

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INformatikos fakultetas

**Komanda: ELITNET1**

**Projektas:** **Application Layer (L7) Firewall component HRPI analyzer**

Modulis: P175B015

Programų sistemų inžinerija  
Laboratorinių darbų dokumentacija

**Laboratorinių darbų dėstytojas**:  
lekt. Andrej Ušaniov

**Vertintojas**

Virginija Limanauskienė

**Studentai:**

Kazimieras Buškus, IFF-6/9 gr.

Šarūnas Andrijauskas, IFF-6/7 gr.

Robertas Strazdauskas, IFF-6/7 gr.

Tomas Jurevič, IFF-6/1 gr.

Kaunas, 2018

Turinys

[Įvadas 3](#_Toc515406500)

[1. Laboratorinių darbų ataskaitos 4](#_Toc515406501)

[1.1. L0 laboratorinio darbo ataskaita 4](#_Toc515406502)

[1.1.1. Bendra informacija 4](#_Toc515406503)

[1.1.2. Kuriamos sistemos idėjos aprašas 4](#_Toc515406504)

[1.2. L1 laboratorinio darbo ataskaita 6](#_Toc515406505)

[1.2.1. Projekto tikslas 6](#_Toc515406506)

[1.2.2. Užduoties analize: techninis pasiūlymas 6](#_Toc515406507)

[1.2.3. Įvardinti projekto funkciniai ir nefunkciniai reikalavimai 9](#_Toc515406508)

[1.2.4. URL ir prisijungimo duomenys į projekto valdymo įrankį/paskyrą. 10](#_Toc515406509)

[1.2.5. URL ir prisijungimo duomenys į pasirinktą projekto kodo saugyklą. 11](#_Toc515406510)

[1.3. L2 laboratorinio darbo ataskaita 11](#_Toc515406511)

[1.3.1. Užduotis 11](#_Toc515406512)

[1.3.2. Projekto eiga 13](#_Toc515406513)

[1.4. L3 laboratorinio darbo ataskaita 17](#_Toc515406514)

[1.4.1. Trečias sprintas 17](#_Toc515406515)

[1.4.2. Testavimas 17](#_Toc515406516)

[1.4.3. Išvados 20](#_Toc515406517)

[2. Dokumentacija naudotojui 21](#_Toc515406518)

[2.1. Įvadas 21](#_Toc515406519)

[2.1.1. Papildoma informacija 21](#_Toc515406520)

[2.2. Programos funkcijos ir diegimas 21](#_Toc515406521)

[2.2.1. Pagrindinės funkcijos 21](#_Toc515406522)

[2.2.2. Diegimas 21](#_Toc515406523)

[2.2.3. Vartotojo sąsaja 23](#_Toc515406524)

[3. Rezultatų apibendrinimas ir išvados 25](#_Toc515406525)

[4. Literatūros sąrašas 26](#_Toc515406526)

[5. Paveikslų sąrašas 27](#_Toc515406527)

[6. Lentelių Sąrašas 28](#_Toc515406528)

# Įvadas

P175B015 modulio „Programų sistemų inžinerija“ tikslas: įgyti bazines programavimo inžinerijos žinias ir praktinius gebėjimus taikyti tinkamus projektavimo metodus konkretiems programų sistemų tipams, organizuoti projektavimo darbus.

Laboratorinių darbų tikslas yra taikant teorijoje gautas žinias, gilintis į projektavimo proceso organizavimą, įgyti komandinio darbo patirtį, parengti reikalavimus projektuojamai sistemai, sukurti sistemos modelį, nubraižyti sistemos architektūrą, sukurti programos kodą, jį ištestuoti, parengti projekto naudotojo dokumentaciją, projekto pristatymą skaidrėmis, projekto tarpinius dokumentus talpinti talpykloje (repozitorijoje).

Šiame dokumente pateikta modulio P175B015 laboratorinių darbų ataskaitos ir sukurtos programų sistemos Dokumentacija naudotojui.

Komanda realizavo įmonės „Elitnet“ pasiūlytą projektą: aplikacijos lygmens DDoS atakų aptikimo modulį tinklo ugniasienei. Projektas rėmėsi matematiniu modeliu aprašytu moksliniame straipsnyje, pagrindinė užduotis buvo išbandyti šį modelį, išsiaiškinti jo panaudojamumo galimybes. Šiame dokumente pateikiamos laboratorinių darbų ataskaitos, apimančios projekto viziją bei apžvalgą, funkcinius bei nefunkcinius programinės įrangos reikalavimus, projekto realizacijos etapai. Taip pat pateikiama informacija apie sistemos diegimą ir naudojimą.

# Laboratorinių darbų ataskaitos

## L0 laboratorinio darbo ataskaita

### Bendra informacija

* Tvarkaraštis:

Semestro projektas: antradienis 15:30

Programų sistemų inžinerija: nelyg. sav. trečiadienis 11:00

* Programos wiki: https://github.com/sarand2/elitnet/wiki
* Komandos narių pareigos:

