KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Justas Umbrasas

Interneto grėsmių analizės karkaso sudarymas ir tyrimas

Baigiamasis magistro darbas

Vadovas
Dr. Dangis Rimkus

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

KOMPIUTERIŲ KATEDRA

TVIRTINU
Katedros vedėjas
(parašas) Prof. dr. Algimantas Venčkauskas
(data)

Interneto grėsmių analizės karkaso sudarymas ir tyrimas

Baigiamasis magistro darbas

Informacijos ir informacinių technologijų sauga (kodas 621E10003)

Vadovas

(parašas) Dr. Dangis Rimkus
(data)

Recenzentas

(parašas) Doc. dr. Rimantas Kavaliūnas

(data)

Projektą atliko
(parašas) Justas Umbrasas
(data)

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS

Informatikos fakultetas

(Fakultetas)

Justas Umbrasas

(Studento vardas, pavardė)

Informacijos ir informacinių technologijų sauga (621E10003)

(Studijų programos pavadinimas, kodas)

Baigiamojo projekto

„Interneto grėsmių analizės karkaso sudarymas ir tyrimas“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 17 m. Gegužės 21 d.

Kaunas

Patvirtinu, kad mano **Justo Umbraso** baigiamasis projektas tema „Interneto grėsmių analizės karkaso sudarymas ir tyrimas“ yra parašytas visiškai savarankiškai, o visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka) *(parašas)*

Umbrasas, J. Tinklalapių apsaugos nuo įsilaužimų modelio sudarymas ir jo efektyvumo įvertinimas. Magistro baigiamasis projektas / vadovas Dr. Dangis Rimkus; Kauno technologijos universitetas, Informatikos fakultetas, Kompiuterių katedra.

Kaunas, 2017. 48 psl.

SANTRAUKA

Dėl nepertraukiamo interneto paslaugų pasiekiamumo iš viso pasaulio jos yra nuolatos atakuojamos. Šios paslaugos gali valdyti konfidencialią informaciją apie bankinius mokėjimus, naudotojų asmeninius duomenis ar kitą reikšmingą informaciją, kurią galima piktavališkai išnaudoti, todėl interneto paslaugų apsauga tampa vienu pagrindiniu rūpesčių. Viešai pasiekiamų paslaugų mastai ir įvairovė yra labai didelė, todėl ypač sudėtinga apsaugoti tinklalapių saugojimo infrastruktūrą nuo galimo įsilaužimo. Šiame darbe išanalizuojamos interneto keliamos grėsmės interneto svetainių saugojimo architektūrai. Peržvelgus pasirinktą Kauno technologijos techninio centro saugomų svetainių architektūrą suprojektuotas grėsmių analizės karkasas ir atlikta jo realizacija. Sukurtoje architektūroje tarpiniai serveriai paskirsto srautą, o galimai pavojingas srautas yra analizuojamas sukonfigūruotos programos lygio ugniasienės. Darbe dalinamasi gautais atliktos realizacijos tyrimo rezultatais.

SUMMARY

Due to continuous availability of Internet services, web applications are constantly under attack. These services often manage bank payments, users' personal data and other significant information; therefore, they need to be protected from any malicious exploitation. As a result, Internet service protection has become one of the main concerns in IT security. Internet services are publicly available and web applications are very diverse, which makes it harder to protect them from a possible break in. In this work, possible threats to Internet hosting architecture are analysed. A selected Kaunas University of technology technical centre Internet hosting architecture was reviewed. Based on the findings Internet threat analysis framework is proposed and implemented in a currently operating infrastructure. In the created architecture, a reverse proxy distributes and balances users' queries to web applications while a configured web application firewall analyses potentially dangerous requests.

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

ĮŽANGA

Baigiamasis projektinis darbas „Interneto grėsmių analizės karkaso sudarymas ir tyrimas“ priklauso Informatikos fakultetui, informacijos ir informacinių technologijų saugos studijų programai.

Interneto paslaugų apsauga yra vienas pagrindinių rūpesčių. Dėl nepertraukiamo paslaugų pasiekiamumo iš viso pasaulio jos yra nuolatos atakuojamos, į jas bandoma įsilaužti. Šios paslaugos gali valdyti konfidencialią informaciją apie bankinius mokėjimus, naudotojų asmeninius duomenis ar kitą reikšmingą informaciją, kurią galima piktavališkai išnaudoti.

Interneto protokolas HTTP (angl. *Hypertext Transfer Protocol*) nebuvo sukurtas sudėtingoms šiuolaikinėms interneto paslaugų struktūroms [1]. Protokolas neturi būsenos, kai reikia sukurti ir palaikyti užmegztą sesiją su paslauga, tam turi būti atliktos atskiros operacijos. Daugumai programų būtina išsaugoti esamą būseną, tai korektiškai turi atlikti naudojama programinė įranga. Šiai funkcijai išpildyti naudojami įvairūs karkasai, programinė įranga, kuri privalo užtikrinti visišką interneto programų saugumą.

Viešai pasiekiamų paslaugų mastai ir įvairovė, interneto tinklalapių scenarijų, skirtingų tinklalapių karkaso naudojimas kuria kompleksiškumą, kuris didina tikimybę, kad bus palikta programavimo spraga. Ne pelno siekianti organizacija „The Open Web Application Security Project“ (OWASP) kas keletą metų sudaro sąrašą kritiškiausių interneto programų saugos problemų, kuris atspindi esminius pažeidžiamumus, kurie išlieka aktualūs [2]. Jau septintus metus pirmoje vietoje išlieka įterpties atakos, antroje vietoje netinkamai veikiantis autentifikavimas ir sesijų valdymas, trečioje – įterptinių komandų atakos (1 lentelė).

1 lentelė. 2017 metų OWASP dešimt kritiškiausių interneto programų rizikų (*Kandidatas publikacijai)* [2]

A1	Kodo įterpimas
A2	Sutrikdytas autentifikavimo ir sesijos valdymas
A3	Sukryžmintas svetainių kodavimas (XSS)
A4	Netinkama prieigos kontrolė
A5	Klaidinga saugumo konfigūracija
A6	Jautrių duomenų atskleidimas
A7	Nepakankama apsauga nuo atakų
A8	Kryžmintas užklausų klastojimas (CSRF)
A9	Komponentų su žinomais pažeidimais naudojimas
A10	Nepakankamai apsaugotos programų programavimo sąsajos (API)

Šiuos pažeidžiamumus būna sudėtinga pastebėti, šiam tikslui yra aibė skirtingų priemonių, kurios skirtos saugumo spragų paieškai. Deja, nėra vieno bendro unifikuoto sprendimo, kuris surastų visas

galimas saugumo spragas. Programinė įranga skirta saugos pažeidžiamumų paieškai gali būti ypač brangi, standartizuoti sprendimai gali būti nepritaikyti konkrečiam panaudojimo atvejui, todėl negalima pasikliauti vienu sprendimu. Rankinis testavimas suranda programos projektavimo spragas, kurių negali aptikti standartizuoti įrankiai. Todėl privalu naudoti priemonių įvairovę, kad būtų galima aptikti daugumą pažeidžiamumų [3].

Apsauga nuo šių atakų turi būti numatyta bei korektiškai įgyvendinta interneto programos, paslaugos programuotojų. Dėl interneto programų skirtingumo ir kompleksiškumo jas apsaugoti yra sudėtinga, jos privalo korektiškai veikti su visomis galimomis įvestimis. Viena svarbiausių technikų kaip galima užtikrinti paslaugos apsaugą yra saugus programavimas. Tam programuotojas privalo žinoti apie galimas saugumo spragas ir kaip jų išvengti. Deja, interneto programos neturi jokio bendro standarto kurio laikantis būtų užtikrintas visiškas paslaugų saugumas. Tai privalo atlikti programuotojai, tačiau dažnai jie nėra gerai apmokyti ir nežino apie galimas rizikas, todėl palieka spragas kurias nesunkiai gali išnaudoti programišiai [4].

Tai ypač didelė problema, nes bendrai paplitusios programavimo klaidos randamos įvairiose programose. Tarp jų paplitę turinio valdymo sistemų karkasai, kurie sudaro didelę dalį viešai pasiekiamų tinklalapių. Pasak interneto technologijų tyrimų rezultatų 2017 metų kovą, turinio valdymo sistema „Wordpress“ sudaro daugiau kaip 27,8% visų internete pasiekiamų paslaugų [5]. Ši ar kita panašaus pobūdžio turinio valdymo sistema gali būti lengvai pritaikoma, turi didelį įskiepių pasirinkimą. Šios sistemos yra atvirosios, įskiepius gali kurti programuotojai nesusipažinę su galimomis saugumo spragomis, sistemos plėtiniai gali būti paliekami be priežiūros, todėl tokių sistemų priežiūra tampa kompleksiniu uždaviniu. Daug viešai pasiekiamų įskiepių yra nekokybiškai suprogramuoti, kartais turi net po keletą skirtingų pažeidžiamumų [6].

Viena esminių saugos problemų yra tai, kad įmonės neįvertina keliamos grėsmės, viešos sistemos prižiūrimos nenuosekliai, neatkreipiamas dėmesys į naujai atrastus pažeidžiamumus, ne laiku vykdomi programiniai atnaujinimai. Specialistai atsakingi už interneto sistemų priežiūrą susiduria su iššūkiu: dėl sistemų architektūros, finansinių suvaržymų ar programuotojų kompetencijos trūkumo nuo visiems žinomo pažeidžiamumo apsiginti nepavyksta. Vienas iš galimų šios problemos programos lygio ugniasienė.

Gintis prieš HTTP lygio atakas yra sudėtinga, įprastos tinklo apsaugai skirtos priemonės yra neefektyvios. Atakas sunku identifikuoti, atskirti nuo įprasto vartotojo elgsenos. Vienas iš sprendimo būdų yra tarp paslaugos naudotojo ir serverio įterpti programos lygio ugniasienę, kuri analizuotų įeinantį ir išeinantį srautą. Pasikliaudama šablonais ji tikrintų visų užklausų tikrumą ir blokuotų bandymus įsilaužti ar kitaip išnaudoti sistemą. Šiam tikslui yra didelis komercinių sprendimų pasirinkimas, kiekvienas iš jų turi skirtingas funkcijas, techninius reikalavimus. Vienos efektyviausių komercinių programos lygio ugniasienių sprendimų yra „F5“, „Barracuda“, „SecurSphere“, ir

„WebDefend“. Joms nenusileidžia ir OWASP fondo projektas „ModSecurity“ [4]. Programos lygio ugniasienes yra gana sudėtinga pritaikyti, tai stipriai priklauso nuo specifinio panaudojimo atvejo.

Interneto grėsmių analizės karkasu šiame darbe vadiname serverių infrastruktūrą, kuri geba analizuoti srautą ir gebėti apsiginti nuo atakų. Interneto grėsmių analizės karkaso sudarymui ir tyrimui atlikti pasirinkta Kauno technologijos universiteto tinklalapių saugojimo infrastruktūra.

Darbo tikslas sudaryti interneto grėsmių analizės karkasą, patikimą virtualių serverių infrastruktūrą, sudarytą iš skirtingų komponentų, atliekančių tinklalapių užklausų filtravimą, analizavimą, aptarnavimą. Sudarytas interneto grėsmių analizės karkasas, kuris padės aptikti saugos incidentus bei ištirtos jo ypatybės.

Siekiant tai užtikrinti reikia įgyvendinti šiuos uždavinius:

1. išanalizuoti tinklalapių prieglobos serverių architektūrą ir jai tinkamas saugumo priemones;
2. suprojektuoti interneto grėsmių analizės karkasą ir parinkti jo komponentus;
3. realizuoti grėsmių analizės karkasą, detalizuoti jo veikimą;
4. ištirti realizuotą grėsmių analizės karkasą;

Pagrindinis planuojamas darbo rezultatas – sukurtas efektyvus interneto grėsmių analizės karkasas, kuris aptarnauja naudotojų užklausas ir geba analizuoti srautą, apsiginti nuo atakų.

1. TINKLALAPIŲ SAUGUMO ANALIZĖ

Pagrindinis darbo uždavinys – sukurti efektyvų interneto grėsmių analizės karkasą ir taip apginti viešai prieinamus tinklalapius nuo kibernetinių įsilaužėlių, nuolatos skanuojančių internetines svetaines bei ieškančių sistemų spragų. Užsibrėžtas tikslas reikalauja peržiūrėti ir išanalizuoti tinklalapių laikymo infrastruktūrą. Tam pasirinkta Kauno technologijos universitete laikomų mokyklų svetainių infrastruktūra.

1.1. Darbo aktualumas

Kibernetiniai nusikaltėliai geba sukompromituoti sistemą nepriklausomai nuo jos buvimo vietos. Užtenka, kad sistema būtų su pažeidžiamumu. Dažnai, atradę pažeidžiamumus kibernetiniai nusikaltėliai interneto sistemose, skanuoja visą internetą ir ieško svetainių, kurios gali turėti šį pažeidžiamumą. Nors atnaujinimas, kuris užtaiso šią saugumo spragą, būna jau išleistas, sistemų administratoriai ne visada suspėja atnaujinti interneto sistemas, kartais sistemų atnaujinimas nėra galimas dėl techninių priežasčių ar tai padaryti yra tiesiog per brangu. Mano siūlomas sprendimas geba analizuoti tinklalapiams siunčiamų ir gaunamų užklausų srautą, geba apsaugoti sistemas su pažeidžiamumais neatnaujinant sistemos kodo, bet tinkamai sukonfigūruojant interneto grėsmių analizės karkasą. Tai yra ypač aktualu, nes interneto sistemos yra nuolatos atakuojamos, atrandamos naujos saugumo spragos, o sukompromituotos svetainės inicijuoja atakas, platina virusus, vykdo kitą žalingą veiklą [7].

1.2. Analizuojama architektūra

Dabartinė puslapių laikymo infrastruktūra yra sukurta atvirojo kodo pagrindu. Visuose paslaugas teikiančiuose serveriuose yra įrašyta atviroji operacinė sistema „Debian“. Sistemą sudaro keturi serveriai, teikiantys duomenų įkėlimo, puslapių apdorojimo ir turinio spartinimo paslaugas. Serverių resursai pavaizduoti lentelėje (2 lentelė). Šioje architektūroje rašymo metu saugomas 1041 tinklalapis.

