ŠIAULIŲ UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

INFORMATIKOS KATEDRA

Artiom Pachomov

Informatikos magistro specialybės II kurso dieninio skyriaus studentas

Dinaminis kompiuterinių sistemų infrastruktūros atnaujinimo modelis, pagrįstas atviro kodo sprendimais

Dynamic update model of computer systems infrastructure based on open source solutions

MAGISTRO DARBAS

Darbo vadovas:

Doc. Dr. L. Kaklauskas

Recenzentas:

_____

Šiauliai, 2014

„Tvirtinu, jog darbe pateikta medžiaga nėra plagijuota ir paruošta naudojant literatūros sąraše pateiktus informacinius šaltinius bei savo tyrimų duomenis“

Darbo autoriaus

_____

(vardas, pavardė, parašas)

Darbo tikslai ir uždaviniai

Tikslas

Parengti dinaminį kompiuterinių sistemų infrastruktūros nepertraukiamo paslaugų tiekimo sprendimą įmonėms ar įstaigoms remiantis šiuolaikiniais nemokamo bei atviro kodo įrankiais.

Uždaviniai

- Išanalizuoti įmonės ar įstaigos naudojančios nemokamo bei atviro kodo kompiuterinių sistemų infrastruktūrą.
- Atlikti nepertraukiamų kompiuterinių paslaugų teikimo analizę.
- Atlikti duomenų replikavimo, sinchronizavimo ir vientisumo užtikrinimo proceso analizę.
- Atlikti vartotojų centralizuoto valdymo analizę.
- Įvertinti sistemų saugumo pažeidžiamumą ir prevencines priemones.
- Sukurti universalų sprendimo modelį, kompiuterinių sistemų infrastruktūros nepertraukiamam saugiam paslaugų tiekimui.

Darbo vadovo _____

(vardas, pavardė, parašas)

Turinys

1. Įvadas

Šiais laikais Lietuvoje sutinkama įmonių, nenorinčių ar negalinčių investuoti į IT infrastruktūrą kurios naudojasi nemokamais atviro kodo sprendimais. Deja, tobulėjant technologijoms dalis įmonių nespėja ar neturi galimybių atsinaujinti esamos infrastruktūros. Kuo ilgiau laukiama, tuo sudėtingesni tampa naujų sprendimų pritaikymai, esamų sistemų-duomenų migravimai. Tokie užsisenėję IT sprendimai tampa itin pažeidžiami saugumo bei duomenų praradimo atžvilgiu (Pachomov A., 2013).

Šio darbo tikslas, išanalizavus tokio tipo įmonę ar įstaigą sukurti universalų sprendimo modelį, leisianti su minimaliomis investicijomis pasiekti šiuolaikinių nemokamų bei atviro kodo sistemų standartų įmonės ar įstaigos IT infrastruktūroje. Kartu su sistemų atnaujinimo galimybėmis, bus aptariami nepertraukiamo paslaugų teikimo (angl. High-Availibility), duomenų replikavimo (angl. replication) bei saugumo priežiūros proceso standartizavimo pritaikymai.

Taipogi šiame darbe analizuojamos standartu tampančios paslaugos, tokios kaip stebėjimo (angl. monitoring), gedimų registravimo, automatinio atsarginių kopijų darymo bei sisteminių įrašų konsolidacija. Šios paslaugos palengvina bendrą IT infrastruktūros priežiūrą bei palaikymą.

2. Teorinė dalis

2.1. Įstaigos analizė

2.1.1. Įstaigos pasirinkimas

Šiame darbe tyrimo duomenų faktinė bazė paremta konkrečios įstaigos esamos kompiuterinių sistemų IT infrastruktūra. Analizuojama įstaiga – Šiaulių valstybinė kolegija, Aušros al. 40, Šiauliai LT-76241, Įstaigos kodas 111968241.

Kolegijos IT infrastruktūrą sudaro apytiksliai 30 virtualių bei fizinių serverių, naudojančių tiek Linux tiek Windows OS. Tiekiama apie 10 skirtingų bazinių IT paslaugų tiek vidiniams tiek išoriniams įstaigos vartotojams. Administruojama apie 500 darbų vietų.

Žemiau išvardytos pagrindinės dėl užsisenėjusios programinės įrangos įstaigos problemos:

- išskirstytos autentifikavimosi sistemos;
- neužtikrintas nepertraukiamas paslaugų teikimas;
- duomenų praradimo rizika;
- necentralizuotas sistemų valdymas;
- pritaikytos IT infrastruktūros priežiūros paslaugos pilnavertiškai neišnaudojamos;
- teikiamų paslaugų nelankstumas;
- sudėtingas naujų paslaugų integravimas.

2.1.2. Tiekiamos paslaugos bei įranga

Kaip minėta ankščiau įstaigoje pagrindinę paslaugų IT infrastruktūrą sudaro apie 30 skirtingų virtualių bei fizinių serverių. Teikiamas paslaugas galima padalinti į žemiau pateiktas pagrindines dalis:

- failų saugykla skirta failams saugoti bei keistis;
- LDAP bei MySQL duomenų bazės;
- tinklinės programinės aplinkos (angl. web-applications);
- prieglobos (angl. hosting) paslaugos;
- el. pašto;
- vidinės tinklo tiekimo paslaugos.

Yra naudojama ir daugiau tinklinių sistemų. Įstaigos analizei bus apsiribota šiomis teikiamomis sistemomis. Reikia įvertinti, jog visa ši tinklinė programinė įranga nebuvo pradėta eksploatuoti vienu metu, pirma buvo pradėtos naudoti vienos sistemos, laikui bėgant, atsirado ir kitų. Didžioji dalis paslaugų veikia atskirai ant jai dedikuotų sistemų. Kai kurios dedikuotos sistemos yra praktiškai neišnaudojamos, kitoms nepakanka resursų, esamame sprendime sudėtinga sumažinti ar padidinti resursų.

2.1.3. Problemų analizė

Išskirstytos autentifikavimosi sistemos problema – tai reiškia, kad, kiekvienai paslaugai, nesvarbu ar tai failų saugykla, el. paštas ar bet kuri tinklinė programinė aplinka, visos jos, turi atskirus prisijungimo duomenis. Paprastu atveju, jeigu įstaigoje priimamas naujas darbuotojas ar studentas, jis registruojamas dešimtyje skirtingų sistemų. Kasmet į kolegiją įstoja apie tūkstantį studentų. Vienu metu besimokančių studentų apytiksliai yra apie tris tūkstančius, tad vienu metu sistemose yra apie 9 tūkstančius registruotų (neskaitant personalo) aktyvių vartotojų kuriuos reikia prižiūrėti. Bendru atveju įvertinant ir personalą, aktyvių vartotojų skaičius gali viršyti 10 tūkstančių aktyvių vartotojų, tarp kurių tūkstantis yra itin dinamiškas (kasmet keičiasi), o realių asmenų – tik apie 3 tūkstančius. Taigi vienas įstaigos darbuotojas privalo atsiminti daugiau negu dešimt skirtingų prisijungimo vardų, slaptažodžių, bei sistemų aplinkų. Administravimo našta taipogi proporcingai didelė, kai reikia registruoti ar šalinti vartotojus iš išskirstytų sistemų. Netgi turint automatizavimo įrankius, laiko sąnaudos vis tiek bus didesnės tiek kartų, kiek yra skirtingų sistemų, taipogi kyla problemų dėl vartotojų vientisumo visuose sistemose.

Neužtikrinto nepertraukiamas paslaugų teikimo problema – bet kuriam atskiram serveriui tapus nepasiekiamu ar nustojus veikti, visiems vartotojams paslauga tampa nepasiekiama. Jeigu tai techninės įrangos gedimas, paslauga gali būti neaktyvi gana nemažą laiką kol įrangos dalys bus pakeistos ir sutrikdyti sklandų įstaigos darbą. Jeigu įmonė ar įstaiga užsiima finansine veikla, kritinės paslaugos sutrikimas gali pridaryti finansinės žalos.

Duomenų praradimo rizikos problema – įmonės bei įstaigos turi svarbius vidinius duomenis, kurių praradimas gali sustabdyti visą darbo veiklą, tad jų nuolatinis pasiekiamumas yra labai svarbus. Įmonės bei įstaigos pasirūpina svarbių duomenų atsarginėmis kopijomis, tačiau, šioje situacijoje yra svarbus ir laikas. Kaip minėta ankščiau, atsarginės kopijos yra svarbu, tačiau dirbant su dinaminiais duomenimis, atsarginės kopijos turi trūkumų, tokių kaip senesnių duomenų

atstatymas (angl. rollback), priklausomai nuo duomenų dydžio atstatymo procesas gali užtrukti, atsarginių kopijų darymo proceso metu apkrauti sistemas ir t.t..

Necentralizuoto sistemų valdymo problema – per didelį laiko tarpą, įmonės ar įstaigos plečia savo IT infrastruktūrą pagal poreikius. Kartais poreikiai iš anksto nebūna numatyti, pavyzdžiui jeigu atsiranda naujo tipo paslaugos pagerinančios darbo sąlygas ar įstaiga nusprendžia tiekti naujas paslaugas. Jeigu infrastruktūra nėra iš anksto parengta plėtimui ir sistema nedokumentuota, naujoms paslaugoms yra imamos naujos platformos, taip sukuriant bendrą infrastruktūrą netvarkingą. IT infrastruktūra gali išsiplėsti iki to, kad administracinis personalas naudoją didelį skaičių įvairių aplinkų su skirtingais saugumo, prieigos bei eksploatacijos parametrais. Tokios sistemos turi tendenciją tapti nestabilios ir palikti įvairių saugumo spragų.

Pritaikytos IT infrastruktūros priežiūros paslaugos pilnavertiško neišnaudojamo problema atsiranda kai, viena ar kita paslauga ar serveris naudoja priežiūros įrankius tik sau. Pavyzdžiui, jeigu kiekvienos sistemos atsarginės kopijos, stebėjimo bei konsolidacijos procedūros atliekamos kiekvienam serveriui atskirai, atitinkamai auga bendras resursų panaudojimo skaičius. Turint *n* serverių, kiekviename reikia atlikti nustatymus individualiai, atitinkamai auga kompiuterinių resursų, elektros bei laiko sąnaudos. Taip pat svarbu įvertinti, jog priežiūros operacijos naudoja tuos pačius resursus kuriuos naudoja ir paslaugos. Taipogi sudėtinga tampa sužiūrėti visų serverių įvykių įrašus, būklę ir t.t..

Teikiamų paslaugų nelankstumo problema – tarkime Linux OS serveryje, yra išleistas atnaujinimas kurioje yra ištaisytos kokios nors saugumo spragos kuriai nors programavimo kalbai (pvz. plačiai interneto tinklalapiams naudojamai PHP). Dažnai programiniai įrankių paketai Linux OS yra siejami su bibliotekomis, kurias naudoja kiti sistemos komponentai, kas gale gali priversti prie bendro visos sistemos naujinimo. Kolegijos atveju, kai kurios mašinos naudoja Slackware Linux OS distribuciją, kuri naudoja „slackbuild“ tipo programinius paketus. Slackware yra laikoma viena greičiausių Linux OS distribucijų, tačiau greitis ateina su naujinimo sunkumais. Beveik bet kuris OS kartos atnaujinimas reikalauja rankinio įsikišimo, ir sistema būna neaktyvi tol, kol atnaujinimas nepabaigiamas bei neištestuojamas. Tokioms problemoms spręsti, greitas būdas yra paketų kompiliavimas tiesiogiai sistemoje, tačiau paslaugų serveriuose, laikyti kompiliavimo įrankius, yra laikoma nesaugu, kadangi įsilaužėliai patekę vidun turėtų galimybes pasigaminti bet kokius jiems reikiamus įrankius. Taipogi nuolatinis paketų kompiliavimas gali palikti sistemoje

daug „šiukšlių“, pavyzdžiui didelį kiekį bibliotekų kurios reikalingos paslaugų ar paketų kompiliavimo, bet ne veikimo metu.

Sudėtingo naujų paslaugų integravimo problema panaši į necentralizuoto sistemų valdymo problemą, tačiau apima kiek daugiau. Esminė problema, tai kad naujoms paslaugoms naudojamos tuščios platformos dėl nesuderinimo su jau esančiomis, taip efektyviai neišnaudojant turimų resursų. Taipogi išlaikoma saugumo spragų bei administravimo laiko sąnaudų problema. Kitais atvejais, dėl neturimo bendro autentifikavimosi modelio naujai kuriamos tinklinėms programinės aplinkos naudoja atskiras vartotojų duomenų bazes, kadangi pagrindinė nėra pritaikyta tokiai eksploatacijai. Taip didėja saugomų duomenų apimtis bei mažėja paslaugos serverio našumas, nors iš esmės yra saugomi tie patys duomenys.

2.2. Nepertraukiamas paslaugos tiekimas (High-Availibility)

2.2.1. Klasteris (angl. cluster)

IT sferoje, terminas klasteris sutinkamas dažnai ir įvairiose srityse. Šio termino apibrėžimas gali skirtis priklausomai nuo jo panaudojimo srities, tačiau yra tam tikros gairės būdingos klasterio apibrėžimui visose jo panaudojimo srityse (Wensong, Z. 2003):

- sudaryta iš dviejų ar daugiau panašių mazgų, turinčių vienodą (ar panašią) OS, nepriklausomai nuo to, kad techniniai mazgų parametrai gali skirtis;
- mazgai sujungti į vieningą tinklą. Nors ir nebūtina, bet efektyviausias našumas bus pasiektas jei mazgai bus sujungti vietiniu tinklu (LAN);
- visi mazgai tarpusavyje yra patikimi (angl. trusted system), kad būtų įmanomas atskirų mazgų valdymas;
- dalis saugyklos resursų yra bendra visiems klasterių mazgams (arba sinchronizuojama papildomais įrankiais);
- jeigu formuojamo klasterio visų mazgų resursai turi būti galimi išnaudoti vienos programos vykdymui, jose turi būti įdiegtos MPI (Message Passing Interface) bibliotekos bei taikomos aplikacijos.

Jeigu paimti terminą globaliai „kompiuterių klasteriu“ vadinama kompiuterių-serverių visuma sujungta į vieningą architektūrą, galutiniam vartotojui matoma kaip viena sistema. Pati klasterio

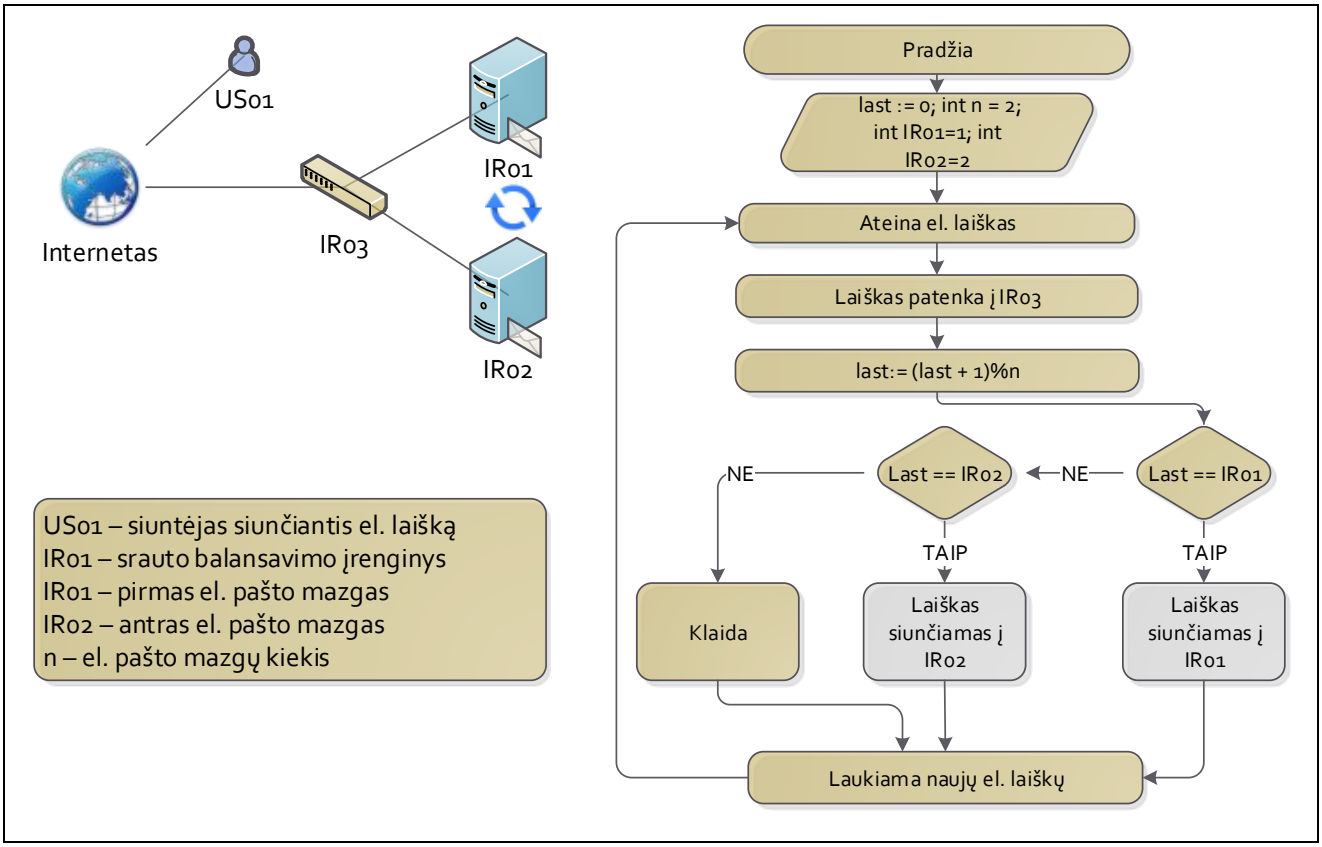
formavimo idėja yra pagrindas daugumai žinomuose sprendimuose tokiuose kaip lygiagrečiųjų skaičiavimų centrai (angl. Grid) arba debesų kompiuterija (angl. Cloud).

