KAUNO TECHNOLOGIJOS UNIVERSITETAS

FUNDAMENTALIŲJŲ MOKSLŲ FAKULTETAS

TAIKOMOSIOS MATEMATIKOS KATEDRA

Tadas Pikutis

E-dokumentų valdymo sistemos, panaudojant grupinius parašus ir Petri tinklus kūrimas ir tyrimas

Magistro darbas

Vadovas

prof. dr. E. Sakalauskas

KAUNAS, 2011

KAUNO TECHNOLOGIJOS UNIVERSITETAS

FUNDAMENTALIŲJŲ MOKSLŲ FAKULTETAS

TAIKOMOSIOS MATEMATIKOS KATEDRA

TVIRTINU
Katedros vedėjas
doc. dr. N. Listopadskis

E-dokumentų valdymo sistemos, panaudojant grupinius parašus ir Petri tinklus kūrimas ir tyrimas

Taikomosios matematikos magistro baigiamasis darbas

Vadovas
prof. dr. E. Sakalauskas
2010 06 01

Recenzentas
SBA IT Skyriaus
direktorius Sigitas Patalauskas
2010 06 01

Atliko
FMMM-9 gr. stud.
T. Pikutis
2010 05 30

KAUNAS, 2011

Kvalifikacinė komisija

Pirmininkas: Leonas Saulis, profesorius (VGTU)

Sekretorius: Eimutis Valakevičius, docentas (KTU)

Nariai: Algimantas Jonas Aksomaitis, profesorius (KTU)

Vytautas Janilionis, docentas (KTU)

Vidmantas Povilas Pekarskas, profesorius (KTU)

Rimantas Rudzkis, habil. dr., vyriausiasis analitikas (DnB NORD Bankas)

Zenonas Navickas, profesorius (KTU)

Arūnas Barauskas, dr., vice-prezidentas projektams (UAB „Baltic Amadeus“)

Santrauka

Šiame darbe pasiūlyta schema, paremta Petri tinklų principais ir elektroniniu grupiniu parašu, leidžianti optimizuoti e-dokumentų judėjimą. Realizavus schemą platformoje SharePoint 2010 ir atlikus tyrimą gauta, jog dokumentai realizavus schemą savo kelią įveikė vidutiniškai viena valanda greičiau. Pateikta elektroninio grupinio parašo saugumo analizė, po kurios nustatyti sistemos saugumo parametrai.

Summary

In this paper i proposed a scheme, based on characteristics of Petri nets and digital group signature, which was implemented on SharePoint 2010. Results showed that solution increased speed of document flow in SharePoint 2010 for averagely one hour per document. Analysis of digital signature scheme indicated what system parameters should be used for secure communications, and how digital signature should be integrated in document management flow.

Turinys

Lentelių sąrašas

Paveikslėlių sąrašas

Įvadas

Šiuolaikinėse organizacijose dėl tobulėjančių technologinių sprendimų atsiranda galimybė perkelti dokumentais paremtus įmonės procesus į kompiuterines sistemas. Siekiama, kad kompiuteriu sukurti e-dokumentai pilnai pakeistu įprastinius dokumentus taip paspartinant įvairius įmonių procesus. Stebėdamos šią tendenciją, didžiosios IT korporacijos siūlo įvairius programinius sprendimus ir platformas, leidžiančias įmonės darbuotojams keistis dokumentais, atitinkančiais įmonės bei valstybės keliamus standartus. Tačiau, šios platformos neturi tam tikrų sprendimų, galinčių optimizuoti tokius procesus. Kadangi įmonėje, kurioje dirbu, buvo nutarta pradėti vystyti e-dokumentų ir įvairių HR procesų sistemą Microsoft SharePoint 2010 platformoje, ir ši užduotis buvo paskirta man, nusprendžiau panagrinėti standartinį platformos funkcionalumą ir esant galimybei jį pagerinti.

Darbo tikslas, kurio teorinis sprendimas ir praktinės realizacijos rezultatai pateikiami šiame darbe, sudarytas iš trijų etapų:

1. efektyvios e-dokumentų sistemos sukūrimas ir ištyrimas,
2. grupinio elektroninio parašo schemos sudarymas ir saugumo tyrimas,
3. aprašytos DVS realizacija SharePoint 2010 platformoje ir jos efektyvumo tyrimas.

Kadangi e-dokumentų sistema įmonėje bus realizuojama SharePoint 2010 platformoje, pateiksiu trumpą sistemos funkcionalumo, sudariusio galimybę realizuoti darbe aprašomus sprendimus, apžvalgą, sprendimą, kaip optimizuoti standartinius sistemos darbo eigos procesus, bei aprašysiu teorinį elektroninio parašo modelį, kurio pagrindu ketinu kurti elektroninio parašo infrastruktūrą įmonėje.

1. Teorinė dalis

1.1 Pagrindiniai terminai, sąvokos ir apibrėžimai

Kadangi tolimesniuose skyriuose kalbėsiu apie sprendimus ir jų realizacijas SharePoint 2010 (arba SP 2010) platformoje, pirmiausiai pristatysiu pagrindines šios platformos technines sąvokas.

Laukas (column) – duomenų tipo vienetas, aprašomas ir registruojamas sistemoje. Laukas gali būti standartinių tipu (skaičius, eilutė, loginis) arba išvestinių tipų (Rich HTML, HTML, bulk) ir t.t. Laukas aprašomas kartą, ir naudojamas kaip šablonas įvairiose sistemos konstrukcijose. Laukų aprašymui ir inicializavimui naudojama MS Visual Studio 2010 programavimo aplinka.

Sąrašas (list) – įvairių laukų rinkinys, skirtas duomenims, prieinamiems tam tikroms asmenų grupėms arba vidinėms aplikacijoms, rinkti ir saugoti. Sąrašai tam tikra prasme primena duomenų bazių lenteles, tačiau tai kur kas sudėtingesnės konstrukcijos su prieigų teisių rinkiniais ir kitu papildomu SP funkcionalumu. Sąrašų aprašymui galima naudoti SP priemones, tokias kaip SP dizaineris arba SP tinklines programas.

Bibliotekos (libraries) – išplėstinė sąrašo struktūra, sudaryta iš laukų, leidžianti saugoti sudėtingus duomenų rinkinius, pvz MS Office dokumentus, ir su jais susijusią informaciją. Bibliotekos turi savo teisių rinkinius skirtus ir pačiai bibliotekai ir jos turiniui. Būtent šias bibliotekos savybes naudosime kaip pagrindinį DVS konstrukcijos elementą. Bibliotekų aprašymui galima naudoti SP priemones, tokias kaip SP dizaineris arba SP tinklines programas.

Šablonai – šabloniniai Office paketo programų failai su integruotais SharePoint laukais. Laukai importuojami SharePoint aplinkoje į dokumentų šablonus, taigi, pildant dokumento šabloną, susietųjų laukų reikšmės perduodamos į SharePoint sąrašus, taip sudarant galimybę praplėsti meta žymas bei išskirti esminę informaciją dokumente.

Darbų eigos (workflows) – vidinės vartotojų kuriamos platformos programos, galinčios panaudoti visus platformos resursus ir atlikti su jais veiksmus. Darbo eigas naudosime modelio, kurį sudarysime SharePoint 2010 dokumentų srautams optimizuoti, realizacijai.

Servisai (services) – sudėtingos standartinės (kartais ir vartotojų) sukurtos programinės sistemos, skirtos tam tikriems veiksmams platformoje atlikti. Nuo darbo eigų servisai skiriasi ne tik sudėtingumo laipsniu, bet ir vykdymo vieta. Darbo eigos vykdomos platformoje, o servisai veikia serverio lygmenyje, todėl jie gali naudoti ne tik platformos, bet ir serverio arba tinklo resursus savo funkcijoms atlikti.

Domeno vartotojų duomenų bazė arba AD (Activ Directory) – pagrindinė įmonės prieigų ir

teisių duomenų bazė, saugoma domeno serveryje. Joje aprašomos darbuotojų teisės visame tinkle. Ad yra pagrindinis SharePoint vartotojų duomenų šaltinis.

Tolimesniuose skyriuose taip pat bus kalbama apie elektroninio parašo modelį, todėl pateikiu bazinę teoriją modelio konstrukcijoms paaiškinti.

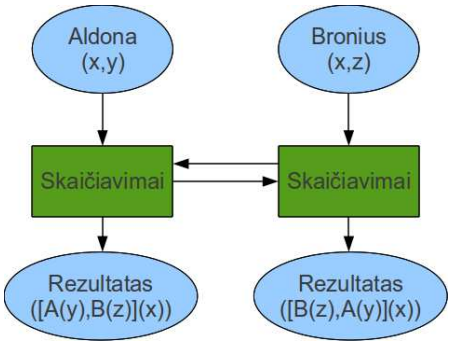
Algoritmų klasifikacija pagal sprendimo laiką.

Algoritmas tai skaičiavimo procedūra kuriai perduodami įvesties duomenys ir gaunami rezultatai. Sudėtingumo teorijoje uždaviniai dažniausiai klasifikuojami pagal efektyviausius žinomus algoritmus jiems spręsti. Algoritmų efektyvumas matuojamas atsižvelgiant į resursus (pvz. laiką, atminties dydį), kuriuos algoritmas naudoja uždaviniui išspręsti. Įveskime funkciją *O(g(n))*, kurios rezultatas laikas (pasirinktais laiko vienetais), per kurį tam tikras algoritmas išsprendžia uždavinį, o *g(n)* – algoritmo įvesties duomenų dydžio *n* funkcija. Tada algoritmus pagal sprendimo laiką suklasifikuosime tokiu būdu:

1. Polinominio laiko – algoritmai, kurie blogiausios baigties atveju uždavinio sprendimui sunaudoja $O\left(n^k\right)$laiko vienetų (čia *k* – tam tikra konstanta).
2. Eksponentinio laiko – algoritmai, kurie blogiausios baigties atveju uždavinio sprendimui sunaudoja $O\left(c^n\right)$laiko vienetų (čia *c* – tam tikra konstanta didesnė už 1).
3. Subeksponentinio laiko - algoritmai, kurie blogiausios baigties atveju uždavinio sprendimui sunaudoja $O\left(\exp\left(c+o(1)*n^{\alpha}*\ln(\alpha)^{1-\alpha}\right)\right)$laiko vienetų (čia c – tam tikra teigiama konstanta, konstanta $0<\alpha<1$, o $o(1)=O(g(n))\quad kai\quad \lim_{n\to\infty} O(g(n))=0$).

Interaktyvus protokolas.

Interaktyvus protokolas gali būti traktuojami kaip žaidimas tarp 2 žaidėju, pvz. Aldonos ir Broniaus, kuriame jie vienas kitam siuntinėja tam tikras žinutes atlikdami žingsnius (skaičiavimus) aprašytus protokole. Įvykdžius protokolą abi šalys gauna tam tikrus rezultatus. Taigi interaktyvus protokolas tai algoritmų *A* ir *B* pora (A,B) skirtų 2 komunikuojančioms pusėms, Aldonai ir Broniui. Panagrinėkime schemą, pavaizduotą paveikslėlyje 1.1.1



Paveikslėlis 1.1.1

Kaip matome, kiekviena šalis turi bendrą įvesties parametrą *x* ir privačiuosius įvesties parametrus *y* ir *z*. Aldonos protokolo rezultate matome, kad rezultatui gauti algoritmas B naudoja Broniaus privačiąją reikšmę, o Broniaus rezultatui gauti algoritmas A naudoja Aldonos privačiąją reikšmę. Visada egzistuoja tikimybė, kad skaičiavimo per komunikaciją gali atsirasti trečioji šalis Zita, kuri gali panaudoti panašų į *B* algoritmą *B'* (jie panašūs savo atsakais į algoritmo A užklausas) ir apgauti Aldoną. Tai vadinama klastojimu (forgery).

Norint išanalizuoti ir suprasti kai kurias savybes, kurios išryškėja bendraujant su Aldona per algoritmą A šiose protokoluose dažnai veikia trečioji šalis, vadinama Valdytoju, kuri, skirtingai nuo Broniaus ar Zitos bet kada turi galimybę nutraukti arba pradėti iš naujo A algoritmą bet kuriuo metu. Toks komunikacijos tipas tarp Aldonos ir Valdytojo vadinamas „Išminčiaus prieiga“ (oracle access).

Grupė

Tarkime, *S* – netuščia aibė, o * - dvinarė operacija, atvaizduojanti S aibės elementų sandaugą į *S* aibės elementą, t.y. $SxS \to S$. $\forall a,b \in S,\quad a*b$ reiškia operacijos * pritaikytos elementams *a* ir *b* rezultatą. Operacija * yra komutatyvi ir asociatyvi operacija aibės S elementams. Elementas $e \in S$ vadinamas vienetiniu, jei teisingi sąryšiai $e*a = a*e = a \quad \forall a \in S$. Atvirkštiniu elementu $b \in S$ elementui $a \in S$ vadinamas toks elementas b, kuriam teisinga $a*b = b*a = e$. Grupe

Apibrėžimas 1.1.1

*Grupe vadinama aibė G su asociatyvia dvinare operacija *. Operacijai * egzistuoja vienetinis elementas, ir bet kuriam G elementui egzistuoja atvirkštinis elementas * operacijos atžvilgiu. Jei operacija * yra komutatyvi, grupė vadinama Abelio grupe.*

Grupę *G* vadinsime cikline, jei ji turi generatorių, kurį pažymėsime *g*, t.y. Jei $g \in G, \quad \forall a \in G \quad teisinga \quad g^x = a, \quad kai \quad x \in Z$.

Grupės Z_m ir Z_m^*.

Z_m tai sveikųjų skaičių, kurių modulis *m* aibė. Ši aibė kartu su sudėties operacija sudaro *m* eilės Abelio grupę.

Z_m^* - grupė, kartu su sandaugos operacija sudaryta iš elementų, pirminių m atžvilgiu ir mažesnių už *m*. Šios grupės eilė apskaičiuojama naudojant Oilerio f-ją: $\varphi(m) = |\{k \mid 1 \ \ k \ \ m \quad ir \quad dbd(k,m) = 1\}|$.

1.2 Trumpa sistemos standartinio funkcionalumo apžvalga

SharePoint 2010 standartinis funkcionalumas dokumentų valdymo srityje ženkliai lenkia SharePoint 2007. Šioje versijoje standartinei DVS sistemai skirta labai daug dėmesio. Smarkiai praplėstos indeksavimo ir paieškos dokumentų kontekste galimybės modernizavus indeksavimo servisus ir meta žymų sinchronizacija. Išplėtotas Bussines Conectivity servisas. Atsirado galimybės ypatingai greitai sinchronizuoti dokumentus vartotojo kompiuteryje su dokumentų bibliotekomis serveryje naudojant Office 2010 paketo programą SharePoint Workspace. Sukurti svetainių dokumentų keitimuisi šablonai, bei dokumentų šablonai bibliotekoms (įskaitant ir aplanko šabloną). Tai, žinoma svarbūs ir vertingi patobulinimai tačiau, standartiniame funkcionalume vis dar nenumatyta gavėju paskirti grupę, išskirti svarbius ar gražinamus dokumentus. Vis dar neatsižvelgiama į resursų, skirtų dokumentų apdorojimui būseną (laisvas ar užimtas), todėl standartinės darbo eigos veikia gana „bukai“. Dokumentai siunčiami tik vienam nurodytam gavėjui, nepaisant to jog gavėjų gali būti ir keli, statomi į eilę pagal gavimo datą o ne pagal svarbą ar būseną. Taigi apžvelgus standartinį funkcionalumą tampa aišku, kad SP 2010 platforma turi galimybes ir įrankius patobulinti ir kurti naujas darbo eigas taip, kad dokumentų apdorojimo laikas būtų trumpesnis. Toliau aprašysime kaip tai padaryti naudojantis minėtu funkcionalumu.

1.3 Petri tinklų savybės, naudojamos sudarant schemą

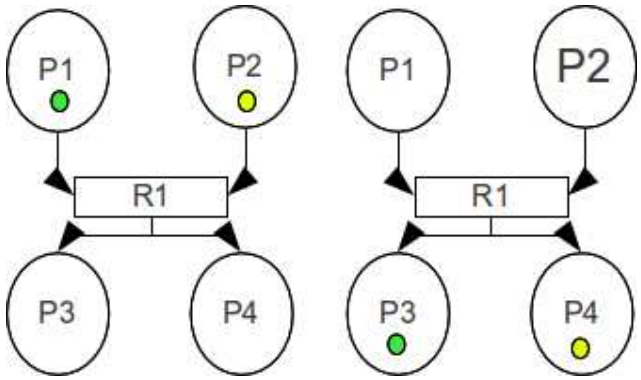
1.2 skyriuje kalbėjome apie tai, jog sudarant dokumentų cirkuliacijos skyrių viduje schemą būtina atsižvelgti į resursų prieinamumą užduočiai atlikti. Šiame skyriuje pateiksiu formalius Petri tinklo apibrėžimus ir savybes, kuriomis naudodamasis sudarysiu schemos algoritmą.

Apibrėžimas 1.3.1

Petri tinklas – dviejų tipų mazgus turintis orientuotasis grafas. Petri tinklų schemose rutuliukais žymimos pozicijos, o stačiakampiais – pareigos. Pozicijas ir pareigas jungia orientuotos grafo

briaunos. Ženklinimui į pozicijas įvedami tam tikri požymiai, vadinami žetonais ir žymimi rutuliukais.

Panagrinėkime paprastą tinklo fragmento pavyzdį, pateiktą paveikslėlyje 1.5.1

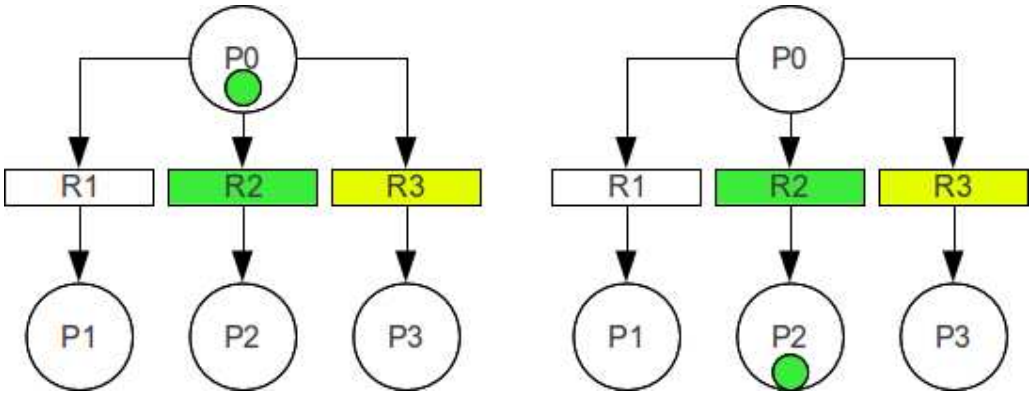


Paveikslėlis 1.3.1

Matome, kad pozicijoje P1 ir P2 yra po vieną žetoną. Kadangi pareigos R1 įėjimo pozicijoje yra po vieną žetoną, žetonai sėkmingai įveikia pareigą ir patenka į pozicijas P3 ir P4. Jeigu bent vienoje iš šių pozicijų žetono nebūtų, pareiga nebūtų įveikta.

Matome, jog ši savybė puikiai atitinka užduoties poreikį – įvertinti resurso prieinamumą. Jei žalią žetoną įsivaizduotumėme kaip dokumentą, o geltoną – kaip žmogų, kuris apdorotų dokumentą, galėtumėm pareigą R1 įvardinti kaip „Žmogus imasi apdoroti dokumentą“. Jeigu pozicijoje P1 būtų keli žetonai (dokumentai) o pozicijoje P2 žetonų (žmonių, galinčių dirbti) nebūtų, pareiga R1 nebūtų įveikiama, t.y. dokumentas nebūtų paimtas apdoroti!