2 lentelė. Serverių resursai

Operacinė sistema	Serverio paskirtis	Operatyvioji atmintis	Branduolių skaičius (3,07 GHz)	Vidinė talpa	Tinklinė talpa
Debian	Spartintuvas	4 GB	1	16 GB	-
Debian	Svetainių serveris	12 GB	4	59 GB	1,3 TB
Debian	Svetainių serveris	16 GB	4	21,5 GB	1,3 TB
Debian	Failų serveris	2 GB	4	12,6 GB	1,3 TB

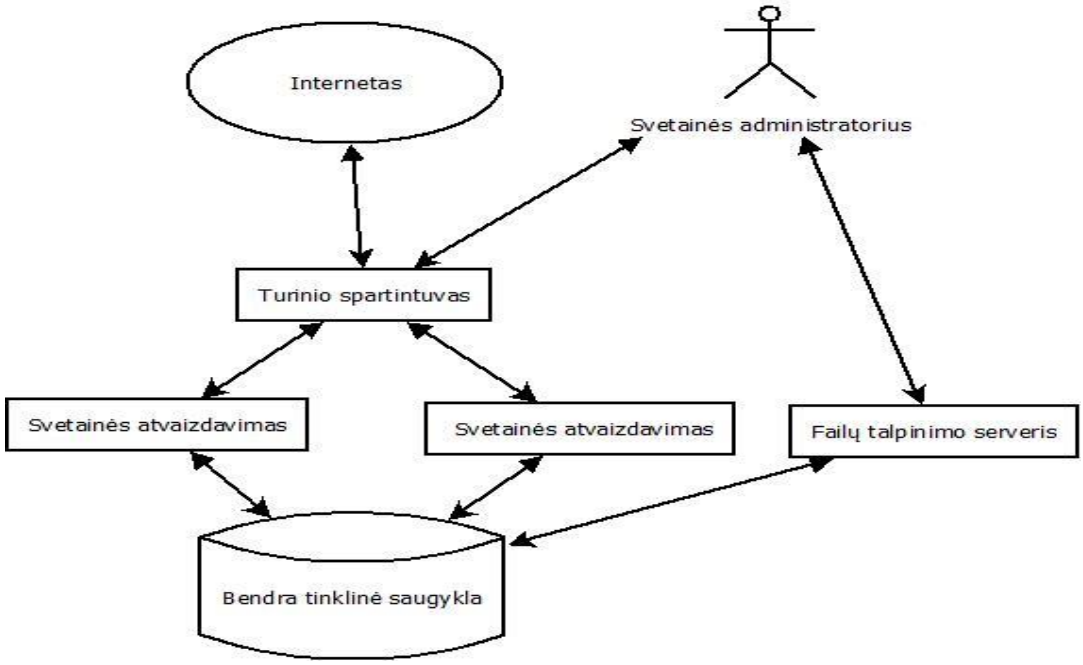
Spartintuvas – pirminis taškas, į kurį atkeliauja užklausos iš interneto. Pagal esamą taisyklių rinkinį patikrinamas užklausos korektiškumas, ar ji yra viena iš galimų kreipinių, kurie yra aprašyti interneto gairėse RFC7230, patikrinama ar užklausos siuntėjas nėra juodajame sąraše. Jeigu užklausa tenkina visus keliamus reikalavimus, išanalizuojama, kuriai svetainei ji buvo skirta, tuomet nusiunčiama į vieną iš svetaines aptarnaujančių serverių.

Dauguma pirmojo serverio svetainių yra pasenusio turinio, mažai prižiūrimos, veikia tik su sena, jau nebepalaikoma programinio paketo „php5“ versija. Antrasis svetainių serveris palaiko naujesnę „php5“ versiją, todėl dauguma naujesnių sistemų naudoja šį serverį. Failų serveris yra skirtas svetainių administratoriams. Per jį įkeliamas ir atnaujinamas svetainės turinys. Šių serverių išdėstymas pavaizduotas 1 pav.

Pasirinkta infrastruktūros apsisaugojimo nuo grėsmių architektūra privalo būti suderinama su naudojamomis atvirosiomis sistemomis, negali būti komercinė. Tiriamoje architektūroje naudojama turinio apdorojimo programa „apache2” [8]. Tai yra viena labiausiai paplitusių atvirųjų programų, skirtų saityno turinio generavimui. Vienas didžiausių konkurentų šiai programai yra taip pat atvira programa „nginx“. Lyginant programų greitaveiką su mažomis apkrovomis „apache2” šiek tiek pirmauja, tačiau serveriams apdorojant didžiulį užklausų skaičių žymiai spartesnis tampa „nginx“ [9]. Siekdami suderinamumo su esama sistema, turinio apdorojimo programos nekeisime.

Turinio spartintuvas naudoja programinį paketą „varnish“. Programinis paketas yra atviras, palaiko daugybę modulių bei suteikia galimybę susiprogramuoti reikiamą funkcionalumą. Vienas iš populiariausių bei reikiamą funkcionalumą turinčių ugniasienės modulių yra OWASP bendruomenės sukurtas įskiepis VFW (Varnish Firewall) [10]. Šis modulis palaiko programos lygio ugniasienę, suteikia galimybę kurti naujas saugumo taisykles bei algoritmus. Tai galėtų būti vienas iš sprendimo būdų, programos lygio ugniasienė pirminiame taške užtikrintų visos infrastruktūros apsaugą.

Failų įkėlimo serveris yra skirtas svetainių administratorių prisijungimui bei duomenų įkėlimui, atnaujinimui. Šis serveris yra sąlyginai saugus, vienintelis galimas pavojus yra prisijungimai su nutekėjusiais svetainių administratorių slaptažodžiais. Ši sritis nėra problematinė, didesnio saugumo užtikrinti nebūtina.



1 pav. Pradinis serverių išdėstymas

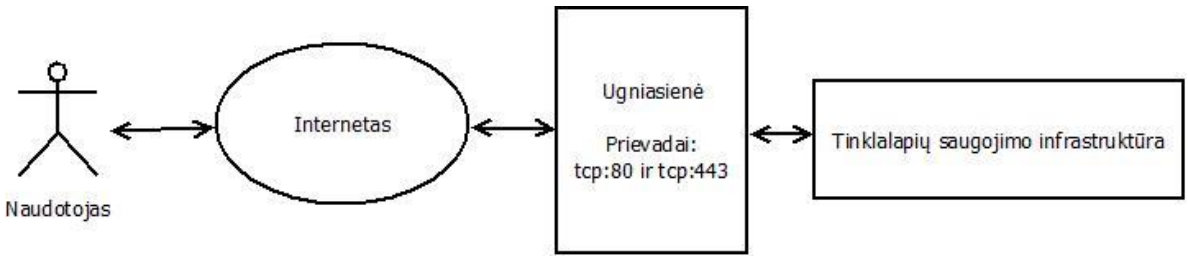
1.3. Analizuojamos architektūros saugos problematika

Laikui bėgant atakos netampa sudėtingesnės, tačiau atakuotojai labiau linkę ieškoti plačiai žinomų paprastų, paplitusių programavimo klaidų. [11]. Tai yra ypač aktualu populiarioms turinio valdymo sistemoms, kurios turi didelius įskiepių kiekius, nes kiekviename įskiepyje gali būti palikta paprasta klaida, kuri gali sukompromituoti visą svetainę.

Darbo rašymo metu tinklalapių laikymo paslauga šioje architektūroje naudojasi 1041 svetainė. Didžioji dauguma svetainių naudoja turinio valdymo sistemas „Joomla“ ir „Wordpress“. Šios sistemos yra atvirosios, labai plačiai naudojamos visame pasaulyje. Kadangi iš interneto serveriai yra pasiekiami tik šiais dviem prievadais „TCP 80“ ir „TCP 443“ tik šie prievadai gali būti naudojami atakoms vykdyti. Pagal HTTP ir HTTPS protokolų veikimą iš tinklo pusės labai sudėtinga atskirti blogą naudotojų srautą nuo gero (2 pav.).

Už šių svetainių programinę priežiūrą yra atsakingi svetainių administratoriai, kurie ne visada yra pakankamai susipažinę su galimomis grėsmėmis ir geriausiomis praktikomis kaip reikėtų užtikrinti tinklalapių saugumą. Atsiradus naujam pažeidžiamumui vienoje iš populiarių turinio valdymo sistemų, jas visas privalu kuo greičiau atnaujinti, tačiau, turint omenyje tokius didelius svetainių kiekius, dalis svetainių lieka neatnaujinta. Net naudodami kokybiškus, gerai prižiūrimus įskiepius administratoriai laiku jų neatnaujindami rizikuoja, kad į jų svetainę bus įsilaužta per jau užtaisytą pažeidžiamumą. Vienas iš sprendimo būdų yra pritaikyti programos lygio ugniasienę, kuri gintų visas svetaines nuo naujai atsiradusio pažeidžiamumo. Tuomet šios svetainės turinio valdymo sistema neprivalo būti

naujausios versijos su ištaisytu pažeidžiamumu, apsaugą nuo šio pažeidžiamumo atliktų programos lygio ugniasienė.



2 pav. Atakos paviršius

1.4. Galimos sistemos komponentų apsaugojimo priemonės

Yra didelis pasirinkimas saugumo sprendimų, kurie gali užtikrinti sistemų saugumą. Tai gali būti ugniasienės, įsilaužimo aptikimo sistemos (IDS), medaus puodynės, programos lygio ugniasienės.

1.4.1. Ugniasienės

Ugniasienė tai techninės ir programinės įrangos derinys, kuris izoliuoja privatų tinklą ar programas praleisdamas reikiamus paketus ir blokuojantis kitus. Yra trys pagrindiniai ugniasienių tipai:

- paketus filtruojantis maršruto parinktuvas;
- programos lygio šliuzas;
- grandinės lygio šliuzas;

Paketus filtruojantis maršruto parinktuvas, kuris pagal įeinančių ir išeinančių IP paketų šaltinius perskirsto arba išmeta paketus. Filtravimo taisyklės remiasi informacija esančia tinklo pakete, kaip siuntėjo IP adresas, paskirties vietos IP adresas, tinklo sąsaja ir kt.

Programos lygio šliuzas (angl. Application level gateway), arba kitaip žinomas kaip įgaliotasis serveris (angl. proxy server). Sistemos naudotojai komunikuoja su tokio tipo ugniasiene ir jeigu naudotojui nepavyksta autentifikuotis arba nepavyksta patenkinti ugniasienės saugumo taisyklių, tokia užklausa tiesiog atmetama ir nepasiekia galutinės programos. Pagrindinis jos minusas yra laikas, kurį užima sujungimo užmezgimas ir patikrinimas prieš perduodant užklausą programai.

Grandinės lygio šliuzas (angl. Circuit level gateway), kuris gali būti kaip atskiras įrenginys arba specializuota programos funkcija. Tokio tipo ugniasienė neleidžia naudotojui komunikuoti su programa tiesiai, bet įsiterpia tarp jų. Vienas sujungimas užmezgamas su šliuzu, tuo tarpu ugniasienė

užmezga sujungimą su programa ir perduotą reikiamą informaciją. Saugumą užtikrina tai, kad ugniasienė nusprendžia, kurie sujungimai bus praleisti, o kurie blokuojami [12].

Gamintojai siūlo specializuotą ugniasienės techninę įrangą, kurios kaina dažnai būna didelė, tačiau siūlomas labai didelis našumas, gebėjimas apsiginti nuo DOS tipo atakų. Dažnai tokios ugniasienės turi techninius apribojimus, kaip maksimali galima procesoriaus apkrova, galimas operatyvinės atminties kiekis. Dėl šių priežasčių virtuali ugniasienė, paskirstyta virtualioje infrastruktūroje tarp daugelio serverių, gali būti pranašesnė. Pagal Ibrahim Waziri (*et al*. 2015) atliktą tyrimą, ugniasienės techninės įrangos veikimas yra geresnis esant normaliomis sąlygomis, tačiau vykstant atakoms debesijos ugniasienės sprendimai yra patikimesni, nes nėra apriboti procesoriaus ir atmintinės resursų [13].

Ugniasienių privalumai:

- geba atmesti neleistiną srautą;
- geba apsaugoti pažeidžiamus protokolus ir paslaugas;
- geba paslėpti paslaugų vardus ir IP adresus nuo išorinių tinklų;
- geba detaliai įrašinėti srauto įvykius serverių žurnalą;

Ugniasienių trūkumai:

- naudojami taisyklių rinkiniai, kurie privalo būti sukonfigūruoti rankiniu būdu, kad būtų galima atskirti teisėtą srautą nuo neteisėto;
- negali reaguoti į tinklo atakas arba imtis priemonių nuo jų apsisaugoti;
- dauguma ugniasienių neanalizuoja srauto turinio;
- negalima apsiginti nuo programos lygio atakų;

1.4.2. Įsilaužimo aptikimo sistemos

Įsilaužimo aptikimo sistemos (angl. Intrusion Detection Systems) padeda informacinėms sistemoms susidoroti su atakomis [14]. Tai pasiekiama surenkant informaciją iš įvairių tinklo sistemų. Surinkta informacija yra analizuojama dėl galimų saugumo problemų. Šios atakos gali būti vykdomos tiek iš organizacijos vidaus arba iš išorinio tinklo.

Įsilaužimo aptikimo sistemų privalumai:

- lengviau įdiegti, nes nepaveikia esamos infrastruktūros;
- analizuojant tinklo paketo turinį tinklo jutikliai geba aptikti atakas vykdomas atakas, kaip TCP SYN, išardytų paketų ir kita;

- geba stebėti tinklo srautą realiu laiku, atpažinti žalingą veiklą jai įvykstant;

Įsilaužimo aptikimo sistemų trūkumai:

- tai nėra sprendimas nuo visų saugumo problemų;
- vykstančią ataką atpažinus ir pranešus, žmogaus įsikišimas privalomas, kad ji būtų ištirta;
- dažni klaidingai teigiami rezultatai, kuomet IDS atpažįsta įprastą naudotojų srautą kaip žalingą;
- dažni klaidingai neigiami rezultatai, kuomet IDS neatpažįsta vykdomos žalingos veiklos;

1.4.3. Medaus puodynės

Tai spąstai, kurių tikslas yra atpažinti, atremti bandymus neteisėtai pasinaudoti programomis ar informacinėmis sistemomis. Dažniausiai tai yra kompiuteris, duomenys arba tinklo vieta, kuri atrodo kaip infrastruktūros dalis ir galimai turi reikalingą informaciją atakuotojui bet iš tikrųjų yra izoliuota ir stebima. Medaus puodynės veikimo principas yra apsimesti tikra sistema, kurią bandytų išnaudoti atakuotojai, nežinodami, kad jie yra stebimi. Kuomet bandoma sukompromituoti medaus puodynę, visa informacija susijusi su ataka yra surenkama. Atlikta veikla pateikia reikšmingą informaciją, kurią išanalizavęs sistemos administratorius gali atsekti atakos šaltinį ir kilmę [15]. Medaus puodynės yra dalijamos į dvi kategorijas.

Produkcinės medaus puodynės (angl. Production Honeypot) yra naudojamos organizacijos lokalaus tinklo infrastruktūros gynybai, sugauna programišius, kurie bando įsilaužti į infrastruktūrą. Šių puodynių įdiegimas yra daug paprastesnis, nei tyrimų medaus puodynių, nes jų tikslas daug paprastesnis ir reikalauja mažiau funkcijų. Todėl jos pateikia mažiau informacijos apie įsilaužėlio veiksmus ir motyvus.

Tyrimų medaus puodynės (angl. Research Honeypots), šios puodynės yra sudėtingos, jos yra suprojektuotos surinkti kuo daugiau informacijos apie įsilaužėlį ir jo veiklą. Jų pagrindinis tikslas yra tirti galimas organizacijos grėsmes. Surinkta tyrimų medaus puodynių informacija padeda organizacijai geriau suprasti įsilaužėlių motyvus ir veikseną, lengviau gintis nuo atakų.

Medaus puodynių privalumai:

- mažas duomenų rinkinys, nes renkami tik tie duomenys, kuomet įsilaužėlis bendrauja su medaus puodyne;
- nedaug klaidingai teigiamų reikšmių;
- gebama aptikti nežinomas atakas, nes bet kokia veikla su medaus puodyne yra anomalija;
- nėra svarbu, ar ataka yra šifruojama, ar ne;
- medaus puodynės yra labai lanksčios;

- reikia nedaug resursų;

Medaus puodynių trūkumai:

- rizika, kad medaus puodynė bus sukompromituota ir atakuos kitas sistemas;
- ribotas atakų matomumo laukas, pastebimos tik atakos, kurios komunikuoja su puodyne;
- patyrę programišiai gali atpažinti medaus puodynę ir daugiau jos nepulti arba pasinaudoti ja, kad nukreiptų dėmesį nuo kitų vykdomų atakų;

1.5. Programos lygio ugniasienės

Programos lygio ugniasienė – tai vienas apsaugos taškas tarp galutinio vartotojo ir interneto programos. Pagrindinis ugniasienės tikslas yra apsaugoti programas nuo atakų, bandymų įsilaužti ir neteisėto prisijungimo. Programos lygio ugniasienės tikslas apginti žiniatinklio programą nekeičiant programos struktūros. Daugeliu atveju programos, kurias gina programos lygio ugniasienė, yra naudojamos produkcijoje ir nėra įmanoma pataisyti problemos arba problemos taisymas reikalauja programą išjungti, tai užima daug laiko ir daug kainuoja. Įprastos ugniasienės, tokios kaip įsilaužimo prevencijos sistema (IPS) arba įsilaužimo aptikimo sistema (IDS), kurios nuolatos stebi tinklą bei atpažįstą galimą incidentą ir jį pažymi žurnalų įrašuose, negali tinkamai apginti saityno programų, nes neturi informacijos apie 7 OSI modelio programos sluoksnį [16]. Tai atskleidžia pateiktas ugniasienių palyginimas saugumo aspektu, pateikta 3 lentelėje.