Klasterio panaudojimas tokioms paslaugoms kaip tinklalapiai, el. paštas, DNS ir kiti yra išskiriamas į dvi pagrindines grupes:

- apkrovos paskirstymas (angl. load balancing);
- nepertraukiamo paslaugos tiekimo klasteris (angl. High-availibility cluster)

Klasterio panaudojimo išskyrimas į dvi grupes yra skirtas gilesnei šių grupių analizei, tačiau tai nereiškia, kad klasteris negali būti vienu metu skirtas balansavimui ir nepertraukiamam paslaugų tiekimui. Praktikoje dažnai sutinkamas būtent šis atvejis, kai formuojami klasteriai yra naudojami balansavimui ir nepertraukiamam paslaugų tiekimui.

Apkrovos paskirstymo klasteris paprastai naudojamas tuo atveju, kai teikiamai paslaugai reikia daugiau resursų nei įmanoma jų turėti viename mazge. Pav.1 pavaizduotas algoritmas atspindi el. pašto srauto apkrovos paskirstymą tarp dviejų el. pašto serverių „Round Robin“ metodu.



Pav. 1. El. pašto srauto balansavimo algoritmas

Kai konfigūruojamas apkrovos paskirstymas, galima atitinkamai pasirinkti balansavimo metodus, įvertinant esančių klasterių mazgų aplinkybėmis. Itin svarbu pasirinkti tinkamą balansavimo metodą norint užtikrinti, kad paslauga stabiliai veiktų visuose mazguose. Balansavimo metodo pasirinkimui įtaką taipogi gali sudaryti skirtingi mazgų techniniai parametrai, inicijuotų užklausų skirtingo resursų kiekio sąnaudos ir t.t. Keli pagrindiniai apkrovos balansavimo metodai plačiau aptarti 1 lentelėje.

1 lentelė. Srauto balansavimo metodai

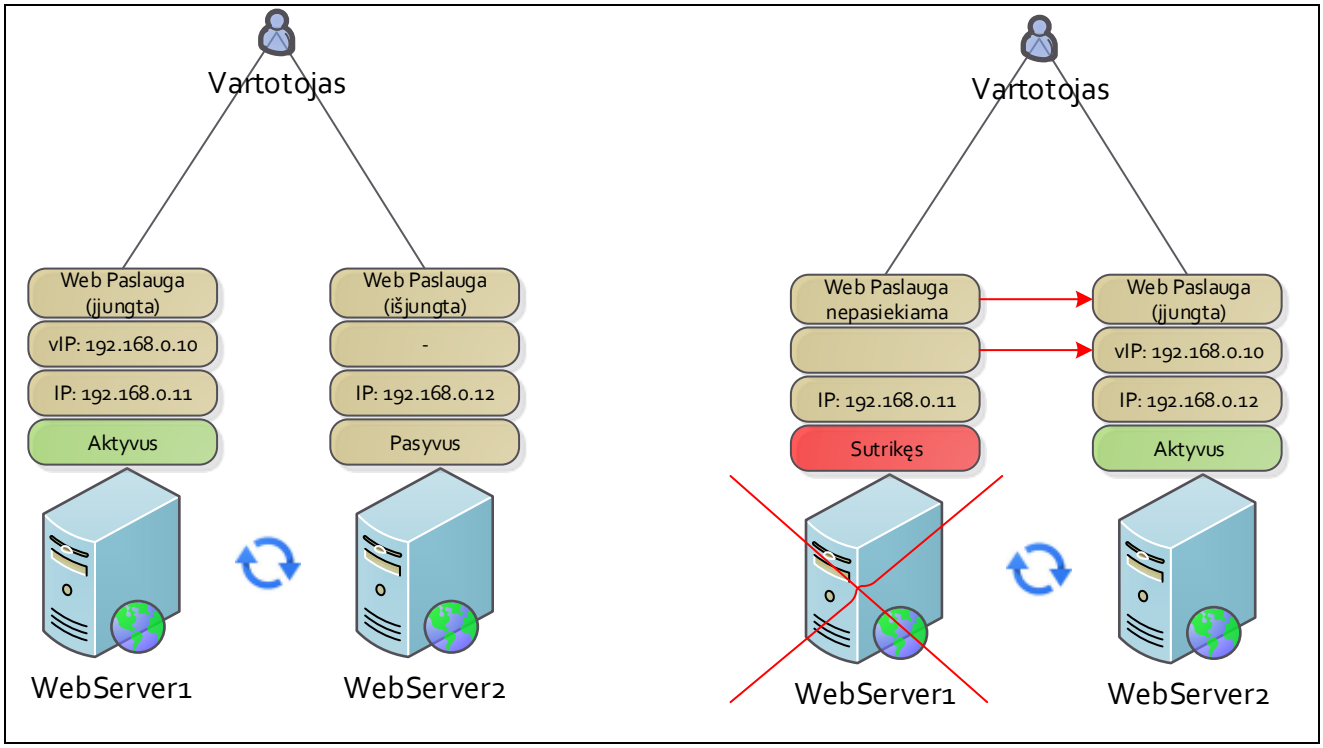
Metodo pavadinimas	Apibūdinimas	Kada naudojamas
Round Robin	Tai standartinis balansavimo metodas, kai kiekviena nauja inicijuojamo ryšio užklausa paduodama mazgams eilės tvarka.	Šis metodas yra patogus daugumoje standartinių konfigūracijų, ypač efektyvus, jeigu klasterio mazgai yra vienodų techninių parametrų.
Ratio (node)	Naudojant šį metodą, vartotojas nustato statinius koeficientus pagal mazgą. Tokiu būdu, mazgo sujungimų skaičius yra proporcionalus nustatytam koeficientui.	Šį metodą naudoja vartotojas atsižvelgdamas į jam reikalingą apkrovos paskirstymą pagal mazgus, pavyzdžiui jei vienas kuris nors klasterio mazgas naudojamas papildomai kitoms paslaugoms.
Dynamic ratio (node)	Naudojant šį metodą paskirstymo koeficientai yra dinaminiai ir keičiasi automatiškai įvertinant mazgo apkrovą.	Šis metodas patogus tuo atveju, kai klasterio mazgai yra skirtingų techninių parametrų ir negali teikti paslaugos taip pat kaip kiti mazgai.

Šiame darbe, bus naudojamas nepertraukiamo paslaugos tiekimo klasteris, taip pat galimai vadinamas automatinio-atstatymo klasteris (angl. failover cluster) arba tiesiog HA klasteris. Esminė HA klasterio idėja yra, tai kad, sutrikus serveriui kurioje yra paslauga, nepasiekiamumo laikas (angl. downtime) bus minimalus ir galutiniam vartotojui praktiškai nepastebimas (Wensong, Z. 2000). HA klasterio valdymas, nustatęs jog kuriame nors mazge atsirado ryšio, techninės įrangos ar pačios paslaugos sutrikimų, automatiškai paslaugą suaktyvuos ant kito „sveiko“ klasterio mazgo, o sugedusį pašalins iš klasterio. Šis procesas yra vadinamas automatiniu paslaugos atstatymu (angl. failover). Šis procesas taipogi gali būti kontroliuojamas ir rankiniu būdu, tai yra naudinga, kai norima atlikti serverio naujinimą, naujų paslaugų diegimą, perkrovimą, testavimą ar bet kokią kitą palaikymo (angl. maintenance) operaciją, neišjungiant paslaugos. Baigus darbus ar atstačius

(sutaisius) mazgą, galima jį grąžinti atgal į klasterį. Pastaba - automatinio-atstatymo klasteris nėra apkrovos balansavimo klasteris ir ne visais atvejai šią funkciją palaiko.

Paprastai serveryje esančiai paslaugai pasiekti yra suteikiamas IP adresas. Kadangi HA klasteriui minimaliai reikia 2 mazgų, tai atitinkamai reikės 3 IP adresų kuriais bus pasiekiama paslauga. Trečiasis IP adresas vadinamas virtualiu IP (vIP). Standartinėje HA klasterio konfigūracijoje vienu metu būna aktyvus tik vienas mazgas ir tas mazgas tuo metu turi priskirtą vIP adresą. Jeigu aktyviajam mazgui kas nors nutiktų, vIP adresas nedelsiant bus perkeltas ant sekančio mazgo. Detalesnis pavyzdys analizuojamas pav. 2.

Pavaizduotoje schemoje matomi du mazgai „WebServer1“ – 192.168.0.11 ir „WebServer2“ – 192.168.0.12. vIP adresas – 192.168.0.10. Tarp mazgų paslaugos resursai yra replikuojami. Ant abiejų mazgų yra sukonfigūruota tinklalapio paslauga intranete puslapiui „www.klasteriopavyzdys.lt“, tačiau paslauga šiuo metu veikia tik ant vieno mazgo - „WebServer1“. Sakykime yra vietinis DNS serveris skirtas vidiniam įmonės naudojimui, kuriame adresas „www.klasteriopavyzdys.lt“ yra susietas su vIP adresu 192.168.0.10.



Pav. 2. Nepertraukiamo paslaugos tiekimo sutrikimo procesas

Pateiktoje schemoje, kairėje aktyvus mazgas yra „WebServer1“ tad naršyklėje kreipiantis tiek 192.168.0.11, tiek 192.168.0.10 IP adresais, tinklalapį atidarys. Tačiau kreipiantis naršyklėje DNS vardu, bus kreipiamasi tik į vIP adresą. Tarkime, „WebServer1“ sugedo procesoriaus ventiliatorius, serveris perkaito ir išsijungė, tada klasterio įranga, startuos tinklalapio paslaugą ant „WebServer2“ mazgo ir vIP adresą perkels taipogi į antrą mazgą, kaip pavaizduota pav.2 dešinėje. Tad kreipiantis DNS vardu, paslauga bus pasiekiama iš antrojo mazgo, nors pagal DNS, kreipiamasi tuo pačiu IP adresu.

2.2.2. Klasterio programinė įranga

Šiuo metu yra žinomi keli HA klasterio programiniai įrankiai pagal skirtingas OS. Taipogi reikia įvertinti kad skirtingai nuo OS pats įrankis gali skirtis, pvz. Linux OS HA klasterį sudaro 3 atskiri įrankiai, o Unix HP-UX OS, klasteriavimo įrankyje yra viskas viename. Juos galima išskirstyti pagal pagrindines OS šeimas:

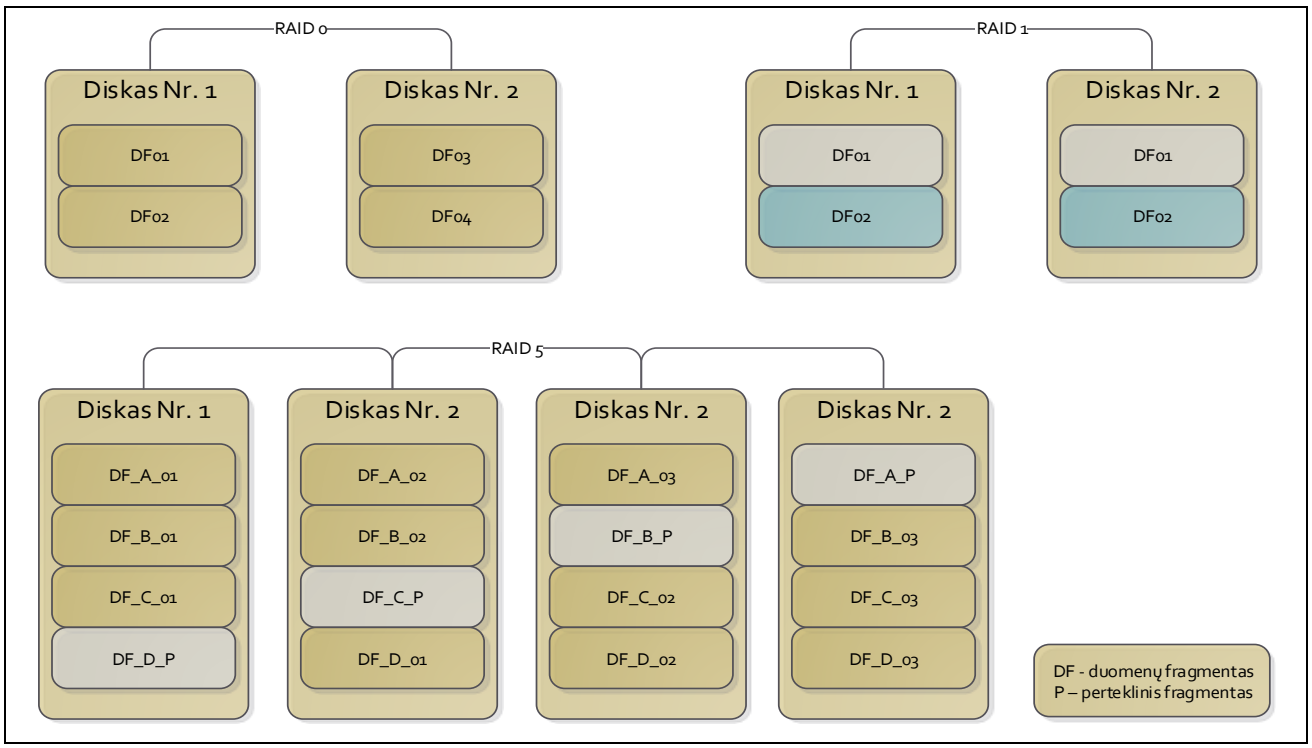
- Linux – Linux OS klasteriui sukurti reikia trijų pagrindinių skirtingų įrankių. Paslaugų valdymo aplinka „Pacemaker“, klasterio pasiekiamumo aplinka „Corosync“ bei klasterio valdymo aplinka „crm“ arba „pcs“. Plačiau apie šiuos komponentus, jų diegimą bei eksploatavimą bus aptarta šio darbo realizacijos dalyje.
- Unix – Unix OS šeima neturi bendro HA programinio paketo, kadangi didžioji dalis Unix distribucijų yra mokamos, tad ir įrankiais pasirūpina OS tiekėjai, pvz. HP-UX – „ServiceGuard“, IBM AIX – „PowerHA“ ir t.t..
- Windows – Windows Server Failover Clustering (WSFC).
- ESX‘i – tai virtualizacijos platforma skirta vmware produktams. Paprastai diegiama į serverį kaip OS, tada susiejama su centralizuoto valdymo stotimi vadinama „vcenter“ kuri valdoma įrankiu vadinamu „vsphere“ kur ir yra galimas HA klasterio realizavimas.

Taipogi svarbu žinoti apie HA klasterio atitvėrimo (angl. fencing) mechanizmą. Atitvėrimo mechanizmas naudojamas tokiu atveju, jei klasteryje dingsta ryšys tarp mazgų, tada kiekvienas mazgas gali „nuspręsti“, kad jis yra pagrindinis, nes kitas nepasiekiamas, anglų kalboje tai vadinama „Split brain scenario“. Linux OS sistemoje atitvėrimo mechanizmas vadinamas „STONITH“ (Shoot The Other Node In The Head). Mechanizmo principas – pagal vartotojo nustatytus papildomus parametrus, mazgas gali nustatyti ar kitas mazgas „neapsimeta“ aktyviuoju,

tokiu atveju atsitveriantis mazgas „apsimetėlį“ perkrauna, tikėdamasis, kad po perkrovimo mazgas automatiškai atsistatys.

2.2.3. Bendra duomenų saugykla (angl. shared storage)

Prieš tęsiant apie bendras saugyklas, atliekama trumpa RAID (Redundant Array of Independent Disks) technologijų apžvalga. Šios technologijos privalumas yra tas, kad galima apjungti du ar daugiau vienodų kietųjų diskų į vieną bendrą saugyklą, ir užtikrinti, kad sugedus vienam iš masyvo diskų, jis gali būti pakeistas nauju neišjungiant sistemos bei neprarandant duomenų. Masyvams sukurti gali būti taikoma tiek techninė tiek programinė įranga, tačiau laiku išbandyta, jog techninis sprendimas yra pranašesnis dauguma atvejų. Techniniam sprendimui pritaikyti yra naudojamas komponentas vadinamas RAID adapteriu. Tokiu atveju, RAID yra kuriamas žemiau OS, ir OS mato įrenginį kaip vienintelį diską nežinodama, kad iš tikro tai daugiau diskų viename. Šiuo metu dažniausiai sutinkami RAID lygmenys yra 0, 1 ir 5 bei jų tarpusavio mišiniai. 2 lentelėje palyginami lygmenys įvairiais aspektais (kur *n* – diskų kiekis, *DRS* – disko skaitymo greitis, *DWS* – disko įrašymo greitis, *span* – minimalus pilnavertiškam RAID sprendimui reikalingas diskų skaičius). Pav. 3 parodytas duomenų paskirstymas tarp diskų naudojant kiekvieną RAID lygmenį atskirai.