Taigi, sudarydami schemą resursus darbui (žmones) ir apdorojimo objektus (dokumentus) žymėsime žetonais. Mūsų atveju dokumentai laikui bėgant keisis, t.y. schemos darbo eigoje keisis tam tikros jų savybės, nurodančios dokumento kelią tiek skyriaus viduje, tiek tarp skyrių. Gali nutikti ir taip, jog dokumente radus klaidą jį teks gražinti prieš tai jį apdorojusiam asmeniui korekcijoms, arba jei dokumentas bus svarbus jį teks pastumti aukštyn eilėje suteikiant didesnį prioritetą. Todėl, galima sakyti keisis ir tam tikros žetono, kuris reprezentuoją dokumentą, savybės. Taigi mes galime praplėsti schemą, generuoti konfliktines situacijas ir aprašyti sąlygines pareigas, atsižvelgiančias į tam tikras raktines žetonų savybes (key values). Panagrinėkime schemos fragmentą, pateiktą paveikslėlyje 1.3.2



Paveikslėlis 1.3.2

Matome, kad turime vieną žetoną pozicijoje P1 ir tris pareigas. Standartiniame Petri tinkle žetonas skiltų į tris žetonus ir atsidurtu visose trijose pozicijose P1, P2, P3. Tačiau mūsų schemos atveju toks scenarijus nėra tinkamas, nes nei resursas, nei dokumentas negali pasidalinti, taigi mes sakysime pareigos gali praleisti tam tikrus žetonus pagal tam tikrus požymius. Pavyzdžiui 1.3.2 schemoje žetonas praeis tik pro pareigą R2, nes žetonas atitinka pareigos sąlygą – jis yra žalias. Konfliktinės situacijos gali atsirasti dėl to, jog žetonų savybės laikui bėgant keičiasi. Jas keičia žmogus, kurio atsakas ne visada nuspėjamas, ir darbo eigos paprogramės, analizuojančios dokumentą, todėl tam, kad nesutriktų schemos veikimas ir neatsirastų ciklų schemoje teks projektuoti konfliktines situacijas.

1.4 Elektroninio parašo schemos parinkimas, integracijos taškai ir saugumo problemos

Kadangi įmonė suformulavo elektroninio parašo dokumentams poreikį, reikia pasirinkti tinkamiausią elektroninio parašo schemą ir numatyti elektroninio parašo integracijos dokumentų judėjimo schemoje taškus, nes vėlesnės schemos modifikacijos gali būti sunkios ir nepraktiškos produkcinėje aplinkoje.

Pasirinkta elektroninio parašo schema – grupinis elektroninis parašas. Kiekvienas pasirašantysis asmuo pasirašo ne savo, bet įmonės organizacinio vieneto vardu, todėl jo anonimiškumas išlaikomas. Tačiau, esant poreikiui grupės valdytojas (dažniausiai organizacinio vieneto vadovas) gali išsiaiškinti pasirašiusiojo tapatybę ir ją paviešinti. Elektroninio parašo modelis darbe bus pateikiamas tik teorinis, nes parašo realizacijai įmonės viduje reikalinga infrastruktūra, kurios paruošimui reikia nemažų laiko ir finansinių resursų. Taigi, schemoje sukurti e-parašo integracijos taškai realizuojant schemą programiškai bus praleidžiami, tačiau juos būtina aprašyti ir sukurti „tuščius“ darbo eigų metodus realizuojant dokumentų judėjimo schema SharePoint platformoje.

Taigi, schemoje turės egzistuoti 2 elektroninio parašo integracijos taškai. Pirmasis taškas – būsenos, į kurias patenka dokumentas ir resursas pasirašant dokumentą po apdorojimo. Dokumentas turėtų patekti arba į trumpo saugojimo biblioteką arba į sistemos „work state“ atmintį ir laukti, kol resursas inicijuos savo elektroninį parašą. Tada pasirašytas dokumentas turėtų keliauti į tolimesnį savo kelio tikslą, o resursas turėtų grįžti į naujų dokumentų laukimo arba apdorojimo būseną.

Antrasis integracijos taškas turėtų būti blogai suformuluoto dokumento (turinio arba sistemine prasme) autoriaus atskleidimo būsena. Į ją dokumentas turėtų patekti ir laukti tol, kol resursas, apdorojantis dokumentą pateiks užklausą grupės vadovui atskleisti pasirašiusiojo tapatybę. Atlikus šį veiksmą, dokumentas keliautų žingsniu atgal į skyrių, iš kurio buvo atsiųstas su dokumento žymomis „Gražintas“, „Svarbus“ ir „Tikslus gavėjas“, o resursas turėtų grįžti prie kitų dokumentų apdorojimo. Pasitarus su įmonės skyrių vadovais preliminariai nutarta, jog atskleidimo procesas būtų atliekamas „pusiau automatiškai“. Pateikus užklausą atskleisti tapatybę, ji būtų atskleidžiama iškart grupės valdytojo vardu, o pats valdytojas apie tai gautu notifikacijas su paaiškinimais. Toks sprendimas buvo pasiūlytas dėl to, jog blogai suformuluotas dokumentas dažniausiai būna pradėjęs kitą procesą arba laukiamas kitame skyriuje, todėl turi būti kaip galima skubiau pataisytas ir persiųstas. Būtent tam, kad neilginti jo kelio laiko, automatiškai išaiškinamas asmuo kuris turi pataisyti klaidas, o dokumentas įgauna statusą „Svarbus“.

Kalbant apie schemos saugumą, reikėtu išskirti du pagrindinius punktus: hash funkcijų šeimos parinkimą, bei grupės eilės parinkimą. Reikės parinkti tokią hash funkcijų šeimą, kuri būtų atspari vadinamosioms „gimimo dienos“ atakoms, kurios dažniausiai naudojamos prieš hash funkcijas. Taipogi reikės įvertinti kokios eilės grupes parinkti, kad būtų išvengta DLP išsprendimo. Visą tai aprašysime tiriamojoje dalyje.

1.5 Diskrečiojo logaritmo problema (DLP)

Diskrečiojo logaritmo apskaičiavimo problema – bazinė problema, kuria paremtas grupinio parašo algoritmas.

Apibrėžimas 1.5.1

Tarkime G – baigtinė ciklinė grupė, kurios generatorius $g \in G$. Diskretusis elemento $a \in G$ logaritmas pagrindu g, žymimas $\log_g(a)$ yra unikalus skaičius $x, 0 \le x \le |G|$, toks kad $a = g^x$.

Jeigu g nėra grupės G generatorius, ieškoma mažiausio x, kuriam galiotų $a = g^x$. Remiantis 1.5.1

apibrėžiamu, DLP problema formuluojama taip:

Apibrėžimas 1.5.2

Turime baigtinę ciklinę grupę G, jos generatorių g ir elementą $a \in G$. *Kaip rasti tokį* $x, 0 \le x \le |G| - 1$, *kad galiotų lygybė* $a = g^x$?

DLP uždavinio sprendimui egzistuoja trys pagrindinės algoritmų grupės: bendrieji, modifikuotieji ir specialieji algoritmai.

Pats paprasčiausias bendrasis algoritmas, grupės generavimas iki tol, kol randamas ieškomas elementas. Tokie algoritmai dažniausiai yra polinominio laiko ir labai neefektyvūs. Dažniausiai naudojami eksponentinio laiko algoritmai, tokie kaip skaičių lauko filtravimo.

1.6 Atvaizdavimo problema (Representation problem arba RP)

Atvaizdavimo problema yra DLP apibendrinimas. Tarkime kad *G* – ciklinė n eilės grupė, o elementai $g_1,...,g_m \in G$ - skirtingi grupės *G* generatoriai.

Apibrėžimas 1.6.1

Tam tikro elemento $a \in G$ *atvaizdavimu vadinamas m dydžio vektorius* $(x_1,...,x_m)$, $0 \le x_i \le n-1, \forall i \in [1;m]$ *toks kad* $a = \prod_{i=1}^{m} g_i^{x_i}$

Šis vektorius dar vadinamas indeksų rinkiniu. Kiekvienam elementui $a \in G$ egzistuoja n^{m-1} rinkinių, su kuriais panaudojant $(g_1,...,g_m)$ gaunamas *a*. Pasinaudojant 1.6.1 apibrėžimu suformuluojame RP uždavinį.

Apibrėžimas 1.6.2

Duota baigtinė ciklinė n eilės grupė G, jos generatorių rinkinys $g_1,...,g_m \in G$ *ir elementas a. Kaip rasti tokį rinkinį* $(x_1,...,x_m)$, $0 \le x_i \le n-1, \forall i \in [1;m]$ *kad* $a = \prod_{i=1}^{m} g_i^{x_i}$?

Jeigu parinktumėme atsitiktinius generatorius, rasti bent 2 reikšmių rinkinį bus taip pat sunku kaip išspręsti DLP uždavinį.

1.7 Diffie-Hellman problema (DHP)

DHP problema glaudžiai susijusi su DLP problema. Ji formuluojama taip.

Apibrėžimas 1.7.1

Duota G – baigtinė ciklinė n eilės grupė, kurios generatorius $g \in G$ ir grupės elementai g^u ir g^v. Kaip rasti elementą g^{uv}?

DHP uždavinio uždavinio sprendimo laikas polinominis, jei per polinominį laiką išsprendžiamas diskretaus logaritmo uždavinys: ieškoma $u = \log_g\left(g^u\right)$ ir apskaičiuojama g^{uv}. Remdamiesi 1.7.1 apibrėžimu suformuluokime DDH sprendžiamumo (Desicion Diffie-Hellman arba DDHP) uždavinį.

Apibrėžimas 1.7.2

Duota G – baigtinė ciklinė grupė, kurios generatorius $g \in G$ ir grupės elementai g^u, g^v ir g^w. Kaip sužinoti, ar elementai g^{uv} ir g^w lygūs?

Šis uždavinys dar prieš jį paskelbiant tam tikra prasme buvo kai kurių kirptografinių sistemų baziniu uždaviniu. Jis taip pat bus svarbus ir šiame darbe.

1.8 Viešojo rakto kriptografinės sistemos

Šifravimo sistemos gali būti skiriamos į dvi grupes: privataus ir viešojo rakto, arba simetrines ir asimetrines. Privačiojo rakto sistemose tas pats privatusis raktas naudojamas ir šifravimui ir dešifravimui. Taigi jeigu dvi šalys nori saugiai komunikuoti, jos turi pasikeisti privačiuoju raktu iš anksto. Viešojo rakto kriptografinėse sistemose, šifravimui ir dešifravimui naudojami du skirtingi raktai. Šifravimui naudojamas viešasis raktas, kuris gali būti publikuojamas, o dešifravimui – privatusis raktas, kurį kiekviena šalis laiko paslaptyje. Viešųjų raktų kriptografines sistemas pristatė kriptografai Diffie ir Hellman 1977 metais. Jie pirmieji iškėlė mintį apie vienkrypčių funkcijų su slaptomis durimis (trap-door one way functions) panaudojimą. Šios funkcijos lengvai skaičiuojamos, tačiau jų atvirkštinės skaičiuojamos sunkiai, nebent žinomos „slaptos durys“. Pavyzdžiui, turime funkciją $f : A \to B$ ir žinutę m. Norėdami užšifruoti žinutę m tiesiog suskaičiuojame $f(m)$, o norėdami ją perskaityti dešifruojame skaičiuodami $f^{-1}(f(m)) = m$. Be šifravimo galimybės, vienkryptės funkcijos su slaptomis durimis sudarė galimybes realizuoti skaitmeninio parašo schemas. Pavyzdžiui, žinutes $m \in B$ parašu gali būti $s = f^{-1}(m)$. Parašas gali būti patikrintas kiekvieno, skaičiuojant ar $m = f(s)$ panaudojant viešąjį raktą.

1.9 Viešojo rakto šifravimas

Viešojo rakto šifravimo schemos sudarytos iš 3 algoritmų: generavimo (gen), šifravimo (enc) ir dešifravimo (dec). Generavimo algoritmas visada tikimybinis, šifravimo – dažniausiai tikimybinis, o dešifravimo – deterministinis. Generavimo algoritmas sugeneruoja privatųjį ir viešąjį raktus tam tikram asmeniui *A*, kuriuos pažymėsime atitinkamai x_a *ir* y_a. Šifravimo algoritmas naudodamas viešąjį raktą užšifruoja tam tikrą žinutę *m*, ir gražiną užšifruotą žinute, kuria pažymime *c*. Dešifravimo algoritmas privačiuoju raktu dešifruoja žinutę *c* ir gražina žinutę *m*. Dešifravimo algoritmui su visomis *m* ir $(x_a; y_a)$ teisinga tokia lygybė: $dec(c,x_a)\begin{Bmatrix} = m, & jei & P(c = enc(m,y_a)) > 0 \\ \neq m & priešingu & atveju \end{Bmatrix}$. Jeigu Šifravimo algoritmas tikimybinis, šifravimo schemą vadinsime tikimybine.

Dažniausiai šifruojamų žinučių ilgis yra ribotas. Be to, simetrinio šifravimo schemos dažniausiai yra greitesnės nei viešojo rakto. Todėl tam, kad būtų sumažinamas šifruojamos žinutės ar duomenų masyvo dydis, bei schema taptų kiek greitesnė, buvo įvestos „Hash“ funkcijos, kurios aprašytos vėlesniuose skyriuose.

1.10 Diffie-Hellman raktų pasikeitimo protokolas

Kriptografai Diffie ir Hellman ne tik pirmieji aprašė viešojo rakto kriptografinę sistemą, bet ir pasiūlė būdą kaip perduoti privatųjį raktą naudojant autentišką, bet nebūtinai slaptą perdavimo kanalą. Perduotas privatusis raktas gali būti panaudotas simetrinio šifravimo algoritme. Aprašysime schemos veikimo principą.

Tarkime *G* – baigtinė ciklinė *q* eilės grupė, o $g \in G$ toks grupės generatorius, kad diskrečiojo logaritmo apskaičiavimas grupėje *G* yra neįvykdomas. Pusių *A* ir *B* (pvz. Aldonos ir Broniaus), besikeičiančių raktais, viešuosius raktus pažymėkime atitinkamai $y_A = g^{x_A}$ *ir* $y_B = g^{x_B}$ o privačiuosius raktus - x_A *ir* x_B. Tam kad gautų bendrojo privačiojo rakto *k* reikšmę, Aldona ir Bronius apsikeičia savo viešaisiais raktais autentišku kanalu, ir pakelia vienas kito viešuosius raktus laipsniu, lygiu savo viešiesiems raktams, t.y. $k = y_A^{x_b} = y_B^{x_a} = g^{x_A x_B}$. Šios schemos saugumas paremtas 1.7 skyriuje aprašyta DHP problema. (Pastaba: dėl šios schemos pavadinimo 1.7 skyriuje aprašyta problema ir buvo pavadinta Diffie-Hellman problema).

1.11 ElGamal šifravimo schema.

Apibrėšime šifravimo schemą, paremtą Diffie-Hellman raktų pasikeitimo protokolu, aprašytu 1.10 skyriuje. Tarkime *G* – baigtinė, *q* eilės ciklinė grupė, o $g \in G$ toks šios grupės generatorius, kad diskrečiųjų logaritmų skaičiavimas grupėje *G* neįvykdomas. Tokia grupe galėtų būti 1.1 skyriuje aprašyta grupė Z_m^*, o $g = p - 1$ kai *p* – didelis pirminis skaičius. Taigi norėdama užšifruoti žinutę $m \in G$ siunčiama Broniui, kurio viešasis raktas yra $y = g^x$, Aldona pirmiausiai renkasi atsitiktinį skaičių $a \in Z_q$ ir apskaičiuoja skaičių porą $(A,B) = (g^a, y^a m)$ kuri ir yra žinutės m šifras. Tada Bronius, gavęs šią skaičių porą ją iššifruoja privačiuoju raktu *x* tokiu būdu: $\frac{B}{A^x} = \frac{y^a m}{g^{ax}} = \frac{g^{xa} m}{g^{ax}} = m$.

Kitas būdas užšifruoti žinutę, kuris bus naudojamas vėlesniuose skyriuose, yra sukeisti viešąjį raktą, ir logaritmo bazę t.y. *y* ir *g* vietomis. Tada analogiškai šifruojant žinutę *m* gauname $(A,B) = (y^a, g^a m)$ ir iššifravimą atliekame taip: $\frac{B}{A^{x^{-1}}} = \frac{g^a m}{y^{ax^{-1}}} = \frac{g^a m}{g^{xax^{-1}}} = m$.

Abiem aprašytais atvejais remiamasi DHP problema. Abi aprašytos schemos – tikimybinės. Jeigu kiekvienam šifravimui rinktumėmės po skirtingą *a*, galiotų DDHP prielaida (tektų tikrinti, ar dvi poros (*A*,*B*) ir (*A*′,*B*′) šifruoja tą pačią žinutę *m*).

1.12 Schnorr identifikacijos protokolas.

Identifikacijos protokolai skirti tam, kad pasirašiusioji Aldona galėtų įtikinti tikrintoją Bronių tuo, kad žinutę pasirašė tikrai ji. Bronius gauna viešąjį Aldonos raktą *Y*, tačiau jei Aldona galėtų parodyti, jog žino privatųjį raktą kuris atitinka Broniaus turimą rakta *Y*, Bronius galėtų būti visiškai įsitikinęs jog viešasis raktas *Y* tikrai priklauso Aldonai. Tokie identifikacijos protokolai turi tenkinti tris pagrindines savybes.

1. Būti saugūs Aldonos atžvilgiu, t.y. Aldonai įrodinėjant kad Broniaus gautas raktas *Y* tikrai jos, nei Bronius nei kas kitas negali nieko sužinoti apie Aldonos privatųjį raktą *x*.
2. Aldona visada turi turėti galimybę vienareikšmiškai įtikinti Bronių kad jo turimas viešasis raktas tikrai jos.
3. Zita, kuri nežino Aldonos privačiojo rakto niekada neturi įtikinti Broniaus, kad jos atsiųstas Broniui viešasis raktas Y iš tiesų priklauso Aldonai.

Šios savybės būdingos neatskleidžiantiems žinių įrodymams, aprašytiems vėlesniuose skyriuose. Šių įrodymų bazinis protokolas – kriptografo C.P. Schnorr pristatytas, vadinamas trijų žingsnių, arba Schnorr protokolu. Protokolo saugumas paremtas diskrečiojo logaritmo problema. Protokolas leidžia įrodyti Aldonai kad ji žino savo viešojo rakto diskretųjį logaritmą. Tarkime G – baigtinė, q eilės ciklinė grupė, o $g \in G$ toks šios grupės generatorius, kad diskrečiųjų logaritmų skaičiavimas grupėje G neįvykdomas. Tarkime $y = g^x$ - Broniaus turimas Aldonos viešasis raktas, o x – Aldonos privatusis raktas. Taigi Aldona žino reikšmes (g, q, y, x) o Bronius (g, q, y). Taigi protokolas vykdomas taip.

1. Aldona pasirenka atsitiktinį $r \in Z_q$, apskaičiuoja $t = g^r$ ir siunčia reikšmę t Broniui.
2. Bronius pasirenka atsitiktinį $c \in \{0,1\}^k$ ir siunčia c Aldonai.
3. Aldona, gavusi reikšmę c skaičiuoja $s = r - cx \pmod q$ ir siunčia Broniui reikšmę s.

Gavęs reikšmę s, Bronius, tikrindamas lygybę $t = g^s y^c$ nusprendžia, ar y tikrai Aldonos viešasis raktas (jei lygybė teisinga, raktas y tikrai Aldonos).

1.13 Elektroninio parašo schemos

Elektroninis parašas tai savotiška ranka pasirašyto parašo alternatyva, realizuota susiejant pasirašomą žinutę su pasirašančiojo viešuoju raktu. Kiekvienas žinutės gavėjas turi turėti galimybę perskaityti žinutę naudodamas pasirašiusiojo viešąjį raktą, o pasirašantysis – suskaičiuoti parašo reikšmę naudodamasis savo privačiuoju raktu. Kaip jau buvo minėta 1.10 skyriuje, pirmoji parašo schema ir jos koncepcija buvo pasiūlyta kriptografų Diffie ir Hellman. Taigi, apibrėžkime elektroninio parašo schemą.