Kuriant programinę įrangą net atlikus daug tokių skirtingų testų, kaip bandymas įsilaužti ar kodo peržiūra, dalis problemų ir saugos trūkumų lieka nepastebėti – tai potenciali grėsmė. Šiais atvejais programos lygio ugniasienės naudojimas gali apsaugoti programą nuo neatrastų saugumo pažeidžiamumų. Be to, programos lygio ugniasienės naudojimas yra privalomas laikantis tokių aukščiausių finansinių mokėjimų internetu standartų, kaip mokėjimo kortelių pramonės duomenų saugumo standartas (PCI DSS).

3 lentelė. IDS/IPS sistemų ir programos lygio ugniasienės saugumo galimybių palyginimas [16]

Apsauga	IPS/IDS ugniasienė	Programos lygio ugniasienė
Kodo įterpimo atakų apsauga (XSS, SQL)	Ribotas	Taip
CSRF apsauga	Ne	Taip
Normalizuoti šifruotą srautą	Ne	Taip
Peržiūrėti HTTPS srauto turinį	Ne	Taip
Sesijos klastojimo/perėmimo apsauga	Ne	Taip
Naršymo per jėgą apsauga	Ne	Taip
Duomenų vagystės apsauga	Ne	Taip
Grubios jėgos atakos apsauga	Ne	Taip
Paslaugų projekcijos apsauga	Ne	Taip
Virusų, kenkėjiškų programų įkėlimo apsauga	Taip	Taip
Programos lygio atsisakymo aptarnauti ataka	Ne	Taip
Dažnio kontrolės apsauga	Ne	Taip
Užklausos, atsakymo perrašymas	Ne	Taip
Programos prieigos ir vartotojų audito žurnalų įrašai	Ne	Taip

1.6. Programos lygio ugniasienių palyginimas

Rinkoje šiuo metu yra labai gausus programos lygio ugniasienių pasirinkimas. Daugelis gamintojų reklamuoja savo produkciją, kaip geriausią, vienintelį sprendimą, kuris atlieka visas reikiamas funkcijas. Programos lygio ugniasienių palyginimas aprašytas 4 lentelė.

4 lentelė Programos lygio ugniasienių palyginimas [17]

Pavadinimas	Funkcijos
Applicure DotDefender	• Apsauga nuo atsisakymo aptarnauti atakos • Apsauga nuo sukryžminto svetainių kodavimo atakos (XSS) • Apsauga nuo SQL įterpimo atakos • Apsauga nuo kelio pakeitimo atakos
Armorlogic Profense	• Apsauga nuo perteklinio duomenų atskleidimo • Tikrinamas HTTP antraščių atitikimas • Tikrinamas sesijų tikrumas • Apsauga nuo sukryžmintų svetainių užklausų padirbimo (CSRF) • Apsauga nuo SQL įterpimo atakos

	• Apsauga nuo atsisakymo aptarnauti atakos (DOS)
Imperva SecureSphere	• Apsauga nuo sukryžminto svetainių kodavimo atakos (XSS) • Apsauga nuo sukryžmintų svetainių užklausų padirbimo (CSRF) • Apsauga nuo SQL įterpimo atakos • Apsauga nuo OWASP dešimt populiariausių atakų
FortiWeb	• Apsauga nuo SQL įterpimo atakos • Apsauga nuo XML schemos atakų • Apsauga nuo sukryžmintų svetainių užklausų padirbimo (CSRF) • Apsauga nuo sukryžminto svetainių kodavimo atakos (XSS) • Apsauga nuo perteklinio duomenų atskleidimo
Barracuda	• Apsauga nuo SQL įterpimo atakos • Apsauga nuo sukryžminto svetainių kodavimo atakos (XSS) • Sausainių apsauga nuo formų klastojimo • Apsauga nuo slaptažodžių spėliojimo
SonicWall	• Apsauga nuo SQL įterpimo atakos • Apsauga nuo sukryžminto svetainių kodavimo atakos (XSS) • Sausainių apsauga nuo formų klastojimo • Apsauga nuo slaptažodžių spėliojimo
Citrix	• Apsauga nuo buferio perpildymo atakų • Apsauga nuo sausainių atakų • Apsauga nuo naršymo per jėgą atakų
ModSecurity	• Apsauga nuo atsisakymo aptarnauti atakos • Apsauga nuo sukryžminto svetainių kodavimo atakos (XSS) • Apsauga nuo SQL įterpimo atakos • Apsauga nuo kelio pakeitimo atakos • Apsauga nuo perteklinio duomenų atskleidimo • Tikrinamas HTTP antraščių atitikimas • Tikrinamas sesijų tikrumas • Apsauga nuo sukryžmintų svetainių užklausų padirbimo (CSRF) • Apsauga nuo XML schemos atakų • Apsauga nuo sukryžminto svetainių kodavimo atakos (XSS) • Sausainių apsauga nuo formų klastojimo • Apsauga nuo buferio perpildymo atakų

Plačiai žinomi, kokybiškiausi komerciniai sprendimai kainuoja labai daug, dažnai perkami didžiulių įmonių tik tam, kad patenkintų keliamus saugumo reikalavimus. Gausu straipsnių su palyginimais tarp įvairių programos lygio ugniasienių. Palyginimai atliekami pagal įvairius kriterijus kaip atsako laikas, efektyvumas, užblokuotas simuliuojamų atakų skaičius ir kt. Tačiau prieinama bendra išvada, kad neegzistuoja vienas įrankis, kuris gali atlikti visas funkcijas ir apginti nuo visų įmanomų pažeidžiamumų. Todėl vienas pagrindinių kriterijų renkantis ugniasienę yra galimybė ją kuo paprasčiau plėsti, lengvai pritaikyti prie iškylančių poreikių, gintis nuo naujai atsiradusių pažeidžiamumų.

Dėl didelio papildinių kiekio, bendruomenės palaikymo, galimybių gausos ir dėl to, kad šis produktas yra visiškai nemokamas pasirinkta naudoti atviro kodo produktą, vieną iš populiariausių, nemokamų programos lygio ugniasienių OWASP (angl. The Open Web Application Security Project) projektą „ModSecurity“. Ši programos lygio ugniasienė nenusileidžia komerciniams sprendimams, yra stipriai palaikoma atviro kodo bendruomenės, todėl prireikus galima gauti techninę pagalbą. Ugniasienė yra pakankamai lanksti, ją galima nesunkiai pritaikyti atsiradus naujoms grėsmėms, užblokuoti piktybinį srautą [18]. Ši ugniasienė yra tinkama, kuomet naudojamas komponentas skirtas svetainių turinio generavimui „apache2“, tačiau tai nėra privaloma, ugniasienę galima iškelti į tam paskirtus serverius, kurie skirti tik taisyklių analizei arba yra galimybė šį modulį įdiegti į kitus turinio generavimo serverius kaip „nginx“, tačiau tai nėra numatytoji konfigūracija.

1.7. Siūlomas metodas

Kadangi nė vienas tinklalapių apsaugos būdas negali apsaugoti nuo visų galimų atakų aibės, siūlomas metodas naudoti kompleksinį sprendimą. Jis susidės iš grandinės lygio šliuzo tipo ugniasienės ir programos lygio ugniasienės, kuri taip pat atliktų įsilaužimo aptikimo sistemos funkciją. Tinklalapių talpinimo infrastruktūroje realizuoti tarpinius serverius, kurie vykdytų ugniasienė funkciją ir programos lygio ugniasienę, kuri analizuotų svetainių srautą ir pranešimus apie galimus įsilaužimus įrašytų į žurnalų įrašus, kas atliktų įsilaužimo aptikimo sistemos funkciją. Medaus puodynės sprendimo pasirinkta nenaudoti, nes tai nėra tiesiogiai susiję su tinklalapių apsauga, bet labiau skirta atlikti atakuotojų veiksmų tyrimui ir analizei, kad pagal tai būtų galima adaptuoti esamą tinklalapių apsaugą.

Šį metodą gali taikyti įmonės teikiančios svetainių prieglobos paslaugas ir norinčios apsaugoti klientų tinklalapius, įmonės siekiančios, kad jų svetainė veiktų sparčiai ir būtų užtikrinta svetainės apsauga nuo įsilaužimo. Taip pat dėl architektūros, kuri susideda iš daugelio komponentų, siūlomas modelis užtikrina paslaugos resursų plėtimo galimybes, pritrūkus procesoriaus, sparčiosios atmintinės ar disko resursų į architektūrą galima įtraukti papildomus serverius. Tai yra paranku bet kokiai įmonei, kurios svetainės lankomumas yra didelis.

1.8. Analizės išvados

Atlikta Kauno technologijos universiteto serverių, kuriuose talpinamos Lietuvos mokyklų svetainės, architektūros analizė. Įvertinus teikiamos tinklalapių laikymo paslaugos serverių architektūrą, nustatyti esminiai taškai, kuriuose galima atlikti grėsmių analizę. Pagrindinis siekis – pasinaudojus atvira programine įranga sukurti centralizuotą saugumo modelį, kuris gebėtų apsiginti nuo galimų grėsmių. Nė vienas sprendimas neapsaugo nuo visų galimų atakų, geriausias būdas užtikrinti tinklalapių saugojimo infrastruktūros apsauga yra naudoti kompleksinę apsaugą naudojant kelias saugumo priemones. Visapusiškai apsaugai tikslingiausia naudoti ugniasienę, programos lygio ugniasienę. Šio darbo aktualumas yra interneto grėsmių analizės karkaso sudarymas, kuris yra aktualus svetainių įmonėms užsiimančioms svetainių priegloba. Siūlomas sprendimas yra skirtas Kauno technologijos universiteto svetainių saugojimo architektūrai, tačiau nesunkiai gali būti pritaikytas bet kokiai tinklalapių prieglobos sistemai.

2. TINKLALAPIŲ GRĖSMIŲ ANALIZĖS KARKASO PROJEKTAVIMAS

Pagal atliktos tinklalapių saugumo analizės rezultatus, aprašomas saugus tinklalapių laikymo sistemos architektūros modelis, kuris išpildo visus keliamus saugumo bei patikimumo reikalavimus.

2.1. Sistemos architektūra

Šia architektūra naudojasi didelis kiekis svetainių, serverių patiriama apkrova gali kisti, saugomų tinklalapių skaičius didėti, todėl ypač privalu keičiantis sąlygoms išlaikyti minimalų užklausų apdorojimo laiką, t.y. architektūroje nesukurti lėtų grandžių, bei išlaikyti aukštą patikimumą, kad vieno serverio klaidos atveju tinklalapiai netaptų nepasiekiami.

Visoje architektūroje dalyvaujantys serveriai yra virtualios mašinos sukurtos universiteto virtualizacijos infrastruktūroje. Tam naudojami hipervizoriai su „vmware“ virtualizacijos sprendimu. Didėjant tinklalapių kiekiui, juos visus būtų galima kelti į vieną serverį, tačiau dėl virtualizacijos ypatybių tai daryti nerekomenduojama. Pagal gamintojų nurodomas technines specifikacijas siekiant užtikrinti geresnį hipervizorių procesoriaus resursų paskirstymą kiekviena virtuali mašina privalo turėti kiek įmanoma mažiau procesoriaus branduolių [19]. Prireikus aptarnauti daugiau tinklalapių reikės daugiau procesoriaus resursų, todėl norint efektyviai išnaudoti hipervizorių reikia numatyti plėtros galimybes ir visas roles užklausų aptarnavimo grandinėje padalinti į kelis serverius. Taip bus tolygiai išnaudotas hipervizorius, bendra architektūra galės naudoti neribotą procesoriaus branduolių skaičių su sąlyga, kad viena virtuali mašina nenaudos daugiau negu keturių branduolių.

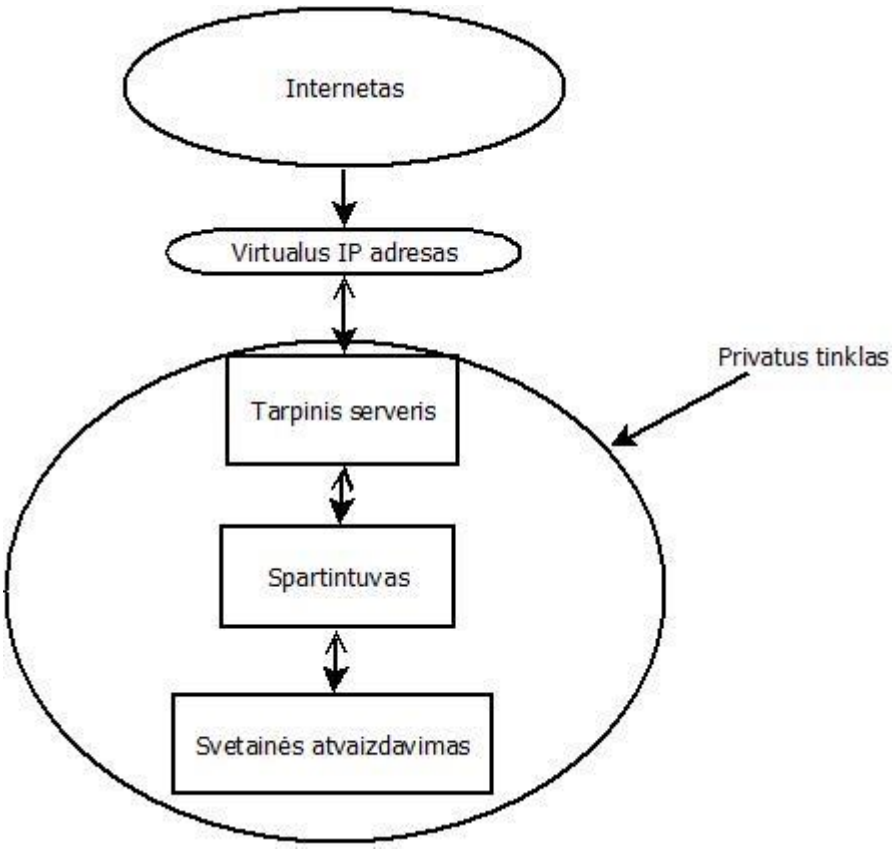
2.2. Sistemos sauga

Sistemos sauga susideda iš keturių skirtingų komponentų grandinės, kuri pavaizduota 3 pav.. Visos sistemos saugumas priklauso nuo visų komponentų tarpusavio sąveikos. Kiekvienas komponentas apdoroja užklausas, taiko iš anksto numatytas saugumo taisykles. Šioje užklausų apdorojimo sekoje kiekvieną iš komponentų galima padauginti iki kelių. Tai užtikrintų paslaugos pasiekiamumą bei palengvinto infrastruktūros plėtimo galimybes padidėjus laikomų tinklalapių skaičiui [20].