Robertas Strazdauskas: GUI kūrimas ir duomenų bazė

Kazimieras Buškus: HRPI logika ir multiproceso įgyvendinimas

Šarūnas Andrijauskas: DDoS atakų generavimas ir multiproceso įgyvendinimas

Tomas Jurevič: paketų iš tinklo plokštės rinkimas

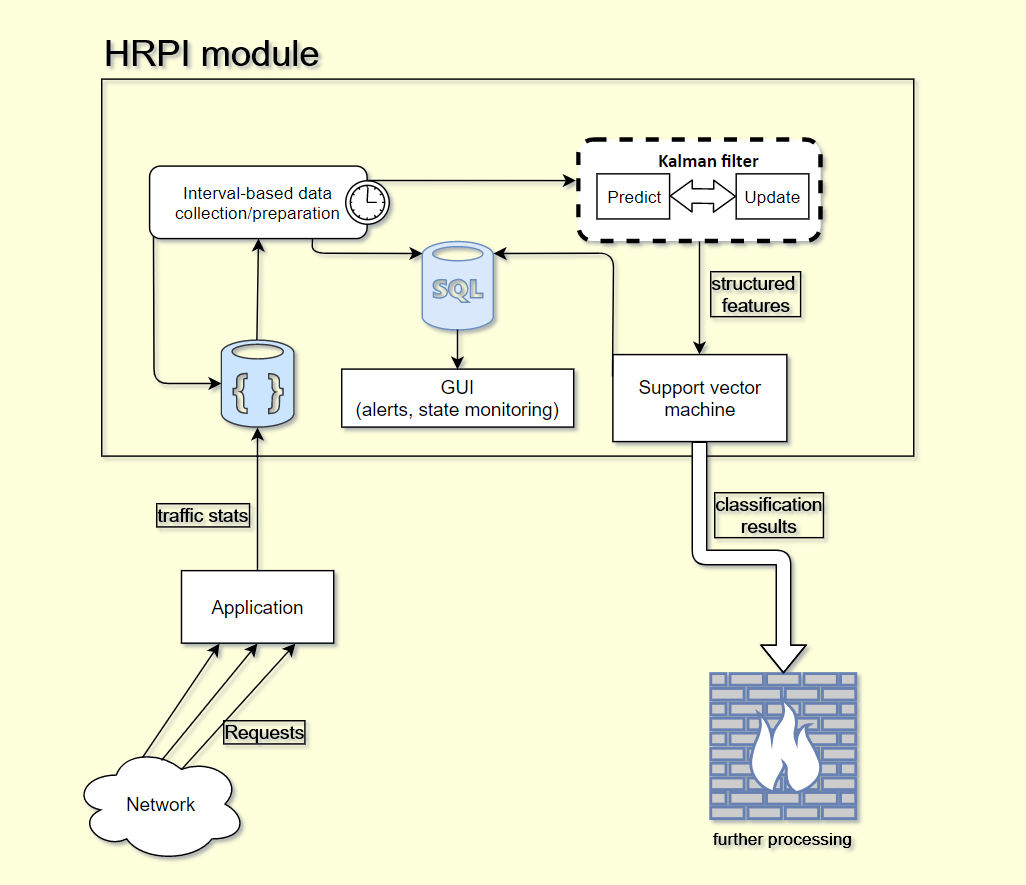
### Kuriamos sistemos idėjos aprašas

HRPI analizatorius, kuriamas pagal pateiktą matematinį modelį [1], leidžia realiu laiku aptikti aplikacijos sluoksnio DDoS atakas ir iš esmės veikia kaip vienas iš L7 ugniasienės modulių, skirtų HTTP protokolu paremtoms atakoms aptikti.

Šio komponento tikslas – pateikti efektyvią, realiu laiku galinčią veikti sistemos struktūrą bei sąveikaujančias apdorojimo funkcijas, klasifikuojančias programos lygio DDoS atakas. Analizatorius taip pat pateikia sistemos administratoriams paprastą naudoti grafinę sąsają tinklo srauto stebėjimui.

Pagrindinės funkcijos:

* Programų lygio DDoS atakų aptikimas. Šis sprendimas turi aptikti tinklo pralaidumą ir serverio resursus švaistančias HTTP GET užklausų atakas, remdamasis tam tikromis statistinėmis atakų savybėmis.
* Būsenos stebėjimas. Analizatorius teikia intuityvias tinklo srauto stebėjimo galimybes, atakos įspėjimo sistemą, ankstesnių būsenų grafikus ir manipuliacijas su istoriniais duomenimis.



pav. Abstrakti sistemos architektūra

## L1 laboratorinio darbo ataskaita

### Projekto tikslas

Sukurti taikomųjų programų lygio (7-o OSI architektūros lygio) ugniasienės komponentą, kuris pateiktų efektyvią, realiu laiku galinčią veikti sistemos struktūrą bei sąveikaujančias apdorojimo funkcijas. Analizatorius taip pat pateikia sistemos administratoriams paprastą naudoti grafinę sąsają tinklo srauto stebėjimui.

Pagrindinės ugniasienės komponento atliekamos funkcijos:

1. Programų lygio DDoS atakų aptikimas. Šis sprendimas turi aptikti tinklo pralaidumą ir serverio resursus švaistančias HTTP GET užklausų atakas, remdamasis tam tikromis statistinėmis atakų savybėmis.
2. Būsenos stebėjimas. Analizatorius teikia intuityvias tinklo srauto stebėjimo galimybes, atakos įspėjimo sistemą, ankstesnių būsenų grafikus ir manipuliacijas su istoriniais duomenimis.