Apibrėžimas 1.13.1

Elektroninio parašo schema tai schema sudaryta iš trijų algoritmų: Generavimo, Pasirašymo ir Tikrinimo (atitinkamai juos pažymėkime gen, sig, ver). Generavimo algoritmas – tikimybinis. Pasirašymo – dažniausiai tikimybinis. Tikrinimo – deterministinis. Generavimo algoritmas sugeneruoja viešąjį ir privatųjį asmens raktus x ir y. Pasirašymo algoritmas, naudodamas sugeneruotą privatųjį raktą x ir pasirašomą žinutę m grąžina žinutes m parašą raktu x, kurį pažymėsime s. Tikrinimo algoritmas, naudodamas parašą s, žinutę m ir viešąjį pasirašiusiojo raktą y patikrina, ar parašas s tikrai pasirašytas asmens, kurio viešasis raktas y.

Atakos prieš parašo schemas dažniausiai klasifikuojamos į tris grupes:

Visiškas įsilaužimas: klastotojas Z, kurį vadinsime Zita, suskaičiuoja pasirašančiojo A, kurį vadinsime Aldona privatųjį raktą, ir gali jos vardu pasirašinėti visas Aldonos siunčiamas žinutes gavėjui B (Broniui).

Atrankinis klastojimas: Zita sugeba suskaičiuoti tam tikrų žinučių arba jų grupių parašus, ir tas arba tų grupių žinutes siųsti Broniui pasirašytas savo vardu.

Egzistencinis klastojimas: Zita sugeba suskaičiuoti paskutinės žinutės parašą ir perdavimo metu turi mažą arba jokios galimybės pakeisti žinutės turinį.

1.14 Hash funkcijos

Elektroninio parašo schemose hash funkcijos skirtos sumažinti pasirašomos žinutės ilgį ir atvaizduoti jas į bitų eilutes, suprantamas pasirašymo algoritmams. Šios funkcijos kai kurias atvejais gali pakeisti ir tikrintojus žinių įrodymo schemose taip jas paverčiant parašo schemomis. Apibrėžkime hash funkciją.

Apibrėžimas

Funkcija H, atvaizduojanti dvejetainių skaičių baigtines eilutes į l ilgio dvejetaines eilutes

$$H:\{0,1\}^{*}\to\{0,1\}^{l}$$

vadinama hash funkcija.

Šioms funkcijoms keliamas efektyvaus skaičiavimo, ir sunkaus atvirkštinės funkcijos radimo reikalavimas, t.y. hash funkcija turi tenkinti bent vieną iš šių trijų savybių:

1. Silpnas sutapimo atsparumas: duotam *x* sunku rasti tokį $x'\neq x$ kad $H(x')=H(x)$;
2. Stiprus sutapimo atsparumas: sunku rasti tokią porą (x,x') *kur* $x\neq x'$ kad $H(x')=H(x)$ kai H parenkama atsitiktinai iš tam tikros hash funkcijų šeimos.
3. Vienkryptiškumas: turint *c* sunku rasti tokį *x* kad $H(x)=c$.

Žinoma šiuo atveju sakant „sunku rasti“ reikia atsižvelgti į konkretaus uždavinio reikalavimus. Jeigu sakytumėm, kad norime išvengti brutalios jėgos atakos (brute force atack), kada iš eilės tikrinamos įvairios hash funkcijos su tam tikrais parametrais, reikėtų tik padidinti rezultato (*c*) dydį bitais. Tačiau koks rezultato dydis užtikrintų saugumą? Egzistuoja daug metodikų skirtų įvertinti šį dydį esant skirtingoms sąlygoms, pavyzdžiui koks turi būti rezultato dydis bitais, kad išvengtumėm vadinamųjų gimimo dienos atakų (birthday atacks). Manoma, kad 160 bitų rezultatas yra pakankamas saugumui užtikrinti, tačiau tai reikės patikrinti vėliau, analizuojant schemos saugumą.

1.15 ElGamal parašo schema

ElGamal parašo schema, paremta 1.11 skyriuje aprašytu ElGamal viešojo rakto šifravimo protokolu. Šią parašo schemą aprašysime tam, kad galėtumėme pristatyti darbe naudojamą Schnorr parašo schemą, kuri yra savotiška šios schemos modifikacija.

Tarkime G – baigtinė, q eilės ciklinė grupė, o $g \in G$ toks šios grupės generatorius, kad diskrečiųjų logaritmų skaičiavimas grupėje G neįvykdomas. Tarkime $y = g^x$ - Broniaus turimas Aldonos viešasis raktas, o x – Aldonos privatusis raktas. Tarkime H – 1.14 skyriuje aprašyta hash funkcija $H : \{0,1\}^* \to Z_q$. Suformuluojame ElGamal parašo apibrėžimą:

Apibrėžimas 2.15.1

ElGamal parašu žinutei $m \in \{0,1\}^*$ *naudojant viešąjį raktą* $y = g^x$ *vadiname porą* $(u,s) \in GxZ_q$, *tenkinančią sąryšį* $g^{H(m)} = y^u u^s$

Apibrėžime y^u iš tiesų reiškia kad grupės elementas u pirma buvo atvaizduotas į sveikųjų skaičių aibę (pažymėkime jį u'), tada apskaičiuotas jo modulis pagrindu q (rezultatas u''), ir tada apskaičiuota $y^u = y^{u''}$.

Tokį parašą žinutei m Aldona gali apskaičiuoti tokiu būdu:

1. Aldona renkasi atsitiktinį $r \in Z_q^*$.
2. Aldona suskaičiuoja $u = g^r$ ir $s = r^{-1}(H(m) - xu)(mod\ q)$.

Jeigu patikrintume šias 2 lygybes pagal apibrėžime duotą sąryšį, iš tiesų gautumėm $y^u u^s = g^{xu} g^{rs} = g^{xu+rs} = g^{H(m)}$.

1.16 Schnorr parašo schema

Kaip jau minėjome 1.15 skyriuje, Schnorr parašo schema yra tam tikra ElGamal parašo schemos modifikacija. Tarkime G – baigtinė, q eilės ciklinė grupė, o $g \in G$ toks šios grupės generatorius, kad diskrečiųjų logaritmų skaičiavimas grupėje G neįvykdomas. Tarkime $y = g^x$ - Broniaus turimas Aldonos viešasis raktas, o x – Aldonos privatusis raktas. Tarkime H – 1.14 skyriuje aprašyta hash funkcija $H : \{0,1\}^* \to Z_q$. Suformuluojame Schnorr parašo apibrėžimą:

Apibrėžimas 1.16.1

Schnorr parašu žinutei $m \in \{0,1\}^*$ *naudojant viešąjį raktą* $y = g^x$ *vadiname porą* $(c,s), c,s \in Z_q$,

tenkinančią sąryšį $c = H(m \| g^s y^c)$.

Tokį parašą žinutei m Aldona gali apskaičiuoti tokiu būdu:

3. Aldona renkasi atsitiktinį $r \in Z_q$.
4. Aldona suskaičiuoja $c = H(m \| g^r)$ ir $s = r - cx(mod\ q)$.

Jeigu patikrintume šias 2 lygybes pagal apibrėžime duotą sąryšį, iš tiesų gautumėm $g^s y^c = g^{s+xc} = g^{r-xc+xc} = g^r$ o iš čia $H(m \| g^s y^c) = H(m \| g^r) = c$.

1.17 Įrodymai apie diskrečiojo logaritmo žinojimą

Ankstesniuose skyriuose buvo minėtos problemos, su kuriomis susiduria Aldona norėdama Broniui, jog Broniaus gautas viešasis raktas Y tikrai jos. Šiame darbe bus tokioms problemoms spręsti bus naudojamos įrodymo sistemos susijusios su diskrečiojo logaritmo problema. Šios sistemos paremtos trijų žingsnių protokolais, paremtais Schnorr protokolu, tačiau reikia atkreipti dėmesį, jog aprašomų protokolų pateikimas labiau primena parašo schemas, todėl šios sistemos vadinamos parašu paremtais žinių įrodymais (signature based proof of knowledge) arba sutrumpintai SPK. Mes sakysime, kad parašas įrodo slaptųjų raktų žinojimą, tačiau šis teiginys galioja tik mūsų nagrinėjamoms sistemoms.

Apibrėžkime schemose naudojamus dydžius ir funkcijas. Tarkime *G* – baigtinė, *q* eilės ciklinė grupė, o $g,h,g_1,..,g_k \in G,k \in Z$ tokie šios grupės generatoriai, kad diskrečiųjų logaritmų skaičiavimas grupėje G neįvykdomas. Grupės generatoriai parenkami atsitiktinai tokiu būdu, kad nė vieno generatoriaus diskretieji logaritmai kito generatoriaus atžvilgiu nebūtų žinomi. Tada grupės elemento atvaizdavimo radimas toks pat sunkus kaip diskrečiojo logaritmo problema. Dažniausiai grupė *G* parenkama kaip grupės Z_p^* pogrupis (*p* – pirminis skaičius). Tegu *H* – 1.14 skyriuje aprašyta stipraus susikirtimo atsparumo hash funkcija. Šalis, kurios dalyvauja protokole vadinsime: pasirašantysis – Aldona, žinutes gavėjas ir parašo tikrintojas – Bronius.

Dabar apibrėšime pirmąjį ir paprasčiausią SPK bazinį komponentą – viešojo rakto *y* diskretųjį logaritmą pagrindu *g*.

Apibrėžimas 1.17.1

Pora $(c,s) \in \{0,1\}^l x\ Z_q$ *tenkinanti sąryšį*

$$c = H(S \| V \| m) \quad kur \quad S = G \| y \quad ir \quad V = g^s y^c$$

vadinama elemento y diskretaus logaritmo pagrindu g žinutei $m \in \{0,1\}^*$ *žinojimo parašu paremtu įrodymu arba SPK, ir žymima*

$$SPK_1\{(\alpha): y = g^{\alpha}\}(m)$$

Matome, kad šis SPK ne kas kita kaip Schnorr parašas, aprašytas 1.16 skyriuje, tik su kiek kitokiais hash funkcijos argumentais.

$SPK_1\{(\alpha): y = g^{\alpha}\}(m)$ reikšmė gali būti apskaičiuojama tik tada, kai žinomas slaptasis raktas $x = \log_g y$, pasirenkant atsitiktinį $r \in Z_q$ ir apskaičiuojant $t = g^r$, o iš čia suskaičiuojama $c = H(g \parallel y \parallel t \parallel m)$ ir $s = r - cx(mod\ q)$. Jeigu g ir y gali būti aiškiai atskleisti kontekste, jie bus perduodami hash funkcijai kaip argumentai. Reikšmę t nuo šiol vadinsime įteikimo reikšme, c – iššūkiu o s – atsaku. Bendruoju atveju paršą, paremtą raktų α, β žinojimu, žymime taip:

$$SPK_i\{(\alpha,\beta): y = g^{\alpha} \wedge z = g^{\beta}h^{\alpha}\}(\mathrm{m})$$

Šis žinojimo įrodymas paremtas logaritmo y pagrindu g žinojimo įrodymui ir elemento *z* atvaizdavimo problemos sprendimui, aprašytiems atitinkamai 1.5 ir 1.6 skyriuose. Indeksas *i* pažymi tam tikrą SPK schemą. Konkrečios indeksais žymimos schemos bus pateiktos vėliau šiame skyriuje. Dažniausiai kuo indeksas i didesnis, tuo sudėtingesnės žinių įrodymo procedūros.

Grįžkime prie 1.17.1 apibrėžimo. Apibrėžime hash funkcijoje (kaip ir tolimesniame darbe) bus naudojami tokie pažymėjimai:

1. *m* – pasirašyta tam tikro ilgio duomenų eilutė (gali būti ir tuščia). Sakysime, kad hash funkcija visada gaus m kaip argumentą (t.y. nebus tokios schemos, kurios kontekste nebūtų m)
2. *S* – duomenų eilutė, kurioje bus saugomi žinojimo (arba teiginio) įteikimo reikšmės.
3. *V* – duomenų eilutė sudaryta iš dalių, galinčių patvirtinti parašo tikrumą.

Iš tiesų, kiekvienas *V* komponentas turi būti lygus tam tikram įteikimo komponentui iš *V*.

Kitas svarbus apibrėžimas – viešojo rakto atvaizdavimo parašu paremtu įrodymu t.y. SPK.

Apibrėžimas 1.17.2

(k+1) eilės vektorius $(c,s_1,..,s_k) \in \{0,1\}^l x (Z_q)^k$ *tenkinantis sąryšį*

$$c = H(S \| V \| m) \quad kur \quad S = g_1 \| ... \| g_k \| y \quad ir \quad V = y^c \prod_{i=1}^{k} g_i^{s_i}$$

vadinamas viešojo rakto y žinutei $m \in \{0,1\}^*$ *atvaizdavimo elementais* $g_1 ... g_k$ *atvaizdavimo žinojimo parašu paremtu įrodymu. Šio įrodymo schemos indeksas i=2 ir žymėjimas*

$$SPK_2\{(\alpha_{1,...}, \alpha_k): y = \prod_{i=1}^{k} g_i^{\alpha_i}\}(m).$$

Apibrėžimuose 1.17.1 ir 1.17.2 pateiktos bazinės SPK schemos, kuriomis remiantis aprašysime schemas, reikalingas grupinio parašo konstravimui. Tačiau prieš tai reikia įvesti papildomus žymėjimus.

Sakysime, kad asmens viešasis raktas y_i gali būti suformuotas naudojant tik tam tikrą generatorių aibės poaibį $g_1,...,g_k$. Tam, kad realizuotumėme šį uždavinį panaudosime indeksų poaibį $I_i \subseteq \{1,...,k\}$, tokį kad $y_i = \prod g_j^{x_{ij}}$, kur x_{ij} - privatusis asmens, kuriam priklauso y_i , raktas. Žymėjimas *ij* parodo kad privatusis raktas atitinka asmens *i* viešąjį raktą y_i ir generatorių g_j . Naudodamiesi šia y_i konstrukcija galime sudaryti SPK schemą, kuri leis įrodyti vieno iš kelių viešųjų raktų žinojimą, neatskleidžiant kurio būtent. Tokiu būdu pasirašančioji Aldona gali paprasčiausiai nurodyti tam tikrą viešąjį rakta y_i , ir įrodyti, kad jo atvaizdavimą ji žino. Tokią schemą pasiūlė kriptografas Cramer. Jo pasiūlytoje schemoje SPK_4 remiamasi SPK_2 schema. Ji bus viena pagrindinių grupinio parašo algoritmo konstrukcijos schemų.

Apibrėžimas 1.17.3

$\left(n + \sum |I_i|\right)$ *eilės vektorius* $\left(c_1,..,c_n,(s_{ij})_{i=1...n; j \in I_i}\right) \in \left(\{0,1\}^l\right)^n x \left(Z_q\right)^{\sum |I_i|}$ *tenkinantis sąryšį*

$$(+)\, c_i = H(S \| V \| m) \quad kur \quad S = g_1 \| ... \| g_k \| y_1 \| y_n \| I_1 \| ... \| I_n$$

$$ir \quad V = y_1^{c_1} \prod_{j \in I_1} g_j^{s_{1j}} \| ... \| y_n^{c_n} \prod_{j \in I_n} g_j^{s_{nj}}$$

vadinamas bent vieno viešojo rakto iš $(y_1,..,y_n)$ *žinutei* $m \in \{0,1\}^*$ *diskrečiojo logaritmo pagrindu*

g žinojimo parašu paremtu įrodymu. Šio įrodymo schemos indeksas i=4 ir žymėjimas

$$SPK_4\{(\alpha_{ij})_{i=1,\dots n;\, j\in I_i} :\vee_{i=1}^{n} y_i = \prod_{j\in I_i} g_j^{\alpha_{ik}}\}(m).$$

Apibrėžime $(+)_{i=1}^{n} c_i$ žymi operaciją „xor“ bitams. Vadinasi, Aldona, įrodinėdama logaritmo žinojimą gali pasirinkti vieną iš visų iššūkio reikšmių atitinkančią vieninteliam viešajam jos pačios raktui. Vadinasi Zitai, norint suklastoti Aldonos parašą bus kur kas sunkiau suskaičiuoti iššūkio reikšme nei SPK_2 schemos atveju.

Matome, kad pastarosios schemos leidžia suformuoti parašus, įrodančius diskrečiojo logaritmo bei atvaizdavimo žinojimus. Tačiau kyla klausimas, kai įrodyti ne tik slaptųjų raktų, bet ir sąryšių tarp jų žinojimą. Apibrėšime parašą, įrodantį kad dviejų viešųjų raktų logaritmai skirtingais pagrindais, lygūs tarpusavyje.

Apibrėžimas 1.17.4

Pora $(c,s)\in\{0,1\}^l xZ_q$ *tenkinanti sąryšį*

$$c = H(S \parallel V \parallel m) \quad kur \quad S = g \parallel h \parallel y \parallel z \quad ir \quad V = g^s y^c \parallel h^s z^c$$

vadinama žinutės $m\in\{0,1\}^*$ *žinojimo parašu paremtu įrodymu arba SPK, įrodanti lygybę* $\log_h z = \log_g y$ *ir žymima*

$$SPK_6\{(\alpha): y = g^{\alpha} \wedge z = h^{\alpha}\}(m)$$

Šį parašą naudosime grupinio parašo schemoje pasirašiusiojo asmens tapatybei atskleisti.

Turėdami SPK_4 parašo apibrėžimą apibrėšime schemą SPK_{10}, kuri bus naudojama vienai iš grupinio parašo komponenčių apskaičiuoti. SPK_{10} gali būti traktuojama kaip du paralelūs SPK_4 parašai $SPK_4\{(\alpha_i)_{i=1,\dots n;\, j\in I_i} :\vee_{i=1}^{n} y_i = g^{\alpha_i}\}$ ir $SPK_4\{(\alpha_i)_{i=1,\dots n;\, j\in I_i} :\vee_{i=1}^{n} z_i = h^{\alpha_i}\}$.

Apibrėžimas 1.17.5

2n eilės vektorius $(c_1,..,c_n,(s_i)_{i=1\dots n;})\in(\{0,1\}^l)^n x(Z_q)$ *tenkinantis sąryšį*

$$(+)\sum_{i=1}^{n} c_i = H(S \| V \| m) \quad kur \quad S = g \| h \| y_1 \| z1 ... \| y_n \| z_n$$

$$ir \quad V = {y_1}^{c_1} g^{s_1} \| {z_1}^{c_1} h^{s_1} \| ... \| {y_n}^{c_n} g^{s_n} \| {z_n}^{c_n} h^{s_n}$$

vadinamas parašu žinutei $m \in \{0,1\}^*$ *, paremtu diskrečiųjų logaritmų* $\log_g y_i$ *ir* $\log_h z_i$ *žinojimų ir jų tarpusavio lygybe bent vienam* $i, 1 \le i \le n$*. Šis parašas žymimas*

$$SPK_{10}\{(\alpha_i)_{i=1,...n;\, j \in I_i} : \bigvee_{i=1}^{n} \left(y_i = g^{\alpha_i}\right) \wedge \left(z_i = h^{\alpha_i}\right)\}(m).$$

2. Tiriamoji dalis

2.1 Dokumentų judėjimo kelių aprašymas

Viena svarbiausių užduočių kuriant DVS schemą – aprašyti dokumentų kelius skaitmenine forma taip, kad po to jie būtų lengvai panaudojami konstruojamuose algoritmuose. Dokumentų kelių sudarymas ir optimizavimas kol kas nebuvo reikalingas, nes įmonėje egzistuoja nusistovėjusi dokumentų perdavimo tvarka, todėl teko jos laikytis.

Kadangi kuriama schema laikui bėgant bus pritaikyta ne tik vienoje grupės įmonėje, ir ne tik SharePoint platformoje, dokumentų kelius aprašyti praktiškiausia SQL duomenų bazėje. Bet kada prireikus su nesudėtinga aplikacija įrašus galima pakeisti, ištrinti ar pridėti naujų. Tam panaudojau Microsoft SQL duomenų bazę. Kadangi SharePoint 2010 standartinėje komplektacijoje įdiegtas 1.2 skyriuje minėtas Bussines Data Conectivity (arba BDC) servisas, galėjau pasinaudoti jo funkcionalumu įrašams iš duomenų bazės perkelti į SharePoint sąrašus. Kitos platformos atveju tektų panaudoti kitokį programinį sprendimą, tačiau įrašus gauti vis tiek pavyktų, todėl kelių aprašymas SQL – pats praktiškiausias.