3 pav. Komponentų sąveika

2.3. Serverių sąveika

Vienas iš veiksnių didinančių teikiamų paslaugų saugumą bei patikimumą yra izoliacija. Didinant patikimumą privalu izoliuoti serverius, paslaugas, procesus, vartotojus, tinklus. Tai užkerta kelią arba sumažina tikimybę bet kokios programinės įrangos veikimo arba sistemos konfigūravimo klaidoms. Siekiant sumažinti galimybę pakenkti sistemai iš išorės, visą sistemos komponentų bendravimą reikia atlikti vidiniuose tinkluose. Numatoma architektūra iš interneto priima užklausas per vieną virtualų IP adresą. Šis adresas yra vienintelis sistemos pasiekimas iš išorės. Visos tolimesnės komunikacijos vykdomos privačiame virtualiame tinkle. Tik virtualus IP adresas priklauso viešai pasiekiamam tinklui, visos vidinės komunikacijos vyksta privačiais IP adresais, kurie nėra išleidžiami į internetą. Tai atvaizduojama 4 pav. Užklausos į serverius pirmiausiai keliauja į virtualų IP adresą. Šį virtualų IP adresą aptarnauja du tarpiniai serveriai aktyviu ir pasyviu režimu. Tai vykdo programinis paketas „keepalived“, kuris veikia „Cisco“ VRRP protokolu ir yra aprašytas RFC5798 [21]. Šis protokolas užtikrina komunikaciją tarp dviejų tarpinių serverių siunčiant „broadcast“ arba „unicast“ tipo paketus. Jeigu vienas iš serverių negauna atsako iš kito, padaro išvadą, kad kitas serveris nepasiekiamas ir prireikus perima virtualų IP adresą.



4 pav. Tinklo izoliavimas

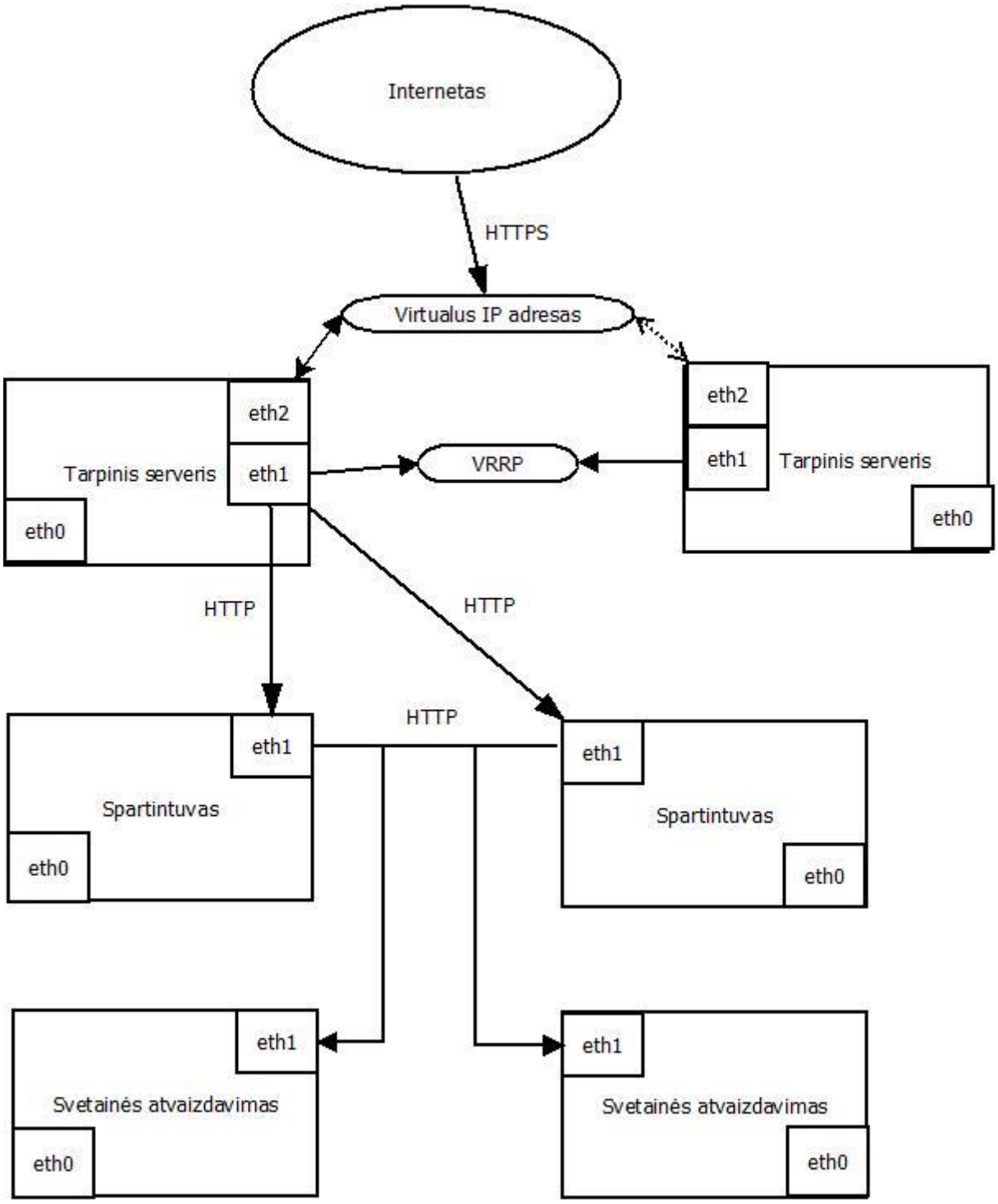
2.3.1. Tarpinis serveris

Tai sistemos pradinis taškas, kuriame vyksta pirminis užklausų apdorojimas. Šią rolę atlieka du tarpiniai serveriai (angl. reverse proxy), veikiantys kaip aktyvus-pasyvus klasteris. Numatytas veikimo principas – sukonfigūruotos trys skirtingos tinklo sąsajos: „eth0“, „eth1“ ir „eth2“. Pirmoji tinklo sąsaja „eth0“ yra skirta serverio valdymui, administratoriaus prisijungimui. Antroji „eth1“ yra privačiame tinkle, o trečioji „eth2“ viešai pasiekiamame tinkle. Aktyvus serveris numatytu intervalu siunčia užklausas pasyviajam. Jeigu per numatytą laiko tarpą antrinis serveris negauna transliavimo individualiais adresais (angl. unicast) tipo pranešimo, priima sprendimą, kad pirmasis serveris nepasiekiamas ir perima virtualaus IP adreso aptarnavimą.

Šie serveriai taip pat atlieka viso HTTP srauto šifravimą, pavertimą HTTPS protokolu – SSL nukrovimą (angl. offload) remiantis SNI protokolu aprašytu RFC3546 [22]. Pagal šį protokolą svetainės, į kurią kreipiamasi, pavadinimas nėra užšifruojamas. Užklausos yra išanalizuojamos bei pagal svetainės pavadinimą parenkamas reikiamas sertifikatas. Visas srautas iki tarpinio serverio yra šifruojamas, o privačiame tinkle keliauja atviru HTTP protokolu. Taip apsaugoma vartotojų informacija, kuri keliauja internetu, o nešifruotą srautą galima analizuoti ir saugoti podėlyje.

Tuomet tarpinis serveris atlieka pradinę užklausų analizę, paskirsto užklausas pagal iš anksto aprašytas taisykles, patikrina užklausos tipą, kryptį, užtikrina visų tinklalapių laikymo serverių apsaugą, apsaugodamas nuo įvairių atakų, kaip slaptažodžių spėliojimo, nekorektiškų kreipinių, DOS tipo atakų.

Visos užklausos ateinančios per viešą tinklą yra nukreipiamos į jas aptarnaujančius serverius privačiais IP adresais. Tarpinis serveris turi virtualų vidinį IP adresą, kuris kitiems serveriams yra nurodytas kaip numatytasis tinklų sietuvas (angl. default gateway). Tarpinio serverio veikimo principas pavaizduotas 5 pav.



5 pav. Sistemos tinklo architektūra

2.3.2. Spartintuvas

HTTP greitintuvas skirtas mažinti tinklalapių atsako laiką, jis žymi tinklalapių turinį, jį išsaugo bei laiko atmintyje. Jeigu šio turinio prireiks netolimoje ateityje, užklausa nebus perduota turinį generuojančiam serveriui, bet paimta iš podėlio. Spartintuvas pagal HTTP protokolo antraštes, aprašytas RFC7230 [1], deda turinį į podėlį numatytam laiko tarpui. Tai yra taikoma statiniam turiniui, kuris nesikeičia: nuotraukoms, paveikslėliams ir kita. Jeigu šis turinys bus užklaustas dar kartą, jo galiojimo laikui dar nepasibaigus, nebus kreipiamasi į galutinį serverį, o atsakyta iš podėlio. Kadangi turinio padėjimas į podėlį priklauso nuo tinklalapio konfigūracijos, galutiniame užklausas aptarnaujančiame serveryje reikėtų įjungti modulį „mod_expires“, kuris reguliuotų bei uždėtų turiniui, kurį reikia padėti į podėlį, atitinkamas antraštes. Be to, HTTP greitintuvas pagal aprašytas taisykles balansuoja tarp turinį generuojančių serverių.

Spartintuvas yra antroji saugumo grandis, kuri patikrina užklausų korektiškumą, atmeta užklausas, kurios nesilaiko RFC protokolo, blokuoja bandymus įsilaužti į sistemas pagal žinomus pažeidžiamumus. Naudojamas vartotojo agento filtras, kuris užblokuoja žinomus blogus vartotojo agentus. Į šį spartintuvą galima įkompiliuoti modulius parašytus C kalba.

Spartintuvas aptarnauja užklausas HTTP protokolu privačiais IP adresais. Užklausas inicijuoja tarpiniai serveriai, o spartintuvas užklausas aptarnauja, nukreipia į turinį generuojančius serverius arba paima informaciją iš podėlio ir jas grąžina per privačią tinklo sąsają tarpiniam serveriui. Tai pavaizduota paveikslėlyje (**Error! Reference source not found.**).

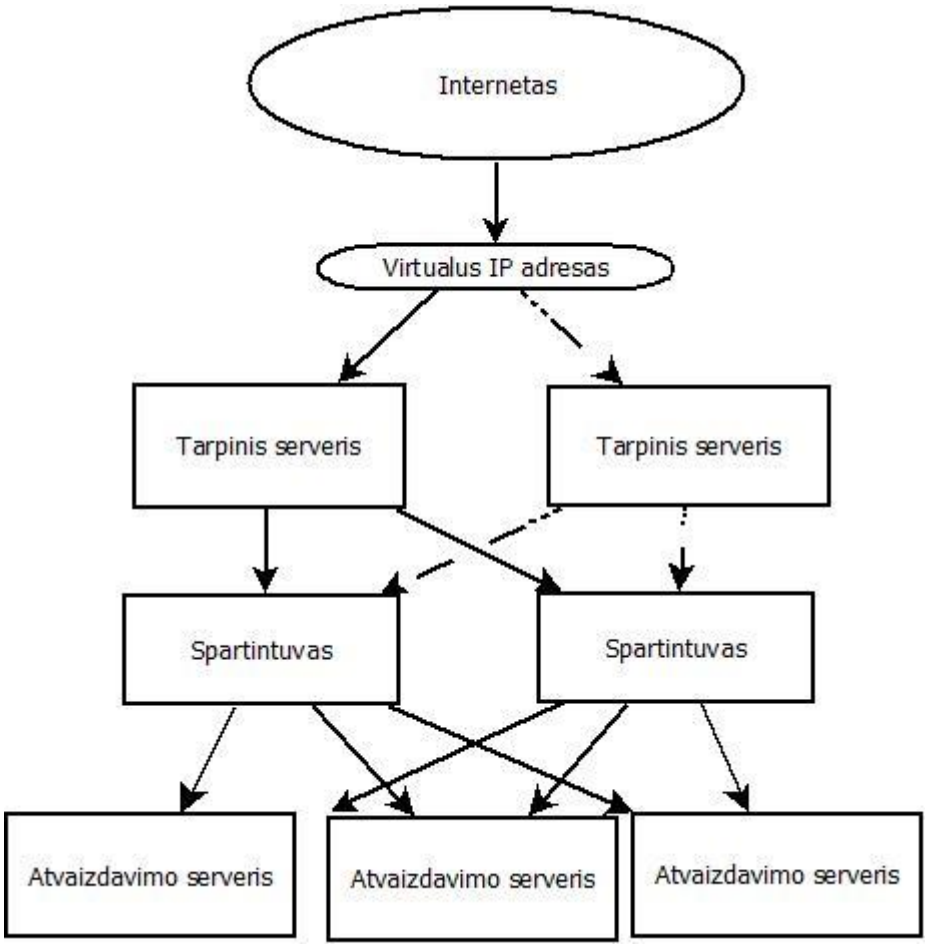
Vienas pagrindinių uždavinių, kuriuos reikia įvykdyti – sukurti ir pritaikyti taisyklių rinkinį, skirtą tinklalapiams. Kadangi svetainių kiekis yra didelis, tai įgyvendinti yra sudėtinga [23].

2.3.3. Tinklalapių serveris

Tai galutinis sistemos taškas aptarnaujantis visas užklausas, kuris generuoja turinį ir jį perduoda užklausos siuntėjui. Šio serverio duomenys privalo būti apsaugoti, kiekvienos svetainės procesas izoliuotas, o svetainių vartotojų teisės apribotos, kad galėtų pasiekti tik failus, priklausančius jų tinklalapiui.

Šioje architektūroje tinklalapių serverių veikimo korektiškumui užtikrinti programa turi matyti originalią užklausą su siuntėjo IP adresu. Tam bus naudojama mūsų sukurta HTTP antraštė „X-REAL-IP“, kuria iš tarpinio serverio bus perduotas realus siuntėjo IP adresas. Tam, kad šią antraštę suprastų tinklalapių serveris, naudojamas „mod_rpaf“ komponentas, kuris vadovaujasi per antraštes perduotu originalaus siuntėjo IP adresu. Kad būtų užtikrinta procesų izoliacija, kiekvieną svetainę aptarnaujantis procesas startuoja tos svetainės vartotojo vardu. Šiam tikslui naudojamas modulis „mod_ruid2“.

Šis serveris aptarnauja visas naudotojų užklausas, klientams perduoda statinį turinį arba sugeneruoja svetainės kodą ir jį grąžina siuntėjui. Suprojektuotoje architektūroje programos lygio ugniasienė įterpiama kaip tinklalapio serverio modulis. Šis modulis įsiterpia prieš serveriui apdorojant užklausą ir iš karto prieš užklausą grąžinant siuntėjui. Bendras sistemos vaizdas pavaizduotas 6 pav.



6 pav. Projektuojama sistemos architektūra

2.3.4. Programos lygio ugniasienė

Programos lygio ugniasienės taikymui pasitelktas OWASP projekto modulis „mod_security“ [24] ir „modsecurity-crs“ taisyklių rinkinys. Dėl didelio užklausų kiekio, kurį reikia išanalizuoti, šiam tikslui reikės daug procesoriaus resursų. Dėl serverių virtualizacijos hipervizoriaus ypatumų daugiau procesoriaus branduolių į virtualų serverį pridėti nereikėtų. Todėl šią programos lygio ugniasienės rolę reikia iškelti į naujai sukurtą serverį. Siekiant užtikrinti didelį visos užklausų apdorojimo grandinės patikimumą, reikia sukurti du atskirus virtualius serverius, kurie bus skirti įeinančių ir išeinančių užklausų analizei pagal nustatytą taisyklių rinkinį. Ugniasienės veikimo principą sudaro penkios fazės, kiekvienoje fazėje analizuojamos jai priskirtos taisyklės.