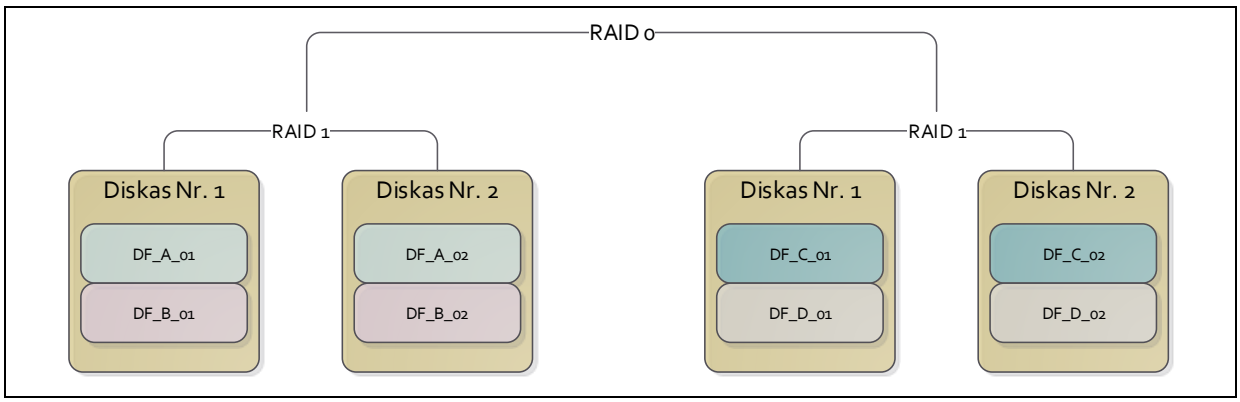


Pav. 3. Praktikoje paplitusių diskų masyvų lygmenų struktūros

2 lentelė. Praktikoje paplitę diskų masyvų lygmenys

Pav.	Minimalus diskų skaičius	Utilizuojama talpa	Kiek diskų galima prarasti	Skaitymo greitis	Įrašymo greitis	Kada naudojamas
RAID 0	2	1	0	n * DRS	n * DWS	Paspartina bendrą skaitymo ir įrašymo operacijų greitį.
RAID 1	2	1 / n	n-1	n * DRS	1 * DWS	Paspartina bendrą skaitymo operacijų greitį, neprarandami duomenys.
RAID 5	3	1 - 1 / n	1	(n -1) * DRS	(n -1) * DWS	Patogu naudoti turint didelį diskų skaičių. Lėtesnis nei RAID 1+0, tačiau maksimaliai utilizuojama talpa.
RAID 1+0	4	2 / n	1 per span	n * DRS	(n / 2) * DWS	Patogu naudoti turint didelį diskų skaičių. Bet kokiu atveju, nuo bendros talpos maksimaliai utilizuoti galima tik pusę, tačiau itin paspartiną įrašymo ir skaitymo operacijų greitį.

Turint didesnį kiekį diskų, sutinkami tarpusavyje sujungti RAID lygmenys, pavyzdžiui kaip pateikta pav. 4 RAID 1+0 lygmuo, kuris yra RAID 1 bei RAID 0 junginys kuris įgyja abiejų lygmenų savybes – diskų gedimų toleravimas bei didesnis įrašymo greitis.



Pav. 4. Pirmojo ir nulinio diskų masyvo lygmenų junginio struktūra

Įvykus automatiniam paslaugos persijungimui klasteryje į kitą mazgą, reikia užtikrinti, kad paslauga dirba su tais pačiais failais. Tam užtikrinti yra naudojama bendros duomenų saugyklos arba sinchronizacijos paslaugos. Kuris iš metodų pasirenkamas priklauso tiek nuo pačios paslaugos, tiek nuo techninių galimybių. Yra išskiriami keli pagrindiniai būdai šiam tikslui pasiekti – atskiri saugyklos įrenginiai, specifinės bendro panaudojimo failinės sistemos (angl. file system) bei failų sinchronizavimas.

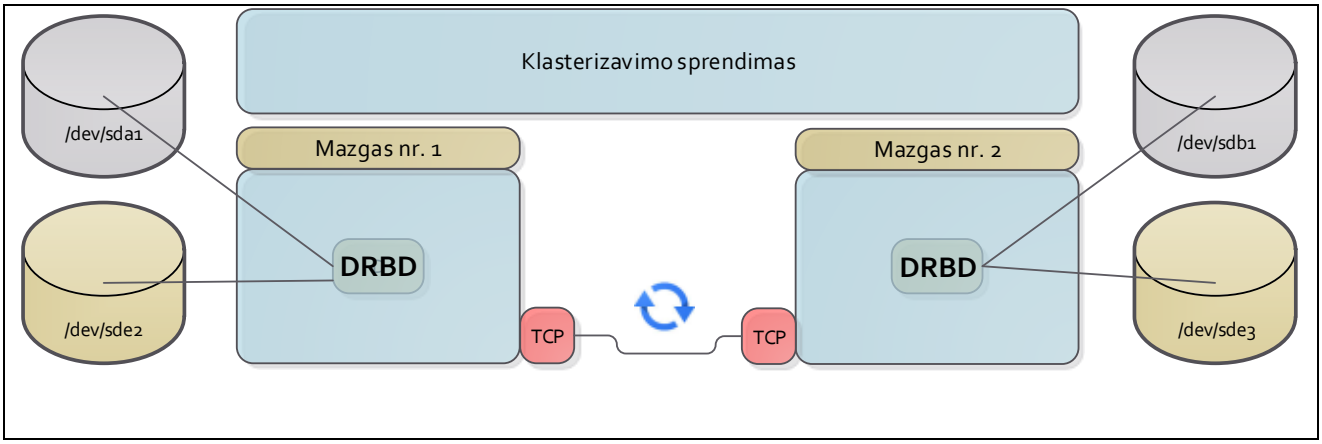
Paduodama duomenų saugykla iš atskiro įrenginiu, specialiuoju saugyklos vietiniu tinklu, vadinamu SAN (Storage Access Network), tačiau šis būdas reikalauja atskiro įrenginio vadinamo saugykla (angl. storage device) bei papildomų tinklo įrenginių ir kitų palaikymų komponentų mazgams vadinamų HBA (Host Bus Adapter) adapteriais. Saugyklos turi integruotus RAID adapterius, kurie užtikrina bendrą diskų greitaveiką bei duomenų saugumą. Vidutinio dydžio talpyklos, gali talpinti apie 50 nepriklausomų kietųjų diskų. Jeigu įvertinti skaitymo bei rašymo greičius *rpm*, tai greitį galima sekančia paskaičiuoti formule:

$$rp = n * diskRPM; \quad wp = \left(\frac{n}{2}\right) * diskRPM; kur\ n - disk\text{ų}\ skai\check{c}ius; diskRPM - disko\ rpm \qquad (1)$$

Šis duomenų saugojimo būdų šiuo metu yra vienas patikimiausių bei efektyviausių, tačiau jeigu įmonė ar įstaiga iki šiol šio įrenginio nenaudojo, šis sprendimas nėra palankus finansiniu atžvilgiu. Saugyklos techniniai sprendimai yra gana brangūs.

Jeigu bendrai saugyklai didelės talpos nereikia bei HA klasteryje mazgų nėra itin daug, galima naudotis ir išskirstytų tinklinių įrenginių sprendimus, tokius kaip SAMBA, NFS arba DRBD (Distributed Replicated Block Device). Tačiau vėl gi kiekvienas sprendimas turi būti įvertintas pagal naudojimo aplinkybes. SAMBA sprendimas yra patogus tuo atveju, jeigu klasteryje yra maišomos OS pvz. Windows ir Linux, tačiau kelia problemų dėl našumo. NFS sistema yra skirta tik Linux OS, tačiau jeigu bendrai klasteriui prieinami duomenys reikalauja įvairių failų savininkų ribojimų, tai vėl gi pasirinkimas ribojantis lankstumą.

DRBD tai išskirstyta replikuojama failinė sistema skirta Linux platformai ir dažnai taikoma HA klasteriuose. Schema pavaizduota pav.5.



Pav. 5. DRBD duomenų replikavimo mechanizmas

DRBD duomenų replikavimo principas yra tas, kad yra sukuriamos dvi failinės sistemos, kurių viena yra aktyvi, o į kitą yra daromos visos I/O (Input - Output) operacijų kopijos. Tokiu atveju sugedus pagrindinei failinei sistemai į jos vietą bus pastatyta pasyvioji failinė sistema. Tai panašu į RAID 1 lygmens technologiją, tačiau tai nėra tas pats. DRDB atveju priešingai nei RAID, yra sukuriamos dvi atskiros failinės sistemos, tad jei sugedus diskui, RAID nukreips I/O operacija į sveiką diską ir aplikacija to nepastebės, DRBD atveju aplikacija nebepasieks savo failų. Būtent šiuo atveju suveiks automatinis paslaugos atstatymas ir paslauga bus aktyvuojame antrame mazge kur buvo DRBD failinės sistemos kopija. Sutvarkius sutrikusio mazgo failinę sistemą, DRBD duomenų replikavimo algoritmas užtikrina kad nepasiekiamumo metu įvykę duomenų pokyčiai būtų sinchronizuoti. Didžiausias DRBD privalumas, kad jis veikia tarp mazgų tinkle. Šis būdas itin patogus, kai duomenys yra labiau statiniai, retai keičiantys. Reikia įvertinti, kad DRBD taipogi sukelia didelę apkrovą sistemoms, tad itin dinaminių duomenų atveju, pavyzdžiui duomenų bazių, daugiau privalumų teikia sprendimas – naudoti pačių DB replikavimo įrankius.

2.3. Duomenų replikavimas

2.3.1. Replikavimo sąvoka

Duomenų replikavimu IT sferoje vadinamas metodas, kai yra realizuojamas informacijos padalinimas tarp išskirstytų programinių ar techninių resursų norint užtikrinti duomenų pasiekiamumą, pasiekiamumo efektyvumą ar nepraradimą (Kopper, K., 2005).

Replikavimo metodai yra skirstomi į aktyvius bei pasyvius. Aktyviu replikavimu vadinamas metodas, kai visuose aplinkose (angl. instance) yra apdorojamos tos pačios užklausos. Pasyvus metodas skiriasi tuo, kad, užklausa apdorojame tik vienoje aplinkoje, o į kitas aplinkas paduodamas

tik rezultatas. Naudojant pasyvų replikavimo metodą yra „taupomi“ procesoriaus resursai, tačiau tam tikrais atvejais tampa sudėtinga, užtikrinti duomenų vientisumą.

2.3.2. Duomenų bazių replikavimas

Prieš tai buvo aptartas DRBD failinės sistemos metodas, kuris replikuoja duomenis tarp dviejų mazgų saugyklų, tačiau šis metodas atsiliepia mazgų našumui jeigu duomenys yra itin dinamiški. DRBD taip pat patogus tuo, kad bendra vieta gali būti skirta neribotam skirtingų paslaugų skaičiui, tačiau jeigu replikuojama paslauga yra tik duomenų bazė, verta paanalizuoti pačių duomenų bazių teikiamus replikavimo įrankius.

Duomenų bazių replikavimo metodai yra išskiriami į pagrindinius du:

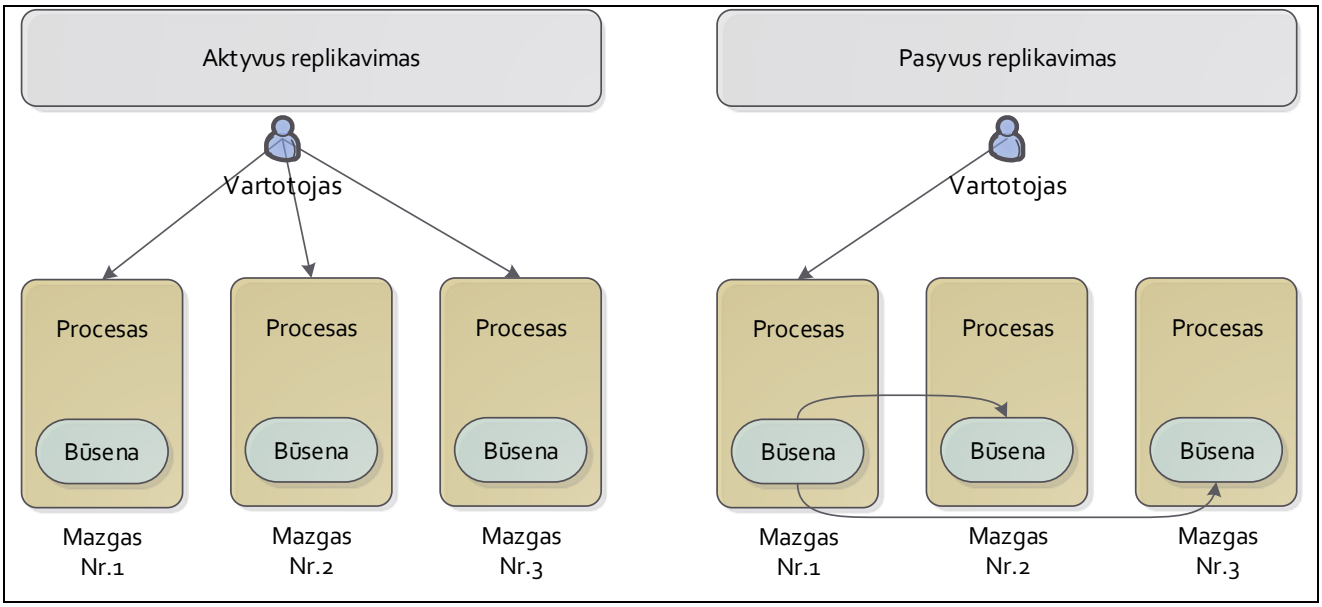
- active-active (arba master-master);
- active-pasive (arba master-slave).

Pav. 6 pavaizduoti šie du replikavimo metodai. Esminis skirtumas, yra tas, kad pasyvaus replikavimo metode, vienu metu, aktyvi duomenų bazė yra viena ir visos įrašymo bei skaitymo operacijos atliekamos tik vienoje aplinkoje vienu metu. Duomenų bazė keičiasi po duomenų įrašymo ar modifikavimo, aktyviajame mazge, o tik tada, replikavimo agentai, informuoja pasyvius mazgus apie pokyčius, kad šie pasivytų tuo metu aktyvios duomenų bazės būseną.

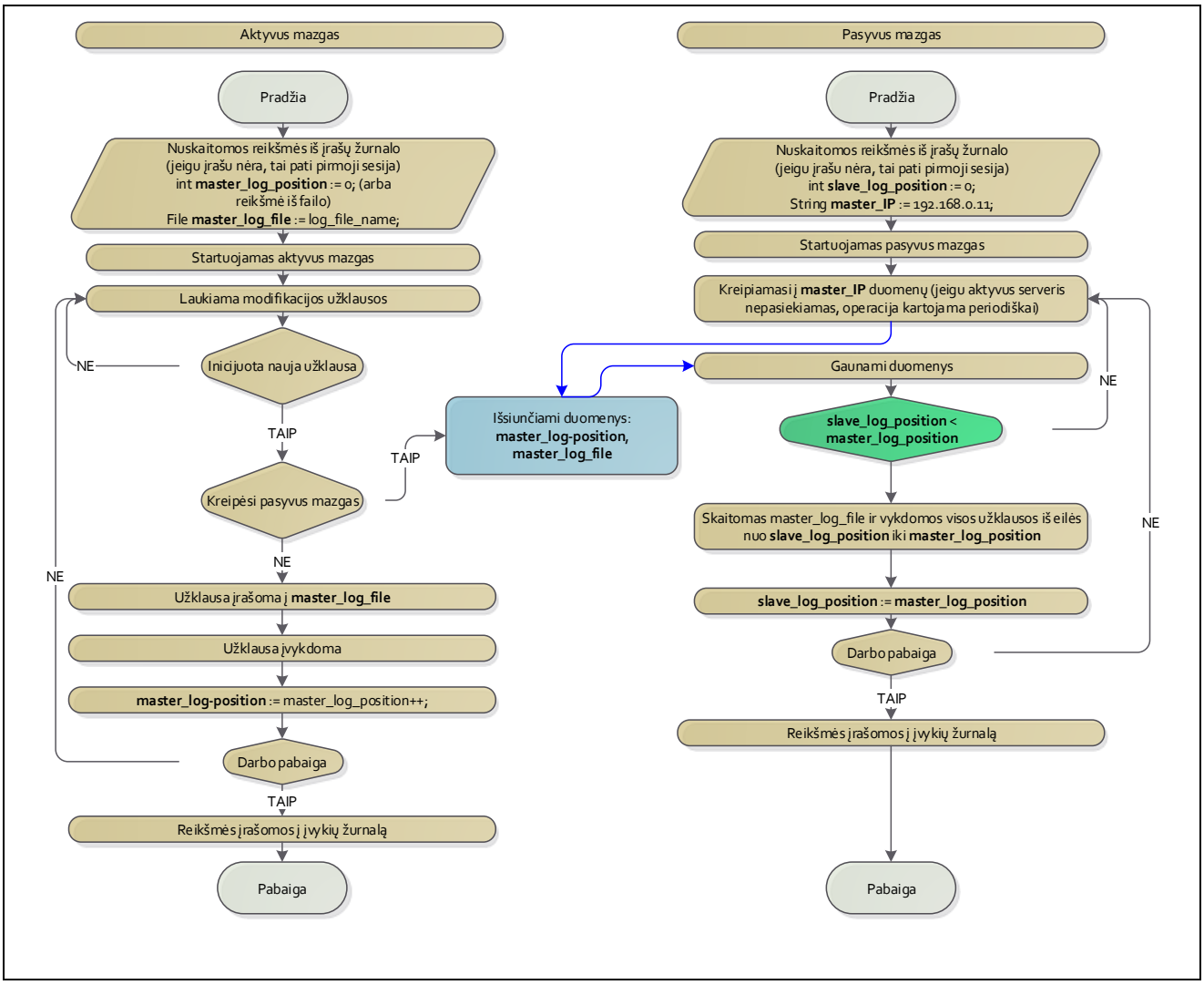
Aktyvaus replikavimo metode, visi duomenų bazių mazgai yra aktyvūs, t.y. įrašymo operacijas galima atlikti bet kuriame duomenų bazės mazge, o šis praneša kitiems mazgams apie pokyčius.