### Užduoties analizė: techninis pasiūlymas

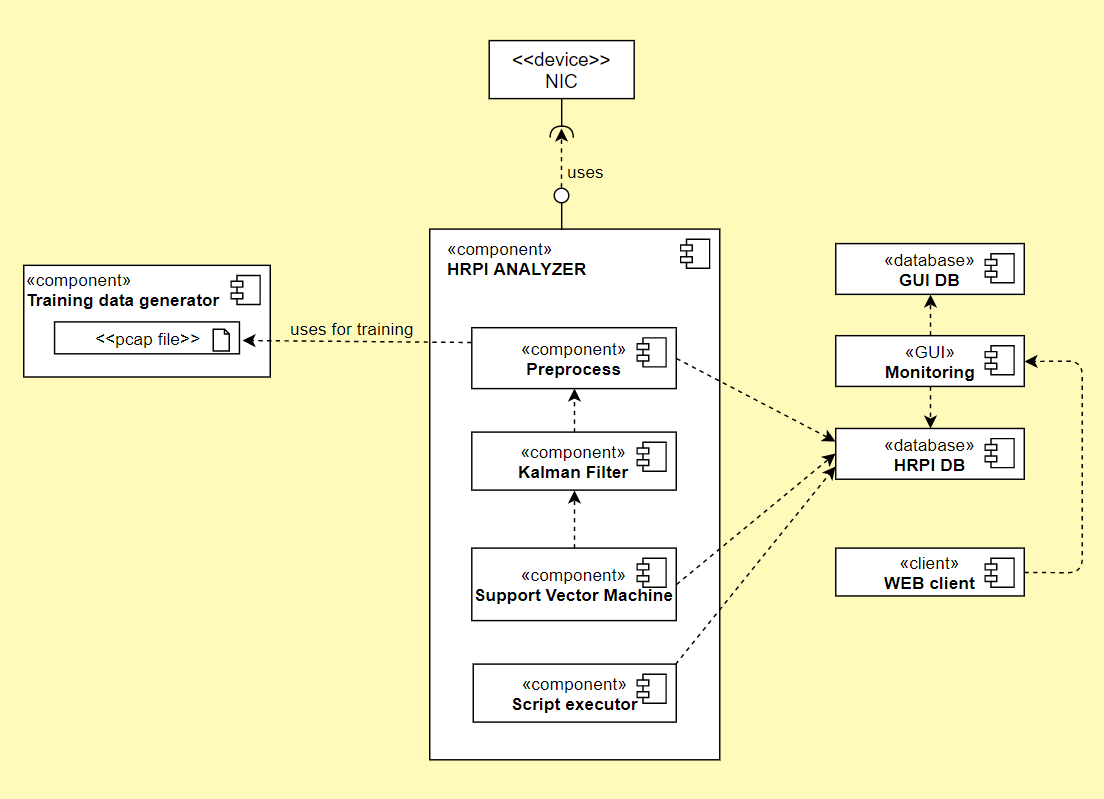
1. **Apibrėžimai bei sutrumpinimai:**
   1. HTTP – HyperText Transfer Protocol
   2. TCP– Transfer Control Protocol
   3. SVM –Support Vector Machine
   4. HRPI – HTTP GET Requests Per IP
   5. DDoS –Distributed Denial of Service
   6. API –Application Programming Interface
2. **Sprendimo apžvalga**

HRPI analizavimo modulis aptinka programos lygio DDoS atakas ir veikia kaip vienas iš L7 lygio ugniasienės komponentų.

Naudojant matematinį metodą[[1]](#footnote-1), ši programa gali efektyviai aptikti atakas ir paleisti atitinkamų veiksmų rinkinį. Sprendimas taip pat pateikia lengvai naudojamą grafinę naudotojo aplinką su programos duomenų srauto stebėjimo bei atakos pranešimais. Grafinė vartotojo sąsaja taip pat gali pateikti istorinius duomenis.

Šis sprendimas susideda iš apmokymo duomenų kūrimo SVM apmokymui aptikti atakas, bei jos testavimui.

