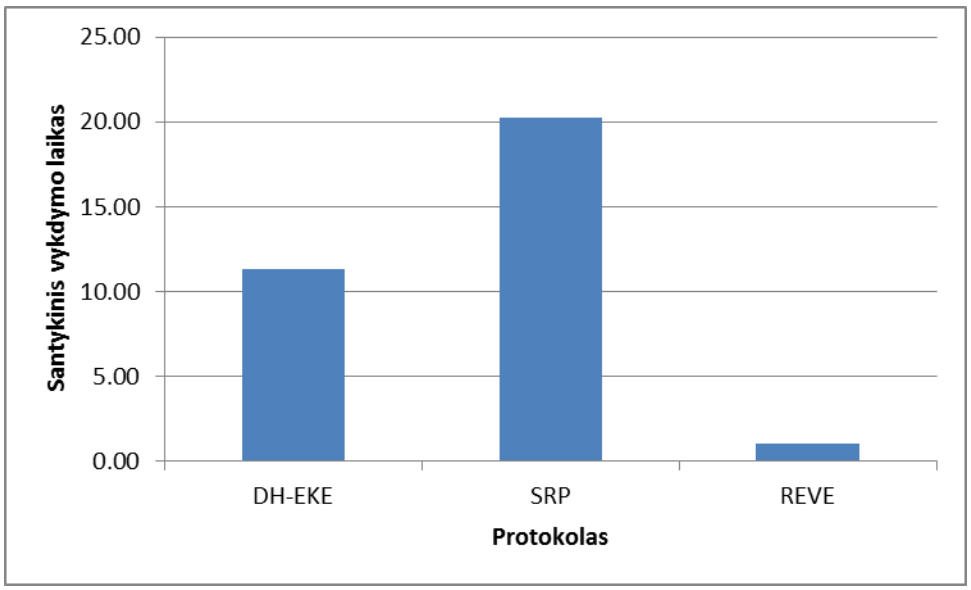
Pats absoliutus dydis (laikas) gal nėra tiek svarbus kiek santykinis vykdymo laikas. Santykinis vykdymo laikas gaunamas visus matavimo laikus padalinus iš mažiausio matavimo laiko reikšmės.

Taigi santykinis vykdymo laikas parodo kiek kartų kiekvienas protokolas yra lėtesnis už REVE protokolą.

Lentelė Nr. 17 Santykinis protokolų vykdymo laikas

Protokolas	Santykinis vykdymo laikas
DH-EKE	11,35
SRP	20,24
REVE	1

Žemiau pateikiamas grafinis santykinio vykdymo laikas.



Lentelė Nr. 18 Santykinis protokolo vykdymo laikas

Taigi iš visų protokolų SRP yra pats lėčiausias, jis už REVE protokolą yra lėtesnis net 20 kartų. Nors DH-EKE protokolas yra beveik dvigubai greitesnis už SRP protokolą jis vis tiek yra 11 kartų lėtesnis už REVE protokolą. Matome kad daugiau saugumo savybių turintys protokolai yra lėtesni, taigi papildomos saugumo savybės lėtina protokolo darbą.

4.5 Išvados

1) Atlikę eksperimentą įsitikinome jog autentifikavimo protokolas tikrai veikia ir suskaičiuojami raktai yra vienodi kliento ir serverio pusėse.
2) Aptarėme skirtingų raktų apsikeitimo saugumo savybės. Mūsų atveju daugiausiai saugumo savybių turi SRP protokolas, mažiausiai – REVE.
3) Aptarėme kokiais atvejais gali praversti protokolo savybė kad raktų apsikeitimo protokolas pasirenkamas dinamiškai.
4) Atlikome protokolų greitaveikos matavimus. Pamatėme kad protokolų greitaveika labai skiriasi, kuo daugiau protokolas turi saugumo savybių tuo jis lėtesnis. Taigi konkrečiai situacijai renkantis protokolą reikėtų atsižvelgti į protokolo greitaveiką ir jo teikiamas saugumo savybes.

5 IŠVADOS

1) Analizės metu buvo išnagrinėtos lustinės kortelės galimybės, autentifikavimo protokolai, saugaus ryšio TLS technologija. Pasinaudojant analizėje sukaupta informacija buvo pasiūlytas autentifikavimo protokolas.
2) Pasiūlytas autentifikavimo protokolas gali autentifikuoti vartotoją panaudojant vieną iš trijų palaikomų autentifikavimo protokolų. Du protokolai autentifikavimui naudoja simetrinį raktą, vienas – vartotojo vardą ir slaptažodį. Visais atvejais vartotojo autentifikavimo duomenys saugomi lustinėje kortelėje.
3) Protokolo emuliatoriaus projekte buvo detalizuota kaip bus realizuojamas protokolo emuliatoriaus. Protokolo emuliatoriaus bus realizuotas programine įranga. Kortelę atitiks kliento programinė įranga. Terminalo įrangą atitiks serverio programinė įranga.
4) Buvo pagamintas protokolo emuliatoriaus kurio dėka buvo išbandytas protokolo veikimas. Atlikę eksperimentą įsitikinome kad protokolas veikia.
5) Tyrimo metu atskleidėme kokiomis saugumo savybėmis pasižymi naudojami raktų apsikeitimo protokolai.
6) Protokolo greitaveikos tyrimas parodė kad protokolo pasirinkimas turi didžiulę įtaką greitaveikai. Kuo protokolas turi daugiau saugumo savybių tuo jis yra lėtesnis. Taigi renkantis koks protokolas turėtų būti naudojamas autentifikavimui reikėtų pagalvoti ar tikrai reikia tokių saugumo savybių kurias suteikia protokolas, gal galima naudoti mažiau saugų bet greitesnį protokolą.