Ugniasienės veikimo principo fazės:

1. užklausos antraštės (angl. Request headers) analizė;
2. užklausos turinio (angl. Request body) analizė;
3. atsakymo antraščių (angl. Response headers) analizė;
4. atsakymo turinio (angl. Response body) analizė;
5. žurnalų pildymas (angl. Logging);

Tiriama gaunamos užklausos antraštė bei vykdomi veiksmai atsižvelgiant į jos tipą. Pavyzdžiui, XML (angl. eXtensible Markup Language) turinys nėra analizuojamas. Jeigu užklausa sėkmingai patikrinama, ji toliau nusiunčiama į tinklalapių saugojimo serverį. Gavus atsakymą užklausa dar kartą apdorojama, tiriamos grįžtančios atsakymo antraštės ir turinys. Neradus pažeidimų, užklausa grąžinama siuntėjui. Jeigu nors kurioje fazėje randamas saugumo pažeidimas, užklausa yra įrašoma į žurnalų įrašus ir siuntėjui pranešama apie klaidą.

2.4. Interneto grėsmių analizės karkaso architektūra

Suprojektuotas interneto grėsmių analizės karkasas susideda iš:

1. tarpinių serverių;
2. spartintuvų;
3. programos lygio ugniasienės tarpinių serverių;
4. turinio generavimo serverių;

Pirminis užklausos įeigos taškas – vienintelis išorinis IP adresas skirtas HTTP ir HTTPS užklausoms apdoroti. Į šio virtualaus IP adreso užklausas atsako tarpinio serverio programinė įranga.

Tuomet užklausa patikrinama pagal RFC standartą ir tik gavus visą užklausos turinį ji persiunčiama toliau. Taip apsisaugoma nuo atsisakymo aptarnauti atakų (angl. Denial of service).

Šių atakų tikslas yra sunaudoti visus galimus prisijungimus prie paslaugos su netikromis naudotojų užklausomis, tuomet tikri klientai negali būti aptarnauti. Tik sulaukus visos užklausos turinio ji yra persiunčiama toliau ir taip neužimama realaus naudotojo vieta. Taip populiarios atsisakymo aptarnauti atakos kaip „Slowloris“ ir „RUDY“ tampa neefektyvios [25].

Tarpinis serveris patikrina užklausos tipą. Jeigu ši užklausa skirta statiniui turiniui ir jos metodas yra „GET“ arba „HEAD“, ji laikoma saugi. Statinio – nekintančio turinio užklausa nėra galimybės manipuliuoti, todėl ji puikiai tinka turinio padėjimui į podėlį. Statinio turinio formatų tipai pavaizduoti 5 lentelėje. Jeigu šio failo dar kartą prireiks sistemos naudotojui, jis bus paimtas iš podėlio. Jeigu užklausa nėra statinio tipo arba panaudotas metodas nėra „GET“ arba „HEAD“, ji perduodama į vieną iš programos lygio ugniasienės tarpinių serverių. Visi galimi metodai aprašyti RFC7230 bei pavaizduoti 6 lentelė.

Programos lygio ugniasienės tarpinis serveris priima užklausą, patikrina jos antraštes ir turinį pagal aprašytas taisykles, tuomet perduoda užklausą turinio generavimo serveriui, kuris ją apdoroja. Užklausos grąžinimo metu jos rezultato antraštės ir turinys dar kartą apdorojami programos lygio ugniasienės. Neradus galimų saugumo pažeidimų, užklausa nusiunčiama į pradinį tarpinį serverį ir grąžinama siuntėjui.

Toks grėsmių analizės karkaso modelis užtikrina programos lygio apsaugą, didelį serverių pasiekiamumą ir galimybę lengvai plėsti infrastruktūrą padidėjus serverių apkrovai, nes architektūros komponentą sudaro daugiau negu vienas serveris. Taip pat sukuriamas bendras apsaugos taškas – programos lygio ugniasienė, kuria galima apginti visus saugomus tinklalapius. Sistemos modelis pavaizduotas 6 pav.

5 lentelė. Statinio turinio formatų tipai

Failo galūnė	Tipas
.jpg	Fotografijos vaizdų išsaugojimo formatas
.jpeg	Fotografijos vaizdų išsaugojimo formatas
.png	Bitų masyvo formatas
.gif	Grafinių vaizdų saugojimo formatas
.tif	Žymėtasis atvaizdų failų formatas
.tiff	Žymėtasis atvaizdų failų formatas
.gz	„gunzip“ programa suglaudinto failo formatas
.tgz	„tar“ ir „gunzip“ programomis suglaudinto katalogo formatas
.bz2	„bzip2“ programa suglaudinto failo formatas
.tbz	„bzip2“ programa suglaudinto katalogo formatas
.rar	„WinRar“ programa suglaudinto failo ar katalogo formatas
.7z	„7-Zip“ programa Suglaudinto failo ar katalogo formatas
.mp3	Srautinis formatas, skirtas perduoti garso signalą
.ogg	Aukštos kokybės multimedijos formatas
.wma	„Microsoft“ multimedijos formatas
.swf	„Adobe flash“ formatas
.css	Šablonų stilių formatas
.js	Programavimo kalbos „javascript“ formatas
.htm	Kompiuterinės žymėjimo kalbos formatas
.html	Kompiuterinės žymėjimo kalbos formatas
.ico	Vaizdo formatas skirtas ikonoms.

6 lentelė. HTTP metodai ir jų paskirtis [1]

Metodas	Paskirtis
OPTIONS	Apibūdinti komunikacijos galimybes
GET	Tik gauti informaciją iš serverio
HEAD	Tik gauti informaciją (antraštes ir rezultato kodą)
POST	Siųsti duomenis į serverį
PUT	Pakeisti visą išteklio turinį į siunčiamą
DELETE	Ištrinti išteklį nurodytą per URI
TRACE	Grąžinti siunčiamus duomenis siuntėjui
CONNECT	Sukurti tunelį su serveriu per nustatytą URI

7 lentelė. Infrastruktūros komponentai ir jų funkcijos

Komponentas	Funkcijos
Tarpinis serveris	• geba užblokuoti užklausas neaptarnaujamiems domenams; • geba užblokuoti „slowloris“, „rudy“, TCP SYN atakas; • geba srautą skirtą generuojamam turiniui nukreipti į programos lygio ugniasienę;
Spartintuvas	• turinys grąžinamas iš podėlio neapkraunant galutinių serverių; • užklausos padalinamos per kelis serverius po lygiai;
Programos lygio ugniasienė	• vykdo įeinančių ir išeinančių užklausų turinio analizę; • praneša apie aptiktas atakas; • geba blokuoti vykdomas atakas;
Turinio generavimo serveris	• visų svetainių procesai veikia svetainei priklausančio vartotojo vardu; • apriboti svetainių vartotojams prieinami katalogai;

2.5. Tinklalapių grėsmių analizės karkaso projektavimo išvados

Išanalizavus tinklalapių apsaugos spragas ir poreikius, sudarytas optimalus sistemos modelio projektas. Sistemos architektūra sudaryta izoliuojant viešus ir privačius tinklus. Tik virtualus IP adresas priklauso tinklui pasiekiamam iš interneto, visi kiti serveriams priskirti IP adresai priklauso privačiam tinklui, kuris neturi išėjimo į internetą, taip užkertamas kelias galimoms saugumo ar konfigūracijos klaidoms atakuojant kitus architektūros komponentus tiesiogiai.

Pradinis taškas, į kurį patenka užklausos, yra tarpinis serveris. Jis užšifruoja vartotojo prisijungimą prie serverio ir pasirūpina pradine apsauga nuo įvairių atakų naudojantis lokaliomis taisyklėmis. Spartintuvas vykdo turinio padėjimą į podėlį. Turinio generavimo serveris yra galutinis sistemos taškas, atliekantis itin nuodugnią užklausų analizę, izoliuojantis svetainių procesus ir apsaugantis nuo neteisėtų vartotojo veiksmų.

Tokia sistema visiškai patenkina svetainių apsaugos poreikius bei optimizuoja jų darbą, o taisykles, kurios įeina į prieš tai paruoštą taisyklių rinkinį, galima pakoreguoti ir pritaikyti pagal iškilusias grėsmes. Apibendrintos infrastruktūros komponentų funkcijos pavaizduotos 7 lentelė.

3. INTERNETO GRĖSMIŲ ANALIZĖS KARKASO REALIZACIJA

Realizuojamas interneto grėsmių analizės karkasas, kuris analizuoja realų tinklalapių srautą, kad būtų galima aptikti vykdomas atakas bei nuo jų apsisaugoti.

3.1. Atlikta realizacija

Atlikta interneto grėsmių analizės karkaso realizacija pagal sukurtą projektą. Į tinklalapių turinio generavimo serverius įdiegtas papildomas programos lygio ugniasienės modulis „mod_security“. Įdiegtas programinio paketo „modsecurity-crs“ taisyklių rinkinys ir pritaikytos visos galimos taisyklės tik analizavimo rėžimu. Pritaikytų taisyklių aprašymas pateiktas 8 lentelė. Įeinantis ir išeinantis srautas yra apdorojamas, užklausos, kurios būtų užblokuotos, įrašomos į sisteminį žurnalą ir yra praleidžiamos.

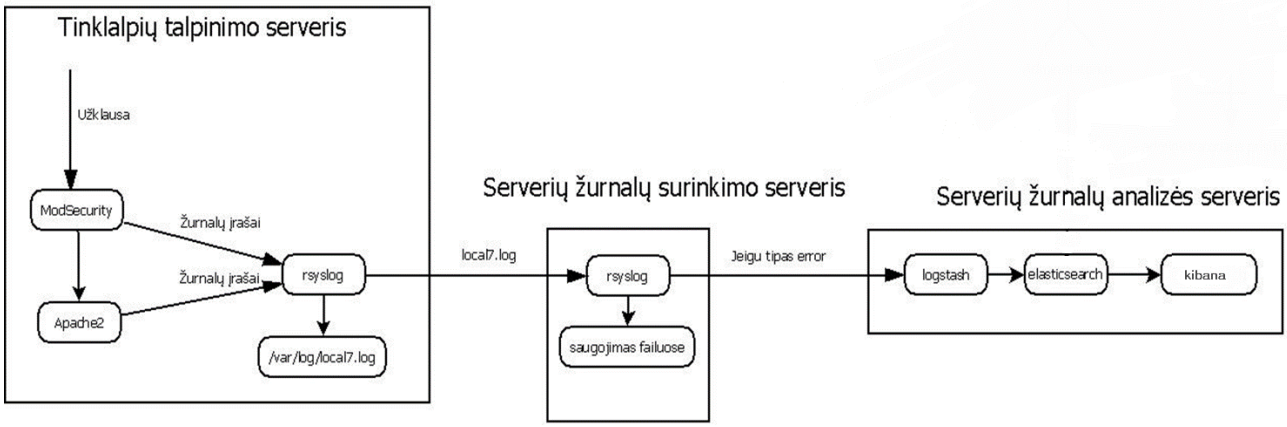
8 lentelė. Bazinių „modsecurity-crs“ taisyklių aprašas

Numeris	Taisyklių failas	Paskirtis
1.	initialization	Įkraunama konfigūracija, patikrinamos užklausos antraštės
2.	drupal-exclusion-rules	Išimtys skirtos „drupal“ turinio valdymo sistemai
3.	wordpress-exclusion-rules	Išimtys skirtos „wordpress“ turinio valdymo sistemai
4.	common-exceptions	Dažnai pasitaikančios išimtys
5.	ip-reputation	Tikrinama IP reputacija pagal anksčiau suveikusias taisykles
6.	method-enforcement	Tikrinami leidžiami metodai
7.	dos-protection	Tikrinamas užklausų kiekis
8.	scanner-detection	Tikrinamas užklausos „user-agent“ reikšmė ir blokuojami žinomi pažeidėjai.
9.	protocol-enforcement	Tikrinama ar laikomasi protokolo pagal RFC
10.	protocol-attack	Apsauga nuo HTTP lygio atakų
11.	application-attack-lfi	Tinklalapių atakų atpažinimas
12.	application-attack-rfi	Failų įterpimo apsauga
13.	application-attack-rce	Komandų vykdymo apsauga
14.	application-attack-php	„php“ įterpimo atakų aptikimas
15.	application-attack-xss	XSS atakų aptikimas
16.	application-attack-sqli	SQL įterpimo atakų aptikimas
17.	application-attack-session-fixation	Sesijų atakų aptikimas
18.	blocking-evaluation	Blokavimas pagal anomalijas, reputaciją
19.	data-leakages	Apsauga nuo tinklalapių saugojimo serverio perteklinės informacijos atskleidimo
20.	data-leakages-sql	Apsauga nuo duomenų bazės serverio perteklinės informacijos atskleidimo
21.	data-leakages-java	Apsauga nuo „java“ perteklinės informacijos atskleidimo
22.	data-leakages-php	Apsauga nuo „php“ perteklinės informacijos atskleidimo
23.	data-leakages-iis	Apsauga nuo IIS perteklinės informacijos atskleidimo
24.	blocking-evaluation	Apsauga nuo išsiunčiamos informacijos pertekliškumo
25.	correlation	Atsižvelgiamą į visų taisyklių koreliaciją

Tam, kad būtų galima apdoroti programos lygio ugniasienės generuojamus žurnalų įrašus, pasitelkta tam sukurta serverių žurnalų analizės architektūra. Ji sudaryta iš atviro kodo produkto „ELK stack“, kuris susideda iš trijų esminių komponentų: „logstash“, „elasticsearch“ ir „kibana“ [26].

- „Logstash“ – tai serverių įrašų analizės įrankis, kuris geba paimti sisteminius žurnalų įrašus iš įvairių formatų, juos išanalizuoti pagal sukurtas taisykles, išskaidyti į skirtingus laukus ir nukreipti į pasirinktą duomenų bazę.
- „Elasticsearch“ – tai indeksuota duomenų bazė, kurioje galima vykdyti greitą paiešką. Didžiausios šios duomenų bazės privalumas yra tas, kad joje galima vykdyti paiešką beveik realiu laiku [27].
- „Kibana“ – tai vienas iš kelių galimų įrankių, skirtų pavaizduoti duomenų bazės „elasticsearch“ sukauptus duomenis sudarant grafikus, lenteles ar kt.

Serverių žurnalai yra surenkami su sistemos programiniu paketu „rsyslog. Pagal „rsyslog“ formatą, aprašytą RFC5424, kiekvienoje žurnalo įrašo eilutėje atsispindi data, laikas, serverio vardas, programos vardas ir programos pranešimas. Šie įrašai siunčiami į centrinį serverių žurnalų surinkimo serverį. Žurnalų serveris pagal įrašo tipą ir prioritetą persiunčia programos lygio ugniasienės įrašus į serverį, skirtą serverių žurnalų analizei atlikti. Šiame serveryje įrašyti įrašus apdorojantys komponentai. Žurnalų įrašo kelias pavaizduotas 7 pav.



7 pav. Žurnalų įrašų kelias

3.2. Realizacijos problemos

Įgyvendinus šį modelį buvo pastebėti neatitikimai su projektu. Svetainių serveriai yra skirtingų operacinių versijų, todėl atsirado skirtumai tarp „modsecurity“ programinio paketo versijų. Nepavyko tinkamai įgyvendinti žurnalų įrašų analizės, nes dėl programinio paketo versijos neatitikimų tarp serverių skyrėsi žurnalų įrašo formatas ir bazinių taisyklių sąrašas. Norint atlikti centralizuotą žurnalų įrašų analizę, privalo būti suvienodintas žurnalų įrašų formatas ir programos lygio ugniasienės vykdomų taisyklių sąrašas. Taip pat įjungtas srauto analizavimas labai stipriai apkrauna serverius, dėl to nukenčia paslauga ir pastebimai sulėtėja svetainių atsako laikas.