**Komponentų apžvalga**

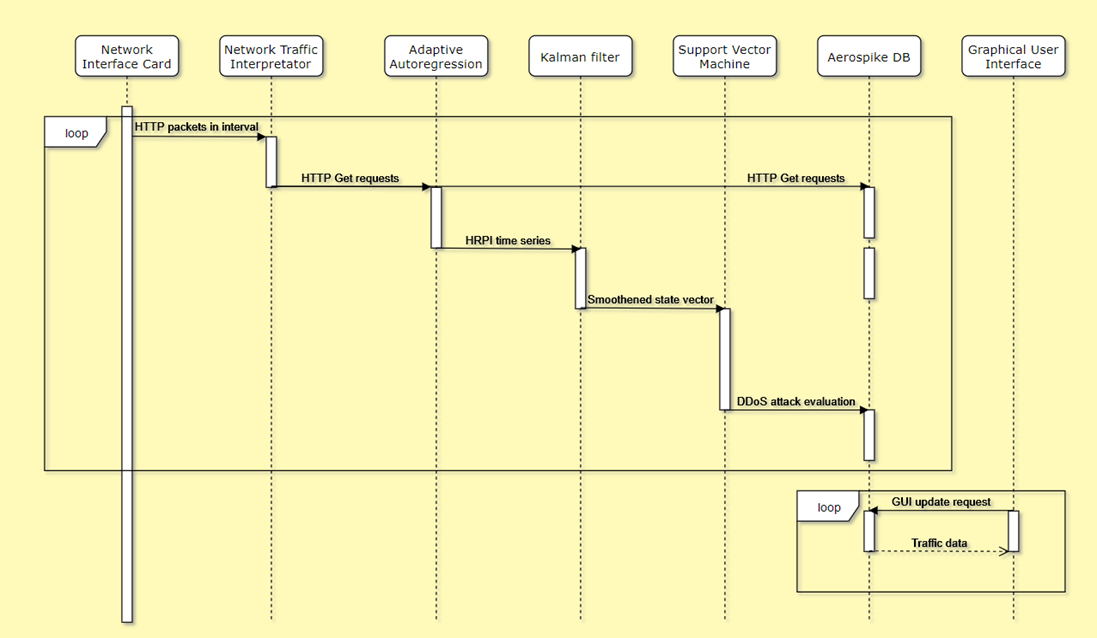


2 pav. Komponentų schema

* NIC – Network Interface Card – prietaisas su kuriuo modulis tiesiogiai bendrauja, kad gautų realaus laiko duomenų srauto duomenis
* Training data generator – programa atsakinga už apmokymo duomenų generavimą SVM klasifikatoriui
* Preprocess – algoritmas skirtas paversti paketinius duomenis iš NIC į HRPI, kurie bus naudojami matematinio modelio
* Kalman filter – algoritmas skirtas įvertinti esamą duomenų srauto būseną
* Support Vector Machine – algoritmas skirtas atskirti atakos duomenų srautą nuo įprasto srauto
* Monitoring –Grafinė vartotojo sąsaja WEB, skirta stebėti esamą srautą ir istorinius duomenis
* Script executor – programa atakos atveju paleidžianti iš anksto numatytą programą.
* GUI DB – Maria DB skirta saugoti GUI informacija, pavyzdžiui prisijungimo informaciją.
* HRPI DB – Aerospike DB skirta realaus laiko duomenų srauto informacijai saugoti
* WEB client – programa veikianti naudotojo naršyklėje

1. **Sąsajos apžvalga**
   1. WIRE – TCP pagrįstas protokolas skirtas bendrauti su Aerospyke duomenų baze
   2. HTTP interface – HTTP protokolas naudojamas bendravimui tarp GUI ir WEB kliento
   3. SQLCNTR – SQL jungties sutrumpinimas – standartinis Python API bendravimui su MariaDB
2. **Veikimo apžvalga**

Vieno laiko intervalo paketų rinkinio įvertinimo vykdymo sekos diagrama:

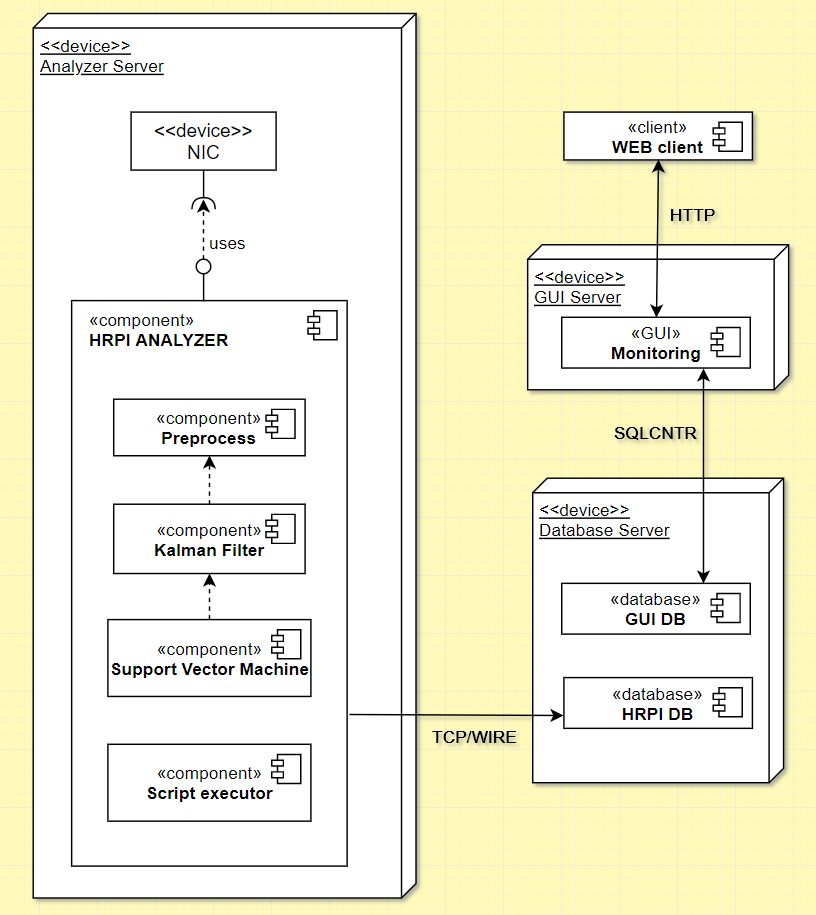


3 pav. Vieno laiko intervalo paketų rinkinio įvertinimo vykdymo sekos diagrama

1. **Pavojaus įspėjimas ir stebėjimas**

Pavojaus įspėjimas yra pateikiamas grafine sąsaja. Programa įrašo duomenų srautą ir jo klasifikavimą, todėl grafikai yra pažymimi atitinkamomis spalvomis.

1. **Įdiegimo rodinys**



4 pav. Įdiegimo rodinio schema

### Įvardinti projekto funkciniai ir nefunkciniai reikalavimai

1. **Funkciniai reikalavimai**
   1. **Bendri reikalavimai**

FBR-1 Šis sprendimas turi suteikti galimybę paleisti iš anksto nustatytus veiksmus / programas pagal esamą duomenų srautą.

FBR-2 Modulis turi palaikyti aukštą pasiekiamumą: programa bus paleista dvejuose mazguose vienu metu.