Duomenų bazių lenteles projektavau tokiu būdu. Kiekviena grupės įmonė turi skyrių kodus, pvz personalas – P, aukščiausio lygio vadovai ALV ir t.t. Kiekvienai grupei priklausantys darbuotojai pagal pareigybės kategorija turi identifikacinį numerį. 1 – aukščiausio lygio pareigybės tarnautojas, 4 – žemiausio, pvz. P1 – personalo vadovas, P4 – atrankų specialistai. Aukštesnės kategorijos darbuotojai gali matyti žemesnės kategorijos darbuotojų dokumentus, bet ne atvirkščiai. Panaši logika naudojama ir SharePoint platformoje – skyriui P suteikus prieigos teises prie bibliotekos „Personalas“, skyriaus vadovas mato visus dokumentus bibliotekoje, tačiau tai nėra gerai, nes „pasimeta“ reikalingi ir svarbūs dokumentai.

Taigi, panaudodamas AD aprašytas vartotojų grupes, suprojektavau atitinkamas lenteles kiekvienai įmonei. Jų struktūra pavaizduota 2.1.1 lentelėje

s_id	*B1*	*B1_L*	*B_2*	*B2_L*	*...*	*B_k*	*B_k_L*	*arch*
1	P	1	B	2		null	null	Ilgalaikis

Lentelė 2.1.1

Aprašysime stulpelių reikšmes ir pagrįsime struktūros pasirinkimą.

s_id – šablono identifikacijos numeris. Visi dokumentai gali būti sugrupuoti į tam tikras grupes, pvz. atostogų prašymas, kuro ataskaitos ir pan. Visų šių grupių dokumentų keliai vienodi, skiriasi tik turinys.

B_k – dokumento paskyrimo vieta, ir jos eilės numeris. Pavyzdyje matome, kad dokumentas sukuriamas personalo skyriuje. Techniškai kalbant tai irgi žingsnis, nes personalo skyriaus asmuo jį nusiunčia į biblioteką „Personalo skyrius“. Šį punktą svarbu pažymėti, nes prireikus gražinti dokumentą adresatas bus žinomas. Po to dokumentas keliauja į buhalteriją, t.y. tampa matomas buhalterijos biblioteką matantiems asmenims, ir t.t.

B_k_L – dokumento paskyrimo vietos teisės lygio filtras. Kaip buvo minėta ankščiau, kyla problema, jog skyriaus vadovas mato labai daug dokumentų, kurių didžiosios dalies matyti jam nereikia. Tam, į dokumentų kelių aprašymą įterpiau pareigybės filtrą kuris veikia -1 principu. Taigi, jeigu dokumentas siunčiamas buhalterijai B, dvejetas indikuoja kad antro pareigybės lygio asmenys pvz. buhalteriai dokumentą mato, o -1 reiškia kad dokumentą gali pamatyti ir B1 pareigybės tarnautojas – vyr. Buhalteris. -1 praktika gali ir išnykti ją pakeičiant papildomu žingsniu, t.y. iš B2 į B1, tačiau darbo su sistema pradžioje ši opcija paliekama dėl papildomos kontrolės.

arch – Nurodomas archyvo tipas. Kai kurie dokumentai privalo būti archyvuojami tam tikrą dienų skaičių, todėl prie dokumentų tipo, kuriuos reikia talpinti į archyvą nurodoma archyvo tipas, pvz. ilgalaikis. Iš tiesų visi dokumentai archyvuojami tačiau skiriasi tik archyvavimo trukmė. („nearchyvuojami“ dokumentai saugumo sumetimais saugomi 30 dienų). Archyvavimo trukmė nurodoma konfigūruojant atitinkamas SharePoint bibliotekas (archyvas tai irgi biblioteka).

Kiekvienai atskirai įmonei kuriama atskira šios struktūros lentelė. Iš tiesų pareigybių tipai skiriasi tik keliose įmonėse pagal įmonės tipą (gamybinė ar prekybos), taigi sukūrus kelias lenteles jų struktūrą galima kopijuoti kuriant naują lentelę, o duomenis tik šiek tiek pakeisti.

2.2 Dokumentų judėjimo skyrių viduje schemos sudarymo prielaidos

Pasinaudodami SP 2010 darbo eigų servisu galime sudaryti sudėtingesnes ir funkcionalesnes programas, skirtas dokumentų srautams optimizuoti. Kaip matėme, dokumentų keliai, aprašyti 2.1 skyrelyje leidžia realizuoti dokumentų persiuntimą iš vieno skyriaus į kitą, tačiau kyla klausimas, kaip dokumentai „cirkuliuos“ skyriaus viduje.

Standartinis SharePoint DVS funkcionalumas šios problemos nesprendžia. Todėl, visi skyriaus darbuotojai mato visus dokumentus, adresuotus skyriui (arba konkrečiai jiems), kurių prieigos teisės pagal valdymo struktūrą, mažesnės arba lygios jų prieigos teisėms. To pasekoje sunkiau pastebėti svarbų ar reikiamą dokumentą, todėl didėjant srautui smarkiai lėtėja aptarnavimo laikas, reikalingos papildomos komunikacijos tarp darbuotojų jiems patiems planuojantis darbą su dokumentu. Ši problema dar aktualesne srityse, kur dokumentą aptarnauja programos, t.y. dokumento mašininio apdorojimo taškuose. Dokumentai apdorojami pagal atsiuntimo laiką visiškai neatsižvelgiant į papildomus jų požymius ar didėjantį poreikį kitose bibliotekose. Ši problema kartais sprendžiama sistemos administratoriui prisijungus prie sistemos ir ranka perdėliojus apdorojimo laikus, tačiau šis sprendimas yra labai nepatogus.

Taigi pagrindinė išryškėjusi problema standartiniame SP DVS funkcionalume – neatsižvelgiama į resurso būseną ir papildomus dokumento požymius. Norėdamas sudaryti efektyvų algoritmą, kurį realizavus kiekviename punkte būtų atsižvelgta į sistemos resursų prieinamumą, dokumentų judėjimo schemą nutariau aprašyti pasinaudojant Petri tinklų savybėmis.

2.3 Schemos bendrojo algoritmo konstravimas

Prieš aprašant galutinę schemą Petri tinklu sudarykime primityvų algoritmą, kuris atspindėtų mums reikalingus veiksmus ir žingsnius schemoje. Primityvaus algoritmo schemą praktiška sudaryti tam, kad konstruojant schemą, paremtą Petri tinklu turėsime rezultato viziją, ir tereikės pasinaudoti Petri tinklų bei SharePoint savybės rezultatui pasiekti.

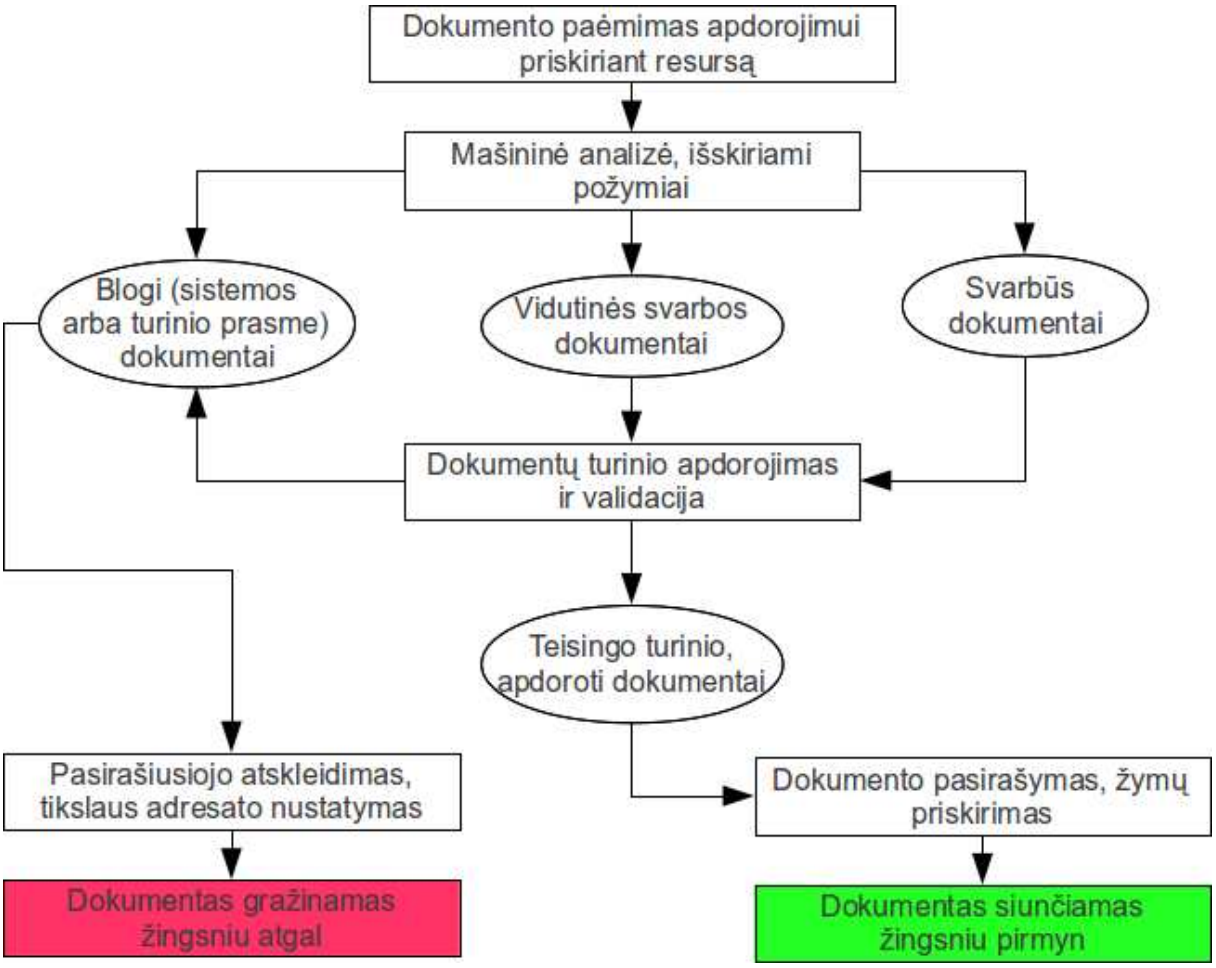
Pirmiausiai reikės realizuoti dokumento paėmimą apdorojimui. Reikia nepamiršti, jog dokumentas bus adresuojamas grupėms asmenų, galinčių atlikti vienodas funkcijas. Jei grupę sudarys tik vienas asmuo, jis vienas ir atliks reikiamą funkciją. Taigi pirmasis žingsnis bus dokumento priėmimas apdorojimui.

Dokumentą priėmus apdorojimui, t.y. paskyrus darbui su juo atitinkamą resursą, dokumentas turi būti analizuojamas mašininiu būdu tam, kad sistema gautu tam tikrus jo požymius, pagal kuriuos su juo bus atliekami tolimesni veiksmai. Skaitmeninės analizės būsenoje bus tikrinama ar dokumentas

turi požymi „svarbus“, gražintas, ir ar dokumentas teisingai suformuluotas. Pastarasis žingsnis tik laikinas. Turbūt iš karto kyla klausimas, kaip dokumentas gali būti blogai suformuluotas, jei naudojami sisteminiai šablonai. Iš tiesų ši problema iškilo jau testuojant schemą produkcinėje aplinkoje. Kadangi dokumentai juda ne tik įmonės viduje, bet ir kai kada tarp įmonių, pasitaiko atveju, kada įvyksta tam tikro dokumento šablono modifikacija siuntėjo arba gavėjo pusėje, ir gautas dokumentas tampa nebe atpažįstamas, t.y. sistema analizuoja dokumentą ir neranda tam tikrų požymių reikšmių nes buvo pakeisti jų vardai arba jų vietos. Kol sistema pilnai nestandartizuota šis punktas paliekamas, ir galbūt, atsargumo dėlei jis bus paliktas ir standartizavus sistemą. Tai priklausys nuo dokumentų apdorojimo laiko esant dideliems dokumentų srautams. Jei šios sąlygos tikrinimo laikas turės įtaką bendram apdorojimo laikui ji bus pašalinta.

Išanalizavus dokumentą mašininiu būdu jis pateks į didelės arba normalios svarbos dokumentų bibliotekas, kur lauks tolimesnės resurso analizės. Pasitarus su įmonės specialistais buvo nutarta suteikti darbuotojui šiek tiek laisvės planuoti savo laiką ir vidutinės svarbos dokumentus apdoroti savo nuožiūra, t.y. leisti jam rinktis, ar dokumentą apdoroti dabar, ar priimti naują vidutinės svarbos dokumentą (t.y. registruoti jį kaip savo užduotį) ir analizuoti bei apdoroti jį (dėl pvz. žodinio susitarimo su dokumentą siuntusio skyriaus vadovu). Tačiau paliktas ir saugiklis. Vidutinės svarbos dokumentų eilei esant didesnei nei 4 dokumentai, seniausias dokumentas įgauna požymi svarbus, ir darbuotojas „verčiamas“ jį apdoroti. Buvo svarstyta ir galimybė tikrinti laiką, kiek dokumentas laukia eilėje, ir pasiekus tam tikra laiko ribą jį padaryti svarbiu, tačiau vieningas sprendimas dėl tokio laiko limito nebuvo priimtas.

Toliau, po dokumentų analizės reikia nutarti, ar dokumento turinys korektiškas, ir jei taip – dėti grupinį parašą ant dokumento ir pagal 1.3 skyriuje aprašytą kelią jį siųsti vienu žingsniu toliau, arba, jei turinys taisomas ne apdorojančio darbuotojo, atskleisti pasirašiusiojo tapatybę ir grąžinti jam dokumentą. Šios schemos grafinis vaizdas pateiktas paveikslėlyje 2.3.1



Paveikslėlis 2.3.1

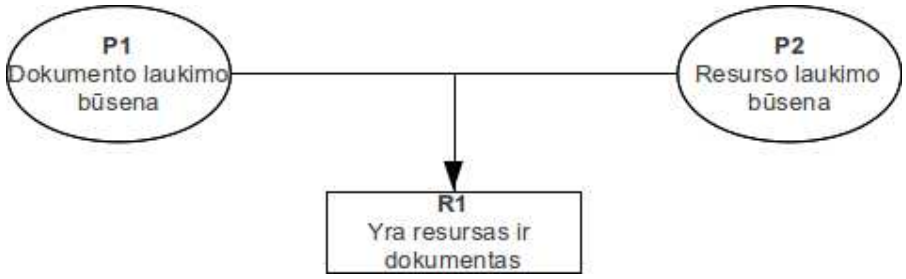
Taigi, turėdami schemos „griaučius“, kiekviename etape pasinaudodami Petri tinklų savybėmis, aprašytomis 1.3 skyriuje bei platformos SharePoint sistema galime suprojektuoti darbinę schemą, pagal kurią bus rašomas programos (t.y. darbo eigos) algoritmų paketas.

2.4 Schemos algoritmo konstravimas

Turėdami bazinę 2.3 skyriuje aprašytą schemos struktūrą, galime konstruoti schemą, pagal kurią sudarysime darbo eigos algoritmus.

Dokumento paėmimas apdorojimui priskiriant resursą

Šiam žingsniui aprašysime dvi pradines būsenas: P1 - „Dokumento laukimo būsena “ ir P2 - „Resurso laukimo būsena“. Šias būsenas sujungsime su pareiga R1 - „Yra resursas ir dokumentas“. Ši schemos dalis atvaizduota 2.4.1 paveikslėlyje.



Paveikslėlis 2.4.1

Į poziciją P1 proceso pradžioje atkeliauja žetonas (dokumentas) sukurtas mašininiu būdu (suformuluota ataskaita, užklausa , ar bet koks kitas dokumentas, sudarytas automatiškai ar žmogaus ir įkeltas į SP dokumentų biblioteką). Reikia atkreipti dėmesį, jog žmogus dokumentą kuria savo kompiuteryje pagal SharePoint pateikiamą šabloną, ir dokumentas į SharePoint sistemą patenka per sinchronizacijos aplinką SharePoint 2010 Workspace, kuri yra Office 2010 programų paketo dalis, arba žmogui rankiniu būdu įkeliant dokumentą į jam priklausančią SharePoint biblioteką. Įkėlus dokumentą registruojamas įvykis, kuris ir iškviečia darbo eigą.

Pozicijoje P2 žetonas reiškia laisvą darbuotoją (žmogų) galinti atlikti veiksmus su dokumentais. Šioje pozicijoje gali būti arba vienas arba daugiau žetonų, priklausomai nuo to, kokio įmonės skyriaus ar baro dokumentų judėjimą schema aprašo. Jeigu skyriuje ar bare yra daugiau nei vienas asmuo, turintis teisę atlikti veiksmus su dokumentais, tada žetonų į poziciją įkeliama tiek, kiek tokių žmonių yra. Kartą priskyrus darbuotojui dokumentą, dokumentų registro sąraše įrašomas žmogus, atsakingas už dokumento apdorojimą, ir dokumentui priskiriamos tokios teisės, jog tik tas darbuotojas galėtų jį matyti ir redaguoti. Iš tiesų tai labai praktiškas triukas, nes nereikia dokumento kilnoti į kitas bibliotekas, jis lieka toje pačioje bibliotekoje, tik būna paslėptas nuo kitų vartotojų bei darbo eigų. Taip, tam tikra prasme, susiejami du žetonai – darbuotojo ir apdorojamo dokumento. Tikrinant šią informaciją registre po to bus galima sekti, ar darbuotojo nelaukia daugiau nei 4 dokumentai ir kurį dokumentą jis peržiūri dabartiniu momentu.

Taigi, į skyrių (t.y. biblioteką, kurios prieigos teises turi skyriaus darbuotojai) atsiuntus dokumentą, ir esant laisvam vartotojui (registro atžvilgiu), pro sąlygą R1 praeina du žetonai, reiškiantys darbuotoją ir su juo susietą dokumentą. Tada startuoja kitos darbo eigos procedūros.

Skaitmeninė analizė, išskiriami požymiai.

Žetonams praėjus per pareigą R1, jie turi patekti į naujas pozicijas. Šias pozicijas pažymėkime P3 - „Dokumento skaitmeninės analizės būsena“ ir P4 - „ Resurso laukimo būsena “.

Kadangi žetonas, patekęs į poziciją P3 šiuo atveju žymi dokumentą, t.y. daugiamatį duomenų rinkinį o mūsų sistema realizuota SP 2010, dalį dokumento analizės galime palikti programinei

įrangai. Ji automatiškai atlieka dokumento pradinę validaciją ir tokių būdu prideda prie dokumento papildomas žymes (meta duomenis). Kaip jau minėjome 1.7 skyriuje svarbu patikrinti ar dokumentas sistemiškai teisingai suformuotas, ar jis buvo gražintas ir kokia jo svarba. Tai atlikus galima nustatyti, kokias paprogrames iškviesti ir į kokią biblioteką dokumentą perkelti. Todėl, P3 jungiama su keturiomis pareigomis: R2 - „Dokumentas blogai suformuotas“, R3 - „Dokumentas buvo gražintas“, R4 – „Dokumentas svarbus“ ir R5 - „Dokumentas normalios svarbos“. Po skaitmeninės analizės dokumentui priskiriami tokie požymiai, kurie atitinka pareigų kriterijus, t.y. žetono savybes pakeičiamos taip, jog jis praeina tik pro vieną pareigą.