Siekiant pataisyti šiuos neatitikimus, sukurtos dvi naujos virtualios mašinos, juose įdiegta naujausia „Debian“ operacinės sistemos versija. Taip pat įdiegtas „apache2“ programinis paketas, kuris veikia kaip tarpinis serveris ir programos lygio ugniasienė. Šie serveriai analizuoja įeinantį ir išeinantį HTTP srautą taip tausodami turinio generavimo serverių procesoriaus resursus. Taip pat dėl serverių vienodumo galima taikyti tas pačias taisykles bei panaikinti žurnalų įrašų formato skirtumus.

3.3. Galutinė realizuota architektūra

Viso interneto grėsmių analizės karkase dalyvauja devynios virtualios mašinos: du pradiniai tarpiniai serveriai, du turinio spartintuvai, du programos lygio ugniasienės serveriai, du turinio generavimo serveriai ir viena mašina skirta žurnalų įrašų surinkimui ir analizei realiu laiku.

3.3.1. Pradiniai tarpiniai serveriai

Du pradiniai tarpiniai serveriai užtikrina paslaugos patikimumą. Neveikiant vienam iš serverių, kitas užtikrins nenutrūkstantį paslaugos veikimą. Įrašytas tarpinio serverio programinis paketas „haproxy“, kuris geba patikrinti užklausą, apsiginti nuo paprastų atakų bei balansuoti užklausas tarp kelių serverių [28]. Šie serveriai veikia aktyviu-pasyviu rėžimu. Serverių techninės informacijos aprašas 9 lentelė.

9 lentelė. Pirminių tarpinių serverių aprašas

Operacinė sistema	„Debian“ 8.7
Procesoriaus branduolių skaičius (3.07GHz)	2
Operatyvioji atmintis	2 GB
Kietojo disko talpa	21 GB
Programinė įranga	
Tarpinis serveris	„Haproxy“ 1.7.2-1~bpo8
Virtualaus IP programinis paketas	„Keepalived“ 1:1.2.13-1

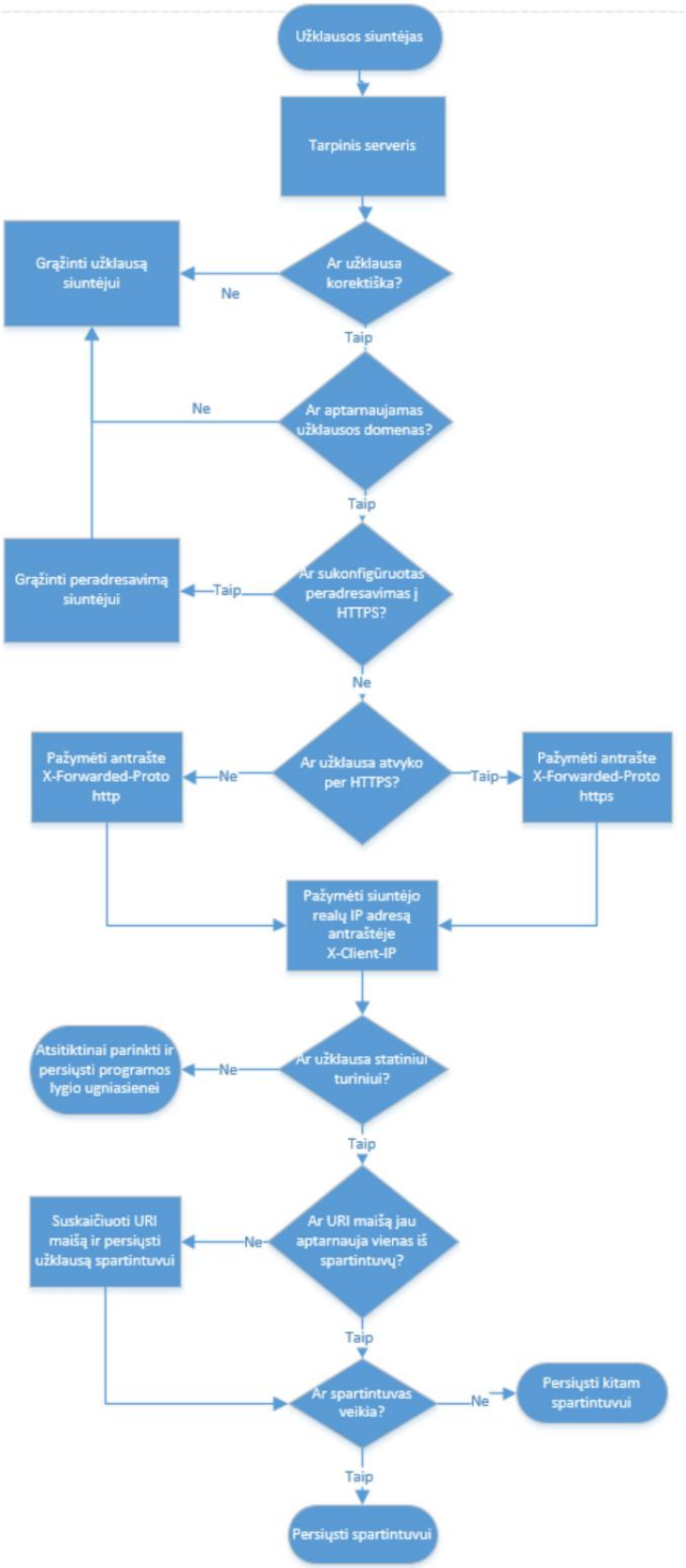
Pradiniai tarpiniai serveriai atsakingi už tai, kad užklausa būtų skirta vienam iš architektūros aptarnaujamų domenų.

Kiekviena užklausa yra patikrinama ir, jeigu ji nesilaiko HTTP ar HTTPS standarto, užklausa yra atmetama. Patikrinama ar domenas, kuriam skirta užklausa, priklauso sukonfigūruotiems domenams, kurie gali būti aptarnaujami. Jeigu sukonfigūruotas domeno peradresavimas, grąžinama užklausa su peradresavimu į šifruotą HTTPS protokolą. Kitu atveju užklausa yra aptarnaujama, papildoma „X-Forwarded-Proto“ antrašte pažymimas protokolas, kuriuo užklausa atvyko iki serverio. Taip pat papildoma „X-Client-IP“ antrašte pažymimas realus siuntėjo adresas. Ši informacija bus perduota turinio generavimo serveriui.

Tuomet vykdomas tikrinimas pagal užklausos tipą. Jeigu užklausa skirta statiniam turiniui, suskaičiuojama užklausos URI maiša. Jeigu suskaičiuota maišos reikšmė jau yra aptarnaujama vieno iš spartintuvų, ši užklausa persiunčiama tam pačiam spartintuvui, kad nebūtų dubliuojama podėlio informacija. Jeigu gautas maišos rezultatas dar nėra aptarnaujamas nė vieno iš spartintuvų, jis parenkamas atsitiktinai.

Tarpinis serveris nuolatos tikrina užklausas aptarnaujančių serverių būseną ir serveriui neatsakius į užklausą, jis yra išimamas iš galimų serverių sąrašo. Pagrindinė tarpinio serverio veikimo struktūrinė schema pavaizduota 8 pav.

Siekiant sumažinti žalingą srautą yra pridėtos dvi papildomos sąlygos. Pirmoji – kad administratoriai gali pridėti IP adresus, kurie nebus aptarnaujami, į juodąjį sąrašą ir tokios užklausos bus tiesiog atmetamos. Antroji – kad visos užklausos į dažniausiai pasitaikančius administratoriaus prisijungimo URI tokius, kaip „/wp-admin/“ ar „/administrator/“ privalo būti kilusios iš Lietuvos IP adresų erdvės.



8 pav. Pradinio tarpinio serverio veikimo struktūrinė schema

3.3.2. Turinio spartintuvai

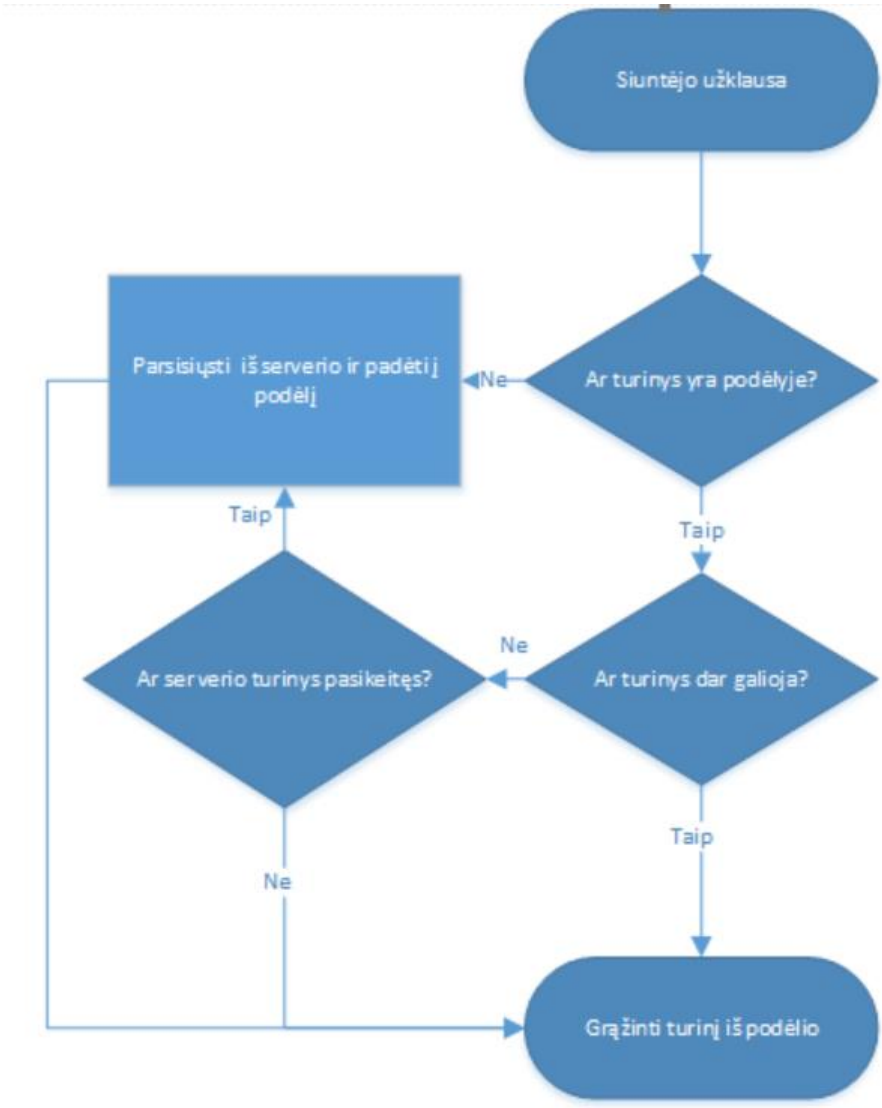
Sukurti du turinio spartintuvai tam, kad tarpiniai serveriai galėtų balansuoti užklausas tarp dviejų spartintuvų ir taip vienam serveriui tapus nepasiekiamam kitas galėtų aptarnautų užklausą. Šie serveriai statinį turinį saugo podėlyje taip pagreitindami užklausos aptarnavimą. Serverių techninės informacijos aprašas

10 lentelė.

10 lentelė. Spartintuvų aprašas

Operacinė sistema	„Debian“ 8.6
Procesoriaus branduolių skaičius (3.07GHz)	1
Operatyvioji atmintis	16 GB
Kietojo disko talpa	5 GB
Programinė įranga	
Turinio spartintuvas	„Varnish“ 4.0.2-1 3.0.0-3

Šie serveriai aptarnauja tarpinio serverio persiųstas užklausas. Jų tikslas paspartinti užklausos aptarnavimo laiką bei perimti dalį krūvio nuo turinį generuojančių serverių. Spartintuvas gavęs užklausą patikrina ar reikiamas turinys jau yra podėlyje. Tuomet patikrinamas turinio galiojimo laikas siunčiant HEAD užklausą. Jeigu turinys nėra pasikeitęs, jis nėra parsisiunčiamas iš serverio. Prailginamas nepasikeitusio turinio galiojimo laikas ir jis grąžinamas iš podėlio. Tuo atveju, jeigu turinys nėra podėlyje, jis atsisiunčiamas ir padedamas į podėlį. Užrašomas turinio generavimo serverio pageidaujamas galiojimo laikas, jeigu to nėra naudojamas sistemos numatytasis nustatymas. Turinio spartintuvo veikimo struktūrinė schema pavaizduota 9 pav.



9 pav. Spartintuvo veikimo struktūrinė schema

3.3.3. Programos lygio ugniasienės

Sukurti du programos lygio ugniasienės tarpiniai serveriai, kurie vykdo užklausų analizę pagal visas rinkinio taisykles ir taip neapkrauna turinį generuojančių serverių procesoriaus resursų. Serveriuose veikia „apache2“ programinė įranga kaip tarpinis serveris, kuris analizuoja įeinančias bei išeinančias užklausas. Šių serverių aprašas 11 lentelėje.

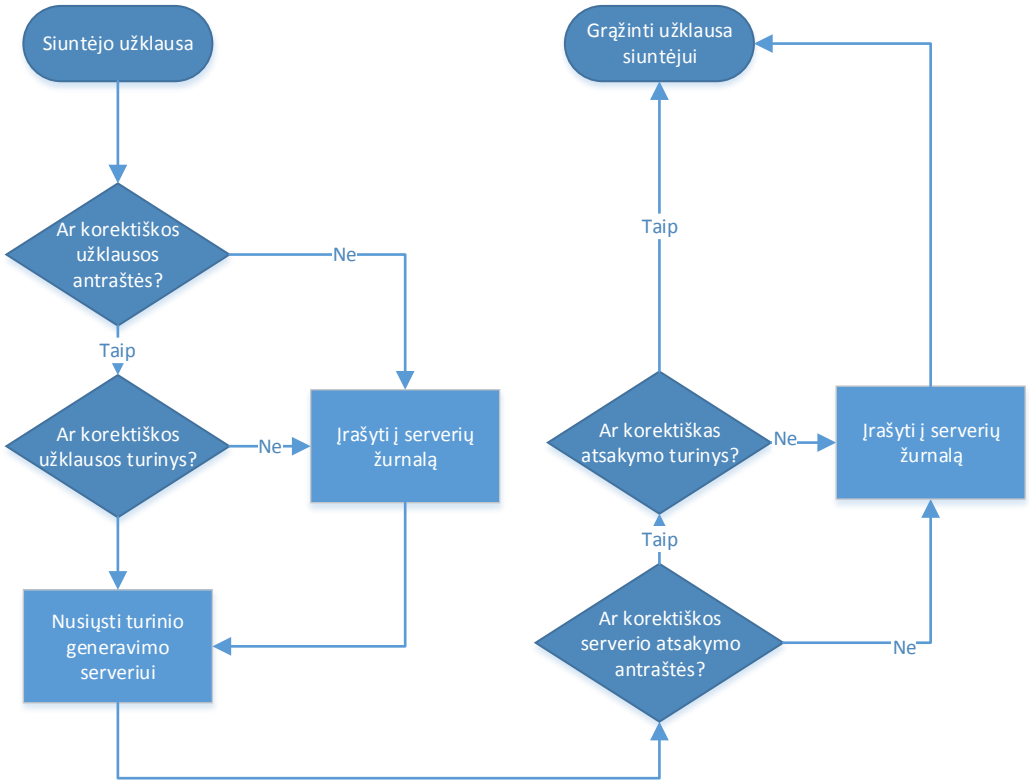
11 lentelė. Programos lygio ugniasienės tarpinio serverio aprašas

Operacinė sistema	„Debian“ 9.0
Procesoriaus branduolių skaičius (3.07GHz)	4
Operatyvioji atmintis	1 GB
Kietojo disko talpa	15 GB
Programinė įranga	
Tarpinis serveris	„Apache2“ 2.4.25-3
Programos lygio ugniasienė	„libapache2-mod-security2“ 2.9.1-2
Taisyklių rinkinys	„Modsecurity-crs“ 3.0.0-3

Programos lygio ugniasienė yra įdiegta per modulį „mod-security“. Šis modulis suteikia galimybę konfigūruoti srauto analizavimo taisykles. Įdiegtas kartu su moduliu siūlomas taisyklių rinkinys „modsecurity-crs“, kuris siūlo naujausias, aktualiausias taisykles. Tinkamai sukonfigūruotos šios taisyklės geba apginti nuo potencialių atakų, įsilaužimų ir kitos kenkėjiškos veiklos.