FBR-3 Modulis turi palaikyti statistikos stebėjimą:

1. Peržiūrėti dabartinę ir istorinę užklausų peržiūrą
2. Peržiūrėti dabartinę ir istorinę duomenų srauto informaciją atakos metu
3. Išpuolio metu pateikti aktyviausių mazgų peržiūrą

FBR-4 Metrikos turi būti prieinamos per interneto GUI

FBR-5 Metrikos istorinis kontekstas turi įtraukti grafiko laiko matavimo vienetus: minutes, valandas, dienas

FBR-6 Modulis turi palaikyti prisijungimo sistemą

FBR-7 Šis sprendimas turi naudoti didelio efektyvumo skaičiavimo metodus ir apdoroti 1Gb duomenų per sekundę.

* 1. **Serverio reikalavimai**

FSR-1 Serveris turi pateikti GUI prieigą per internetą

FSR-2 MariaDB duomenų bazė turi laikyti: GUI indentifikavimo informaciją

FSR-3 Aerospyke duomenų bazė turi saugoti periodinę duomenų tinklo srauto ir klasifikavimo informaciją.

* 1. **Kliento pusės reikalavimai**

FKR-1 Prisijungimas naudojantis vartotojo vardą ir slaptažodį

FKR-2 Atsijungimo nuo kliento funkcija

FKR-3 Realaus laiko tinklo srauto metrinė grafika

FKR-4 Įtartinos veiklos, atakų statistikos metrika

* + 1. **GUI reikalaviami**

FKR-5 Realiu laiku atnaujinamos, lengvai suprantamos diagramos, kurios turi įtraukti:

1. HRPI srauto informaciją
2. Bendrą tinklo srautą
3. Gautas užklausas ir paketus

FKR-6 Grafinė įspėjimo pranešimo funkcija

1. **Nefunkciniai reikalavimai**

NFR-1 Komponento stebėjimo GUI turi būti naršyklės programa, galinti veikti „Mozilla Firefox“ ir „Google Chrome“ aplinkose (atitinkamai 66 ir 60 versijos).

NFR-2 Šio modulio įgyvendinimas apima:

1. Atakų užklausų klasifikavimą ir treniravimą naudojant PCAP failus
2. Infrastuktūros struktūrizavimą: naudojamos Virtualios mašinos imituoja normalų ir DDoS srautą.

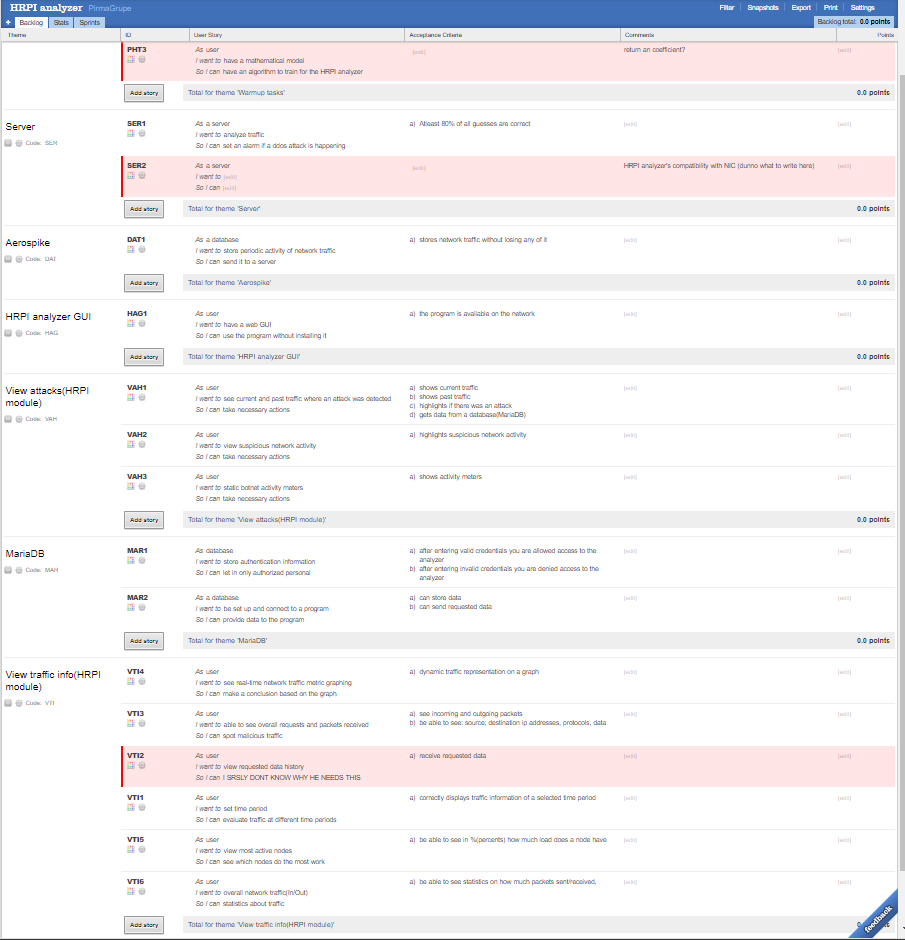
### URL ir prisijungimo duomenys į projekto valdymo įrankį/paskyrą.