MySQL ir LDAP yra skirtingos architektūros duomenų bazės, MySQL – reliacinė, LDAP – hierarchinė. Duomenų bazių skirtinga architektūra ne vienintelis dalykas kuris įtakoja replikavimo technologijų sprendimą, tačiau paaiškina kai kuriuos pasirinktus sprendimus.

Prieš analizuojant abiejų duomenų bazių replikavimo metodikas, apibrėžiamas standartinis aktyvaus-pasyvaus mazgų replikavimo mechanizmas pav. 7.



Pav. 6. Aktyvaus (kairėje) ir pasyvaus (dešinėje) duomenų replikavimo schema



Pav. 7. Duomenų replikavimo algoritmas master-slave režime

Kurį metodą pasirinkti, priklauso nuo panaudojimo paskirties bei aplinkybių. 3 lentelėje įvardinami metodų privalumai ir trūkumai gali padėti priimti sprendimą, kuris replikavimo metodas atsižvelgiant į tuometines aplinkybes ar poreikius (Kopper, K., 2005).

3 lentelė. Replikavimo režimų privalumai be trūkumai

	Privalumai	Trūkumai
Master-master	Vienam iš mazgų nustojus veikti neįtakoja bendro paslaugos veikimo, paslauga būna pilnai prieinama. Galima pritaikyti balansavimo metodą, kad paskirstyti apkrovą mazgams.	Yra rizika, kad bus sugadintas duomenų vientisumas. Nuolatinė apkrova visiems mazgams. Įvykus įrašų kolizijai, replikavimo procesas sustoja, kol nebus pratęstas rankiniu būdu.
Master-pasive	Pasyviajame mazge duomenų bazėje draudžiama atlikti įrašymo ar modifikavimo operacijos, tad rizika sugadinti duomenų vientisumą kur kas mažesnė. Nėra operacijų kolizijos rizikos. Papildomas skaitymo paslaugas (pvz. atsarginių kopijų darymo) galima nukreipti į pasyvųjį mazgą, taip leidžiant pagrindinei paslaugai utilizuoti aktyvaus mazgo resursus pilnai.	Aktyviam mazgui nustojus veikti paslauga galės naudotis tik skaitymo iš duomenų bazės paslauga. Pasyvaus ir aktyvaus mazgų vaidmenų sukeitimo procesas reikalauja paslaugos sustabdymo. Sudėtingesnis duomenų bazės iš atsarginės kopijos atstatymo procesas.

Abiejų duomenų bazių atveju itin svarbus yra nepasiekiamumo (angl. downtime) aspektas. Sustabdžius pasyviosios duomenų bazės paslaugą ir suaktyvavus po laiko, pasyvioji duomenų bazė „pasiveja“ aktyviosios duomenų bazės pokyčius. Kadangi replikavimo procesas gali būti pristabdytas, tai išlaiko pasyviųjų mazgų privalomus, kurie leidžia mazgą išimti atlikti sistemos palaikymo darbus.

MySQL duomenų bazės replikavimo mechanizmui svarbiausi elementai yra pokyčių įvykių žurnalas bei skaitymo pozicijos. Aktyviajame mazge yra vedamas SQL užklausų įvykių žurnalas iš kurio skaito pasyvusis mazgas. Sustabdžius pasyvųjį mazgą, jis atsimena kurioje vietoje jis nustojo skaityti pagal poziciją ir grįžęs į darbo režimą, pratęs skaitymą iki tol, kol pasivys einamąją poziciją.

Kadangi įvykių žurnalas yra pildomas visomis užklausomis (tiek įrašų pridėjimo, šalinimo ir modifikacijų), tik laiko klausimas, kada šie įrašai perpildys sistemą. Deja, taupant resursus, yra nustatomos maksimalios įvykių įrašų saugojimo datos, ir viršijusios šį laiką, yra šalinamos. Jeigu pasyvaus mazgo neaktyvumo laikotarpis viršija šią datą, tolimesnė replikacija nebeįmanoma. Tokiu

atveju, yra daroma nauja atsarginė kopija kuri yra atstatoma į pasyvųjį mazgą ir replikacija pratęsiama.

LDAP duomenų bazės replikavimo mechanizmas yra paprastesnis nei MySQL, tačiau itin remiasi mazgų laiku. Laikas tarp replikuojamų duomenų bazių mazgu privalo sutapti, tad yra rekomenduojama naudoti ntp (Network Time Protocol) servisus mazguose. Taipogi replikuojant LDAP duomenų bazę, galima pasirinkti, kokiu būdų apie replikavimo pokyčius bus informuotas pasyvus mazgas. Informacijos stūmimo (angl. push) būdas, iš aktyviojo mazgo kas tam tikrą laiko intervalą informuoja apie pokyčius, o traukimo (angl. poll) metodu į aktyvųjį mazgą kreipiasi pasyvusis mazgas pats. Priešingai nei MySQL duomenų bazės replikavimas, naujas LDAP pasyvus mazgas neskaito įvykių žurnalo, o tiesiog „perskaito“ visus įrašus nuo pradžių. Šioje situacijoje tai yra patogu, dėl to, kad LDAP duomenų bazė yra hierarchinė. Tik po to, kai naujai atreplikuota duomenų bazė pasiveja aktyviosios įrašus, naujiems įrašams atlikti, yra naudojami aktyviojo mazgo įvykių žurnalo įrašai.

Prieš tai 3 lentelėje buvo pateikti teiginiai, apie tai, kad yra rizika prarasti duomenų vientisumą, tačiau to galima išvengti naudojantis papildomais duomenų vientisumo užtikrinimo duomenų bazėse įrankiais. Šie įrankiai nebuvo pristatyti kartu su replikavimo sprendimais, kadangi atsirado vėliau. MySQL ir LDAP duomenų bazių atveju, šiuos įrankius teikia ne pačios paslaugos tiekėjai (trečios šalies programinė įranga) ir dar nėra pilnai išbandyta, tačiau laikui bėgant tampa vis labiau patikima. MySQL atveju yra vis labiau besiplečiantis „Percona“ įrankių rinkinys.

2.4. Programinės įrangos autentifikavimo centralizavimas

2.4.1. Tapatybės valdymas ir vieninga autorizacija

Per pastarąjį laikotarpį, įvairiems verslo sprendimams tinklinių programinių aplinkų (angl. web-applications) atsiranda vis daugiau, kartu su šia programine įranga padidėjo vartotojų, grupių ir vaidmenų apimtis kuri turi būti administruojama. Taip pat atsiranda papildomos informacijos saugumo spragos, kadangi, tokios programinės įrangos dalinis administravimo funkcionalumas dažnai perkeliamas kitiems vidiniams įmonės darbuotojams (Pachomov, A., 2013).

Administravimo našta vis didėja su tokios programinės įrangos plėtimusi ir panaudojimu įmonėse ir įstaigose. Būtent šiuo atveju IdM (Identity Managment) gali sumažinti bendro administravimo darbą, kai visa vartotojų informacija: vartotojų vardai, slaptažodžiai, grupės, vaidmenys ir kiti atributai yra valdomos centralizuotoje sistemoje.

Tapatybės valdymo sprendimai finansiškai prieinami ne visoms įstaigoms, be to kyla problemų su jau nusistovėjusia įranga, kadangi dauguma įmonių susikūrusios ankščiau nei atsirado šie sprendimai.

Tapatybės valdymas – gali būti traktuojamas kaip individualių tapatybių valdymas vieningoje sistemoje, tokioje kaip įmonė, tinklas ar net šalis. Verslo įmonėse IT srityje, tapatybės valdymas yra svarbiausias aspektas valdant vartotojų vaidmenis ir jų prieigos privilegijas individualių tinklų naudojimo atžvilgiu. Šios tapatybės valdymo sistemos apjungia įvairius įrankius ir technologijas vartotojų bei su jais susijusios informacijos kontroliavimui organizacijos viduje.

Tapatybės valdymas neatskiriamai susietas su saugumu bei produktyvumu bet kurios organizacijos susijusios su el. komercija. Organizacijos naudoja tapatybės valdymo sistemas ne tik kontroliuoti savo vartotojų bazę, bet ir verslo produktyvumui didinti. Centralizuota prieiga sumažina esminių procesų kainą bei sudėtingumą.

Tapatybės valdymo sistemų privalumai:

- Ši sistema organizacijoms suteikia galimybę plėsti prieigą prie savo informacinės bazės nekompromituojant saugumo. Tapatybių kontroliavimas potencialus tiekimo atžvilgiu asmenims iš išorės (angl. outsiders).
- Dalį sistemos aspektų, įrankių galima taikyti pagal organizacijos poreikius, taip automatizuojant daug laiko užimančias užduotis.
- Gerai parengta tapatybių valdymo sistema reiškia, kad yra platus vartotojų kontroliavimas, kas priveda į sumažintą vidinių bei išorinių atakų riziką.

Tapatybės valdymo trūkumai:

- Integravimas su nusistovėjusiomis sistemomis – ne visos įmonės pradėjusios kurtis ir plėsti savo tinklinių sistemų struktūras numatė, jog ateityje bus tokios galimybės kaip IdM, tad tarp naudojamos programinės įrangos, visada atsiras tokios, kurios naudoja savotiškus autentifikavimo metodus, bei turi tik savo vidines vartotojų bazes. Jeigu tokia sistema tapo nepakeičiama organizacijos dalis, tai tampa dideliu iššūkiu, migruoti tokią sistemą į vieną, bendrą, vieningą sistemą.
- Vartotojų privatumo informacija – kai kurie vartotojai (vadinami paranojiškais), prisibijo, jog įmonės turinčios didelį kiekį jų asmeninės informacijos, gali ją panaudoti savo

tikslams, pavyzdžiui nutekinti, taip sudarant galimybę opozicijai sukompromituoti pačius vartotojus.

- Saugumo pažeidžiamumas – kadangi IdM sistemos turi stipriai palengvintas ir puikiai susistemintas valdymo bazes, kurios vilioja internetinius nusikaltėlius. Internetinių nusikaltėlių pagrindiniai tikslai IdM – registruotų vartotojų informacija, bei „atsarginių durų“ tipo vartotojai su aukštomis privilegijomis sistemoje.

Tinklinių sistemų, vartotojų didelio kiekio problemai yra jau taikomi sprendimai, kurie mažina darbo administratorių darbo laiko sąnaudas, didiną administravimo ir pačių vartotojų terpės patogumą.

IdM sistemos kaip ir visos kitos, turi savo trūkumų, tačiau prieš darant išvadas, reikia įvertinti, koks reikšmingumo skirtumas tarp šios sistemos privalumų bei trūkumų.

Platesnis IdM aspektas, kai pradedama apjungti įvairių įmonių ar įstaigų IdM sprendimus, kai vienos įmonės ar įstaigos vartotojai gali autorizuotis kitoje, patikimoje įmonėje ar įstaigoje. Jungimas (angl. federation) skirtos dalintis skaitmeniniais identifikatoriais su patikimais partneriais. Kitaip tariant, tai autentifikavimo-bendrinimo (angl. authentication-sharing) mechanizmas sukurtas leisti prieiti prie ne savo įstaigos ar įmonės sistemų su savo įstaigos ar įmonės vartotojo vardu, slaptažodžiu ar kitais prisijungimo duomenimis. Šis mechanizmas dar žinomas kaip - Single Sign-On. Vienas iš SSO vartotojams globaliai prieinamų pavyzdžių yra „Google“ bei „Youtube“ vartotojų bazės.

3. Projektinė dalis

3.1. Įrankių ir priemonių pasirinkimų analizė

Šiame darbe panaudojama tiek įstaigoje esanti tiek sprendimui pritaikymui reikalinga programinė įranga. Įrankiai išskirstyti į kategorijas:

- operacinės sistemos – Slackware linux v13, CentOS 6.5;
- programavimo kalbos – PHP, Bash shell scripting, Perl;
- paslaugų tiekimo programinė įranga – mysql server, mysql client, ldap server, ldap client tools, pacemaker, corosync, crm shell, mdadm, http server, postfix, drdb utils, holland_backup, opennms, syslog-ng.

Iš operacinės sistemos Slackware linux bus migruojama į CentOS 6.5. CentOS operacinė pasirinkta dėl kelių priežasčių. Pagrindinė priežastis ta, kad OS yra paremta RHEL (Red Hat Enterprise Linux) operacinės sistemos atviru kodu. RHEL Linux distribucija yra komercinė ir turi nuolatinį palaikymą bei yra pripažinta viena geriausių verslo klasės (angl. enterprise) linux distribucijų. Taipogi, RHEL finansuoja CentOS distribucijos palaikymą, kad pastaroji galėtų konkuruoti su kitomis verslo klasės linux distribucijomis. Sekanti priežastis lėmusi pasirinkimą tai, kad CentOS naudoja rpm (Red Hat Package Manager) programinius paketus, kurie yra suderinami su dauguma verslo klasės distribucijomis tokiomis kaip Fedora, SLES ir kitomis. Paskutinė priežastis lėmusi šį pasirinkimą yra tai, kad OS yra nesudėtingai naujinama bei atnaujinimai dėl spragų yra pakankamai greitai pateikiami. Kaip pavyzdžiu galima remtis pastaruoju įvykiu susijusiu su „openssl“ bibliotekos vadinamu „Heartbleed“ pažeidžiamumu viešai paskelbtu 2014-04-07 18:39 , jau kelių dienų bėgyje, RHEL ir CentOS išleido atnaujintus programinius paketus pažeidžiamumui ištaisyti.

Programavimo kalbos pasirinktos dvi. PHP kalbos pasirinkimą lėmė įstaigoje esančios tinklinės programinės aplinkos. Ši kalba šiame darbe itin naudojama nebus, tiesiog bus analizuojamas autentifikavimo testams (standartinis PHP autentifikavimo testavimo kodas pateiktas priede nr. 6). Bash shell scenarijų rašymo kalba pasirinkta OS lygmenyje automatizavimo tyrimo testo procesams atlikti. Perl programavimo kalba naudojama įvairiais aspektais. Šiame darbe didžiausi Perl programavimo kalbos privalumai yra tai, kad programavimo kalba palaiko tiek LDAP tiek MySQL modulius leidžiančius nesudėtingai dirbti su duomenų bazių duomenimis. Kitas svarbus aspektas, Perl yra glaudus su pačios Linux OS valdymo procesais.

Programinės įrangos pateikti tik pagrindinių paketų pavadinimai, reikia atkreipti dėmesį jog priklausomų paketų bei bibliotekų nurodyta nėra. Paprastai, diegiant paketą, per programinių paketų valdiklį (CentOS vadinamas „yum“), jis automatiškai parinks visus reikiamus pagalbinius programinius paketus bei bibliotekas. Paketų kompiliavimo atsisakyta dėl įvairių priežasčių, beveik visi sprendimai bus diegiami tik per oficialias distribucijos repozitorijas.

Minėta ankščiau, kad klasterio valdymui Linux OS yra kelios skirtingos aplinkos – „pcs“ ir „crm“. Kuris įrankis tiekiamas kaip standartinis priklauso nuo distribucijos. CentOS atveju, tai „pcs“, tad norint naudoti „crm“ gali prireikti pasinaudoti papildomomis ne distribucijos repozitorijomis. Šiuo atveju, klasterio valdymo įrankių tiekėjai, turi oficialią repozitoriją kuria bus naudojamasi šiame darbe. Klasterio formavimui bei valdymui pagrindiniai skirti paketai yra: pacemaker, corosync, crm shell.

Likusi prieš tai įvardyta programinė įranga yra skirta bazinėms paslaugomos tiekti arba RAID bei bendros saugyklos formavimui bei valdymui.