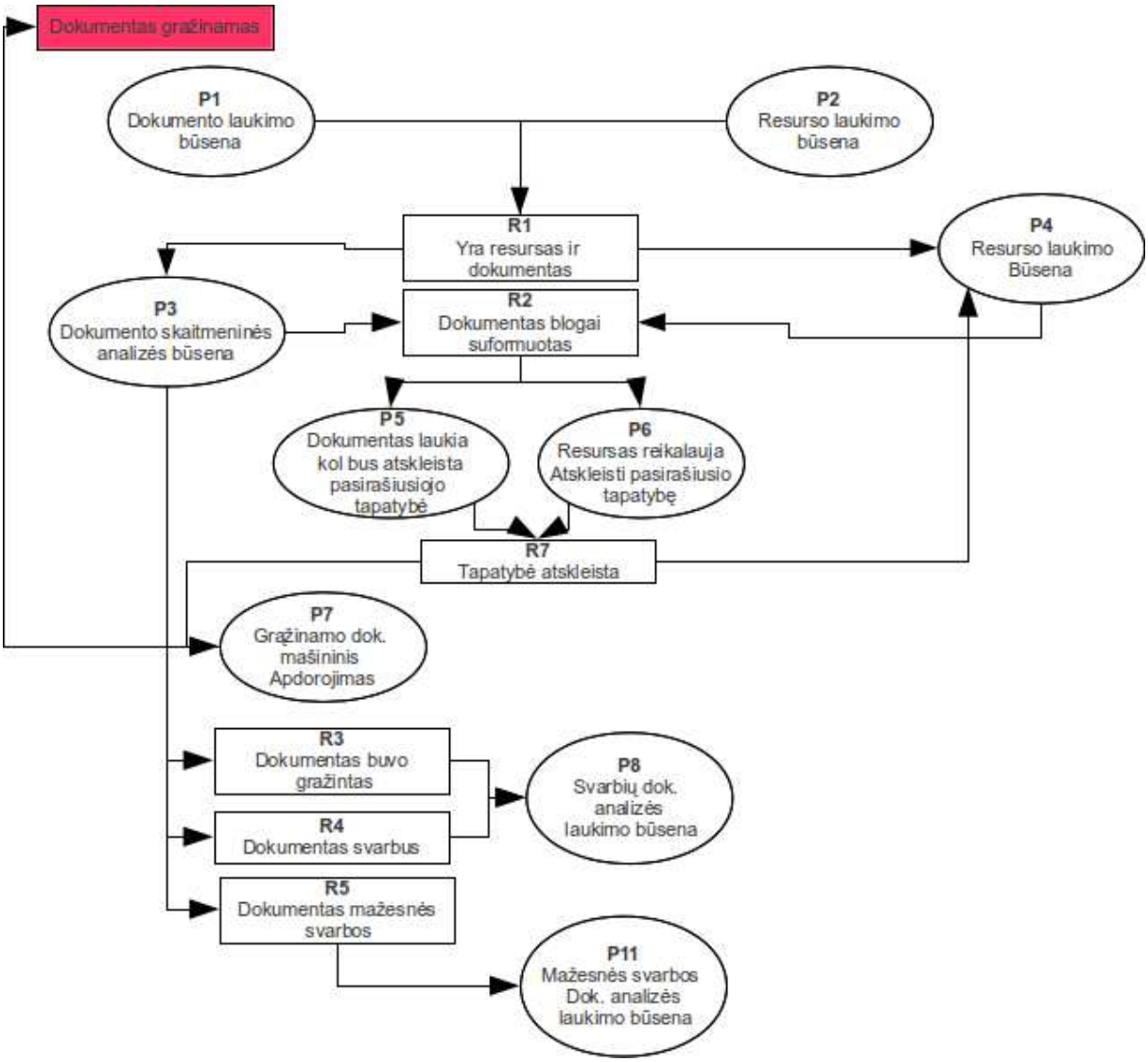
Kol atliekama dokumento skaitmeninės analizės būsena resursas patenka į laukimo būseną, P4. Iš šios būsenos resursas bus paskirtas atlikti tam tikrus veiksmus su dokumentais, priklausomai nuo to, pro kokią iš pareigų R2, R3, R4 ar R5 praeis dokumentą žymintis žetonas po mašininio apdorojimo. Apie šias galimybes pakalbėsime toliau nagrinėdami schemą.

Blogi (sistemos arba turinio prasme) dokumentai.

Norint realizuoti blogų dokumentų gražinamą, schemoje aprašysime dvi būsenas: P5 - „Resursas reikalauja atskleisti pasirašiusio tapatybę“ ir P6 - „ Dokumentas laukia kol bus atskleista pasirašiusiojo tapatybė “. Į šias būsenas dokumento ir resurso žetonai pateks per pareigą R2, jei dokumentas blogas sistemine prasme arba per pareigą R16, kai dokumento turinys reikalaus pataisymų, kuriu resursas pats atlikti negali, todėl dokumentą privalo gražinti. SharePoint sistemoje tai bus realizuota pakeičiant dokumento teises, t.y. jis nebus keliamas į biblioteką atitinkančią poziciją P6, jis liks pradinėje bibliotekoje su kitu teisių rinkiniu ir savybėmis. Kaip jau minėjome ankščiau 1.6 skyriuje tapatybės atskleidimas bus vykdomas grupės valdytojo vardu apie tai jį notifikuojant, taigi tai automatizuotas procesas, po kurio dokumento ir resurso žetonai pro pareigą R7 - „Tapatybė atskleista“ patenka atitinkamai į būsenas P7 - „Grąžinamo dok. mašininis apdorojimas“ ir P4.

Iš tiesų būsena P7 ne kas kita kaip skyriaus, iš kurio buvo gautas dokumentas biblioteka, į kurią perkeliamas dokumentas su teisių rinkiniu jį sudariusiam asmeniui. Jis atsidurs asmens, padariusio klaidą, svarbių dokumentų sąraše nueidamas kelią schema P1 – R1 – P3 – R3 – P8.

Patekęs į P4 resursas vėliau galės rinktis, ar apdoroti vidutinės svarbos dokumentus ar laukti naujų dokumentų. Aprašytas pozicijas ir pareigas prijunkime prie 2.4.1 paveikslėlyje pavaizduotos schemos ir gausime schemą, pavaizduotą 2.4.2 paveikslėlyje.

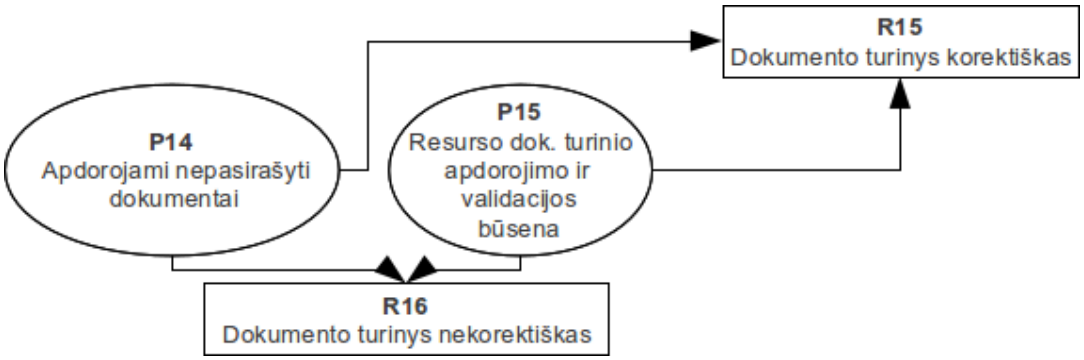


Paveikslėlis 2.4.2

Dokumentų turinio apdorojimas ir validacija

Dokumentų turinio validacijai realizuoti įvesime dvi būsenas P14 - „Apdorojami nepasirašyti mažesnės svarbos dokumentai“ ir P15 - „Resurso dok. turinio apdorojimo ir validacijos būsena “. Jose atitinkamai atsidurs dokumentą ir resursą žymintis žetonai, po to kai resursas imsis dokumento apdorojimo. Sistemoje tai bus realizuota tikrinant, ar resursas „atidarė“ dokumentą ieškant „Taken“ požymio registre arba „View State“ būsenoje. Apdorojimo metu dokumentas nematomas jokioms kitoms darbo eigoms ar vartotojams. Šios dvi pozicijos jungiamos su pareigomis R15 - „Dokumento turinys korektiškas“ ir R16 - „Dokumento turinys nekorektiškas“, vieną iš kurių žetonai praeis po analizės. Jeigu resursas nuspręs kad dokumento turinys nėra korektiškas ir jis nėra įgaliotas jo pataisyti, praėję R16 pareigą dokumento ir resurso žetonai atsidurs pozicijose P5 ir P6, kurios aprašytos ankščiau. Jeigu resursas nutars, kad dokumento turinys korektiškas, dokumentą ir

resursą žymintis žetonai keliaus į būsenas P9 ir P10 dokumento pasirašymui. Dokumento validacijos schemos dalis pavaizduota 2.4.3 paveikslėlyje



Paveikslėlis 2.4.3

Svarbūs dokumentai

2.4.2 paveikslėlyje matome, kad pro pareigas R3 ir R4 praėję žetonai patenka į būseną P8 - „Svarbių dok. analizės laukimo būsena“. Tai yra svarbių dokumentų eilė, kurios elementai susieti su tam tikrais resursais (kaip minėta ankščiau). Šie dokumentai reikalauja greito apdorojimo, todėl jei iš P3 pro R2 nepraeina nė vienas žetonas, t.y. nėra blogai suformuotų dokumentų, o P8 pozicijoje žetonas yra, sudarius pareigą R6 - „Svarbaus dok. apdorojimas“ pro ją iš P4 ir P8 praeis žetonai (prasidės svarbaus dokumento apdorojimas), ir pateks į pozicijas P14 ir P15, kur bus atliktas jų apdorojimas ir validacija (aprašyta ankščiau).

Vidutinės arba normalios svarbos dokumentai

2.4.2 paveikslėlyje matome, kad pro pareigą R5 praėję žetonai patenka į poziciją P11 - „Mažesnės svarbos dok. skaitmeninės analizės laukimo būsena “ . Ši pozicija ypatinga tuo, jog kelias iki jos šiek tiek skiriasi nuo kelių iki kitų pozicijų. Kadangi pozicija P4 sujungta su Pareigomis R2 ir R6, resursas pirma skiriamas į pozicijas P14 ir P15 su svarbiais arba gražintais dokumentais. Ir tik jei tokių nėra, iš P4 resursas, pro naujai aprašytą pareigą R10 - „Nėra nei svarbių, nei blogai suformuluotų dok.“ pateks į poziciją P12 - „Resurso pasirinkimo būsena“. Schemą projektuojame šitokiu būdu tam, kad darbuotojui būtų galimybė mokytis planuoti savo laiką ir tobulėti profesinėje srityje. Jeigu darbuotojas mato, jog dokumento turinio analizė jam aktuali, jis gali „rezervuoti“ sau dokumentą, tačiau jį apdoroti kiek vėliau. Taigi, darbuotojas „rezervavęs“ dokumentą gali grįžti prie bendros dokumentų eilės peržiūros, tada jo žetonas įveiktų pareigą R11 - „Laukti naujų dokumentų“ ir patektų į poziciją P2, arba peržiūrėti tik ką gautą (arba gautus seniau, jei į šią poziciją jis atkeliavo iš P4) dokumentus, įveikdamas pareigą R12 - „Peržiūrėti žemesnės svarbos dokumentus“ ir patekęs į poziciją P13 - „Resurso žemesnės svarbos dokumentų analizės būsena“. Šioje būsenoje resursas peržiūri visą sąrašą dokumentų, kuriuos jis rezervavo. Reikia atkreipti dėmesį, jog dokumentų

sąraše jis mato ne tik dokumento pavadinimą, bet ir dokumento meta duomenų reikšmes, todėl gali suvokti dokumento turinio esmę bei įvertinti jo svarbą. Keliant šią schemą į kitą sistemą (sistemą SAP) šioje vietoje gali atsirasti ne žmogus, bet programa, tačiau ji analogiškai, pagal kriterijų rinkinius galės pasirinkti, kokį veiksmą atlikti

Iš pozicijos P13 resursas per pareigas R13 - „Neapdoroti dokumentų“ ir R14 - „Apdoroti dokumentus“, (R14 sujungtas su pozicija P11) gali pasirinkti tolimesnę veiksmų eigą. Mes neverčiame resurso apdoroti dokumentų, jis gali tiesiog peržiūrėti jų sąrašą, įvertinti situaciją ir laukti naujų dokumentų grįždamas į P2, arba pasirinkti vieną iš P11 laukiančių dokumentų, ir jį apdoroti. Tokiu atveju iš P11 ir P13 per R14 iššaunami du žetonai, kurie patenka į P14 ir P15 apdorojimui (aprašyta ankščiau). Tačiau šioje schemoje dalyvaujant žmogui, svarbu įvertinti scenarijų, kada žmogus sudarys per didelę „rezervuotų“ dokumentų eilę. Kaip kalbėjome ankščiau, norint to išvengti reikia riboti eilės dydį iki 5 dokumentų. Tai padarysime įvesdami pareigą R8 - „Mažesnės svarbos susietų dok. daugiau nei 3“ tarp pozicijų P11 ir P8. Tokiu būdu, pozicijoje P11 atsiradus penktam dokumentui, kuris susietas su tuo pačiu resursu, pirmo dokumento (seniausio) žetonas šaunamas per R8 į P8, t.y. į svarbių dokumentų biblioteką. Tokiu būdu kitame cikle resursui priėmus penktą dokumentą, jis bus priverstas apdoroti arba penktą dokumentą (jei jis gražintas), arba ankščiau perkeltą į P8 pirmąjį, o po to tik ką gautą penktą dokumentą, jei jis svarbus.

Taigi, galutinė schema pagal aprašytas savybes pavaizduota 1 priede.

Dokumentų pasirašymas.

Kadangi elektroninio parašo infrastruktūros realizacijai reikalingi papildomi resursai, tolimesniuose skyriuose pristatysiu grupinio parašo modelį, kurį, esant galimybei, ketinu integruoti į schemą. Grupinio parašo konstrukcija pasirinkau todėl, jog atsiranda galimybė kiekvienam skyriaus atstovui pasirašinėti dokumentą skyriaus vardu. Tai gana aktualu pasirašant „jautraus“ turinio dokumentus, įvairius komercinius pasiūlymus ir pan.

2.5 Grupinio parašo schemos modelis

Tarkime kad $P = \{P_1,..P_n\}$- grupės, kuriai konstruojamas parašas, narių aibė, o M – tam tikras asmuo su aukštesnėmis privilegijomis, kurį vadinsime „Grupės valdytoju“. Tegul Γ - aibės $2^{\{1,...n\}}$ poaibis. Tada aibę $\{P_i \mid i \in S, s \in \Gamma\}$ vadinsime autorizuota koalicija, o Γ - valdančiąja struktūra. Ši struktūra privalo būti monotonine (t.y. dviems aibės $2^{\{1,...n\}}$ poaibiams S ir S', kai $S \in \Gamma \quad ir \quad S' \supseteq S$ turi galioti ir $S' \in \Gamma$).

Pastaba: Specialusis atvejis, kada ši struktūra nėra monotonine ($\Gamma = \{\{1\},\{2\},...,\{n\}\}$) vadinama paprastojo grupinio parašo schema.

Mūsų nagrinėjama apibendrinta grupinio parašo schema susideda iš keturių pagrindinių procedūrų.

1. Parengties (setup) – tikimybinė interaktyvaus protokolo procedūra, veikianti tarp grupės valdytojo M ir grupės narių P. Procedūros parametras Γ, gražinami rezultatai - grupės viešasis raktas Y, kiekvieno grupės nario slaptieji raktai $x_i \quad (atitinkama\, i \quad kiekvienam \quad P_i \in P)$ ir valdytojo M atskleidimo slaptasis raktas w.
2. Pasirašymo (sign) – tikimybinė interaktyvaus protokolo procedūra, veikianti tarp grupės narių ($P_i \mid i \in S, S \in \Gamma$). Procedūros parametrai: m – pasirašoma žinutė, grupės viešasis raktas Y, struktūra Γ, koalicija S ir slaptasis raktas x_i, priklausantis atitinkamiems grupės nariams $P_i \mid i \in S, S \in \Gamma$. Procedūra gražina žinutės m parašą s.
3. Tikrinimo (verify) – procedūros parametrai: žinutė m, grupės viešasis raktas Y, struktūra Γ ir parašas s. Procedūra gražina logines reikšmes „taip“ jei parašas teisingas ir „ne“ priešingu atveju.
4. Atskleidimo (open) – algoritmas, skirtas išaiškinti pasirašiusįjį grupės narį. Paramterai: žinutė m, viešasis grupės raktas Y, struktūra Γ, parašas s ir atskleidimo slaptasis raktas w. Jei s tikras m parašas, pasirašytas grupės viešuoju raktu Y, algoritmas grąžina $S \in \Gamma$ ir įrodymą, kad grupės nariai $P_i \mid i \in S$ tikrai pasirašė žinutę m.

Grupė publikuoja savo viešąjį raktą Y, struktūrą Γ ir kai kuriuos sistemos parametrus. Nagrinėjama parašo schema tenkina tokias savybes:

1. Nesuklastojamumas – tik autorizuotų koalicijų grupės nariai $P_i \mid i \in S, S \in \Gamma$ gali

pasirašyti žinutes. Parašo tikrumas bet kada gali būti patikrintas viešai prieinamais Y ir Γ.

2. Anonimiškumas ir neatsekamumas – turint grupinį parašą (arba kelis tos pačios grupės parašus) neįmanoma patikrinti nei kurios koalicijos grupės nariai jį sugeneravo (anonimiškumas), nei ar du skirtingi parašai buvo sugeneruoti tos pačios koalicijos narių (neatsekamumas).
3. Atskleidžiamumas ir neapšmeižiamumas – grupės nariai negali nei uždrausti ar sutrukdyti grupės valdytojui atskleisti pasirašiusiojo koalicijos nario tapatybės (atskleidžiamumas), nei pasirašyti žinutės kito grupės nario vardu (neapšmeižiamumas).
4. Grupės valdytojas gali dalyvauti tik vykdant algoritmus „Parengiamumas“ (setup) bei atskleidimas (open)

Šios keturios savybės pagrindinės grupinio parašo savybės. Kitos trys savybės, kurias pateikiu, yra papildomos probleminio pobūdžio savybės, kurios realizuojamos tam, kad grupės valdytojo funkcionalumas būtų tiksliau ir aiškiau apibrėžtas.

5. Grupės valdytojas privalo atskleisti pasirašiusiojo tapatybę tik būtiniausiais atvejais, tačiau būtiniausius atvejus jis įvardina pats, o tam reikalinga papildoma kontrolė.
6. Grupės valdytojai turėtų būti atskirti, valdytojas turi galėti arba tik atskleisti pasirašiusįjį, arba tik suteikti prieigos teises prie grupės naujiems nariams.
7. Tam kad išvengti valdytojo sukčiavimo, valdytojo teisės aprašytos 6 punkte turėtų priklausyti skirtingiems tos pačios grupės nariams.

Dabar aprašysime schemų konstravimo prielaidas.

2.6 Efektyvi paprasto grupinio parašo schema

Tarkime G – baigtinė, q eilės ciklinė grupė, o $g \in G$ toks šios grupės generatorius, kad diskrečiųjų logaritmų skaičiavimas grupėje G neįvykdomas. Tarkime H – 1.14 skyriuje aprašyta hash funkcija $H : \{0,1\}^* \to \{0,1\}^l$.

Aprašykime inicijavimo procedūrą. Tarkime grupės valdytojas M atsitiktinai pasirenka savo privatųjį raktą $w \in {}_q$ ir apskaičiuoja viešąjį raktą $z = g^w$ ir įteikimo reikšmę $SPK_1\{(\alpha): z = g^\alpha\}(M)$ jam. Ši įteikimo reikšmė išsiunčiama visiems grupės nariams. Tada, kiekvienas grupės P

narys P_i atlieka tokias operacijas:

- pasirenka savo privatųjį raktą $x_i \in Z_q$,
- suskaičiuoja savo viešuosius raktus $y_i = g^{x_i}$,
- suskaičiuoja savo viešojo rakto įteikimo reikšmę $SPK_1\{(\alpha): y_i = g^{\alpha}\}(P_i)$,
- išsiunčia savo viešąjį raktą ir rako įteikimo reikšmę grupės valdytojui.

Grupė publikuoja viešųjų raktų rinkinį $Y = \{y_1, y_2, .., y_n\}$ kartu su grupės valdytojo viešuoju raktu z ir sisteminiais parametrais *G,g,q* ir *H*. Taigi, grupės, norėdamas pasirašyti žinutę *m*, užšifruoja vieną iš *Y* elementų (savo viešąjį raktą), ir pateikia parašą (SPK) žinutei m įrodydamas kad

- tikrai jis užšifravo viešąjį raktą (naudodamasis grupės valdytojo viešuoju raktu),
- jis žino užšifruoto viešojo rakto diskretųjį logaritmą.