[https://easybacklog.com](https://easybacklog.com/)

E-mail: gaming\_fun@mail.com

Password: labasrytas

Backlog veiklos



5 pav. Projekto valdymo įrankio iškarpa

### URL ir prisijungimo duomenys į pasirinktą projekto kodo saugyklą.

<https://github.com/sarand2/ELITNET1>

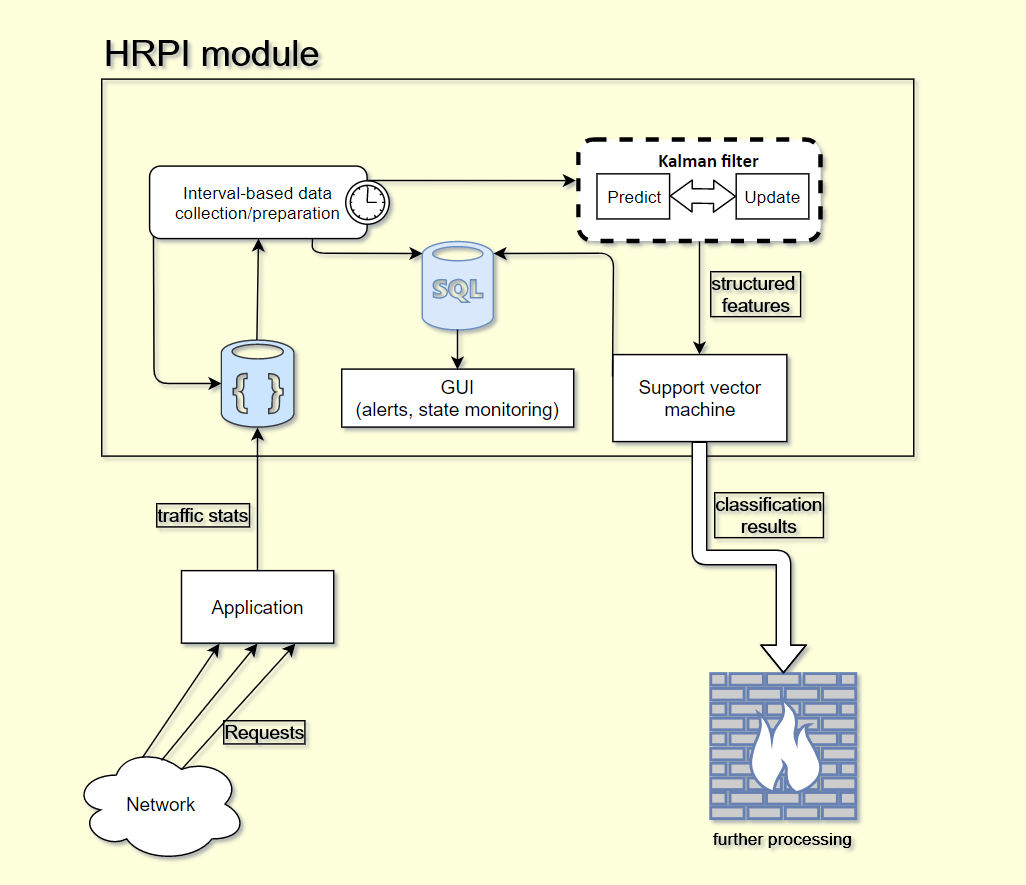
## L2 laboratorinio darbo ataskaita

### Užduotis

Pagal nurodytą matematinį metodą, sukurti HRPI (HTTP *get* Requests Per Ip) analizavimo program, leidžiančią realiu laiku aptikti aplikacijos sluoksnio DDoS atakas. Iš esmės programa turi veikti kaip vienas iš L7 ugniasienės modulių, skirtų HTTP protokolu paremtoms atakoms aptikti.

Šio komponento tikslas – pateikti efektyvią, realiu laiku galinčią veikti sistemos struktūrą bei sąveikaujančias apdorojimo funkcijas, klasifikuojančias programos lygio DDoS atakas. Programa taip pat turi pateikti sistemos administratoriams paprastą naudoti grafinę sąsają tinklo srauto stebėjimui.

Pagrindinės funkcijos:

* Programų lygio DDoS atakų aptikimas. Šis sprendimas turi aptikti tinklo pralaidumą ir serverio resursus švaistančias HTTP GET užklausų atakas, remdamasis tam tikromis statistinėmis atakų savybėmis.
* Būsenos stebėjimas. Analizatorius teikia intuityvias tinklo srauto stebėjimo galimybes, atakos įspėjimo sistemą, ankstesnių būsenų grafikus ir manipuliacijas su istoriniais duomenimis.