3.2. Pradinis projekto aprašymas

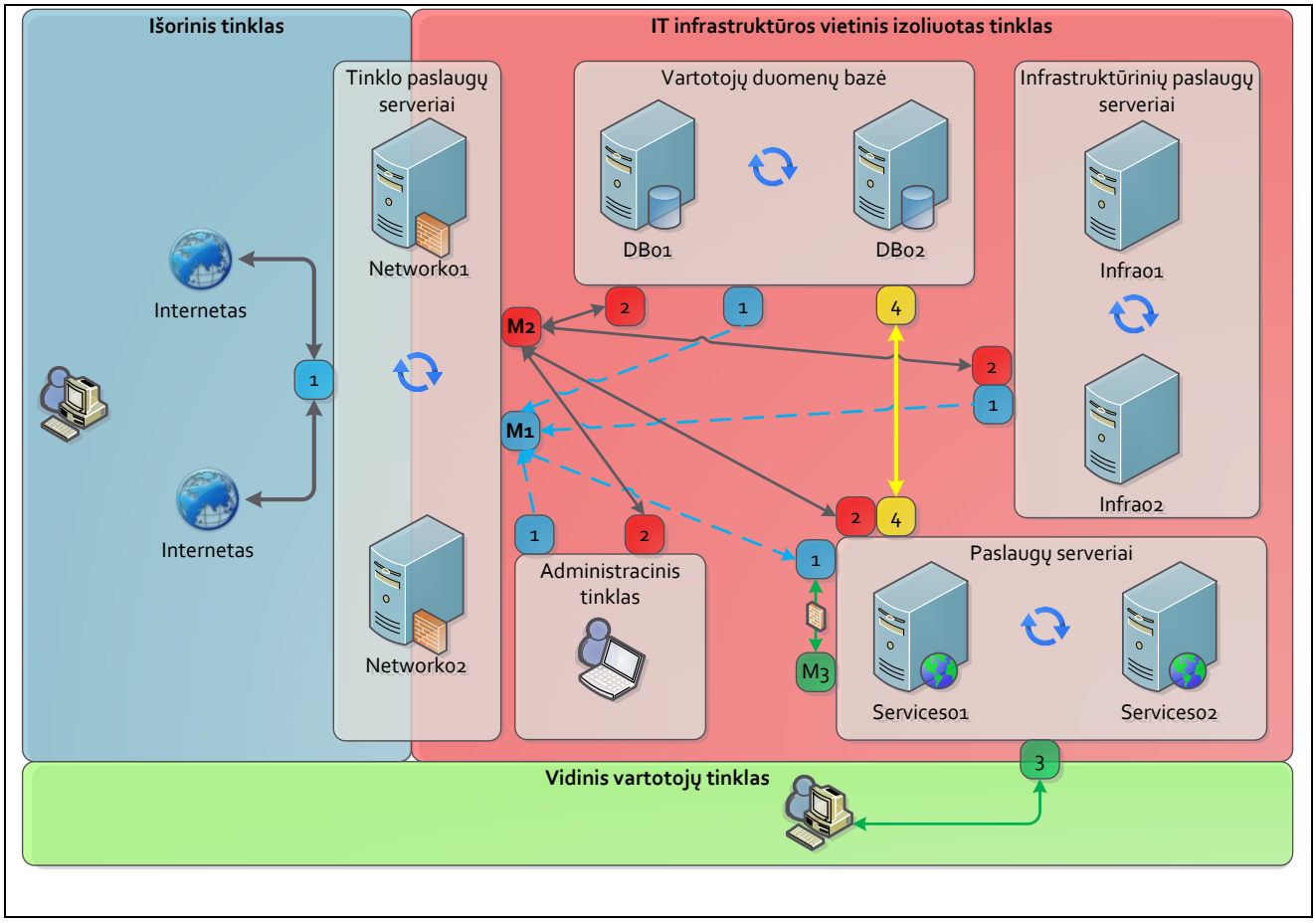
Šiame darbe kuriamas sprendimas yra skirtas Šiaulių valstybinės kolegijos IT infrastruktūros naudojantis esančia programine ir technine įranga su minimaliomis investicijomis pasiekti šiuolaikinių nemokamų bei atviro kodo sistemų standartų bei išnaudoti jų privalomus.

Naudojantis kuriamu sprendimu siekiama, jog įstaigoje tiekiamos paslaugos bus nepertraukiamos (techninės ar programinės įrangos gedimo atveju nebus prastovų), bus likviduotos darbo pradžioje pateiktos problemos. Taip pat siekiama minimizuoti IT infrastruktūrą tiek prižiūrinčių tiek besinaudojančių priežiūros bei naudojimosi apkrovą. Sprendimo modeliavimo metu, bus aptariamos saugumo gairės, leisiančios geriau suprasti galimus infrastruktūros pavojus bei kaip jų išvengti.

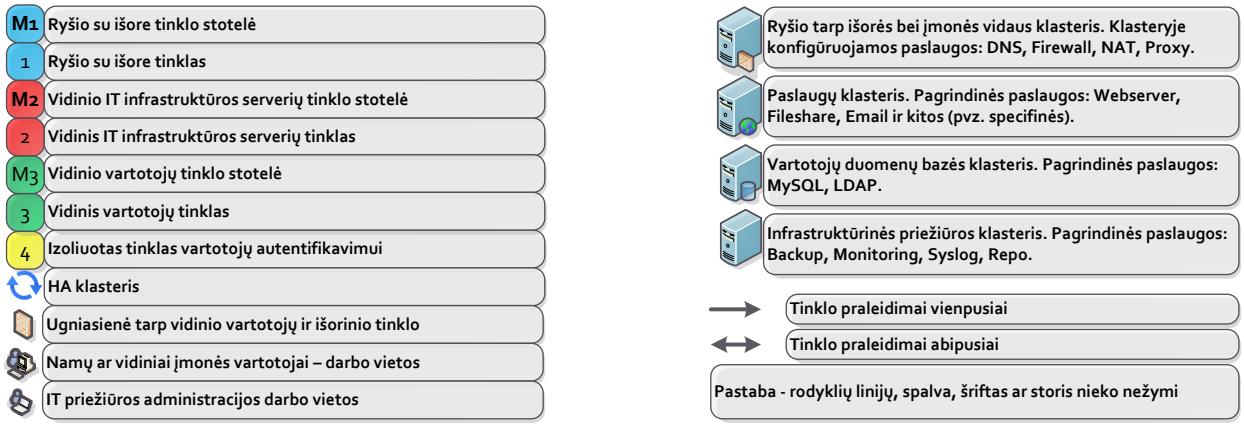
Galutiniame šio projekto sprendimo modelyje bus pateikti du pagrindiniai aspektai, kuriais vadovaujantis įstaiga turės galimybę atsinaujinti esamą infrastruktūrą:

- IT paslaugų infrastruktūros architektūra;
- Centralizuota nepertraukiamo tiekimo vartotojų duomenų bazė.

Įstaigos IT infrastruktūra pateikta pav. 8. Infrastruktūra yra išskaidoma trejais pagrindiniais aspektais: potinkliais, paslaugų serverių klasterių bei ryšio kanalais. Schemos žymėjimo paaiškinimai pateikti pav. 9.

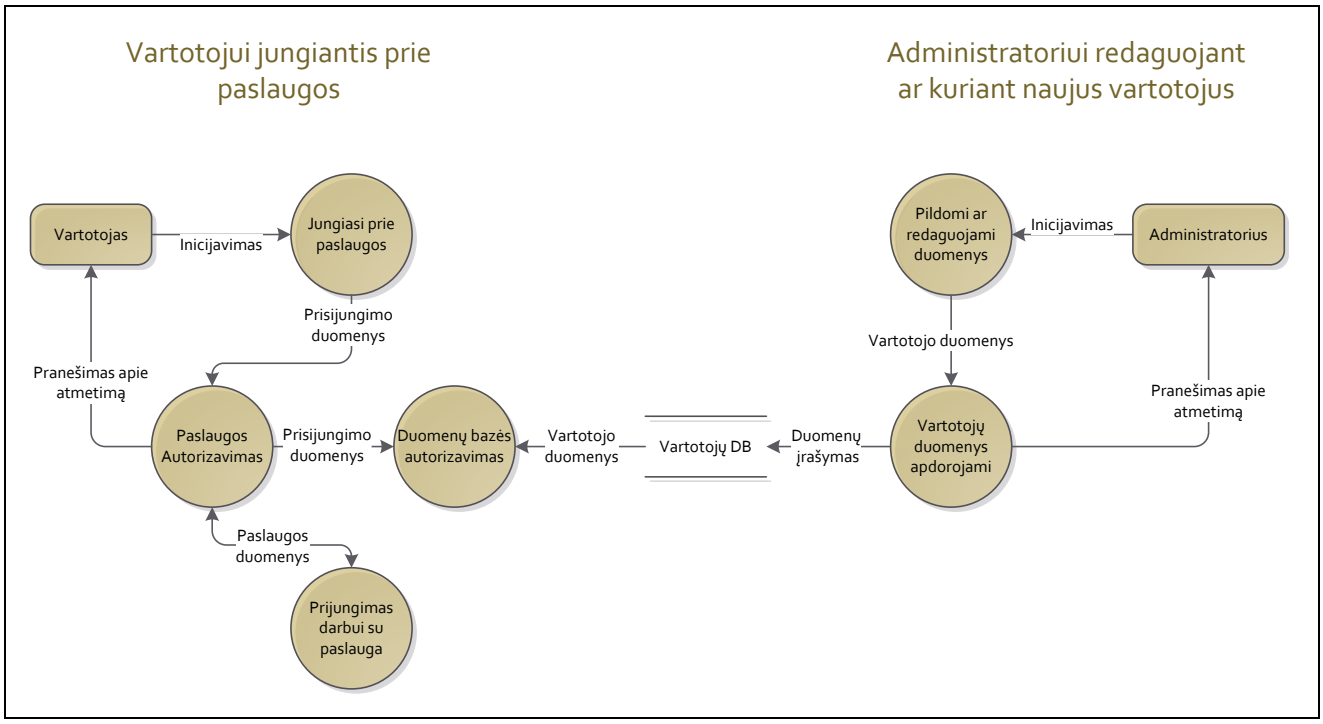


Pav. 8. Izoliuoto autentifikavimosi bei nepertraukiamų paslaugų įmonės ar įstaigos infrastruktūra su priežiūra



Pav. 9. Žymėjimų paaiškinimai iš pav. 8

Sekančiame pav. 10 pateikta duomenų srautų diagrama atspindinti centralizuoto autentifikavimo duomenų bazės veikimą.



Pav. 10. Autentifikavimosi duomenų srauto diagrama

Projekto galutiniame modelyje, bus aptarti svarbūs saugumo aspektai, susiję su praleidimais vidinio IT infrastruktūros tinkle, kurių pagalba visiškai izoliuojamas nepageidaujamas įeinantis ryšys.

3.3. Projekto vykdymo planas

		1	2	3	4	5	6	7	8
2012	rūgsėjis	+		+	+				
2012	spalis	+			+				
2012	lapkritis	+			+				
2012	gruodis	+					+		+
2013	sausis	+		+					
2013	vasaris	+	+						
2013	kovas	+	+						
2013	balandis	+	+						
2013	gegužė	+	+	+			+		+
2013	birželis		+						
2013	liepa		+						
2013	rugpjūtis		+			+			
2013	rugsėjis			+		+			
2013	spalis					+			
2013	lapkritis					+			
2013	gruodis					+		+	
2014	sausis			+			+	+	+
2014	vasaris							+	
2014	kovas							+	
2014	balandis							+	
2014	gegužė			+					+

4. Realizacinė dalis

4.1. Darbų eigos grafas

		1	2	3	4	5	6	7	8
2012	rūgsėjis	+		+	+				
2012	spalis	+			+				
2012	lapkritis	+			+				
2012	gruodis	+					+		+
2013	sausis	+		+					
2013	vasaris	+	+						
2013	kovas	+	+						
2013	balandis	+	+						
2013	gegužė	+	+	+			+		+
2013	birželis		+						
2013	liepa		+						
2013	rugpjūtis		+			+			
2013	rugsėjis			+		+			
2013	spalis					+			
2013	lapkritis					+			
2013	gruodis					+		+	
2014	sausis			+			+	+	+
2014	vasaris							+	
2014	kovas							+	
2014	balandis							+	
2014	gegužė			+					+

4.2. Tinklo topologija

Kadangi darbe atliekamas esamų sistemų migravimas, reikia atkreipti dėmesį į tai, ar migravimo metu senųjų serverių pasiekiamumas būtinas, jeigu taip, naujoji infrastruktūra integruojama į esamą tinklo topologiją. Jeigu yra galimybės rekomenduojama kurti naują infrastruktūrą lygiagrečiai.

Tinklo topologiją įstaigos infrastruktūroje sudaro sekantys elementai:

- LAN adresavimas;
- ugniasienė;
- NAT.

Pradžioje pasirenkama kokie potinkliai bus naudojami vidiniam vartotojų, vidiniam klasterių bei ryšio tarp paslaugų ir duomenų bazių klasterių tinklui. Rekomenduojamos gairės į kurias reikėtų atsižvelgti pasirenkant naują tinklo topologiją:

- Nenaudoti pirmojo C klasės IP adresų potinklio. Šis potinklis dažnai naudojamas kaip standartinis nedidelio aptarnavimo mąsto įrenginiuose, pvz. bevieliame maršrutizatoriuose. Panašaus įrenginio pajungimas į jau esamą tinklą su šiuo potinkliu priveda prie sutrikimų tinkle.
- Tarp vidinio vartotojų ir klasterių serverio rekomenduojama naudoti skirtingos klasės IP adresų potinklius, tai supaprastina NAT paketų transliavimo sprendimą.
- Jeigu įstaigoje yra numeruojami kabinetai, galima patogiai pritaikyti A klasės IP adresų potinklius kaip kabinetų ženklinimą. Pavyzdžiui, 1 a. 2 kab. 5 komp. – 10.1.2.5, 3 a. 7 k. 2 komp. – 10.3.7.2 ir t.t.

Kuriant lygiagrečiai naują infrastruktūrą šiame darbe rekomenduojama IT infrastruktūros vietiniam potinkliui naudotis B klasės, o vartotojų vietiniam potinkliui A klasės IP adresavimu. B klasės IP potinkliai turi didesnį kiekį adresų nei C, nors 255 IP adresų per potinklį atrodo pakankamai, tačiau susiduriama su problemomis kai norima izoliuoti vienų serverių potinklius nuo kitų, arba surišti adresus su serverių pavadinimu ar kontekstu.

4.3. Vartotojų valdymo centralizavimas

Tapatybės valdymo esamų sprendimų modeliai reikalauja didelių kaštų, bei valstybinės įstaigos tiesiogiai įsigyti pasirinktos produkcijos negali, dėl viešųjų pirkimų. Įprastu atveju, įstaigos taiko atviro kodo nemokamą programinę įrangą, arba įrangą, kuri privalo būti pagal pasirašytas partnerystes sutartis. Tapatybės valdymo esantys sprendimai taip pat gali būti atmetami ir dėl esančio pernelyg didelio funkcionalumo ar kaip minėta prieš tai, dėl sudėtingo senųjų sistemų migravimo.

Atviro kodo tinklinės programinės įrangos autentifikavimosi būdus galima išskirti į pagrindines tris grupes:

- Savadarbės (angl. custom) sistemos arba sistemos naudojančios autentifikavimuisi duomenų bazes – įvairaus pritaikomumo sistemos kurių vartotojo autentifikavimosi duomenys saugomi duomenų bazėse. Į šia grupę įeina ir tokios sistemos kaip – įvairūs TVS (turinio valdymo sistemos) sprendimai, el. paštas ir kiti.
- Sistemos integruotos su Linux OS – tokiose sistemose autentifikavimas vyksta remiantis pagrindinėje OS sistemoje registruotais vartotojais. Į šia grupę įeina ir tokios sistemos kaip – Samba (failų bendrinimo servisas), FTP (failų perdavimo sistema) ir kiti.

- Unikalios sistemos – tokiose sistemose autentifikavimas gali būti įvairus – nestandartizuotas ar unikalus, priklausomai nuo kūrėjo, pavyzdžiui, kai sistemos vartotojai saugomi atskirame faile. Į šia grupę įeina ir tokios sistemos kaip – Ezproxy (įgaliotojo serverio servisas), smbldap (Perl LDAP sprendimas) ir kiti.

Įprastu atveju didžiausią dalį tinklinės programinės įrangos įstaigose bei įmonėse apima savadarbės arba duomenų bazes autentifikavimui naudojančios sistemos.

Remiantis Šiaulių valstybinės kolegijos duomenimis, šiuo metu didžiausią dalį apima informacinės sistemos, jų yra daugiau nei dešimt, ir autentifikavimosi duomenys saugomi atskirose duomenų bazėse (žr. 4 lentelę).

4 lentelė. Pavyzdinės įstaigos dabartinė autentifikavimosi situacija

E. paslauga	Prisijungimo vardas	Slaptažodis	Galimybė pasikeisti slaptažodį
El. paštas	v.pavarde@domenas.lt	Generuojamas (komplikuotas)	Yra
Studentų darbo vieta	varpav[n]	Paskutiniai šeši asmens kodo skaičiai	Yra
Darbuotojų darbo vieta	vardas	Nėra	Yra
Intranetas	vardpa[m]	Neviešinama	Nėra
Gedimų sistema	Nesusisteminta	Asmeninis	Yra
Įgaliotasis bibliotekos serveris	0ASMENSKODAS	Neviešinama	Nėra
Moodle	Nesusisteminta	Nesusisteminta	Yra
Informacinės sistemos	Nesusisteminta	Nesusisteminta	Nėra
Kitos, ITC skyriaus nevaldomos tinklinės sistemos	Nesusisteminta	Nesusisteminta	Yra

Moodle skirtingų sistemų kolegijoje yra daugiau nei trys. Informacinių sistemų – daugiau nei dešimt, ir visos jos turi atskiras autentifikavimosi duomenų bazes. Kai kuriomis sistemomis kai kurie darbuotojai nebesinaudoja vien dėl to (pavyzdžiui gedimų), kad nepamena prisijungimų ar net pačių sistemų. Kai kuriomis sistemos, skirtomis naudotis dideliam kiekiui vartotojų, naudojasi vos keli asmenys.

4 lentelėje pateiktas reikšmė „Nesusisteminta“ yra bet kokia simbolių seka (gali būti atsitiktinė) ir nėra lygi kitai, taip pat įvardytai reikšmei. Reikšmės „Neviešinama“ paslėptos, dėl galimo slaptažodžių sistemos pažeidimo, reikia įvertinti tik tai, kad jose naudojamos slaptažodžių sistema taip pat skiriasi nuo visų kitų ir nelygios tarpusavyje.