Iš šių dviejų išvadų seka, kad grupės nariui užšifravus savo viešąjį raktą grupės valdytojo viešuoju raktu, grupės valdytojas gali lengvai nustatyti pasirašiusįjį grupės narį.

Grupės nario P_j pasirašymo procedūra žinutei m vykdoma tokiu būdu.

- Narys pasirenka atsitiktinį $a \in Z_q$.
- Užšifruoja savo privatųjį raktą y_j apskaičiuodamas $A = z^a \quad ir \quad B = y_j g^a$.
- Skaičiuoja parašą $V_1 := SPK_{10}\{(\alpha_i)_{i=1,...n; j \in I_i} : \bigvee_{i=1}^{n} \left(A = z^{\alpha_i}\right) \wedge \left(\frac{B}{y_i} = g^{\alpha_i}\right)\}(m)$
- Skaičiuoja parašą $V_2 := (c', s') := SPK_1\{(\alpha): B = g^{\alpha}\}(V_1)$

Taigi parašas sudaromas iš keturių komponenčių: (A, B, V_1, V_2)ir gali būti patikrintas tikrinant (V_1, V_2). Parašas V_1 užtikrina, kad (*A, B*) yra vieno iš grupės viešųjų raktų *Y* rakto šifras. Parašas V_2 užtikrina, kad pasirašiusysis iš tiesų žino viešojo rakto, kurį panaudojo pasirašydamas, diskretųjį logaritmą. Vadinasi, pasirašiusysis įrodo diskrečiojo logaritmo tam tikram raktui iš raktų rinkinio *Y* žinojimą, vadinasi jis iš tiesų yra grupės *P*, kuriai priklauso raktai *Y* narys.

Tam kad atskleistų grupės nario, pasirašiusio žinutę *m* tapatybę, grupės valdytojas iššifruoja (*A, B*) ir gauna pasirašiusiojo grupės nario viešąjį raktą. Tam, kad vienareikšmiškai įrodytų jog šis viešasis raktas tikrai priklauso tam tikram *j*-tajam grupės nariui, grupės valdytojas gali paskaičiuoti

$SPK_6\{(\alpha): z = g^{\alpha} \wedge A = \left(\frac{B}{y_j}\right)^{\alpha}\}(P_j)$, ir įrodyti, kad raktas tikrai nario *j*.

2.7 Teorinė grupinio parašo saugumo analizė

Kadangi grupinio parašo modelis nebuvo programiškai realizuotas, pateikiu tik jo teorinę saugumo analizę.

Mūsų parašo schemos saugumas, kaip ir daugelio kripografinių schemų atvejų, paremtas DLP uždavinio sudėtingumu ir Hash funkcijų parinkimu.

Kalbant apie hash funkcijas, galime išskirti dažniausiai praktikoje naudojamas MD5 ir SHA-1 hash funkcijų šeimas, tačiau prieš maždaug 6 metus Microsoft korporacija oficialiai paskelbė, jog šios funkcijos nebebus naudojamos kompanijos kuriamų produktų saugumo schemose. Kyla klausimas kodėl. Kaip matėme mūsų schemoje, kaip ir kitose schemose, elektroniniu parašu pasirašomas hash funkcijos rezultatas. Taigi, jei pasirašyto dokumento turinys tam tikrame etape pakeičiamas, pasikeičia, perdavus dokumentą kaip argumentą hash funkcijai gausime kitą rezultatą, kuriam ankstesnis parašas nebebus teisingas.

Dabar įsivaizduokime tokią schemą. Zita siunčia Broniui pasirašyti dokumentą *m*. Ji nori Bronių apgauti, ir dokumento m parašą pritaikyti kitam dokumentui m'. Tačiau Bronius pasirašo *H(m)* rezultatą, taigi Zita turi sugeneruoti tokį *m'* kad *H(m)=H(m').* Sugeneravusi m', kurio turinys kitoks, bet hash reikšmė tokia pati, Zita gali uždėti m dokumento parašą dokumentui m' ir jis bus laikomas teisėtai pasirašytu Broniaus. Šis atvejis vadinamas hash funkcijos sutapimu (collision), dėl to hash funkcijoms ir keliamas atsparumo sutapimui reikalavimas (1.14 skyrius, 2 savybė).

Taigi, sukčių tikslas rasti du hash funkcijos argumentus, kurių rezultatas būtų toks pat. Tai atrodytu sudėtingas uždavinys, tačiau sukčiams jį palengvina tikimybinis paradoksas, vadinamas „gimimo dienos problema“.

Įsivaizduokime, jog jums reikia sudaryti tokią grupę kurioje būtų bent du žmones, kurių gimimo diena sutampa. Turbūt galima intuityviai nuspėti, jog subūrus 367 žmonių grupę tikimybė, jog bent du iš jų gimė tą pačią dieną bus lygi 1 (100%). Tačiau intuityviai nuspėti negalime to, jog surinkus tik 23 žmonių grupę ši tikimybė tampa lygi 0,5 (50%). Vadinasi, tokiu būdu galime generuoti tam tikro dydžio pseudoatsitiktinius rinkinius, kuriuos naudodami kaip hash funkcijos argumentus galime rasti sutapimą. Panagrinėkime lentelę paveikslėlyje 2.7.1

Bitai	Galimų hash rezultatų skaičius	Fiksuota tikimybė gauti dviem vienodoms reikšmėms (susikirtimui)				
		0.1%	**1,00 %**	**25,00 %**	**50,00 %**	**75,00 %**
16	6.6×10^{4}	11	36	1.9×10^{2}	3.0×10^{2}	4.3×10^{2}
32	4.3×10^{9}	2.9×10^{3}	9.3×10^{3}	5.0×10^{4}	7.7×10^{4}	1.1×10^{5}
64	1.8×10^{19}	1.9×10^{8}	6.1×10^{8}	3.3×10^{9}	5.1×10^{9}	7.2×10^{9}
128	3.4×10^{38}	8.3×10^{17}	2.6×10^{18}	1.4×10^{19}	2.2×10^{19}	3.1×10^{19}
256	1.2×10^{77}	1.5×10^{37}	4.8×10^{37}	2.6×10^{38}	4.0×10^{38}	5.7×10^{38}
384	3.9×10^{115}	2.8×10^{56}	8.9×10^{56}	4.8×10^{57}	7.4×10^{57}	1.0×10^{58}
512	1.3×10^{154}	5.2×10^{75}	1.6×10^{76}	8.8×10^{76}	1.4×10^{77}	1.9×10^{77}

Lentelė 2.7.1

Matome, kad turėdami 128 bitų hash funkciją turėsime $2.2\text{x}10^{19}$ galimų rezultatų, ir parinkę $2.2\text{x}10^{19}$ argumentų, tikimybę gauti du vienodus rezultatus lygi 50%. Matome tendenciją, naudojant „gimtadienio“ ataką su 50% tikimybe rasti susikirtimą, tam tikra prasme silpniname funkcijos saugumą, rakto ilgį mažindami per pusę, t. y. rasti susikirtimą 128 bitų rakto ilgio hash funkcijai reikia skaičiuoti beveik tiek pat reikšmių, kiek brutalios jėgos ataką naudojant prieš 64 bitų funkciją. Detalesnė tikimybių analizės lentelė pateikta 4 priede.

Taigi, siekiant užtikrinti saugumą gimtadienio atakų atžvilgiu, realizuodamas grupinį parašą ketinu naudoti vieną iš SHA-2 hash šeimos funkcijų. Pasak literatūros šaltinių šios šeimos funkcijoms kol kas nerastas būdas apskaičiuoti sutapimui.

Kitas svarbus sistemos saugumo aspektas – DLP neišsprendžiamumas. Kyla klausimas, kodėl ši problema bazinė, ir kodėl tokia didelė jos svarba. 2.6 skyriuje aprašyta, kaip grupės valdytojas pasirenka savo privatųjį raktą w ir susiskaičiuoja viešąjį raktą *z*: $z = g^{w}$. Grupė publikuoja grupės generatorių *g*. Vadinasi sukčiai gali paskaičiuoti $\log_g z = \log_g g^{w} = w$ t. y. Sukčiai gali surasti grupės valdytojo privatųjį raktą! Analogiškai sukčiai gali suskaičiuoti ir bet kurio grupės nario privačiuosius raktus. Štai todėl labai svarbu parinkti tokią grupės eilę, kad DLP taptų neišsprendžiama.

Tokiems uždaviniams spręsti šiuo metu naudojami funkcijų lauko rėčio algoritmai bei jų kombinacijos. Naudojant šiuos algoritmus, geriausi kol kas pasiekti rezultatai buvo kriptografų Antoine Joux ir Reynald Lercier. Jiems, sujungus keturis šešiolikos 1,3 GHz branduolių kompiuterius Teranova kompiuterius, 2005 metų rugsėjo 23 dieną, po 17 dienų skaičiavimų pavyko apskaičiuoti diskrečiuosius logaritmus elementams iš grupės, kurios eilė 2^{613}. Manoma, kad parinkus grupę, kurios eilė 2^{2048} arba 2^{1024}, net sujungus visus žemėje esančius kompiuterius nepavyktų išspręsti DLP. Taigi savo schemoje, DLP saugumui užtikrinti naudosiu grupes, kurių eilė 2^{1024}.

2.8 Eksperimentinis tyrimas SharePoint aplinkoje

Schema buvo realizuota SharePoint 2010 platformoje ir išbandyta administracinėje įmonėje. Ji tapo pilnai funkcionali produkcinėje sistemoje 2001-04-08. Sistema naudojosi 23 darbuotojai. Didžioji dauguma darbuotojų – vadovai, todėl jiems schema veikė kaip „skyriui iš vieno asmens“. Panaudoti penki pagrindiniai dokumentų šablonai, dvi prašymų formos ir trys ataskaitų. Ilgiausias dokumento kelias – keturi skyriai, trumpiausias – trys. Dviems dokumentų šablonams buvo galimas gražinimas.

Rezultatais laikysime dokumentų šablonų aptarnavimo trukmes prieš, ir po schemos realizacijos produkcinėje sistemoje. Rezultatai pateikiami tokiu formatu:

1. Šablonas – šablono identifikacijos numeris.
2. P1 – P7 pozicijų, kuriose buvo šablonas, apdorojimo pabaigos laikai.
3. Archyvas – laikas, kada dokumentas buvo patalpintas į archyvą (sutampa su paskutinės pozicijos laiku arba vėluoja viena minute dėl apvalinimo).

Šie rezultatai gauti iš proceso eigos įrašų duomenų bazės. Nepaimti tie rezultatai, kurie dėl įvairių priežasčių, pateikiamų interpretacijoje, galėjo būti iškreipti. Papildoma rezultatų analizė, atlikta skaičiuokle pateikiama šiuose stulpeliuose:

4. Bendra Trukmė – laiko skirtumas tarp dokumento sukūrimo datos ir jo suarchyvavimo datos.
5. Korekcija darbo laike – korekcija, reikalinga norint paskaičiuoti darbo valandas. Jei procesas pradėtas vieną darbo dieną o baigtas kitą, reikia išskaičiuoti nakties ir nedarbo laiko tarpą.
6. Trukmė darbo valandomis – darbo laikas, sugaištas dokumento apdorojimui.

Rezultatai buvo renkami tokiu būdu: tam tikroje pozicijoje baigus dokumento apdorojimą, laiko žymė, kada tai atlikta, buvo dedama į duomenų bazės stulpelį. Jei dokumentas buvo nusiųstas toliau ar grąžintas, laiko reikšmė dedama į kitos pozicijos stulpelį. Taigi jei dokumento kelią sudaro 3 skyriai turėsime 3 laiko įrašus ir archyvavimo datą, jei 3 skyriai, tačiau buvo vienas gražinimas vienu žingsniu atgal – šešias laiko žymas. Analizuojami neapdoroti ir apdoroti rezultatai pateikiami atitinkamai antrame ir trečiame prieduose.

2.9 Rezultatų analizė ir interpretacijos

Gautus stebėjimų rezultatus analizavau tokiu būdu:

1. suskirsčiau gautus dokumentų aptarnavimo laikus pagal šablonus

2. stebėjau aptarnavimo laikų pokyčius prieš ir po schemos realizacijos sistemoje.

Žemiau pateikiu dokumentų aptarnavimo laikų grafikus. *X* ašyje pažymėtos aptarnavimo datos, *Y* - aptarnavimo laikai.

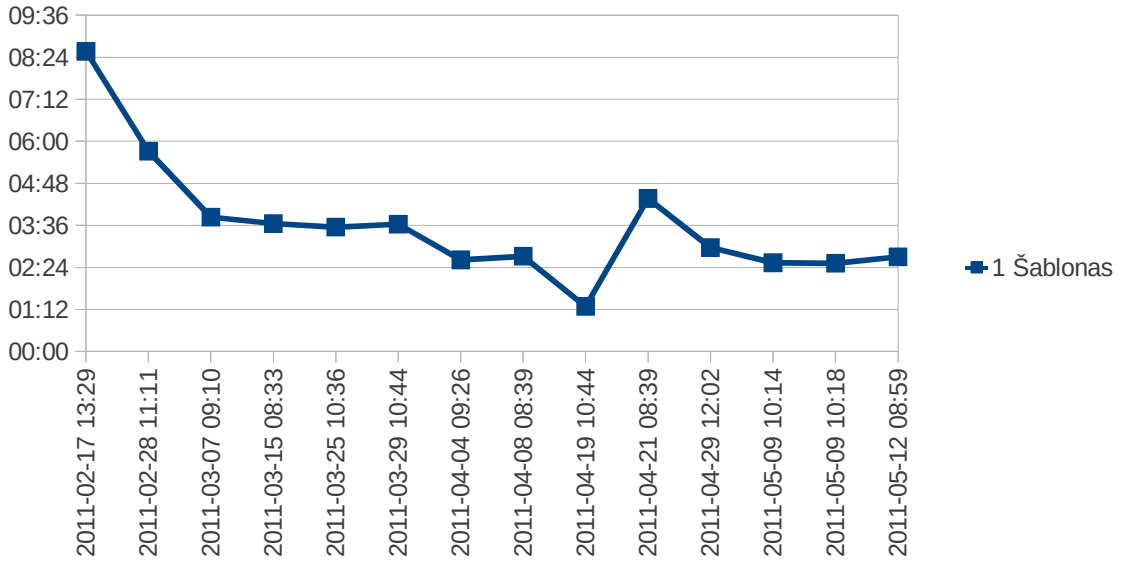


Diagrama 2.9.1

Diagramoje 2.9.1 matome, kad pradėjus veikti schemai dokumentų aptarnavimas šiek tiek pagreitėjo. Aptarnavimo laiko vidurkis iki schemos veikimo datos 4 h 17 min, po – 2 h 43 min. Iki aptarnavimo schemos realizavimo datos 1 šablono dokumentams įvykdyti du gražinimai, kurie užtruko labai ilgai, dėl to smarkiai įtakojo vidurkį. Po schemos realizavimo datos buvo tik vienas gražinimas. Matoma aptarnavimo laiko tendencija, bet dėl palyginti nedidelio stebėjimo skaičiaus ji nėra labai ryški. Panagrinėkime kitų šablonų dokumentų aptarnavimo tendencijas.

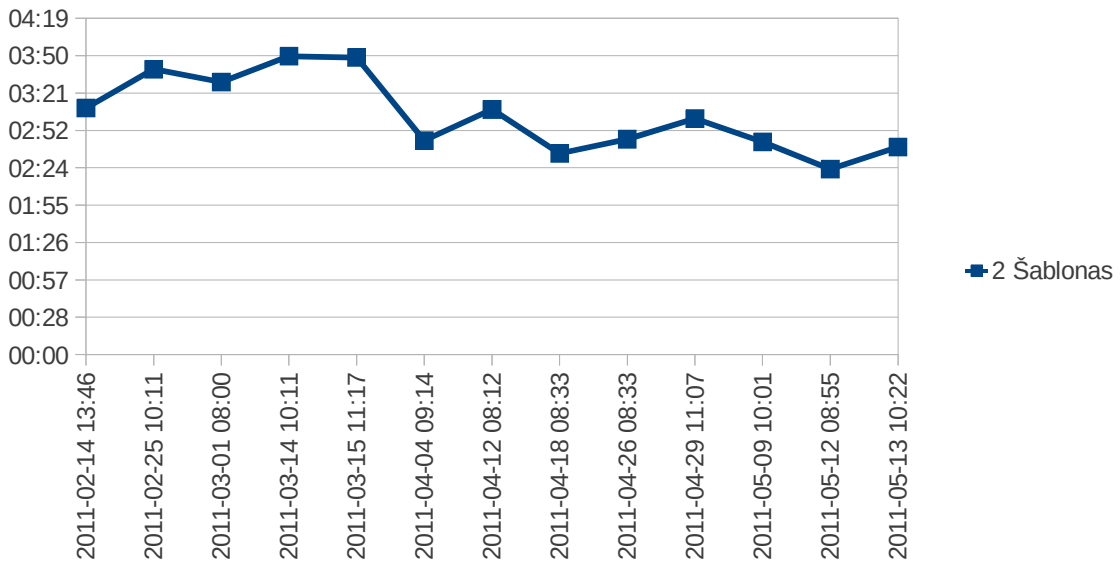


Diagrama 2.9.2

Diagramoje 2.9.2 matome, kad dokumentų aptarnavimo laikas po schemos realizacijos sistemoje taip pat pagreitėjo, bet šis pagreitėjimas vizualiai nėra labai ryškus. Aptarnavimo laikų vidurkiai iki ir po schemos realizacijų atitinkamai 3 h 27 min ir 2 h 45 min. Dokumentų grąžinimų nebuvo. Manau tai gana geras rezultatas, taipogi galima sakyti, jog šablonas paruoštas gana gerai, nes nebuvo dokumentų gražinimo nei dėl sisteminių, nei dėl turinio klaidų.

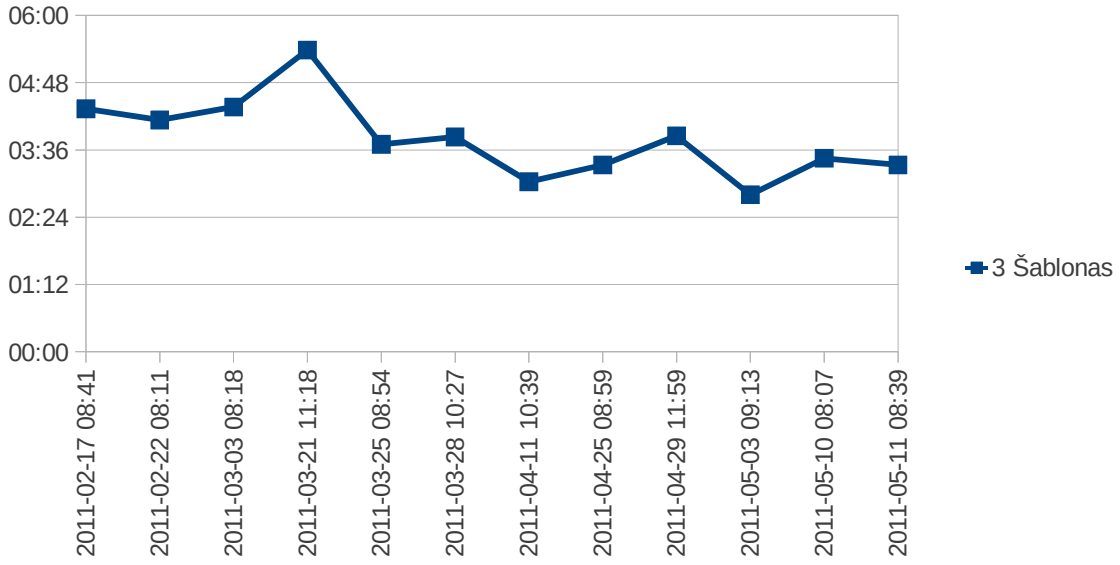


Diagrama 2.9.3

Diagramoje 2.9.3 kaip ir diagramose 2.9.1 ir 2.9.2 matyti tendencingas aptarnavimo laiko sumažėjimas įsigaliojus schemai. Vidutiniai aptarnavimo laikai iki ir po schemos įsigaliojimo atitinkamai 4 h 17 min ir 3 h 18 min.