6 LITERATŪRA

1 Marc Witteman. Advances in Smartcard Security, 2002 [žiūrėta 2009-12-25]. Prieiga per internetą: http://www.riscure.com/fileadmin/images/Docs/ISB0707MW.pdf

2 Chew Keong TAN. C# BigInteger Class, 2002 [žiūrėta 2010-03-15]. Prieiga per internetą: http://www.codeproject.com/KB/cs/biginteger.aspx

3 Calypso Networks Association. Calypso functional specification Card Application, 2010 [žiūrėta 2010-12-25]. Prieiga per internetą: http://www.calypsotechnology.net/index.php?option=com_docman&task=doc_download&gid=3&Itemid=40

4 F. Levy. Calypso functional specification SAM and Key management, 2010 [žiūrėta 2010-12-26]. Prieiga per internetą: http://www.calypsotechnology.net/index.php?option=com_docman&task=doc_download&gid=77&Itemid=40

5 Calypso Networks Association. Calypso Handbook, 2010 [žiūrėta 2012-12-26]. Prieiga per internetą: http://www.calypsotechnology.net/index.php?option=com_docman&task=doc_download&gid=77&Itemid=40

6 S. Murdoch, S. Drimer, R. Anderson, M. Bond. Chip and PIN is broken, 2010 [žiūrėta 2010-03-26]. Prieiga per internetą: www.cl.cam.ac.uk/~sjm217/papers/oakland10chipbroken.pdf

7 S. Bellovin, M. Merritt. Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks, 1992 [žiūrėta 2009-10-12]. Prieiga per internetą: http://www.cse.iitm.ac.in/~ragav/btp/pake/1.pdf

8 Ph. Proust, J.P. Tual, L. Sourgen, F. Germain. High Security SmartCards, 2004 [žiūrėta 2009-12-27].

Prieiga per internetą:

http://www.computer.org/portal/web/csdl/doi/10.1109/DATE.2004.1268853

9 MasterCard. History of the Card Payment System, [žiūrėta 2009-12-23].

Prieiga per internetą:

http://www.mastercard.com/us/company/en/docs/history%20of%20payments.pdf

10 Andrew Patrick. Human Factors of Security Systems: A Brief Review, 2002 [žiūrėta 2009-12-26].

Prieiga per internetą: http://www.andrewpatrick.ca/passwords/passwords.pdf

11 nCipher. NCIPHER netHSM Technical architecture, 2004 [žiūrėta 2009-12-28].

Prieiga per internetą: http://www.asiapeak.com/download/nethsm_arch_issue_1.pdf

12 Y. Oren, A. Wool. RFID-Based electric voting: What could possibly go wrong?, 2010 [žiūrėta 2010-03-15].

Prieiga per internetą: http://iss.oy.ne.ro/e-Voting-RFID-Relay-IEEE.pd

13 Smart card alliance. Smart card FAQ. [žiūrėta 2009-12-27].

Prieiga per internetą: http://www.smartcardalliance.org/pages/smart-cards-faq

14 K. Mayes, K. Markantonakis. Smart Cards, Tokens, Security and Applications. London: Sprinter, 2007. 392 p.

15 David P. Jablon. Strong Password-Only Authenticated Key Exchange, 1996 [žiūrėta 2009-12-25].

Prieiga per internetą:

http://grouper.ieee.org/groups/1363/passwdPK/contributions/jablon.pdf

16 T. Dierks, E. Rescorla. The Transport Layer Security (TLS) Protocol, 2008 [žiūrėta 2009-12-12].

Prieiga per internetą: http://tools.ietf.org/html/rfc5246

17 D. Taylor, T. Wu, N. Mavrogiannopoulos, T. Perrin. Using the Secure Remote Password (SRP) Protocol for TLS Authentication, 2007 [žiūrėta 2009-12-25].

Prieiga per internetą: http://tools.ietf.org/html/rfc5054#section-2.5.3

7 SANTRUMPŲ ŽODYNAS

RSA – (angl. RSA) asimetrinio šifravimo algoritmas

DESX (angl. Data Encryption Standart X) patobulintas DES algoritmas

TDES – (angl. Triple DES) trigubas DES, šifravimo algoritmas

DES – (angl. Data Encryption Standart) duomenų šifravimo standartas

RFID – (angl. Radio Frequency ID) komunikacijos technologija

NFC – (angl. Near field communications) artimo lauko komunikacijos technologija

HSM – (angl. Hardware security module) aparatinis saugumo modulis

ROM - (angl. Read only memory) atmintis iš kurios galima tik skaityti

EMV – (angl. Eurocard MasterCard Visa) kortelių leidėjų grupė

SSL - (angl. Secure socket layer) saugaus ryšio technologija

TLS – (angl. Transport layer security) saugaus ryšio technologija

SAM – (angl. Secure Application Module) taikomosios programos saugos modulis