Siekiama pertvarkymu sukurti sistemą, kurioje prisijungimo vardas bei slaptažodis yra susieti tarp visų sistemų, t.y. ar vartotojas jungiasi į „Moodle“ ar į kokią informacinę sistemą, naudoja tą patį vartotojo vardą ir slaptažodį (žr. 5 lentelėje):

5 lentelė. Pavyzdinės įstaigos pritaikius sprendimą autentifikavimosi situacija

E. paslauga	Prisijungimo vardas	Slaptažodis	Galimybė pasikeisti slaptažodį
El. paštas	Vartotojo_ID@domenas.lt	Vartotojo_Pass	Yra
Studentų darbo vieta	Vartotojo_ID	Vartotojo_Pass	Yra
Darbuotojų darbo vieta	Vartotojo_ID	Vartotojo_Pass	Yra
Intranetas	Vartotojo_ID	Vartotojo_Pass	Yra
Gedimų sistema	Vartotojo_ID	Vartotojo_Pass	Yra
Įgaliotasis bibliotekos serveris	Vartotojo_ID	Vartotojo_Pass	Yra
Moodle	Vartotojo_ID	Vartotojo_Pass	Yra
Informacinės sistemos	Vartotojo_ID	Vartotojo_Pass	Yra
Kitos, ITC skyriaus nevaldomos tinklinės sistemos	Vartotojo_ID	Vartotojo_Pass	Yra

Visą unifikacijos procesą galima atlikti, tiesiogiai nekeičiant esamų sistemų. Darbo pradžiai svarbu apsibrėžti reikiamą techninę ir programinę įrangą. Kadangi planuojama sistema, turi būti centralizuota, rekomenduojama ją diegti švariame serveryje ar virtualioje mašinoje. Reikalinga atviro kodo programinė įranga – operacinė sistema su užduočių valdymo įrankiu (angl. job scheduler), sinchronizacijos įrankis bei duomenų bazė. Tinkamos programinės įrangos pavyzdys: Linux OS (CentOS distribucija), „cron“, „rsync“ , „mysqld“ ir „PHP“.

Siekiant unifikuoti visas sistemas, iš pradžių reikia sumodeliuoti unifikacijos sprendimus kiekvienai tinklinės programinės įrangos autentifikavimosi grupei atskirai.

Savadarbės (angl. custom) sistemos arba naudojančios autentifikavimuisi duomenų bazes sistemos – kaip aptarta ankščiau, visos šio pobūdžio sistemos autentifikavimuisi naudoja duomenų bazes, tik kiekviena – skirtingas. Tokiu atveju, kuriama nauja duomenų bazė, kuri bus bendra visoms sistemoms. Reikia įvertinti, kad duomenų bazės architektūra, turi būti tinkamai parengta – joje turi būti visoms sistemoms reikalinga informacija apie vartotojus, bei informacija apie sistemas, kuriomis vartotojas gali naudotis (žr. pav. 11). Lengviausias pritaikymas yra savadarbėms sistemoms, sudėtingesnis atvejis darant su kitų kūrėjų įranga – reikės papildomų laiko sąnaudų pritaikyti kitų kūrėjų tinklinę programinę įrangą pagal pertvarkomą sistemą, tačiau kadangi įranga atviro kodo, tai yra įmanoma. Esant tokiai duomenų bazei, nesudėtinga sukurti primityvų tinklalapį, kuriame vartotojai prisijungę galės pasikeisti slaptažodžius.

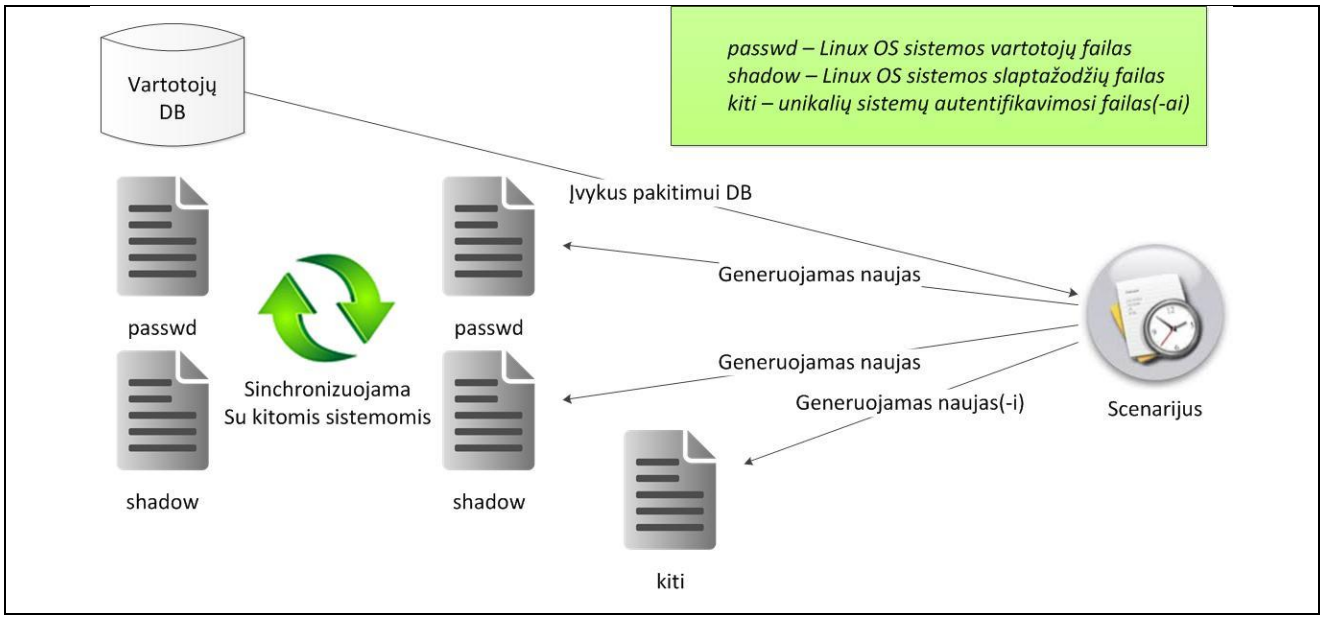
ID	Vardas	Pavardė	Vartotojo_ID	Vartotojo_Pass	Veikla	Kita_Informacija	Sistema001	Sistema002	Sistema003
1	Vardenis	Pavardenis	19829876	74b87337454200d4d33f80c4663dc5e5	Studentas	Esant poreikiui tokių laukų gali būti daugiau	0	1	0
2	Vardenė	Pavardenė	19829817	7815696ecbf1c96e6894b779456d330e	Darbuotojas	Esant poreikiui tokių laukų gali būti daugiau	1	1	0
3	Administratorius	Administrius	19828223	b5b037a78522671b89a2c1b21d9b80c6	Darbuotojas	Esant poreikiui tokių laukų gali būti daugiau	1	1	1

Pav. 11. Unifikuotos autentifikavimosi pavyzdinis duomenų bazės fragmentas

Integruotos su Linux OS sistemos – pačios OS sistemos vartotojai paprastai saugomi „passwd“ faile, o slaptažodžių maišos (angl. hash) – faile „shadow“. Įvykus pakitimams duomenų bazėje, OS scenarijų bei užduočių valdymo pagalba, šie failai pergeneruojami naudojantis duomenimis iš duomenų bazių. Tokios sistemos kaip FTP, ar „Samba“, su papildomomis komandomis, gali užregistruoti – išregistruoti vartotojus servisam naudotis, tad pavyzdžiui keičiantis „passwd“ failui, jis lyginamas su prieš tai esančiu, ir jei atsiranda vartotojų, kurių jau nebereikia, scenarijuje įvykdoma to serviso išregistravimo komanda (jeigu norima atskirti naudojimosi servisus (pvz. vartotojui norima suteikti naudotis OS viena bet ne kita sistema). Kad, išlaikyti viską vientisai visose sistemose, pasikeitus „passwd“ ir „shadow“ failams, su „rsync“ pagalba, galima atnaujinti šiuos failus ir kituose serveriuose. Išimtis, kai serveriams nereikia autentifikavimosi failų sinchronizacijos, savarankiški (dirba tiesiogiai su duomenų baze – pavyzdžiui LDAP).

Unikalios sistemos – sukurti universalių autentifikavimosi galimybių šioms sistemoms nėra, kitaip jos nebūtų unikalios. Kiekvienai tokiai sistemai, automatizuotus vartotojų sąrašą turinčius failus kurti reikia atskirai, prieš tai pilnai išsiaiškinus įsigalinus į pateiktą atvirą kodą kaip veikia sistemos autentifikavimas.

Unifikavus grupes atskirai, galima jas jungti į bendrą sistemą. Unifikuotos sistemos pagrindinio mechanizmo modelis sudarytas iš šių dalių: vartotojų duomenų bazės, užduočių valdymo įrankio fiksuojančio pakitimus duomenų bazėje, scenarijų, Linux OS sistemos vartotojų autentifikavimosi failų, bei unikalių sistemų autentifikavimosi failų. Įvykus pakitimui duomenų bazėje, užduočių valdymo įrankis paleidžia pagrindinį scenarijų, kuris remiantis informacija iš duomenų bazės sugeneruoja naujus operacinės ir unikalių sistemų autentifikavimosi failus. Šiais failais sistema pakeičia senus autentifikavimosi failus, o sinchronizacijos įrankis pastebėjęs pakitimus, sinchronizuoja naujus autentifikavimosi failus su nutolusiomis sistemomis (žr. pav. 12).



Pav. 12. Bendro visų sistemų autentifikavimosi mechanizmas

Scenarijai sudaryti iš OS komandų ir PHP programavimo kalbos darbui su duomenų bazėmis elementais. Scenarijaus tikslas, sugeneruoti taisyklingus Linux OS bei unikalių sistemų autentifikavimosi failus, užregistruoti naujus sistemos vartotojus, išregistruoti senus ar nereikalingus sistemos vartotojus, pakeisti senus naujai sugeneruotais autentifikavimosi failais. Kaip aptarta integruotomis su Linux OS sistemų unifikavimo modelyje sudaryme, galima užregistruoti – išregistruoti specifinius vartotojus, tik pasirinktoms sistemoms. Pavyzdžiui vartotojui pakeitus slaptažodį bus atliekami tokie žingsniai:

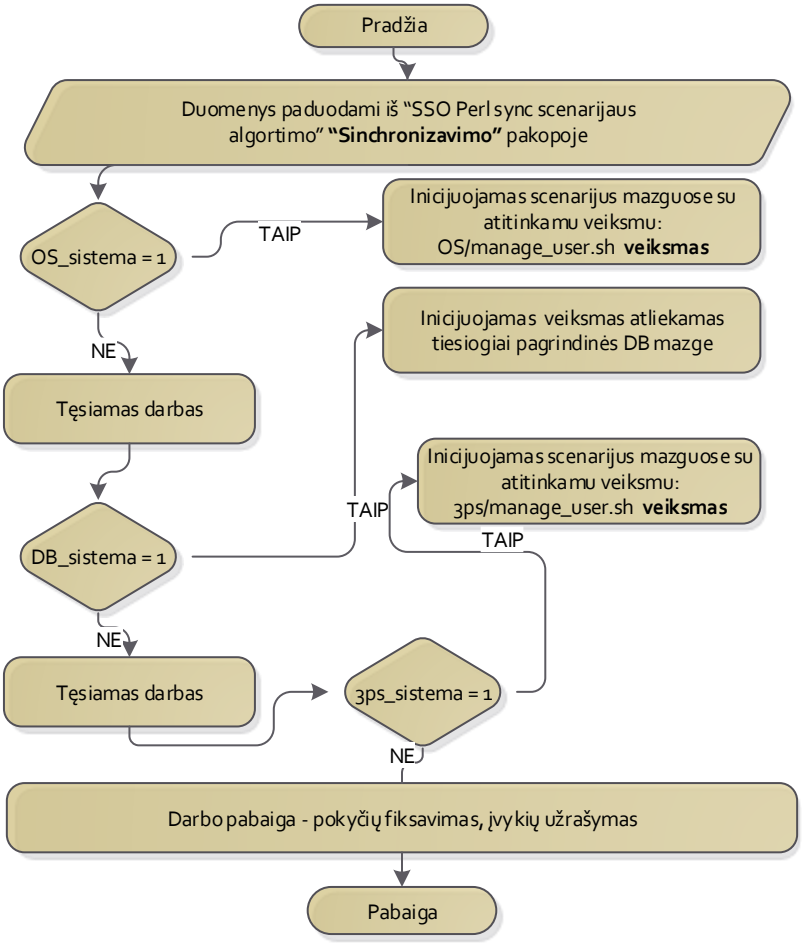
1. Duomenų bazėje pagal vartotoją pakeičiama slaptažodžio maiša.
2. Įvykus DB pakitimams OS užduočių tvarkyklė paleidžia integruotos su Linux OS vartotojų registravimo scenarijų.
3. Scenarijus generuoja naujus „passwd“ , „shadow“ ir unikalių sistemų failus.
4. Sugeneravęs failus, tikrinama ar reikia išregistruoti kuriuos nors vartotojus iš kurių nors servisų, jei tokių randa, išregistruoja ir tęsia darbą, priešingu atveju tiesiog tęsia darbą.
5. Pakeičia senus failus naujais.
6. „rsync“ pastebėjęs pasikeitusius failus, sinchronizuoja pokyčius su kitomis sistemomis.
7. Visų sistemų duomenys atnaujinti.

Analogiški žingsniai galioja ir naujų vartotojų registravimui, išregistravimui, prieigos prie sistemų suteikimui bei draudimui.

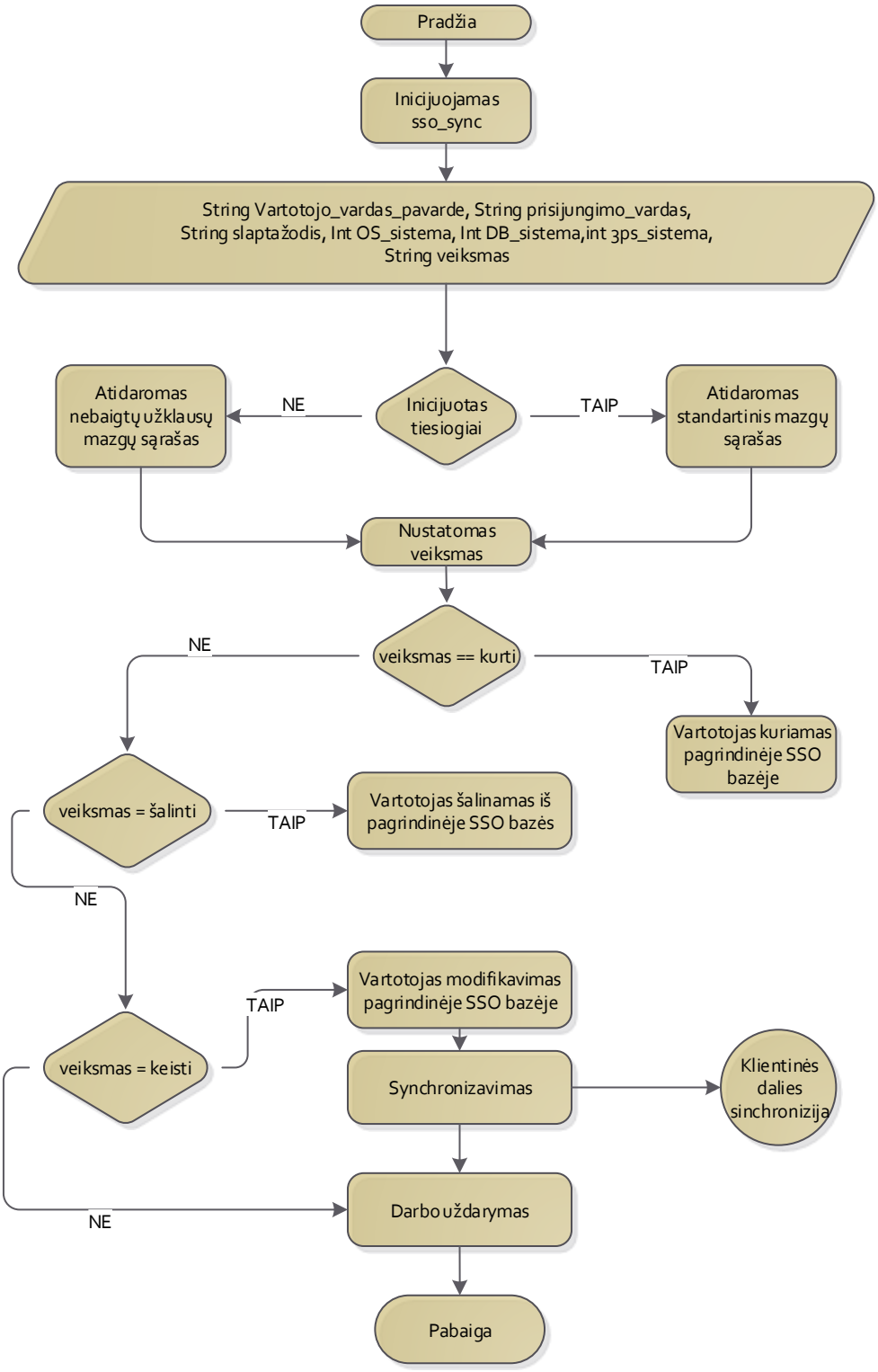
Buvo atliktas praktinis testas su 4 mazgais nustatyti, ar toks mechanizmas įmanomas. Pilnas praktinio testo atviras kodas pateiktas priede nr. 6. Pabrėžiama, kad kodas buvo pritaikytas pagal minimalią infrastruktūrą ir tik testavimui, tad šis kodas tiesiogiai naujoje infrastruktūroje taikymui netinka. Šis kodas tinkamas tik kaip pavyzdys kuriant analoginį taikytina scenarijų įmonės ar įstaigos esamai-kuriamai infrastruktūrai.