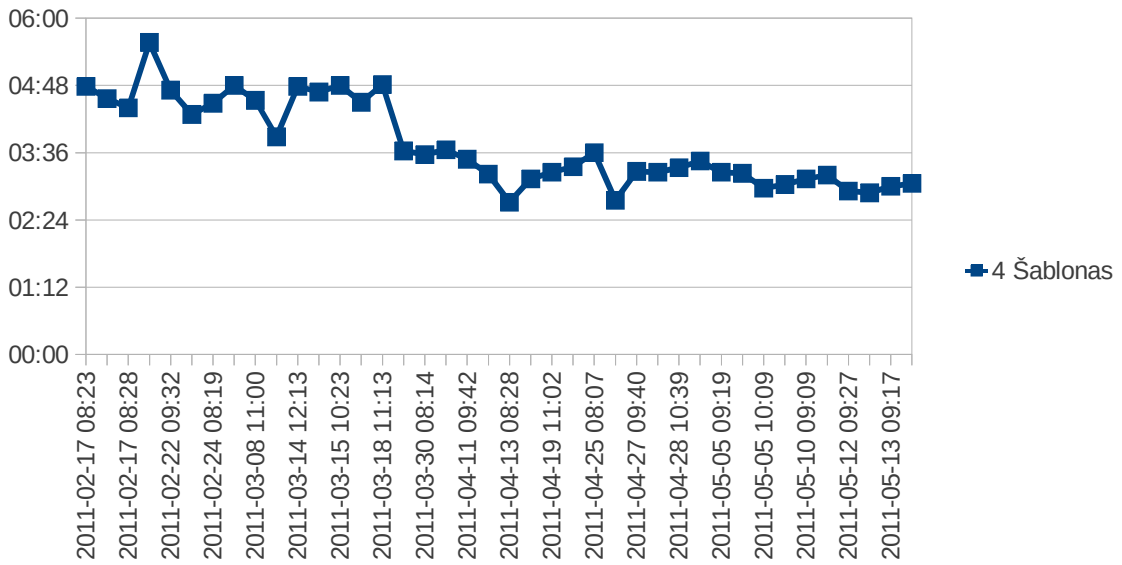


Diagrama 2.9.4

2.9.4 diagramoje matome, kad 4 dokumento šablono dokumentų sistemoje cirkuliavo kur kas daugiau nei pirmųjų trijų dokumentų šablonų. Analogiškai ankščiau aptartoms schemoms matyti tendencingas aptarnavimo laiko mažėjimas po schemos įsigaliojimo. Šioje schemoje, skirtingai nei prieš tai buvusiose matyti aptarnavimo laiko sumažėjimas dar iki schemos įsigaliojimo. Tai susiję su darbuotoju „apsimokymu“ ir susitarimus. Tačiau pradėjus veikti optimizavimo schemai, net ir šis, nusistovėjęs laikas tendencingai sumažėjo. Tai labai geras rezultatas parodantis, jog optimizavimo schema veiksminga. Vidutiniai aptarnavimo laikai iki ir po schemos įsigaliojimo atitinkamai 4 h 28 min ir 3 h 9 min. Grąžinimu nebuvo, todėl galime daryti prielaidą, jog dokumento šablonas paruoštas gerai.

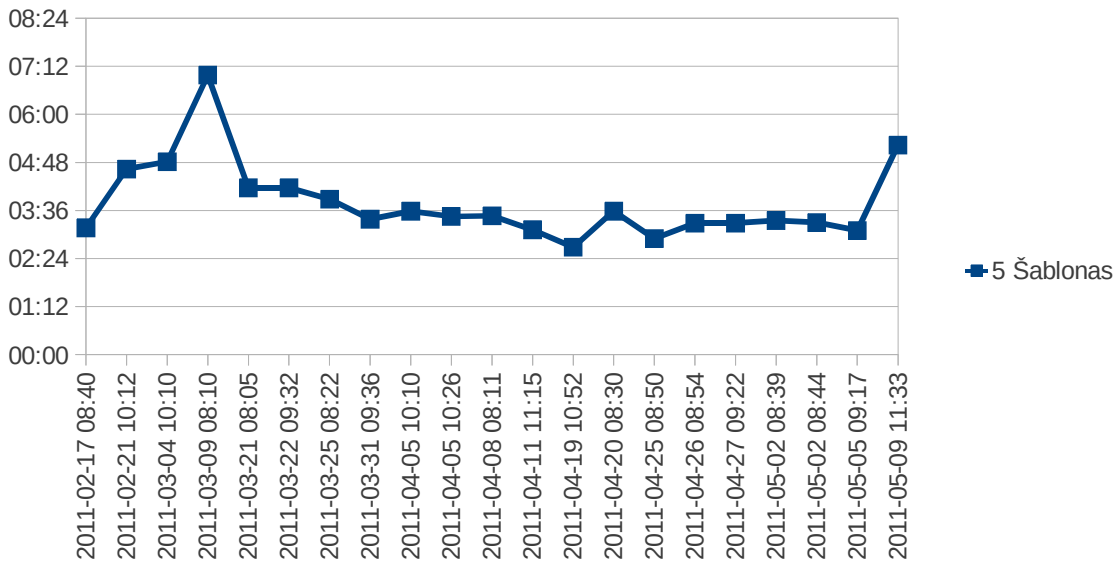


Diagrama 2.9.5

Diagramoje 2.9.5 kaip ir diagramoje 2.9.4 matyti dokumentų aptarnavimo spartos padidėjimo bei darbuotojų apsimokymo tendencijos, tačiau net ir darbuotojams apsimokius, aptarnavimo laikas pagreitėjo pradėjus veikti schemai. Aptarnavimo laikų vidurkiai prieš ir po schemos pritaikymo 4h 13 min. ir 3 h 23 min. Buvo atlikti du gražinimai, kurie įtakojo vidurkius, tačiau aptarnavimo spartos tendencija vis tiek matoma.

Schemos efektyvumas turėtu dar labiau išryškėti pritaikius ją gamybinėse įmonėse. Jose didesni ir dokumentų srautai, ir skyriai, į kuriuos dokumentai keliauja, taigi dokumentui aptarnauti bus daugiau laisvų resursų.

3. Išvados

Darbe pasiūlyta ir realizuota e-dokumentų valdymo schema, paremta Petri tinklų principais ir perkelta į SharePoint 2010 platformą. Realizuota sistema leido ženkliai sutrumpinti dokumentų cirkuliacijos laiką lyginant su standartiniu platformos sprendimu. Dokumentų cirkuliacijos laikas vidutiniškai sutrumpėjo viena valanda kiekvienam tirtam šablonui.

Darbe sudarytas grupinio e-parašo modelis, numatyta ir aprašyta efektyvi jo integracija į darbe realizuotą e-dokumentų valdymo schemą. Atlikta schemos saugumo analizė ir parinkti saugūs e-parašo schemos parametrai.

4. Rekomendacijos

Rekomenduojama darbe aprašytą e-dokumentų valdymo sistemą integruoti į kitas dokumentų valdymo platformas tam, kad būtų pagerintos sistemų funkcinės charakteristikos.

Rekomenduojama integruoti pasiūlytą grupinį e-parašą kartu su pasiūlyta dokumentų valdymo schema tam, kad užtikrinti dokumentų autentiškumą ir integralumą.

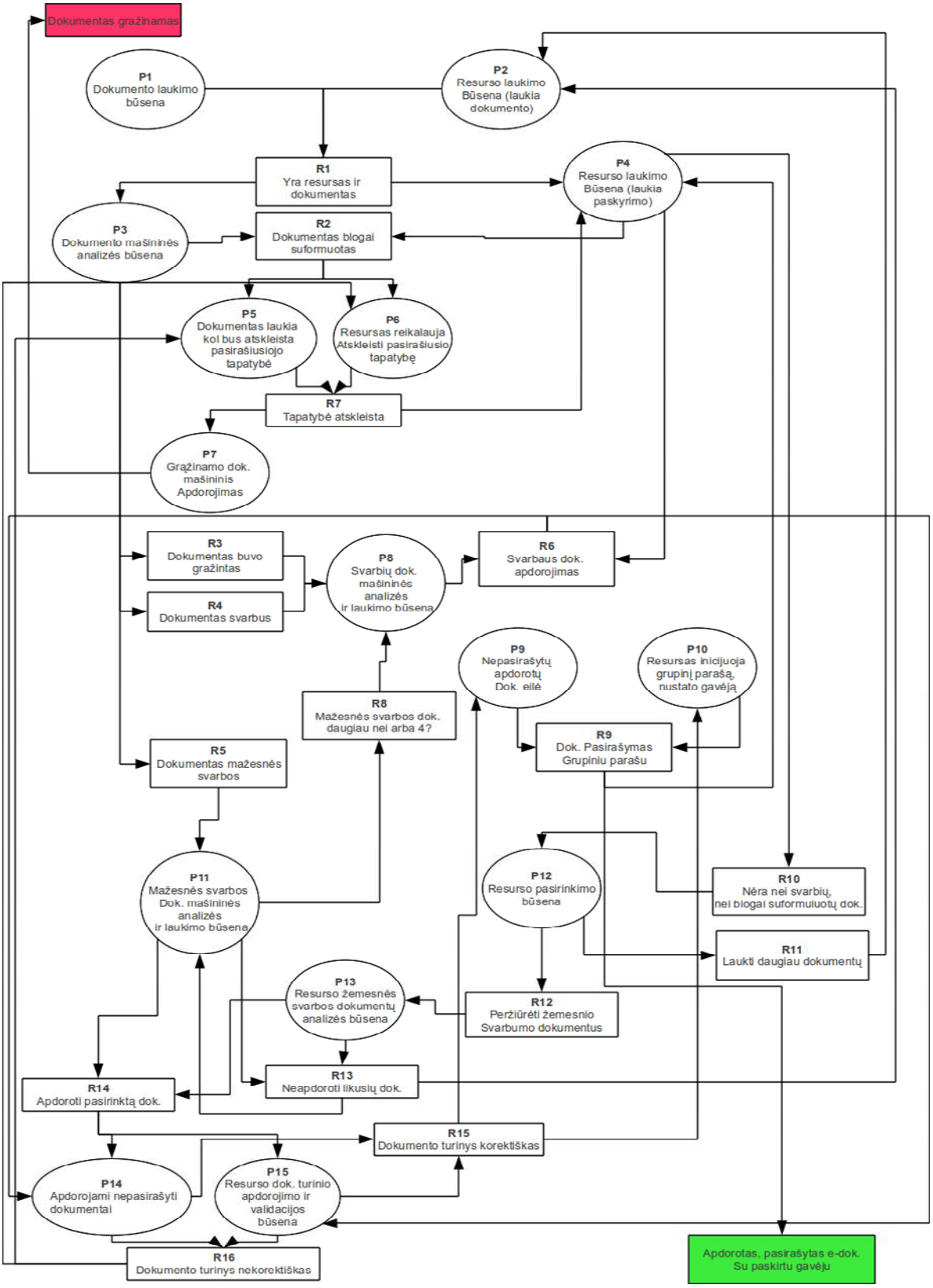
Literatūros sąrašas

1. Henrikas Pranevičius. Sudėtingų sistemų formalizavimas ir analizė. 219-225 p. Kauno technologijos niversitetas, 2008.
2. Paul A. Fishwick. Simulation Model Design and Execution. Pearson Education POD; 1st edition. 1995.
3. Chris Ling. Lecture on Petri Nets Method. 2001
4. Neal Koblitz. A course in number theory and cryptography. Springer-Verlag, second edition, 1994.
5. Evangelos Kranakis. Primality and Cryptography. Wiley- Teubner Series in Computer Science, 1986.
6. Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Van- stone. Handbook of Applied Cryptography. CRC Press, Boca Raton, FL, 1997.
7. Kevin McCurley. The discrete logarithm problem. In Carl Pomerance, editor, Cryptology and computational number theory, volume 42 of Proceedings of Symposia in Applied Mathematics, p. 49–74. American Mathematical Society, 1990.
8. Andrew M. Odlyzko. Discrete logarithm and smooth polynomials. In Gary L. Mullen and Peter Jau-Shyong Shiue, editors, Finite Fields: Theory, Applications and Algorithms, vol. 168 of Contemporary Mathematics, p. 269–278. American Mathematical Society, 1994.
9. Henri Cohen. A Course in Computational Algebraic Number Theory. Number 138 in Graduate Texts in Mathematics. Springer-Verlag, Berlin, 1993.
10. Stefan Brands. An efficient off-line electronic cash system based on the representation problem. Technical Report CS-R9323, CWI, April 1993.
11. Whitfield Diffie and Martin E Hellman. New directions in cryptography. IEEE Trans. on Information Theory, IT- 22(6):644–654, Nov. 1976.
12. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In George Robert Blakley and David Chaum, editors, Advances in Cryptology — CRYPTO ’84, volume 196 of Lecture Notes in Computer Science, p. 10–18. Springer Verlag, 1985.
13. Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete

logarithms. IEEE Trans. On Information Theory, IT-31(4):469–472, July 1985.

14. David Chaum, Jan-Hendrik Evertse, and Jeroen van de Graaf. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In David Chaum and Wyn L. Price, editors, Advances in Cryptology — EUROCRYPT ’87, volume 304 of Lecture Notes in Computer Science, p. 127–141. Springer-Verlag, 1988.
15. Claus P. Schnorr. Efficient signature generation for smart cards. Journal of Cryptology, 4(3):239–252, 1991.
16. Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, p. 33–43, Seattle, Washington, 15–17 May 1989. ACM.
17. Ivan Bjerre Damgård. Collision free hash functions and public key signature schemes. In David Chaum and Wyn L. Price, editors, Advances in Cryptology — EURO- CRYPT ’87, volume 304 of Lecture Notes in Computer Science, p. 203–216. Springer-Verlag, 1988.
18. John M. Pollard. Monte Carlo methods for index computation (mod p). Mathematics of Computation, 32(143):918–924, July 1978.
19. Function field sieve http://en.wikipedia.org/wiki/Function_field_sieve. 2011
20. Discrete logarithm records. http://en.wikipedia.org/wiki/Discrete_logarithm_records. 2011