pav. Sistemos architektūra

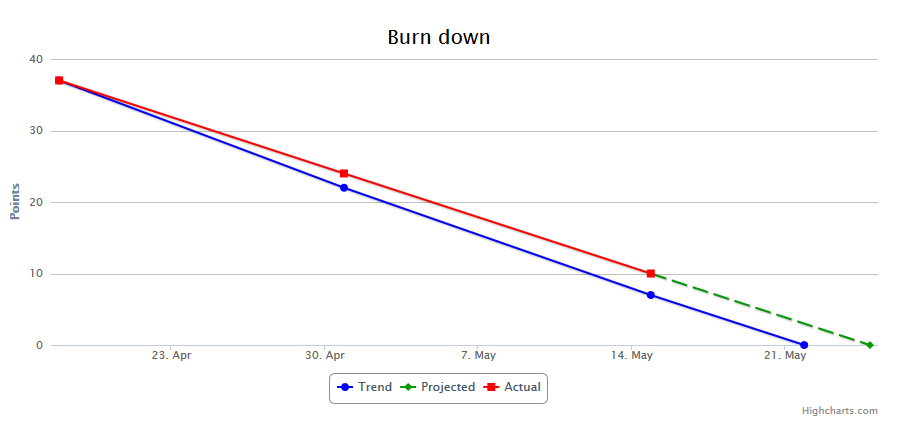
### Projekto eiga

Pirmas Sprintas

pav. Pirmo sprinto atliktų darbų sąrašas

Antras Sprintas

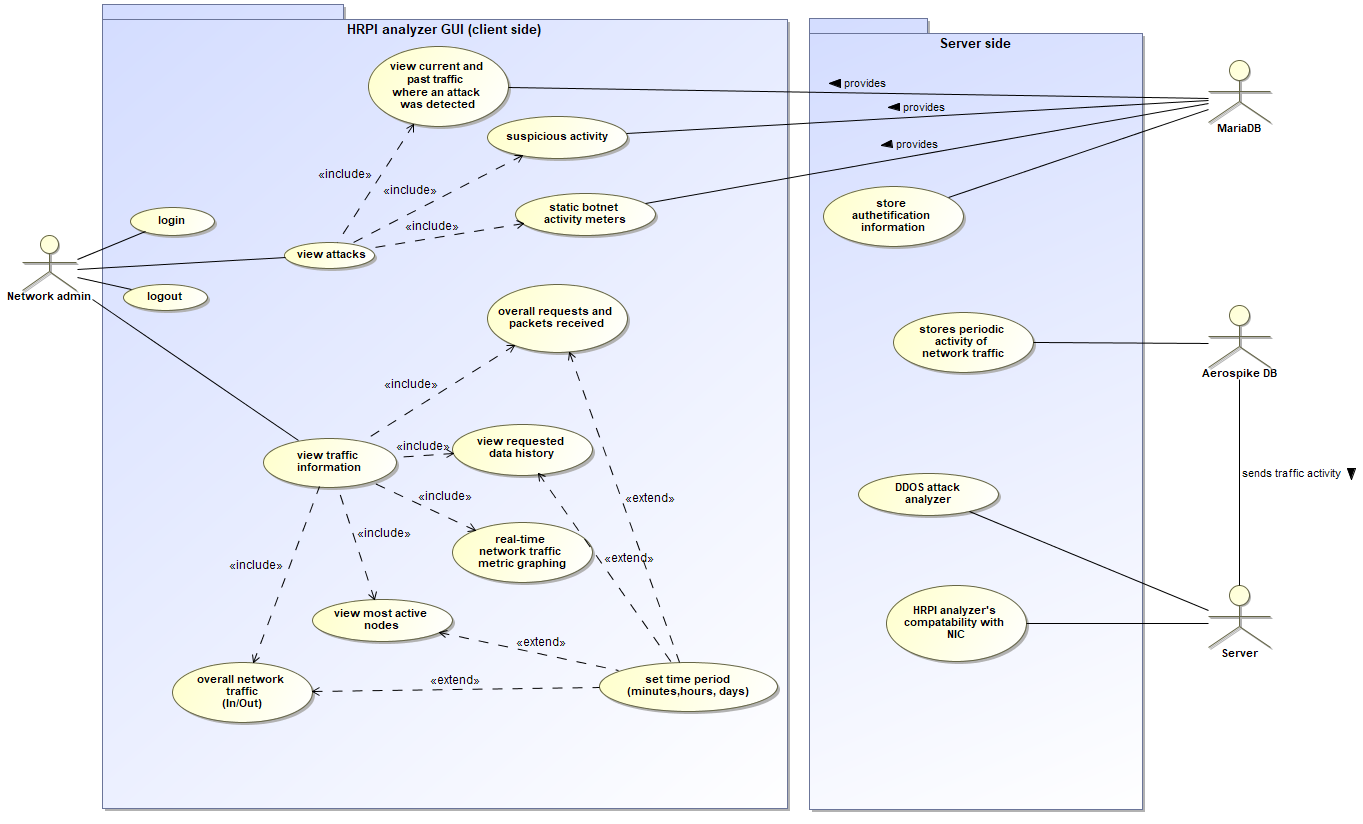
pav. Antro sprinto atliktų darbų sąrašas

Burn down grafikas

pav. „Burn-Down“ grafikas

Retrospektyva

* Individualus darbas prie atskirų sistemos komponentų leido susikoncentruoti į sritį ir pasiekti pakankamai gerus rezultatus kodo įgyvendinimo atžvilgiu, perprasti galimas problemas sąveikoje/komunikacijoje su kitais sistemos komponentais.
* Nestruktūruotas planavimas ir aiškių *deadline* nebuvimas lėmė, kad prie komponentų buvo praleidžiama daugiau laiko, sudaroma daugiau specifinės paskirties kodo, kuris nėra esminis sistemos komponentams.
* Reikėtų daugiau komunikacijos tarp užduočių ir dažnesnio esamo projekto dizaino/veikimo/tobulinimo aptarimo bendrai architektūrai įvertinti.
* Reikėtų mažiau koncentruotis ties naudojamomis technologijomis ir metodais, o bandyti kuo greičiau įgyvendinti specifinį projekto dalies funkcionalumą.

UML panaudos atvejų diagrama

pav. UML panaudos atvejų diagrama

## L3 laboratorinio darbo ataskaita

### Trečias sprintas

Pagrindinės užduotys

Trečiajame sprinte buvo siekiama iki galo užbaigti visus sistemos komponentus ir juos sujungti į vieną visumą. Pagrindiniai darbai buvo:

* sukurti multiprocesinio paketų apdorojimo logiką
* pritaikyti HRPI entropijos skaičiavimą treniravimo duomenis realaus laiko veikimui
* sukurti efektyvius metodus dedančius bei imančius informaciją į/iš duomenų bazės
* sukurti minimalistinę GUI programos realaus laiko veikimui pavaizduoti

Retrospektyvos rezultatai

* Dirbti poroje prie probleminių kodo dalių pasirodė ne tik lengviau, bet ir efektyviau.
* Iš anksto nesusipažinus su sudėtingesnių programos metodų testavimu, programiniai testai pasirodė užimsią, per daug laiko, jų atsisakymas šiuo sprintu baigus projektą atrodo pagrįstas.
* Kuriant GUI nuspręsta: geriau be daugiau galimybių suteikiantis *framework* ar *template*, o tas, kuris turi daugiau pavyzdžių.
* Derinant programą dėl *memory leak,* pirmiausia reiktų atsižvelgti į išorinių, mažiau pažįstamų API kvietinius.