Interneto grėsmių analizei atlikti, buvo įjungtos visas programos lygio ugniasienės „modsecurity“ taisyklių rinkinys. Kad šios taisyklės negalėtų neigiamai paveikti paslaugų kokybės, jos įjungtos tik tyrimo rėžimu. Į serverių žurnalų įrašomos visos užklausos, kurios būtų užblokuotos dėl kurios nors iš taisyklių. Įrašoma taisyklė, kuri sukeltų blokavimą, pažymima priežastis ir taisyklės numeris. Taip pat į papildomos saugumo įrašus išsaugoma visa užklausa su įeities ar išeities visomis antraštėmis. Paprastai tokios informacijos tinklalapių atvaizdavimo serveriai nesaugo, nes ji yra perteklinė, tačiau programos lygio ugniasienė ją geba saugoti analizės tikslais.

Pirmiausiai patikrinamos siunčiamos užklausos antraštės nuo galimo protokolo nesilaikymo, bandymo įterpti kodą ar kitos kenkėjiškos veiklos. Toliau patikrinamas užklausos turinio korektiškumas ir užklausa persiunčiama turinio generavimo serveriui. Grįžtanti užklausa siuntėjui apdorojami lygiai taip pat. Patikrinamos užklausos antraštės ir turinys. Kiekviena užklausa atitikusi bent kurią iš taisyklių yra įrašoma į serverio žurnalą. Taip pat pažymimas visas užklausos turinys su antraštėmis saugumo žurnaluose. Programos lygio ugniasienės veikimo struktūrinė schema pavaizduota 10 pav.



10 pav. Programos lygio ugniasienės veikimo schema

3.3.4. Žurnalų analizės serveris

Sukurtas serveris skirtas žurnalų surinkimui ir analizei. Šiam tikslui panaudota programinė įranga žinoma kaip „ELK stack“. Analizė realiu laiku reikalauja sąlyginai daug resursų, ypač stipriai išnaudojami saugyklos pajėgumai ir vieta. Serverio žurnalo įrašams saugoti skirta 100GB vietos. Visą vietą išnaudoja tik vienos savaitės žurnalų įrašai. Serverio techninės specifikacijos aprašytos 12 lentelė.

12 lentelė. Žurnalų analizės serverio aprašas

Operacinė sistema	„Debian“ 9.0
Procesoriaus branduolių skaičius (3.07GHz)	4
Operatyvioji atmintis	5 GB
Kietojo disko talpa	105 GB
Programinė įranga	
Tarpinis serveris	„Apache2“ 2.4.25-3
Žurnalų įrašų saugojimo vieta	**„E**lasticsearch“ 5.4.0
Žurnalų įrašų priėmimas ir analizė	**„L**ogstash“ 1:5.4.0-1
Žurnalų įrašų atvaizdavimas	**„K**ibana“ 5.4.0

Visų serverių žurnalų įrašai yra kaupiami vietiniame diske, kur jie saugomi tris mėnesius, ir rsyslog protokolu persiunčiami centrinį žurnalų serverį. Centriniame serveryje žurnalai šie įrašai

saugomi pusantrų metų. Iš čia įrašai pažymėti local7.info, local7.warn arba local7.error etiketėmis, kurios reiškia programos apache2 pranešimus, yra persiunčiami į žurnalų analizės serverį.

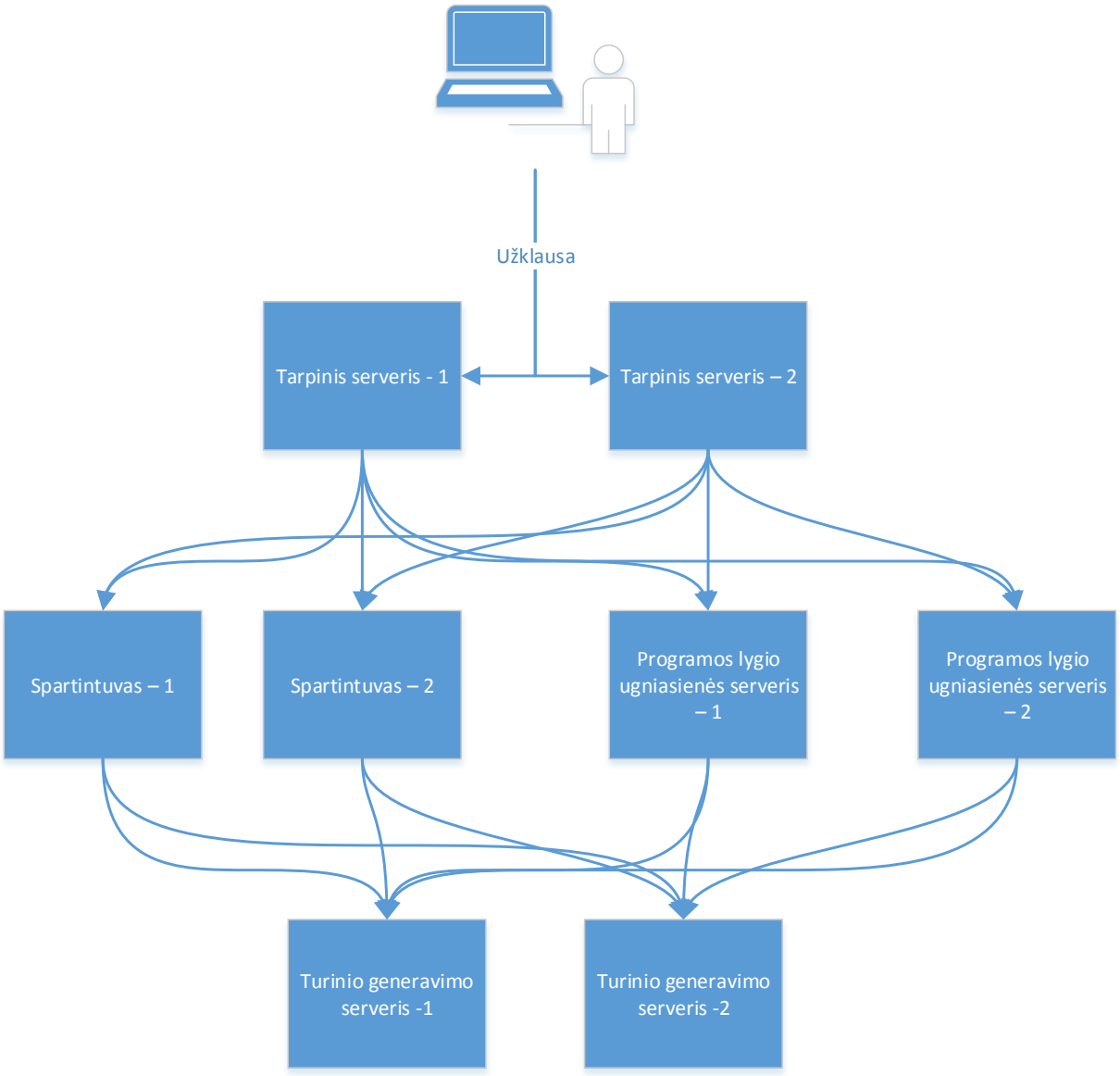
Analizės serveris priima įrašus su programiniu paketu „logstash“, kuris išanalizuoja kiekvieną įrašą. Analizuojami pradinio tarpinio serverio, programos lygio ugniasienės ir turinį generuojančio serverio žurnalų įrašai. Dvi pagrindinės programos, kurių įrašai yra analizuojami yra „haproxy“ ir „apache2“. Serverių žurnalų įrašai išskaidomi į skirtingus laukus tam, kad būtų galima lengviau atlikti analizę ir paiešką. Išskaidymo taisyklėms aprašyti naudojama „oniguruma“ reguliariųjų išraiškų, šešta versija (Angl. Oniguruma Regular Expressions Version 6.0.0). Aprašytos pradinio tarpinio serverio, turinį generuojančio serverio ir programos lygio ugniasienės reguliariosios išraiškos. Įrašai skaidomi ir surašomi į „elasticsearch“ duomenų bazę, kurioje yra suindeksuojami greitai paieškai realiu laiku. Supaprastinta žurnalų analizės modelio schema pavaizduota 11 pav.

11 pav. Žurnalų įrašo apdorojimo schema

3.3.5. Bendra architektūra

Realizuota architektūra su keletu patobulinimu išpildo interneto grėsmių analizės karkaso projektą. Du tarpiniai serveriai veikia aktyvus – pasyvus rėžimu. Vienas iš jų aptarnauja visas užklausas, tuo tarpu kitas užimtų pirmojo vietą šiam neveikiant. Tarpiniai serveriai, spartintuvai bei programos lygio ugniasienės turi visų aptarnaujamų domenų konfigūraciją, todėl kiekvieną užklausą perduota ją aptarnaujantiems serveriams. Realizuota architektūra pavaizduota 12 pav. Visos pavojingos užklausos keliauja pro programos lygio ugniasienės serverius, gauti rezultatai užrašomi į serverių žurnalų įrašus, atliekama detali įeinančios bei išeinančios užklausų antraščių ir turinio analizė. Architektūra gali būti nesunkiai plečiama, nes nesvarbu kiek yra spartintuvų, programos lygio ugniasienių ar turinio generavimo serverių. Prireikus daugiau resursų tereikia pridėti papildomą serverį ir jį įtraukti į bendrą architektūra. Tačiau tai nepavyktų įgyvendinti su tarpiniais serveriais, nes jie veikia aktyvus – pasyvus rėžimu. Galimi keli sprendimo variantai kaip paslaugoms suteikti daugiau negu po vieną IP adresą ir atlikti balansavimą pasikliaujant DNS „round-robin“ technika [29] , naudoti

tinklo įrangą, pagal kurią visi tarpiniai serveriai turėtų tą patį bendrą IP adresą ir kiekvieną užklausą dalintų tarp visų tarpinių serverių [30] ir kt. Tai nėra problema, nes labai sudėtinga sugeneruoti tokią apkrovą, kurios negalėtų aptarnauti vienas tarpinis serveris.



12 pav. Interneto grėsmių analizės karkaso realizacija

.

3.4. Rezultatai

Sukurtas interneto grėsmių analizės karkasas, kuris turi didelį patikimumą, visos karkaso dalys yra dubliuotos ir gali išgyventi vieno iš dviejų komponentų neveikimą. Gauta infrastruktūra geba nufiltruoti užklausas nepriklausančias nė vienam iš aptarnaujamų domenų, tai nublokuoja dalį atsitiktinių skanavimų iš interneto. Infrastruktūroje vykdoma užklausų skirtų generuojamam turiniui analizė, išanalizuojamas užklausos turinys ir atsakymas. Programos lygio ugniasienė veikia tyrimo rėžimu, visa informacija apie užklausos turinį yra įrašoma į serverių žurnalus tolimesnei analizei, kurią atlieka sistemos administratorius. Dėl didelio svetainių skaičiaus galimų reikšmių aibė yra labai didelė ir dalis korektiškų užklausų yra pažymimos kaip žalingos. Todėl sudėtinga įjungti blokavimo taisykles ir būtina užtikrintam, kad ateityje neiškils problemų nors vienam klientui įdiegiant naują įskiepį ar atnaujinant svetainę. Tokia sistema yra patikima blokavimui atlikti prieš žinomus pažeidžiamumus, atradus naują pažeidžiamumą, apsaugą nuo jo galima užtikrinti visoms svetainėms įterpiant taisykle programos lygio ugniasienėje ir nekeičiant svetainių programinio kodo. Sistema būtų veiksmingiausia, jeigu saugomų svetainių kiekis būtų mažas, o programinis kodas ir svetainių apsaugą užtikrinanti programos lygio ugniasienė būtų administruojama tų pačių asmenų. Tokiu atveju būtų galima sudaryti visų galimų klaidingai teigiamų reikšmių aibę sistema keletą mėnesių pasinaudojus tik įrašymo į serverių žurnalus rėžimu, ištyrus visas klaidingai teigiamas reikšmes ir tik tuomet įjungus sistema blokavimo rėžimu.

4. REALIZUOTO INTERNETO GRĖSMIŲ ANALIZĖS KARKASO TYRIMAS

Realizuota architektūra analizuoja srautą, tačiau jo neblokuoja. Neteisingai įjungtos blokavimo taisyklės gali paveikti tinklalapių saugojimo paslaugos veikseną, užblokuoti tinkamas Tam, kad būtų galima vykdyti srauto blokavimą reikia detaliai ištirti esamą srautą.

4.1. Aptarnautų užklausų tyrimas

Pasirinktas tyrimo laiko tarpas yra savaitė, nes tai labiausiai atspindi serverių krūvį. Savaitgaliais serverių krūvis yra mažiausias, o savaitės dienomis svetainės lankomos labiausios. Per savaitę, kuomet buvo vykdomas tyrimas buvo aptarnauti 41 milijonas užklausų. Iš jų net 21 milijonas buvo nufiltruotos pradiniame taške, pirmajame tarpiniame serveryje. Likę 14 milijonų užklausų buvo paskirstyti tarp serverių, turinio spartintuvų ir programos lygio ugniasienės, kurie jas aptarnavo, turinį paimant iš turinį generuojančių serverių. Vos 1,9 milijono užklausų pasiekė galutinius turinį generuojančius serverius, vadinasi viskas buvo užblokuota pradinių tarpinių serverių arba reikiama informacija grąžino turinio spartintuvai paėmę ją iš podėlio. Užklausų pasiskirstymas pavaizduotas 13 lentelė.

13 lentelė. Užklausų pasiskirstymas

Serveris	Serverį pasiekusios užklausos	Užklausų pasiskirstymas procentais
Pradinių tarpinių serverių aptarnautos užklausos	41161674	-
Pradinių tarpinių serverių užblokuotos užklausos	21636995	52,57%
Turinio spartintuvas – 1	7086617	17,22%
Turinio spartintuvas – 2	5839809	14,19%
Programos lygio ugniasienės serveris – 1	2786150	6,77%
Programos lygio ugniasienės serveris – 2	2786075	6,77%
Užklausos pasiekusios turinį generuojančius serverius	1026028	2,49%

Pradiniai tarpiniai serveriai nufiltravo 21 milijoną užklausų. Tai įvyko, nes jos tenkino vieną iš šių sąlygų:

1. sujungimo nepavyko užmegzti per 15 sekundžių;
2. užklausa buvo skirta neaptarnaujamam domenui;
3. užklausa skirta svetainės administravimo adresui ir nepriklauso vienam iš Lietuvos IP adresų;
4. siuntėjo IP adresas administratoriaus įtrauktas į juodąjį sąrašą;

4.1.1. Tarpiniai serveriai

Beveik milijonas užklausų buvo neaptarnautos, nes nepavyko sujungimas su serveriu. Klientas dėl kažkokių priežasčių neatsiuntė visos užklausos per nustatytas 15 sekundžių, todėl jos buvo atmestos. Todėl nesutampa serverį pasiekusių užklausų ir aptarnautų užklausų kiekiai. Beveik visos užklausos buvo „GET“ arba „HEAD“ tipo ir vos 373 tūkstančiai užklausų buvo „POST“. Tai pavaizduota 14 lentelė.