Scenarijai rašyti perl bei bash programavimo(scenarijų) kalbomis. Pagrinde išskiriami du rinkianiai įrankių:

a. master_sso – pagrindinis perl scenarijus ir jo papildiniai skirti centralizuotam vartotojų valdymui iš centrinio mazgo. Diegiama vartotojų DB klasteryje.
b. client_sso – bash scenarijai, atliekantis vartotojų modifikacijų darbus sistemos viduje, pagal master_sso sync inicijavimą.



Algoritmas 1. Klientinės dalies sinchronizacijos algoritmas



Algoritmas 2. SSO perl sync scenarijaus algoritmas

Pirmasis bei antrasis pateikti algoritmai, atspindi centralizuoto vartotojų valdymą. Scenarijai naudojami išskirstytuose mazguose. Tarpusavyje mazgai turi būti „patikimi“ (pvz. surišti RSA

raktais specifinio vartotojo privilegijomis). Scenarijai išskaidomi į mazgus, pagal jų paskirtį. Centralizuotos duomenų bazės mazge įkeliami „master_sso“ scenarijai ir sutvarkoma užduočių tvarkyklė „cron“ pagal priede pateiktas instrukcijas. Valdomuose mazguose, visuose įkeliam „client_sso“ scenarijai“, kurios įvykus pokyčiams inicijuos centralizuotas duomenų bazės mazgas.

4.4. Klasterio formavimas

Šiame darbe pasirinktas aktyvaus-pasyvaus HA klasterio formavimas su dviem mazgais. Pradinis pasirengimo žingsnis yra operacinės sistemos diegimas. Diegiama CentOS 6.5 distribucija. Operacinės sistemos diegimas viršija šio darbo taikymo srities rėžius, tad detalus diegimo procesas neaptariamas. Daugiau informacijos dėl diegimo pateikta 1-ajame priede. Toliau darbe apibrėžiami diegiamų serverių pavadinimai: **mazgas_01**, **mazgas_02**.

Sekantys pasirengimo žingsniai yra serverių pajungimas į vieną potinklį. Įprastu atveju užtenka rezervuoti tris IP adresus paslaugos pasiekimui. Nors nebūtina, tačiau yra rekomenduojama paduoti papildomą izoliuotą potinklį su dar dviem IP adresais. Įvardyti trys adresai bus naudojami serverių bei paslaugos pasiekimui, o papildomi du „širdies plakimui“ (angl. heartbeat) stebėti. „Širdies plakimu“ HA klasterių terminologijoje vadinamas procesas klasterių mazgų bei jais kontroliuojamų resursų pasiekiamumui stebėti. Paprastai šis procesas yra atliekamas izoliuotame tuščiame potinklyje „unicast“ arba „multicast“ tinklo protokolu. Kadangi potinklis izoliuotas, tuščias ir niekam daugiau nenaudojamas, tokiu atveju yra užtikrinamas itin greitas klasterio mazgų pasiekiamumo patikrinimas tinkle.

```
# Rezervuojami adresai pirmajame potinklyje
mazgas_01 = 172.16.10.11/24
mazgas_02 = 172.16.10.12/24
vIP = 172.16.10.10/24
# Rezervuojami adresai heartbeat potinklyje
mazgas_01 = 172.16.15.11/24
mazgas_02 = 172.16.15.12/24
```

Įdiegus OS mazguose, sekantys veiksmai yra klasterio paketų diegimas. Pastaba – diegimo ir konfigūracijos veiksmai atliekami abiejuose mazguose. CentOS paketų diegimui naudojamas ankščiau minėtas įrankis „yum“. Toliau įvardijamų paketų diegimui galima naudoti standartinę oficialią CentOS paketų repozitoriją, tačiau joje nebus „crm“ klasterio įrankio, kadangi tiekiama alternatyvi „pcs“. Norint naudoti „crm“ reikia papildyti sistemą papildoma repozitorija. Tai galima atlikti sukūrus naują failą kataloge /etc/yum.repos.d/clusterlabs, ir į jį įrašius:

```
[network_ha-clustering_Stable]
name=Stable High Availability/Clustering packages (CentOS_CentOS-6)
type=rpm-md
baseurl=http://download.opensuse.org/repositories/network:/ha-
clustering:/Stable/CentOS_CentOS-6/
gpgcheck=1
gpgkey=http://download.opensuse.org/repositories/network:/ha-
clustering:/Stable/CentOS_CentOS-6/repodata/repomd.xml.key
enabled=1
```

Toliau inicijuojamas įrankių diegimas:

```
yum install corosync pacemaker crmsh
```

Sekantys žingsniai, yra programinės įrangos konfigūravimas klasterio formavimui. Svarbu prieš pradedant konfigūruoti klasterio įrangą, suderinti ntp servisą, kad abiejų mazgų laikas būtų visiškai vienodas. Suderinus ntp, pereinama prie corosync konfigūravimo failo /etc/corosync/corosync.conf (šalia kataloge galima rasti pavyzdinį failą corosync.conf.example). Į failą įrašome, heartbeat potinklio sąsają su mazgais.

```
compatibility: whitetank

aisexec {
        user:        root
        group:       root
        }
totem {
        version: 2
        secauth: off
        interface {
                member {
                        memberaddr: 172.16.15.11
                }
                member {
                        memberaddr: 172.16.15.12
                }
                ringnumber: 0
                bindnetaddr: 172.16.15.0
                mcastport: 5405
                ttl: 1
        }
        transport: udpu
}
logging {
        fileline: off
        to_logfile: yes
        to_syslog: yes
        logfile: /var/log/cluster/corosync.log
        debug: off
        timestamp: on
        logger_subsys {
                subsys: AMF
                debug: off
        }
}
```

Rekomenduojama kad abu mazgai būtų pasiekiami DNS vardais. Tą galima atlikti IT infrastruktūros DNS serveryje arba lokaliai serveryje, įrašius sąsajas /etc/hosts faile.

Prieš startuojant klasterį, jeigu yra naudojama ugniasienė, reikia pridėti taisyklę, praleisiančią „heartbeat“ užklausas. CentOS tai galima atlikti įvykdžius šią komandą:

```
iptables -A INPUT -p udp -m multiport --dports 5404,5405 -j ACCEPT
# Norint kad ši taisyklė išliktų po ugniasienės perkrovimo, ją reikia pridėti į
/etc/sysconfig/iptables failą.
```

Abiejuose mazguose paleidžiamas corosync servisas. Šį servisą (bei kitus svarbius klasterio formavimui, pvz. ntp) reikia įjungti, kad būtų paleidžiamas sistemos startavimo metu. Tokiu atveju, sutrikęs mazgas, po perkrovimo galės vėl prisijungti prie HA klasterio.

```
chkconfig ntp on
chkconfig corosync on
chkconfig pacemaker on
chkconfig | egrep 'ntp|corosync|pacemaker'
corosync        0:off   1:off   2:on    3:on    4:on    5:on    6:off
ntpd            0:off   1:off   2:on    3:on    4:on    5:on    6:off
ntpdate         0:off   1:off   2:off   3:off   4:off   5:off   6:off
pacemaker       0:off   1:off   2:on    3:on    4:on    5:on    6:off

/etc/init.d/corosync start

# Tikriname ar mazgas sėkmingai prisijungė prie klasterio
corosync-cfgtool -s

Printing ring status.
Local node ID 873638080
RING ID 0
       id      = 172.16.15.11
      status  = ring 0 active with no faults

Printing ring status.
Local node ID 873638080
RING ID 0
       id      = 172.16.15.12
      status  = ring 0 active with no faults
```

Svarbi saugumo išimtis „selinux“ perjungimas į leistiną (angl. permissive) režimą. Šis įrankis yra CentOS dalis kuris numatytai veikia priverstinio vykdymo (angl. enforcing) režimu. Įrankis skirtas papildomos programinės įrangos stebėjimui ir jos veiksmų ribojimui, jei veiksmai pagal „selinux“ politiką yra draudžiami. Kartai sutinkama programinė įranga, kuri pažeidžia „selinux“ numatytą politiką, tad tenka redaguoti politiką arba įrankį išjungti. Dėl sudėtingo „selinux“ konfigūravimo sistemų administratoriai dažnai pasirenka šį įrankį tiesiog išjungti. Jeigu „selinux“ trukdė klasterio formavimui ir pasirinkta įrankį išjungti, svarbu palikti leistiną režimą, o

neišjungti įrankį aplamai. Tokiu atveju, draustini įvykiai bus įrašyti į /var/log/audit/audit.log failą bet nebus pritaikyti, tad ieškant saugumo kompromitavimo įvykių sistemoje galima bus naudotis šiais įrašais. Išjungti įrankį galima atlikus:

```
setenfore 0
# norint įjungti šį režimą visam laikui, faile /etc/selinux/config nustatome atributą SELINUX=permissive
```

Šioje pakopoje turimas suformuotas dviejų mazgų klasteris, tačiau kol kas nesukonfigūruoti automatinio perjungimo įrankiai. Startuojamas pacemaker servisas ir atliekama klasterio būsenos patikra.

```
/etc/init.d/pacemaker start
crm status

Last updated: Thu May  8 21:09:19 2014
Last change: Wed Apr 30 13:56:06 2014 via crm_attribute on mazgas_01
Stack: classic openais (with plugin)
Current DC: mazgas_01 - partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
2 Nodes configured, 2 expected votes
0 Resources configured

Online: [mazgas_01 mazgas_02]
```

Pagal statusą matoma, kad suformuotame klasteryje yra 2 mazgai, tačiau nėra sukonfigūruoti jokie resursai. Resursais vadinamas servisai ar paslaugos stebimos bei valdomos „pacemaker“, pavyzdžiui, kad užtikrinti jog resursai bus perkelti į kitą mazgą sutrikus mazgui ant kurio jis buvo aktyvus. Sukuriamas naujas resursas vIP adresui priskirti.

```
crm configure primitive ClusterIP ocf:heartbeat:IPaddr2 \
     params ip=172.16.10.10 cidr_netmask=32 \
     op monitor interval=30s

# patikriname ar atsirado resursas
crm status
Last updated: Thu May  8 21:16:51 2014
Last change: Wed Apr 30 13:56:06 2014 via crm_attribute on mazgas_01
Stack: classic openais (with plugin)
Current DC: mazgas_01- partition with quorum
Version: 1.1.10-14.el6_5.3-368c726
2 Nodes configured, 2 expected votes
1 Resources configured


#Online: [mazgas_01 mazgas_02]
 Resource ClusterIP
     ClusterIP (ocf::heartbeat:IPaddr2):       Started mazgas_01

# patikriname kaip atrodo priskirti IP ant mazgas_01
```

```
ip a
1: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
qlen 1000
    link/ether 00:15:5d:5c:26:0b brd ff:ff:ff:ff:ff:ff
    inet 172.16.10.11/24 brd 172.16.15.255 scope global eth0
    inet 172.16.10.10/32 brd 172.16.15.255 scope global eth0
    inet6 fe80::215:5dff:fe5c:260b/64 scope link
       valid_lft forever preferred_lft forever

# patikriname kaip atrodo priskirti IP ant mazgas_02
ip a
1: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
qlen 1000
    link/ether 00:15:5d:5c:26:0b brd ff:ff:ff:ff:ff:ff
    inet 172.16.10.12/24 brd 172.16.15.255 scope global eth0
    inet6 fe80::215:5dff:fe5c:260b/64 scope link
       valid_lft forever preferred_lft forever
```

Šioje pakopoje sukonfigūruotas dviejų mazgų klasteris su vIP adresu, kuris bus automatiškai perskirtas iš mazgas_01 į mazgas_02 pirmojo mazgo sutrikimo atveju. Dabar, kai HA klasteris suformuotas, klasterį galima papildyti paslaugų resursais, tokiais kaip web servisas, duomenų bazės ir t.t. Naujų resursų pridėjimas yra atliekamas analogiškai vIP resurso sukūrimui. Priede nr.2 ir nr.3 yra pateikti rankinio „corosync“ crmsh ir pcs aplinkų valdymo komandų rinkiniai bei pavyzdžiai kaip pridėti kitų paslaugų resursus.

4.5. Replikavimo formavimas

Nors yra ir daugiau atviro kodo nemokamų duomenų bazių sprendimų, šiame darbe yra nagrinėjamos tik MySQL ir LDAP. Prieš tai minėtą jog MySQL ir LDAP yra skirtingos architektūros duomenų bazės, MySQL – reliacinė, LDAP – hierarchinė. MySQL replikavimo formavimo procesas sudėtingesnis nei LDAP, tad toliau bus pateiktas tik šios duomenų bazės replikavimo metodas. Priede nr. 4 yra papildomos informacijos kaip atlikti LDAP duomenų bazės replikavimą.

Prieš pradedant MySQL duomenų bazės replikavimą, abiejuose mazguose reikia atlikti šiuos pasirengiamuosius veiksmus:

- sukurti replikavimo vartotoją;
- ugniasienėje atlikti MySql prievado praleidimą;
- suderinti ntp servisą.

Pasirinktas replikavimo metodas master-master, iš tiesų yra master-slave, tiesiog abu mazgai vienu metu vienas kitam ir master ir slave. Taigi iš pradžių bus padarytas vienpusis master-slave

režimas ir tada papildytas iki master-master. Nors ir ne būtina, bet darbą palengvina, jei abiejų mazgu replikavimo vartotojas su tuo pačiu slaptažodžiu. Nauju replikavimo vartotojo sukūrimas, vykdyti abiejuose mazguose:

```
# mazgas_01
mysql> grant replication slave on *.* to 'replica'@'172.16.10.12' identified by
'7=ZRqWgyd';
mysql> flush privileges;

# mazgas_02
mysql> grant replication slave on *.* to 'replica'@'172.16.10.11' identified by
'7=ZRqWgyd';
mysql> flush privileges;
```

Toliau kuriamas master tiekėjo režimas abiejuose mazguose. Ši konfigūracija atliekama /etc/my.cnf faile, svarbu, kad kiekvienas replikuojamas mazgas konfigūracijoje turi savo ID, ir šie ID negali sutapti su kitų mazgais.

```
# mazgas_01 /etc/my.cnf

[mysqld]

server-id=1
log-bin=node_01_mysql-bin

# jeigu vartotojų slaptažodžiai skirtingi, svarbu, kad MySQL vartotojų duomenų
bazė nebūtų sinchronizuojama
binlog-ignore-db=mysql

# mazgas_02/etc/my.cnf

[mysqld]

server-id=2
log-bin= node_02_mysql-bin

# jeigu vartotojų slaptažodžiai skirtingi, svarbu, kad MySQL vartotojų duomenų
bazė nebūtų sinchronizuojama
binlog-ignore-db=mysql
```

Atlikus šiuos pakeitimus ir startavus MySQL duomenų bazę, šioje pakopoje, abu mazgai jau veikia master režimu, t.y. veda įrašus apie ateinančias užklausas ir pasiruošusios perduoti informaciją slave mazgams. Šioje pakopoje, rekomenduojama atlikti master duomenų bazės atsarginę kopiją ir ją išskleisti į slave mazgą. Tokiu atveju, svarbu užfiksuoti šiuos parametrus: *binlog file, binlog file position*. Šie duomenys bus reikalingi inicijuojant slave mazgą. Rekomenduojamas įrankis atlikti atsarginei kopijai yra „holland_backup“ apie jį bus pateikta daugiau informacijos „Paslaugų migravimas“ skiltyje.

```
mysql> show master status\G;
*************************** 1. row ***************************
            File: node_01_mysql-bin.000001
        Position: 103
    Binlog_Do_DB:
Binlog_Ignore_DB: mysql
```

Jeigu matomas panašus rezultatas abiejuose mazguose, reiškia master replikavimas jau veikia. Taigi, šioje pakopoje, beliko įjungti replikavimą. Slave mazge, atliekama šį komanda:

```
mysql> CHANGE MASTER TO MASTER_HOST='172.16.10.11', MASTER_USER='replica',
MASTER_PASSWORD='2f5ccee#>vr)', MASTER_LOG_FILE=' node_01_mysql-bin.000001',
MASTER_LOG_POS=103;

mysql>  start slave;
mysql>  show slave status\G;

*************************** 1. row ***************************
Slave_IO_State: Waiting for master to send event
Master_Host: 172.16.10.11
Master_User: replica
Master_Port: 3306
Connect_Retry: 60
Master_Log_File: node_01_mysql-bin.000001
Read_Master_Log_Pos: 103
Slave_IO_Running: Yes
Slave_SQL_Running: Yes
Replicate_Do_DB:
Replicate_Ignore_DB:
...
```

Jeigu matomas panašus vaizdas, reiškia kad replikavimas jau vyksta. Šioje pakopoje mazgas_01 yra master, o mazgas_02 slave. Pirmajame mazge atliekami analogiški veiksmai slave režimo inicijavimui. Atlikus šiuos veiksmus, turimas master-master režimo replikavimas tarp dviejų mazgų. Kaip atlikti replikavimo patikrinimą plačiau priede nr. 5.