1 Priedas. DVS Schema



2 Priedas. Pradiniai duomenys

sablonas	P1	P2	P3	P4	P5	P6	P7	archyvas
2	2011-02-14 13:46	2011-02-14 14:37	2011-02-14 16:55	null	null	null	null	2011-02-14 16:56
4	2011-02-17 08:23	2011-02-17 09:19	2011-02-17 12:10	2011-02-17 13:09	null	null	null	2011-02-17 13:10
4	2011-02-17 08:26	2011-02-17 09:20	2011-02-17 12:18	2011-02-17 12:59	null	null	null	2011-02-17 13:00
4	2011-02-17 08:28	2011-02-17 09:00	2011-02-17 11:52	2011-02-17 12:51	null	null	null	2011-02-17 12:52
5	2011-02-17 08:40	2011-02-17 09:38	2011-02-17 11:49	null	null	null	null	2011-02-17 11:50
3	2011-02-17 08:41	2011-02-17 11:12	2011-02-17 13:00	null	null	null	null	2011-02-17 13:01
1	2011-02-17 13:29	2011-02-17 14:25	2011-02-17 17:00	2011-02-18 09:35	2011-02-18 10:28	2011-02-18 13:02:00	null	2011-02-18 13:03
4	2011-02-21 08:12	2011-02-21 09:11	2011-02-21 11:12	2011-02-21 12:45	null	null	null	2011-02-21 13:46
5	2011-02-21 10:12	2011-02-21 11:02	2011-02-21 14:00	2011-02-21 14:50	null	null	null	2011-02-21 14:50
4	2011-02-22 09:32	2011-02-22 10:25	2011-02-22 13:15	2011-02-22 14:14	null	null	null	2011-02-22 14:15
3	2011-02-22 08:11	2011-02-22 10:29	2011-02-22 12:19	null	null	null	null	2011-02-22 12:19
4	2011-02-23 08:23	2011-02-23 09:01	2011-02-23 11:41	2011-02-23 12:39	null	null	null	2011-02-23 12:40
4	2011-02-24 08:19	2011-02-24 09:07	2011-02-24 11:51	2011-02-24 12:48	null	null	null	2011-02-24 12:48
2	2011-02-25 10:11	2011-02-25 11:11	2011-02-25 13:51	null	null	null	null	2011-02-25 13:51
1	2011-02-28 11:11	2011-02-28 11:58	2011-02-28 14:22	2011-02-28 14:52	2011-02-28 16:54	null	null	2011-02-28 16:54
2	2011-03-01 08:00	2011-03-01 09:00	2011-03-01 11:30	null	null	null	null	2011-03-01 11:30
4	2011-03-02 08:12	2011-03-02 09:07	2011-03-02 12:01	2011-03-02 13:00	null	null	null	2011-03-02 13:00
3	2011-03-03 08:18	2011-03-03 10:50	2011-03-03 12:40	null	null	null	null	2011-03-03 12:40
5	2011-03-04 10:10	2011-03-04 11:00	2011-03-04 14:00	2011-03-04 14:58	null	null	null	2011-03-04 14:59
1	2011-03-07 09:10	2011-03-07 10:03	2011-03-07 13:00	null	null	null	null	2011-03-07 13:00
4	2011-03-08 11:00	2011-03-08 12:00	2011-03-08 14:39	2011-03-08 15:32	null	null	null	2011-03-08 15:32
5	2011-03-09 08:10	2011-03-09 09:08	2011-03-09 11:28	2011-03-09 12:11	2011-03-09 15:09	null	null	2011-03-09 15:09
4	2011-03-10 11:13	2011-03-10 11:13	2011-03-10 14:08	2011-03-10 15:06	null	null	null	2011-03-10 15:06
2	2011-03-14 10:11	2011-03-14 11:09	2011-03-14 14:01		null	null	null	2011-03-14 14:01
4	2011-03-14 12:13	2011-03-14 13:13	2011-03-14 16:02	2011-03-14 17:00	null	null	null	2011-03-14 17:00
4	2011-03-14 13:09	2011-03-14 14:03	2011-03-14 17:00	2011-03-15 08:50	null	null	null	2011-03-15 08:50
4	2011-03-15 10:23	2011-03-15 11:22	2011-03-15 14:21	2011-03-15 15:11	null	null	null	2011-03-15 15:11
1	2011-03-15 08:33	2011-03-15 09:23	2011-03-15 12:12	null	null	null	null	2011-03-15 12:12
2	2011-03-15 11:17	2011-03-15 12:11	2011-03-15 15:06	null	null	null	null	2011-03-15 15:06
4	2011-03-18 08:43	2011-03-18 09:33	2011-03-18 12:14	2011-03-18 13:13	null	null	null	2011-03-18 13:13
4	2011-03-18 11:13	2011-03-18 12:10	2011-03-18 15:08	2011-03-18 16:02	null	null	null	2011-03-18 16:02
5	2011-03-21 08:05	2011-03-21 08:55	2011-03-21 11:35	2011-03-21 12:15	null	null	null	2011-03-21 12:15
3	2011-03-21 11:18	2011-03-21 14:01	2011-03-21 16:41	null	null	null	null	2011-03-21 16:41
5	2011-03-22 09:32	2011-03-22 10:12	2011-03-22 12:52	2011-03-22 13:42	null	null	null	2011-03-22 13:42
4	2011-03-22 10:48	2011-03-22 11:38	2011-03-22 13:54	2011-03-22 14:26	null	null	null	2011-03-22 14:26
5	2011-03-25 08:22	2011-03-25 09:14	2011-03-25 11:44	2011-03-25 12:15	null	null	null	2011-03-25 12:15
3	2011-03-25 08:54	2011-03-25 10:42	2011-03-25 12:36	null	null	null	null	2011-03-25 12:36
1	2011-03-25 10:36	2011-03-25 11:48	2011-03-25 14:09	null	null	null	null	2011-03-25 14:09
3	2011-03-28 10:27	2011-03-28 12:37	2011-03-28 14:17	null	null	null	null	2011-03-28 14:17
1	2011-03-29 10:44	2011-03-29 11:58	2011-03-29 14:22	null	null	null	null	2011-03-29 14:22
4	2011-03-30 08:14	2011-03-30 08:54	2011-03-30 10:34	2011-03-30 11:48	null	null	null	2011-03-30 11:48
5	2011-03-31 09:36	2011-03-31 10:06	2011-03-31 12:16	2011-03-31 12:59	null	null	null	2011-03-31 12:59
4	2011-04-01 08:26	2011-04-01 09:19	2011-04-01 11:27	2011-04-01 12:05	null	null	null	2011-04-01 12:05
2	2011-04-04 09:14	2011-04-04 09:54	2011-04-04 11:59	null	null	null	null	2011-04-04 11:59
1	2011-04-04 09:26	2011-04-04 09:56	2011-04-04 12:03	null	null	null	null	2011-04-04 12:03
5	2011-04-05 10:10	2011-04-05 10:59	2011-04-05 13:14	2011-04-05 13:45	null	null	null	2011-04-05 13:45
5	2011-04-05 10:26	2011-04-05 11:00	2011-04-05 13:22	2011-04-05 13:53	null	null	null	2011-04-05 13:53
5	2011-04-08 08:11	2011-04-08 08:44	2011-04-08 11:08	2011-04-08 11:39	null	null	null	2011-04-08 11:39
1	2011-04-08 08:39	2011-04-08 09:19	2011-04-08 11:22	null	null	null	null	2011-04-08 11:22
4	2011-04-11 09:42	2011-04-11 10:18	2011-04-11 12:33	2011-04-11 13:10	null	null	null	2011-04-11 13:11
4	2011-04-11 10:10	2011-04-11 10:52	2011-04-11 12:53	2011-04-11 13:23	null	null	null	2011-04-11 13:23
3	2011-04-11 10:39	2011-04-11 12:35	2011-04-11 13:40		null	null	null	2011-04-11 13:41
5	2011-04-11 11:15	2011-04-11 12:00	2011-04-11 14:01	2011-04-11 14:22	null	null	null	2011-04-11 14:22
2	2011-04-12 08:12	2011-04-12 09:01	2011-04-12 11:21		null	null	null	2011-04-12 11:21
4	2011-04-13 08:28	2011-04-13 08:57	2011-04-13 10:13	2011-04-13 11:10	null	null	null	2011-04-13 11:11
4	2011-04-15 11:59	2011-04-15 12:33	2011-04-15 14:11	2011-04-15 15:07	null	null	null	2011-04-15 15:07
2	2011-04-18 08:33	2011-04-18 09:02	2011-04-18 11:08	null	null	null	null	2011-04-18 11:08
1	2011-04-19 10:44	2011-04-19 11:25	2011-04-19 12:00	null	null	null	null	2011-04-19 12:01
5	2011-04-19 10:52	2011-04-19 11:27	2011-04-19 13:17	2011-04-19 13:33	null	null	null	2011-04-19 13:33
4	2011-04-19 11:02	2011-04-19 11:46	2011-04-19 13:54	2011-04-19 14:17	null	null	null	2011-04-19 14:17
5	2011-04-20 08:30	2011-04-20 09:08	2011-04-20 11:44	2011-04-20 12:04	null	null	null	2011-04-20 12:05
4	2011-04-20 08:47	2011-04-20 09:13	2011-04-20 11:22	2011-04-20 12:07	null	null	null	2011-04-20 12:08
1	2011-04-21 08:39	2011-04-21 09:21	2011-04-21 11:08	2011-04-21 11:22	2011-04-21 13:01	null	null	2011-04-21 13:01
4	2011-04-25 08:07	2011-04-25 09:00	2011-04-25 11:07	2011-04-25 11:43	null	null	null	2011-04-25 11:43
5	2011-04-25 08:50	2011-04-25 09:11	2011-04-25 11:11	2011-04-25 11:44	null	null	null	2011-04-25 11:44
3	2011-04-25 08:59	2011-04-25 11:06	2011-04-25 12:18	null	null	null	null	2011-04-25 12:19
2	2011-04-26 08:33	2011-04-26 09:09	2011-04-26 11:18	null	null	null	null	2011-04-26 11:19
4	2011-04-26 08:49	2011-04-26 09:34	2011-04-26 11:07	2011-04-26 11:34	null	null	null	2011-04-26 11:34
5	2011-04-26 08:54	2011-04-26 09:45	2011-04-26 11:46	2011-04-26 12:10	null	null	null	2011-04-26 12:11
5	2011-04-27 09:22	2011-04-27 10:00	2011-04-27 12:09	2011-04-27 12:39	null	null	null	2011-04-27 12:39
4	2011-04-27 09:40	2011-04-27 10:15	2011-04-27 12:25	2011-04-27 12:55	null	null	null	2011-04-27 12:56
4	2011-04-27 10:14	2011-04-27 11:00	2011-04-27 13:07	2011-04-27 13:29	null	null	null	2011-04-27 13:29
4	2011-04-28 10:39	2011-04-28 11:23	2011-04-28 13:29	2011-04-28 13:58	null	null	null	2011-04-28 13:59
2	2011-04-29 11:07	2011-04-29 12:01	2011-04-29 14:09	null	null	null	null	2011-04-29 14:09
3	2011-04-29 11:59	2011-04-29 13:48	2011-04-29 15:49	null	null	null	null	2011-04-29 15:50
1	2011-04-29 12:02	2011-04-29 13:02	2011-04-29 15:00	null	null	null	null	2011-04-29 15:00
5	2011-05-02 08:39	2011-05-02 09:17	2011-05-02 11:24	2011-05-02 12:00	null	null	null	2011-05-02 12:00
5	2011-05-02 08:44	2011-05-02 09:23	2011-05-02 11:24	2011-05-02 12:02	null	null	null	2011-05-02 12:02
4	2011-05-02 08:48	2011-05-02 09:33	2011-05-02 11:27	2011-05-02 12:15	null	null	null	2011-05-02 12:15
3	2011-05-03 09:13	2011-05-03 11:11	2011-05-03 12:00		null	null	null	2011-05-03 12:01
5	2011-05-05 09:17	2011-05-05 09:44	2011-05-05 11:35	2011-05-05 12:23	null	null	null	2011-05-05 12:23
4	2011-05-05 09:19	2011-05-05 09:52	2011-05-05 11:53	2011-05-05 12:34	null	null	null	2011-05-05 12:34
4	2011-05-05 09:46	2011-05-05 10:08	2011-05-05 12:23	2011-05-05 13:00	null	null	null	2011-05-05 13:00
4	2011-05-05 10:09	2011-05-05 10:49	2011-05-05 12:44	2011-05-05 13:07	null	null	null	2011-05-05 13:07
2	2011-05-09 10:01	2011-05-09 10:36	2011-05-09 12:45	null	null	null	null	2011-05-09 12:45
1	2011-05-09 10:14	2011-05-09 10:43	2011-05-09 12:45	null	null	null	null	2011-05-09 12:46
1	2011-05-09 10:18	2011-05-09 10:50	2011-05-09 12:49	null	null	null	null	2011-05-09 12:49
5	2011-05-09 11:33	2011-05-09 11:58	2011-05-09 14:01	2011-05-09 14:22	2011-05-09 14:52	2011-05-09 16:27	2011-05-09 16:47	2011-05-09 16:47
3	2011-05-10 08:07	2011-05-10 10:08	2011-05-10 11:33	null	null	null	null	2011-05-10 11:34
4	2011-05-10 09:01	2011-05-10 09:36	2011-05-10 11:44	2011-05-10 12:02	null	null	null	2011-05-10 12:03
4	2011-05-10 09:09	2011-05-10 09:40	2011-05-10 11:48	2011-05-10 12:17	null	null	null	2011-05-10 12:17
4	2011-05-11 08:21	2011-05-11 08:55	2011-05-11 10:59	2011-05-11 11:33	null	null	null	2011-05-11 11:33
3	2011-05-11 08:39	2011-05-11 10:43	2011-05-11 11:58	null	null	null	null	2011-05-11 11:59
2	2011-05-12 08:55	2011-05-12 09:20	2011-05-12 11:17	null	null	null	null	2011-05-12 11:18
1	2011-05-12 08:59	2011-05-12 09:27	2011-05-12 11:41	null	null	null	null	2011-05-12 11:41
4	2011-05-12 09:27	2011-05-12 09:58	2011-05-12 11:58	2011-05-12 12:22	null	null	null	2011-05-12 12:22
4	2011-05-13 08:14	2011-05-13 08:46	2011-05-13 10:49	2011-05-13 11:07	null	null	null	2011-05-13 11:07
4	2011-05-13 09:17	2011-05-13 09:49	2011-05-13 11:52	2011-05-13 12:17	null	null	null	2011-05-13 12:17
4	2011-05-13 09:28	2011-05-13 10:02	2011-05-13 12:02	2011-05-13 12:31	null	null	null	2011-05-13 12:31
2	2011-05-13 10:22	2011-05-13 10:57	2011-05-13 13:01	null	null	null	null	2011-05-13 13:02

3 priedas. Apdoroti duomenys

Šablonas	P1	Archyvas	Bendra trukmė	Korekcija darbo laike	Trukmė darbo valandomis	Periodo vidurkis
1	2011-02-17 13:29	2011-02-18 13:03	23:34	15:00:00	08:34	
1	2011-02-28 11:11	2011-02-28 16:54	05:43	00:00:00	05:43	
1	2011-03-07 09:10	2011-03-07 13:00	03:50	00:00:00	03:50	
1	2011-03-15 08:33	2011-03-15 12:12	03:39	00:00:00	03:39	
1	2011-03-25 10:36	2011-03-25 14:09	03:33	00:00:00	03:33	
1	2011-03-29 10:44	2011-03-29 14:22	03:38	00:00:00	03:38	
1	2011-04-04 09:26	2011-04-04 12:03	02:37	00:00:00	02:37	
1	2011-04-08 08:39	2011-04-08 11:22	02:43	00:00:00	02:43	04:17
1	2011-04-19 10:44	2011-04-19 12:01	01:17	00:00:00	01:17	
1	2011-04-21 08:39	2011-04-21 13:01	04:22	00:00:00	04:22	
1	2011-04-29 12:02	2011-04-29 15:00	02:58	00:00:00	02:58	
1	2011-05-09 10:14	2011-05-09 12:46	02:32	00:00:00	02:32	
1	2011-05-09 10:18	2011-05-09 12:49	02:31	00:00:00	02:31	
1	2011-05-12 08:59	2011-05-12 11:41	02:42	00:00:00	02:42	02:43
2	2011-02-14 13:46	2011-02-14 16:56	03:10	00:00:00	03:10	
2	2011-02-25 10:11	2011-02-25 13:51	03:40	00:00:00	03:40	
2	2011-03-01 08:00	2011-03-01 11:30	03:30	00:00:00	03:30	
2	2011-03-14 10:11	2011-03-14 14:01	03:50	00:00:00	03:50	
2	2011-03-15 11:17	2011-03-15 15:06	03:49	00:00:00	03:49	
2	2011-04-04 09:14	2011-04-04 11:59	02:45	00:00:00	02:45	03:27
2	2011-04-12 08:12	2011-04-12 11:21	03:09	00:00:00	03:09	
2	2011-04-18 08:33	2011-04-18 11:08	02:35	00:00:00	02:35	
2	2011-04-26 08:33	2011-04-26 11:19	02:46	00:00:00	02:46	
2	2011-04-29 11:07	2011-04-29 14:09	03:02	00:00:00	03:02	
2	2011-05-09 10:01	2011-05-09 12:45	02:44	00:00:00	02:44	
2	2011-05-12 08:55	2011-05-12 11:18	02:23	00:00:00	02:23	
2	2011-05-13 10:22	2011-05-13 13:02	02:40	00:00:00	02:40	02:45
3	2011-02-17 08:41	2011-02-17 13:01	04:20	00:00:00	04:20	
3	2011-02-22 08:11	2011-02-22 12:19	04:08	00:00:00	04:08	
3	2011-03-03 08:18	2011-03-03 12:40	04:22	00:00:00	04:22	
3	2011-03-21 11:18	2011-03-21 16:41	05:23	00:00:00	05:23	
3	2011-03-25 08:54	2011-03-25 12:36	03:42	00:00:00	03:42	
3	2011-03-28 10:27	2011-03-28 14:17	03:50	00:00:00	03:50	04:17
3	2011-04-11 10:39	2011-04-11 13:41	03:02	00:00:00	03:02	
3	2011-04-25 08:59	2011-04-25 12:19	03:20	00:00:00	03:20	
3	2011-04-29 11:59	2011-04-29 15:50	03:51	00:00:00	03:51	
3	2011-05-03 09:13	2011-05-03 12:01	02:48	00:00:00	02:48	
3	2011-05-10 08:07	2011-05-10 11:34	03:27	00:00:00	03:27	
3	2011-05-11 08:39	2011-05-11 11:59	03:20	00:00:00	03:20	03:18

4	2011-02-17 08:23	2011-02-17 13:10	04:47	00:00:00	04:47	
4	2011-02-17 08:26	2011-02-17 13:00	04:34	00:00:00	04:34	
4	2011-02-17 08:28	2011-02-17 12:52	04:24	00:00:00	04:24	
4	2011-02-21 08:12	2011-02-21 13:46	05:34	00:00:00	05:34	
4	2011-02-22 09:32	2011-02-22 14:15	04:43	00:00:00	04:43	
4	2011-02-23 08:23	2011-02-23 12:40	04:17	00:00:00	04:17	
4	2011-02-24 08:19	2011-02-24 12:48	04:29	00:00:00	04:29	
4	2011-03-02 08:12	2011-03-02 13:00	04:48	00:00:00	04:48	
4	2011-03-08 11:00	2011-03-08 15:32	04:32	00:00:00	04:32	
4	2011-03-10 11:13	2011-03-10 15:06	03:53	00:00:00	03:53	
4	2011-03-14 12:13	2011-03-14 17:00	04:47	00:00:00	04:47	
4	2011-03-14 13:09	2011-03-15 08:50	19:41	15:00:00	04:41	
4	2011-03-15 10:23	2011-03-15 15:11	04:48	00:00:00	04:48	
4	2011-03-18 08:43	2011-03-18 13:13	04:30	00:00:00	04:30	
4	2011-03-18 11:13	2011-03-18 16:02	04:49	00:00:00	04:49	
4	2011-03-22 10:48	2011-03-22 14:26	03:38	00:00:00	03:38	
4	2011-03-30 08:14	2011-03-30 11:48	03:34	00:00:00	03:34	
4	2011-04-01 08:26	2011-04-01 12:05	03:39	00:00:00	03:39	04:28
4	2011-04-11 09:42	2011-04-11 13:11	03:29	00:00:00	03:29	
4	2011-04-11 10:10	2011-04-11 13:23	03:13	00:00:00	03:13	
4	2011-04-13 08:28	2011-04-13 11:11	02:43	00:00:00	02:43	
4	2011-04-15 11:59	2011-04-15 15:07	03:08	00:00:00	03:08	
4	2011-04-19 11:02	2011-04-19 14:17	03:15	00:00:00	03:15	
4	2011-04-20 08:47	2011-04-20 12:08	03:21	00:00:00	03:21	
4	2011-04-25 08:07	2011-04-25 11:43	03:36	00:00:00	03:36	
4	2011-04-26 08:49	2011-04-26 11:34	02:45	00:00:00	02:45	
4	2011-04-27 09:40	2011-04-27 12:56	03:16	00:00:00	03:16	
4	2011-04-27 10:14	2011-04-27 13:29	03:15	00:00:00	03:15	
4	2011-04-28 10:39	2011-04-28 13:59	03:20	00:00:00	03:20	
4	2011-05-02 08:48	2011-05-02 12:15	03:27	00:00:00	03:27	
4	2011-05-05 09:19	2011-05-05 12:34	03:15	00:00:00	03:15	
4	2011-05-05 09:46	2011-05-05 13:00	03:14	00:00:00	03:14	
4	2011-05-05 10:09	2011-05-05 13:07	02:58	00:00:00	02:58	
4	2011-05-10 09:01	2011-05-10 12:03	03:02	00:00:00	03:02	
4	2011-05-10 09:09	2011-05-10 12:17	03:08	00:00:00	03:08	
4	2011-05-11 08:21	2011-05-11 11:33	03:12	00:00:00	03:12	
4	2011-05-12 09:27	2011-05-12 12:22	02:55	00:00:00	02:55	
4	2011-05-13 08:14	2011-05-13 11:07	02:53	00:00:00	02:53	
4	2011-05-13 09:17	2011-05-13 12:17	03:00	00:00:00	03:00	
4	2011-05-13 09:28	2011-05-13 12:31	03:03	00:00:00	03:03	03:09
5	2011-02-17 08:40	2011-02-17 11:50	03:10	00:00:00	03:10	
5	2011-02-21 10:12	2011-02-21 14:50	04:38	00:00:00	04:38	
5	2011-03-04 10:10	2011-03-04 14:59	04:49	00:00:00	04:49	
5	2011-03-09 08:10	2011-03-09 15:09	06:59	00:00:00	06:59	
5	2011-03-21 08:05	2011-03-21 12:15	04:10	00:00:00	04:10	
5	2011-03-22 09:32	2011-03-22 13:42	04:10	00:00:00	04:10	
5	2011-03-25 08:22	2011-03-25 12:15	03:53	00:00:00	03:53	
5	2011-03-31 09:36	2011-03-31 12:59	03:23	00:00:00	03:23	
5	2011-04-05 10:10	2011-04-05 13:45	03:35	00:00:00	03:35	
5	2011-04-05 10:26	2011-04-05 13:53	03:27	00:00:00	03:27	04:13
5	2011-04-08 08:11	2011-04-08 11:39	03:28	00:00:00	03:28	
5	2011-04-11 11:15	2011-04-11 14:22	03:07	00:00:00	03:07	
5	2011-04-19 10:52	2011-04-19 13:33	02:41	00:00:00	02:41	
5	2011-04-20 08:30	2011-04-20 12:05	03:35	00:00:00	03:35	
5	2011-04-25 08:50	2011-04-25 11:44	02:54	00:00:00	02:54	
5	2011-04-26 08:54	2011-04-26 12:11	03:17	00:00:00	03:17	
5	2011-04-27 09:22	2011-04-27 12:39	03:17	00:00:00	03:17	

4 priedas. Detalesnė hash funkcijų tikimybinė analizė

Bitai	Galimų hash rezultatų skaičius	Fiksuota tikimybė gauti dviem vienodoms reikšmėms (susikirtimui)									
		10^{-18}	10^{-15}	10^{-12}	10^{-9}	10^{-6}	0.1%	1,00 %	25,00 %	50,00 %	75,00 %
16	6.6×10^{4}	2	2	2	2	2	11	36	1.9×10^{2}	3.0×10^{2}	4.3×10^{2}
32	4.3×10^{9}	2	2	2	2,9	93	2.9×10^{3}	9.3×10^{3}	5.0×10^{4}	7.7×10^{4}	1.1×10^{5}
64	1.8×10^{19}	6,1	1.9×10^{2}	6.1×10^{3}	1.9×10^{5}	6.1×10^{6}	1.9×10^{8}	6.1×10^{8}	3.3×10^{9}	5.1×10^{9}	7.2×10^{9}
128	3.4×10^{38}	2.6×10^{10}	8.2×10^{11}	2.6×10^{13}	8.2×10^{14}	2.6×10^{16}	8.3×10^{17}	2.6×10^{18}	1.4×10^{19}	2.2×10^{19}	3.1×10^{19}
256	1.2×10^{77}	4.8×10^{29}	1.5×10^{31}	4.8×10^{32}	1.5×10^{34}	4.8×10^{35}	1.5×10^{37}	4.8×10^{37}	2.6×10^{38}	4.0×10^{38}	5.7×10^{38}
384	3.9×10^{115}	8.9×10^{48}	2.8×10^{50}	8.9×10^{51}	2.8×10^{53}	8.9×10^{54}	2.8×10^{56}	8.9×10^{56}	4.8×10^{57}	7.4×10^{57}	1.0×10^{58}
512	1.3×10^{154}	1.6×10^{68}	5.2×10^{69}	1.6×10^{71}	5.2×10^{72}	1.6×10^{74}	5.2×10^{75}	1.6×10^{76}	8.8×10^{76}	1.4×10^{77}	1.9×10^{77}