14 lentelė. Tarpinio serverio aptarnautų užklausų tipai

Metodas	Užklausų skaičius	Užklausų pasiskirstymas procentais
OPTIONS	1935	0.00%
GET	21933651	54,31%
HEAD	18078786	44,76%
POST	373535	0,92%
PUT	154	0,00%
DELETE	99	0,00%
TRACE	0	0,00%
CONNECT	0	0,00%
Viso:	40388160	-

4.1.2. Turinio spartintuvai

Turinio spartintuvai atliko statinio turinio saugojimą podėlyje, dauguma šių užklausų buvo grąžintos iš podėlio papildomai neapkraunant turinį generuojančio serverio. Jeigu nenurodyta kitaip numatytasis statinio turinio galiojimo laikas yra 24h. Jeigu turinys jau yra podėlyje, bet jo galiojimo laikas pasibaigęs į turinį generuojantį serverį nusiunčiama vos viena „HEAD“ tipo užklausa, kuri patikrina ar turinys nepasikeitęs. Taip nuimama didelė krūvio dalis nuo turinį generuojančio serverio. Iš viso statiniui turiniui buvo skirti beveik 13 milijonų užklausų. Iš jų beveik visų užklausų tipas „GET“, kuris skirtas turiniui paimti, tai pavaizduota 15 lentelė.

15 lentelė. Turinio spartintuvo aptarnautų užklausų tipai

Metodas	Užklausų skaičius	Užklausų pasiskirstymas procentais
OPTIONS	13	0,00010%
GET	12916922	99,93434%
HEAD	8444	0,06533%
POST	17	0,00013%
PUT	10	0,00008%
DELETE	0	0,00000%
TRACE	3	0,00002%
CONNECT	0	0,00000%
Viso:	12925409	

4.1.3. Programos lygio ugniasienės

Programos lygio ugniasienės serveriai aptarnavo 5,5 milijono užklausų. Visos šios užklausos buvo skirtos dinaminiam turiniui, todėl jos gali būti pavojingos ir nėra tinkamos turinio padėjimui į podėlį. Dauguma jų taip pat buvo „GET“ tipo, tai pavaizduota

16 lentelė.

16 lentelė. Programos lygio ugniasienės aptarnautų užklausų tipai

Metodas	Užklausų skaičius	Užklausų pasiskirstymas procentais
OPTIONS	1479	0,02654%
GET	5317580	95,43466%
HEAD	7096	0,12735%
POST	245672	4,40908%
PUT	46	0,00083%
DELETE	81	0,00145%
TRACE	2	0,00004%
CONNECT	3	0,00005%
Viso:	5571959	

Tarp aptarnautų užklausų vos apie tris procentus buvo tokių, kurios įjungus programos lygio ugniasienės blokavimo rėžimą būtų užblokuotos, tai pavaizduota 17 lentelė.

17 lentelė. Aptarnautų užklausų ir rastų anomalijų santykis

Serveris	Aptarnautos užklausos	Rastos anomalijos	Rastos anomalijos procentais nuo visų užklausų
Programos lygio ugniasienės serveris – 1	2786150	90530	3,24%
Programos lygio ugniasienės serveris – 2	2786075	89085	3,19%
Viso:	5844732	179615	3,07%

4.2. Išanalizuotas srautas

Programos lygio ugniasienei išanalizavus šias užklausas, vos 3% jų būtų užblokuotos, nes tenkintų bent vieną iš programos lygio ugniasienės blokavimo taisyklių. Nors srautas, kuris būtų blokuojamas yra sąlyginai mažas, tai nereiškia, kad programos lygio ugniasienė prastai veikia. Nežinomas realiai įvykusių atakų kiekis ir ar visos aptiktos žalingos užklausos iš tikrųjų buvo dėl vykdomos atakos, o ne dėl netaisyklingai padaryto tinklalapio, kuris iššaukė klaidingai neigiamą atsakymą.

5. IŠVADOS

Šiame darbe atlikta pasirinktos Kauno technologijos universitete laikomos mokyklų svetainių infrastruktūros tinklalapių laikymo paslaugos saugumo analizė.

Už svetainių apsaugą ir atnaujinimą yra atsakingi svetainių administratoriai, kurie ne visada yra susipažinę su galimomis grėsmėmis. Taip pat svetainių architektūros atakas labai sudėtinga atskirti nuo naudotojų srauto, nes jos vykdomos tais pačiai prievadais ir protokolais. Susiduriant su šiomis problemomis pasirinkta perdaryti svetainių saugojimo architektūrą. Vienas iš apsigynimo nuo svetainių architektūros atakų būdų yra programos lygio ugniasienė, kuri analizuoja įeinantį ir išeinantį srautą.

Projektuojant sistemos architektūrą atsižvelgta į „VMware“ hipervizorių resursų paskirstymą taip užtikrinant architektūros patikimumą ir plėtimo galimybės. Iškilus didesniam resursų poreikiui virtualių serverių skaičius gali būti padidintas ir jie nesudėtingai įsilies į sukurtą architektūrą. Numatytas įeinančio srauto šifravimas HTTPS protokolu, turinio padėjimas į podėlį, įeinančio ir išeinančio srauto analizavimas su programos lygio ugniasiene.

Realizuotame grėsmių analizės karkase dalyvauja devyni virtualūs serveriai. Tarpiniai serveriai paskirsto užklausas, spartintuvai geba turinį grąžinti iš podėlio, programos lygio ugniasienės analizuoja srautą ir praneša apie galimus mėginimus įsilaužti. Analizės serveris skirtas žurnalų įrašų analizei ir grėsmių analizės karkaso veikimo įvertinimui.

Tyrimo dalyje apžvelgtas realizuotos architektūros aptarnautų užklausų pasiskirstymas. Efektyviausiai apsaugą užtikrino tarpiniai serveriai, kurie blokavo net 52,57% visų tinklalapių užklausų. Programos lygio ugniasienės užfiksavo vos 3% galimai žalingų užklausų, kurios buvo skirtos dinaminiam turiniui. Išanalizuotas normalios serverių apkrovos srautas, todėl palyginus su visu užklausų kiekiu, žalingų užklausų skaičius yra mažas, tačiau realus atakų kiekis nežinomas.

Pagal gautus rezultatus programos lygio ugniasienė yra veiksminga siekiant svetaines apginti nuo jau žinomų pažeidžiamumų, tačiau įjungtas bendras taisyklių rinkinys netinka visapusiškai svetainių apsaugai. Dėl didelio klaidingai teigiamo atsakymų kiekio programos lygio ugniasienės nepatartina įjungti blokavimo rėžimu. Šią ugniasienę tikslingiausia naudoti atlikus detalų suveikiančių taisyklių tyrimą per ilgą laiko tarpą. Galimų įvesčių ir suveikiančių taisyklių aibė turi būti mažesnė, todėl tikslingiausia ją naudoti apsaugant vieną specifinę programą.

NUORODOS

[1] I. E. T. F. (IETF), „Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing,“ 06 2014. [Tinkle]. Available: https://tools.ietf.org/html/rfc7230. [Kreiptasi 15 01 2016].

[2] O. W. A. S. Project, „OWASP,“ 12 04 2017. [Tinkle]. Available: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project#tab=OWASP_Top_10_for_2017_Release_Candidate. [Kreiptasi 17 04 2017].

[3] X. Y. F. A. LaShanda Dukes, „A Case Study on Web Application Security Testing with Tools and Manual Testing,“ 2013.

[4] A. H. S. S. M. M. H. F. A. Abdul Razzaq, „Critical Analysis on Web Application,“ *IEEE Eleventh International Symposiumon Autonomous Decentralized Systems (ISADS),* 2013.

[5] W3Techs, „w3techs.com,“ 17 04 2017. [Tinkle]. Available: https://w3techs.com/technologies/details/cm-wordpress/all/all. [Kreiptasi 17 04 2017].

[6] P. I. a. V. K. Teemu Koskinen, „Quality Of WordPress Plug-Ins:,“ *ASE/IEEE International Conference on Social Computing and ASE/IEEE International Conference on Privacy, Security,,* 2012.

[7] S. S.-B. T. C. Huw Fryer, „Malicious web pages: What if hosting providers could actually do something…,“ *Computer Law & Security Review,* t. 31, nr. 4, pp. 490-505, 2015.

[8] „Apache2,“ [Tinkle]. Available: https://httpd.apache.org/. [Kreiptasi 15 01 2016].

[9] B. R. M. K. Prakash P, „Performance Analysis of Process Driven and,“ *IEEE Sponsored 9th International Conference on Intelligent Systems and Control (ISCO),* 2015.

[10] „OWASP VFW Project,“ 23 01 2014. [Tinkle]. Available: https://www.owasp.org/index.php/OWASP_VFW_Project. [Kreiptasi 15 01 2016].

[11] D. B. E. K. Theodoor Scholte, „Have things changed now? An empirical study on input,“ *Computers & Security,* t. 31, nr. 3, p. 344–356, 2012.

[12] V. M. D. D. S. Tejvir Kaur, „Comparison of network security tools- Firewall,,“ *International Journal of Enhanced Research in Science Technology & Engineering,* t. 3, nr. 2, pp. 200-204, 2014.

[13] D. J. S. Ibrahim Waziri, „A Heuristic Migration Logic Between Firewalls in a Federated Cloud Network,“ *IEEE SoutheastCon,* 2015.

[14] C.-H. R. L. Y.-C. L. K.-Y. T. Hung-Jen Liao, „Intrusion detection system: A comprehensive review,“ *Journal of Network and Computer Applications,* pp. 16-24, 2013.

[15] M. K. T. V. Roman Jasek, „APT detection system using honeypots,“ *Recent Advances in Automatic Control, Information and Communications,* pp. 25-29, 2013.

[16] Y. R. H. G. M. H. A. Z. Ghanbari, „Comparative approach to web application firewalls,“ *International Conference on Knowledge-Based Engineering and Innovation (KBEI),* 2015.

[17] M. Vaishali, „Web Application Firewall to Protect Against Web Application Vulnerabilities: A Survey and Comparison,“ *Int. J. Computer Technology & Applications,* t. 4, pp. 141-144, 2013.

[18] R. S. P. M. R. G. Rama Koteswara Rao, „Neutralizing Cross-Site Scripting Attacks Using Open Source Technologies,“ *ICTCS '16 Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies,* 2016.

[19] G. S. A. H. C. W. M. J. X. Z. Ajay Gulati, „VMware Distributed Resource Management: Design, Implementation, and Lessons Learned,“ *VMTJ Spring 2012,* pp. 45-64, 2012.

[20] S. N. S. M. D. Martti Vasar, „Framework for Monitoring and Testing Web Application,“ *Proceedings of the WICSA/ECSA 2012 Companion Volume,* pp. 53-60 , 2012 .

[21] I. E. T. F. (IETF), March 2010. [Tinkle]. Available: https://tools.ietf.org/html/rfc5798. [Kreiptasi 15 04 2016].

[22] S. Blake-Wilson, M. Nystrom, D. Hopwood, J. Mikkelsen ir T. Wright, Network Working Group, June 2013. [Tinkle]. Available: https://www.ietf.org/rfc/rfc3546.txt. [Kreiptasi 22 04 2016].

[23] J. B. J. W. X. Y. L. F. Sen Wang, „On adapting HTTP protocol to content centric networking,“ *CFI '12 Proceedings of the 7th International Conference on Future Internet Technologies* , pp. 1-6, 2012.

[24] „Modsecurity,“ [Tinkle]. Available: https://www.modsecurity.org/. [Kreiptasi 15 01 2016].

[25] J. D. E. L. J. M. N. L. R. W. Evan Damon, „Hands-on denial of service lab exercises using SlowLoris and RUDY,“ *InfoSecCD '12 Proceedings of the 2012 Information Security Curriculum Development Conference,* pp. 21-29, 2012.

[26] S. P. S. D. a. T.-S. M. Melody Moh, „Detecting Web Attacks Using,“ *IEEE 6th International Conference on Advanced Computing,* pp. 733-738, 2016.

[27] J. Bai, „Feasibility analysis of big log data real time search based on Hbase and ElasticSearch,“ *Ninth International Conference on Natural Computation (ICNC,* pp. 1166-1170, 2013.

[28] I. C. Anju Bala, „Fault Tolerance Challenges, Techniques and Implementation in Cloud Computing,“ *IJCSI International Journal of Computer Science Issues*, t. 9, nr. 1, pp. 288-293, 2012.

[29] H. E. B. M. Z. D. E. K. A. Khiyaita, „Load Balancing Cloud Computing: State of Art,“ *Network Security and Systems (JNS2), 2012 National Days of Network Security and Systems*, pp. 106-109, 2012.

[30] C. P. O. B. Gregory Detal, „Multipath in the middle(box),“ *HotMiddlebox '13 Proceedings of the 2013 workshop on Hot topics in middleboxes and network function virtualization* , pp. 1-6, 2013.

[31] L. L. QIAN Jing-hui, „Realization of Dynamic Floating IP Cluster Based on Keepalived,“ *Control and Instruments in Chemical Industry* , 2012.

[32] R. A. M. U. O. A. a. M. A.-H. Nur Amalina, „Enhanced Network Security System Using Firewalls,“ *ARPN Journal of Engineering and Applied Sciences*, t. 8, nr. 12, pp. 999-1004, 2013.

6. TERMINŲ IR SANTRUMPŲ ŽODYNAS

Santrumpa, terminas	Paaiškinimas
HTTP	Protokolas skirtas turiniui internete pasiekti (angl. Hypertext Transfer Protocol)
HTTPS	Protokolas skirtas turiniui internete pasiekti papildytas srauto šifravimu (angl. HTTP Secure)
OWASP	Atviras interneto programų saugumo projektas (angl. Open Web Application Security Project)
RFC	Interneto publikacijos tipas, kuris aprašo techninę specifikaciją (angl. Request for Comments)
VFW	Programinio paketo „varnish“ ugniasienė (angl. Varnish FireWall)
TCP	Vienas pagrindinių interneto protokolų (angl. Transmission Control Protocol)
IDS	Įsibrovimo aptikimo sistema (angl. Intrusion Detection System)
IPS	Įsibrovimo sutrukdymo sistema (angl. Intrusion Prevention System)
IP	Interneto protokolas (angl. Internet Protocol)
DOS	Atsisakymo aptarnauti ataka (angl. Denial-of-service attack)
XSS	Sukryžminto kodavimo ataka (angl. Cross-Site Scripting)
SQL	Kalba skirta duomenų bazių valdymui (angl. Structured Query Language)
VRRP	Kompiuterių tinklo protokolas skirtas IP adresų paskirstymui tarp dalyvaujančių serverių (angl. Virtual Router Redundancy Protocol)
SSL	Saugumo technologija skirta srauto šifravimui (angl. Secure Sockets Layer)
TLS	Šis protokolas pakeitė SSL, protokolas skirtas užtikrinti saugų srauto šifravimą tarp kliento ir serverio (angl. Transport Layer Security)
SNI	TLS protokolo išplėtimas, kuomet klientas atskleidžia domeną, jo nešifruodamas (angl. Server Name Indication)
XML	Aprašų kalba skirta dokumentams formuoti (angl. Extensible Markup Language)
URI	Simbolių seka skirta atpažinti reikiamam turiniui (angl. Uniform Resource Identifier)
DNS	Srities vardų struktūra skirta be skaitinių IP adresų naudoti simbolinius (angl. Domain Name System)
SYN	Paketo tipas siunčiamas užmezgant sujungimą (angl. Synchronize)