4.6. Paslaugų migravimas

Prieš tai realizacinėje dalyje yra aptartos: centralizuoto vartotojų valdymo, HA klasterio bei replikuojamos duomenų bazės įgyvendinimo metodikos. Realizavus šias dalis, iki pateikto galutinio stovio modulio lieka dvi dalys – bazinių paslaugų migravimo bei IT infrastruktūros priežiūros diegimo procesai. Paslaugų migravimui poreikis atsiranda, kai norima esamas sistemas perkelti į naujesnes, patikimesnes platformas, kas šio darbo atveju ir yra siekiama. Paslaugos sudarytos iš dviejų pagrindinių komponentų – darbui bei atvaizdavimui reikalingų duomenų bei duomenų bazės. Migruojant bet kokią paslaugą svarbu atsižvelgti į tai, kaip bus atliekamas šių dviejų komponentų migravimas. Taipogi turi būti įvertinta, kad platforma į kurią migruojama jau yra atitinkamai sukonfigūruota, pvz. konfigūracijos nustatymai ir pan. Rekomenduojama kopijuojant paslaugų

failus bei eksportuojant duomenų bazes paslaugą sustabdyti, tam kad išsaugoti duomenų vientisumą (t.y. kad duomenis nebūtų pakeisti kopijavimo proceso metu).

Darbui ir atvaizdavimui duomenų perkėlimas yra atliekamas itin paprastai. Užtenka failus tiesiog nukopijuoti į naują migruojamą platformą. Svarbu atsižvelgti į kopijuojamų failų ir katalogų teises. Paprasčiausias būdas užtikrinti kad įrašymo/skaitymo teisės nepakistų, suarchyvuoti visus paslaugos failus į bendrą archyvą. Jeigu archyvavimo metodas nepriimtinas, galima naudotis kopijavimo papildomais atributais, pvz. linux „cp -p, scp -p“, kur „-p“ reiškia išlaikyti esamas failų ar katalogų prieigos teises.

Duomenų MySQL ar LDAP bazės migravimas gali būti kiek sudėtingesnis procesas nei failų kopijavimas, tačiau metodas visiškai tas pats. Stabdoma paslauga bei jos duomenų bazė, duomenų bazės įrankiu pagalba atliekama duomenų bazės kopija. Atlikta kopija perkeliama į naują platformą, kurioje taipogi duomenų bazės įrankiais yra importuojama į naująją duomenų bazę. Šis metodas reikalauja paslaugos sustabdymo, bei kol paslauga nebus paleista ant naujos platformos, senosios naudojimas tampa beprasmis, kadangi nebus išlaikyti migravimo procese daromi pakeitimai. Jeigu paslaugos pasiekiamumas migravimo metu labai svarbus, galima pasinaudoti prieš tai šiame darbe aptarta replikavimo metodika duomenų bazėms migruoti. Tokiu atveju, įmanoma užtikrinti, kad kol senoji paslauga veiks, tuo pačiu metu naujai migruojama paslauga turės tuos pačius duomenis su visais pakeitimas. Tada prastovos laikas skiriamas tik paslaugos prieinamumui perjungti (kuris paprastai trumpa itin nedaug laiko).

Apibendrinti paslaugos migravimą galima atliekant šiuos žingsnius:

1. atliekama analogiška paslaugos konfigūracija platformoje į kurią bus migruojama;
2. stabdoma senoji paslauga;
3. perkeliami senosios paslaugos duomenys;
4. startuojama nauja paslauga;
5. atliekamas paslaugos prieinamumo pasiekiamumo perjungimas.

Šiuo metodu galima perkelti visas bazines paslaugas teikiamas šio darbo analizuojamoje įstaigoje, tokias kaip:

- failų saugykla skirta failams saugoti bei keistis;
- LDAP bei MySQL duomenų bazės;

- tinklinės programinės aplinkos (angl. web-applications);
- prieglobos (angl. hosting) paslaugos;
- el. paštas;
- vidinės tinklo tiekimo paslaugos.

4.7. IT infrastruktūros priežiūra

IT infrastruktūros priežiūrai išskiriamos trys pagrindinės paslaugos:

- paslaugų prieinamumo priežiūra;
- atsarginės kopijos;
- sisteminių įrašų konsolidacija.

Šių paslaugų turėjimas infrastruktūroje užtikrina, kad tiekiamos paslaugos yra pasiekiamos maksimaliai įmanomą laiką, neprarandami duomenys, įmanomas duomenų atstatymas, yra centralizuotas įvykių žurnalas bei sistemų našumo grafikai kuriais remiantis galima lengviau nustatyti sistemos gedimus bei poreikius.

Paslaugų prieinamumo priežiūros įrankiai, tai įrankiai, kurie dažniausiai naudoja standartinius SNMP (Simple Network Management Protocol) sprendimus rinkti tam tikrus duomenis apie sistemas bei standartinius tinkle pasiekiamumo įrankius tokius kaip „ping“, „netcat“ ir pan. Šių įrankių junginys, suvedamas į vieningą sistemą, kurioje galima stebėti sistemų našumo grafikus bei susikonfigūruoti pranešimus, jei mazgas arba vienas jo prievadų tinkle tampa nepasiekiamas. Šiuo metu vienas didžiausių tokio tipo verslo klasės atviro kodo įrankių yra „OpenNMS“. Mažesnio mąsto, tačiau patenkinanti minimalius poreikius stebėjimui yra „Cacti“ bei „Zabbix“ įrankiai. Kiekvieno šio įrankio diegimas yra specifiškas, tad prieš diegiant reikėtų susipažinti su įrankių dokumentacija.

Atsarginių kopijų darymas yra būtinas dalykas, norint užtikrinti, kad paslaugų duomenys nebūtų prarasti įvykus nenumatytam sutrikimui. Tačiau kuo dažniau daromos atsarginės kopijos sistemoje, tuo daugiau resursų yra tam sunaudojama. Jeigu padarytos kopijos laikomos toje pačioje sistemoje, tad sutrikus sistemai yra galimybė, kad visos atsarginės kopijos dings kartu su sistema. Atsarginių kopijų darymo dažnumo reikalavimą galima sumažinti iki karto per dieną, kadangi replikavimo sprendimas pasirūpina tuo, kad duomenys būtų vientisi. Tokiu atveju, turint sistemų našumo stebėjimo grafikus, galima nustatyti, kuriuo paros metu, sistemoje atlikti atsargines kopijas

geriausia. Atsarginėms kopijoms yra įvairių įrankių, leidžiančių procesą atlikti centralizuotai ir pagal nustatytus grafikus, tačiau kartais paslaugos būna itin specifiškos, o visos sistemos atsarginės kopijos darymas, gali būti traktuojamas kaip vietos švaistymas. Tokiu atveju, galima tiesiog parašyti scenarijus sistemose, kuriuos nutolusi sistema inicijuos pagal nustatytą tvarkaraštį.

Sisteminių įrašų konsolidacija, tai kai, visų infrastruktūroje esančių sistemų parinktų įvykių žurnalų kopijos yra siunčiamos į centralizuotą serverį. Įrašų kopijų turėjimas atskiroje sistemoje yra itin patogus dauguma atveju, tačiau šiame darbe išskiriami tik šie du – kai įvykių įrašų failai tampa itin dideli, apdorojimui OS sunaudoja didelius kiekius resursų bei sutrikus sistemai, apie sutrikimą bus rasti įrašai kitoje, veikiančioje sistemoje. Linux OS tokiai paslaugai teikti yra atviro kodo nemokamas „syslog-ng“ programinis paketas. Taipogi yra šio paketo agentai skirti ir kitoms OS, tokioms kaip Windows ar Unix.

Visų šių paslaugų turėjimas, palengvina bendrą naštą paslaugų sistemoms bei administruojančiam personalui dauguma atveju. Tačiau tokios sistemos dažnai naudoja apsaugos trumpinius (angl. shortcuts), pvz. išjungta ugniasienė, prieiga prie administracinio vartotojo tarp mazgų be slaptažodžio. Patekus į vieną iš tokių sistemų, galimas pavojus visai bendrai infrastruktūrai. Norint minimizuoti šiuos pavojus rekomenduojama:

- specifinėms operacijoms atlikti, sistemoje kurti atskirus privilegijuotus vartotojus neturinčius pilnų teisių, ir leisti be slaptažodžio nutolusioms sistemoms jungtis tik prie jų, o ne prie pilnas teises turinčių vartotojų;
- išjungti nuotolinę prieigą prie pilnas administracines teises turinčių vartotojų (pvz. root), naudoti privilegijuotus vartotojus;
- sukonfigūruoti ugniasienę kiekvienam mazgui atskirai gali tapti nemažai laiko reikalaujantis bei painus procesas, tačiau svarbus sistemų saugumui.

4.8. Galutinis projekto stovio aprašymas

Realizacinėje šio darbo dalyje yra parodyta, kaip atliekami sprendimai, kurie atitinka pradinio projekto aprašymo schemą pateiktą pav. 8. Išanalizuota kur pritaikyti ir kaip realizuoti: HA klasterį, duomenų replikavimą, paslaugų migravimą, vartotojų centralizavimą bei IT infrastruktūros priežiūros sprendimą. Darbo pradžioje pateikiamos įstaigoje tyrimo metu aptiktos problemos dėl pasenusios programinės įrangos, yra išsprendžiamos pateiktas realizacijos sprendimais. Sekančiame sąraše pateiktos problemos priminimui, bei paaiškinimai, kurios realizacijos dalys jas išsprendžia:

- išskirstytos autentifikavimosi sistemos – išsprendžiamas pritaikius vartotojų centralizavimo sprendimą;
- neužtikrintas nepertraukiamas paslaugų teikimas – išsprendžiamos pritaikius HA klasterį bei replikavimą;
- duomenų praradimo rizika – išsprendžiamos pritaikius HA klasterį, replikavimą, bei atsarginių kopijų darymo centralizavimą;
- necentralizuotas sistemų valdymas – išsprendžiamas pritaikant pav.8 infrastruktūros schemą;
- pritaikytos IT infrastruktūros priežiūros paslaugos pilnavertiškai neišnaudojamos – išsprendžiamos įdiegus centralizuotą IT infrastruktūros priežiūrą infrastruktūroje;
- teikiamų paslaugų nelankstumas – išsprendžiamas atsisakius senųjų sistemų;
- sudėtingas naujų paslaugų integravimas – išsprendžiamas atsisakius senųjų sistemų.

4.9. Rekomendacijos

Šio darbo pritaikymas yra paremtas atviro kodo nemokamais sprendimais stengiantis kuo daugiau sumažinti įmonių ar įstaigų išlaidas atnaujinant savo IT infrastruktūrą. Žemiau pateiktų punktų gilesnė analizė ir pritaikymas gali praplėsti šio darbo pritaikymo galimybes:

- Autentifikavimo duomenų bazės formavimas bei optimizavimas darbe nėra išsamiai analizuojamas, tačiau duomenų bazės susiejimas su paslaugomis, jų prieiga, bazės architektūros bei panaudos efektyvumo optimizavimas gali itin paspartinti visos infrastruktūros darbą bei palengvinti migravimo procesus.
- Darbe minima aukštesnė IdM pakopa vadinama Single Sign on (SSO). SSO sprendimas, leidžia autentifikuotis savo įmonės ar įstaigos vartotojais kitose patikimose įstaigose ar įmonėse. Jeigu darbe tinkamai panaudotas IdM sprendimas, jis tinkamas federaciniam apjungimui su kitomis įstaigomis (Vilda, S, 2012). Rekomenduojama tolimesnė šio aspekto analizė šiame darbe.
- Darbe „vartotojų valdymo centralizavimo“ skiltyje aprašomas realizacijos mechanizmas yra pavyzdinio tipo. Galima atlikti išsamią analizę ir kodą perrašyti optimizuojant saugumo bei vientisumo vietas. Realizacijos modelį galima standartizuoti ir sukurti visus reikalavimus atitinkančius paketus bei galimybę diegti per repozitorijas.

5. Išvados

Remiantis atlikta įstaigos analize, galima teigti, jog Lietuvoje yra įstaigų bei įmonių, kurios savo IT infrastruktūrai stengiasi naudoti tik nemokamą programinę įrangą. Taipogi remiantis tolimesne analize, galima teigti, jog ilgalaikis tokios įrangos eksploatavimas be atnaujinimų IT infrastruktūrą padaro nestabilią, pažeidžiamą bei nelanksčią.

Atlikus tyrimą paaiškėjo, kad galima dinamiškai atnaujinti - permigruoti visą IT sistemų infrastruktūrą remiantis tik atviro kodo nemokamais sprendimais, bei nepertraukiant arba pertraukiant labai trumpam paslaugų tiekimą.

Atlikus tyrimą paaiškėjo, kad remiantis tik atviro kodo nemokama programine įranga, galima sukurti patikimą, saugią bei nepertraukiamo paslaugų tiekimo IT infrastruktūrą.

Darbe pateiktas dinaminis infrastruktūros modelis yra rodiklis, kuriuo remiantis galima planuoti įstaigos ar įmonės pilną IT infrastruktūros atnaujinimą bei migravimą.

6. Literatūros ir informacijos šaltinių sąrašai

- Barauskas, N. „Debesų kompiuterijos pritaikymas, internetinei matematinio programavimo ir modeliavimo sistemai“. *Magistro darbas*, 2013.
- Clusterlab team. „Documentation“. *Pacemaker*. [interaktyvus][žiūrėta: 2014.05.11] http://clusterlabs.org/doc/.
- Dinker, D. „MySQL Replication: Pros and Cons“. *Schooner*. [interaktyvus][žiūrėta: 2014.05.11]http://www.percona.com/files/presentations/percona-live/PLMCE2012/PLMCE2012-SchoonerSQL_PerconaLive2012.pdf
- Kopper, K. „The Linux Enterprise Cluster: Build a Highly Available Cluster with Commodity Hardware and Free Software“. San Francisco: No Starch Press, 2005.
- Pachomov, A., Kaklauskas, L. „Tinklinės programinės įrangos autentifikavimo unifikacijos tyrimas“. *PROFESINĖS STUDIJOS: teorija ir praktika*, 2013.
- Vilda, S. „Naudotojų elektroninės tapatybės valdymo Vilniaus Gedimino technikos universitete tyrimas“. *Magistro darbas*, 2012.
- Wensong, Z. „Linux Virtual Server for Scalable Network Services“. *Ottawa Linux Symposium 2000*, 2000.
- Wensong, Z. Wenzhuo, Z. „Linux Virtual Server Clusters: Build highly-scalable and highly-available network services at low cost“. *Linux magazine*, 2003.

7. Anotacija

Pachomov A. Dinaminis kompiuterinių sistemų infrastruktūros atnaujinimo modelis, pagrįstas atviro kodo sprendimais. Vadovas doc. Dr. Liudvikas Kaklauskas – Šiaulių universitetas Matematikos ir informatikos fakultetas, 2014.

Šiame darbe analizuojamos įmonės su užsistovėjusia bei pasenusia programine įranga dinaminis atnaujinimas utilizuojant naujos atviro kodo nemokamos įrangos galimybes. Formuojamas dinaminis modelis, kuriame pritaikomi nepertraukiamų paslaugų, vartotojų centralizuoto valdymo bei neprarandamų duomenų sprendimai. Taip pat pateikiama analizė, kaip atlikti paslaugų įrangos migravimą bei sukurti pagalbinę IT infrastruktūros dalį, optimizuojančia sistemų priežiūrą bei našumą.

Raktiniai žodžiai: duomenų replikavimas, klasteris, nepertraukiamos paslaugos tiekimas, paslaugų migravimas.

8. Summary

Pachomov A. Dynamic update model of computer systems infrastructure based on open source solutions. Tutor doc. Dr. Liudvikas Kaklauskas – Šiauliai university, Faculty of Mathematics and informatics, 2014.

This paper analyzes, dynamic systems software updates for institution with deprecated system infrastructure using free, open source based solutions using most of it possibilities. Dynamic model is formed, which includes identity management, high availability clustering, data replication and data integrity solutions. Also additional analysis is included for IT infrastructure usage optimization.

Keywords: clustering, high-availability (HA) clustering, date replication, uninterrupted service supply, service migration.

9. Priedai

2. CentOS diegimo instrukcija pateikta papildome priedų kataloge „addon_01“. Diegimo instrukcija paimta iš http://www.tecmint.com/centos-6-3-step-by-step-installation-guide-with-screenshots/
3. „crmsh“ bei „pcs“ valdymo komandų rinkinys bei paaiškinimai pateikti papildome priedų kataloge „addon_02“. Paimta iš https://gist.github.com/beekhof/5589599.
4. Pacemaker resursų pridėjimas, bei „heartbeat“ palaikomų resursų biblioteka pateikti papildome priedų kataloge „addon_03“ . Paimta iš http://www.linux-ha.org/wiki/Resource_Agents.
5. LDAP replikavimo instrukcija pateikta papildome priedų kataloge „addon_04“. Paimta iš http://www.openldap.org/doc/admin24/replication.html.
6. Replikavimo testai atliekami su replikuojamos duomenų bazės įrankiais. Replikavimo patikros mechanizmas bendras - įrašyti „master“ mazge įrašą, ir patikrinti ar „slave“ mazge atsiranda įrašas. Jeigu tai MySQL, tai galima atlikti su standartiniais „SELECT“, „INSERT“ bei „UPDATE“ įrankiais. Jeigu LDAP, tada su „ldapadd, ldapmodify“ komandomis.