### Testavimas

Naudojama programinė įranga

* „Wireshark“ paketų gaudymo programa testuoti
* „Pylint“ statinei Python logikos kodo analizei
* „Resource“, „Objgraph“ Python moduliai *memory leak* diagnozuoti
* Chrome (version 66) bei Mozilla naršyklės (version 60) GUI komponentų vaizdavimui tikrinti

Testavimo planas bei rezultatai

1 lentelė. Grafinės vartotojo sąsajos testavimo planas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testo ID**  **(GUI)** | **Scenarijus** | **Abstraktūs žingsniai** | **Tikėtinas rezultatas** | **Gautas rezultatas** | **Praėjo/nepraėjo** |
| **TEG01** | Įvedami neteisingi prisijungimo duomenys GUI | 1. Paleisti GUI naršyklėje 2. Įvesti neteisingus prisijungimo   duomenys | Neleidžiama prisijungti ir matyti GUI | Neleidžiama prisijungti | **Praėjo** |
| **TEG 02** | Į GUI įvesti istorijos intervalą ateityje arba tolimoje praeityje | 1. Paleistį GUI naršyklėje 2. Į laukus įvesti ateities/praeities intervalą | GUI grafike nevaizduoja traffic‘o  parametrų | Traffic‘o parametrai nevaizduojami | **Praėjo** |
| **TEG 03** | GUI turi atvaizduoti atpažintą atakos traffic‘ą | 1. Paleisti GUI naršyklėje 2. Paleisti ataką vykdantį scipt‘ą | GUI grafe turi po 10-20 sekundžių pradėti vaizduoti ataką reiškainčius taškus | Vaizduojami DDoS traffic‘o parametrai | **Praėjo** |
| **TEG 04** | GUI pasirenkant *Live* tab‘ą, turi būti vaizduojami realaus laiko traffic‘o parametrai | 1. Paleisti GUI naršyklėje 2. Pasirinkti *Live graph* skiltį | Vaizduojamas realaus laiko (su atsilikimu 1-2 s.) traffic‘o parametrų grafikas | Vaizduojami dabartinio traffic‘o parametrai | **Praėjo** |
| **TEG05** | GUI pasirenkant *History* tab‘ą, turi būti vaizduojami pasirinkto intervalo parametrai | 1. Paleisti GUI naršyklėje 2. Pasirinkti *History* skiltį 3. Pagal 3 įvedimo formas(data, valanda, valandos 5 minučių intervalas) pasirinkti norimą | Vaizduojamas pasirinkto intervalo traffic‘o parametrų grafikas | Vaizduojami pasirinkto intervalo traffic‘o parametrai | **Praėjo** |
| **TEG06** | GUI be anomalijų veikia ant Chrome 66, Mozilla 60 naršyklių | 1. Paleisti GUI iš atitinkamos naršyklės 2. Bandyti TEG01 TEG02 TEG03 TEG04 TEG05 atvėjus 3. Stebėti įvedimo/išvedimo komponentų anomalijas | Vaizduojamas GUI skirtingose naršyklėse skiriasi tik įvedimo formomis, anomalijų nerasta | Veikia kaip buvo tikėtina | **Praėjo** |

2 lentelė. Duomenų srauto testavimo planas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testo ID**  **(traffic script)** | **Scenarijus** | **Abstraktūs žingsniai** | **Tikėtinas rezultatas** | **Gautas rezultatas** | **Praėjo/nepraėjo** |
| **TED01** | Paleistas DDoS/Normal sciptas IP adresų kiekiu bei iš vieno IP siunčiamų requestų kiekiu atitinka parametrus 10% ribose | 1. Paleisti traffic scriptą su norimais parametrais 2. Stebėti traffic‘ą *Wireshark* pagalba | *Wireshark* pateikiami duomenys nenukrypsta nuo parametrų per daugiau nei 10 % | Priklausomai nuo vykdymo laiko, pastebėtos didesnės nei scenarijuje nurodytos paklaidos | **Dalinai**  (priklauso nuo *proxy* serverių apkrautumo) |

3 lentelė. HRPI įvertinimo testavimo planas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testo ID**  **(HRPI logic)** | **Scenarijus** | **Abstraktūs žingsniai** | **Tikėtinas rezultatas** | **Gautas rezultatas** | **Praėjo/nepraėjo** |
| **TEH01** | HRPI entropijos stebėjimo procesas sugeba atskirti DDoS bei normalų traffic‘ą | 1. Paleisti traffic scriptą su norimais parametrais 2. Stebėti traffic‘ą *Wireshark* pagalba 3. Stebėti traffic‘o klasifikavimą GUI, serveryje | GUI po 10-20 s. pradeda signalizuoti atakuojantį traffic‘ą, serveris vykdo nurodytus attack scriptus | Laikas, po kurio pradedama signalizuoti ataka, priklauso nuo proxy serverių apkrautumo, tačiau tiek GUI vaizdavimas, tiek script iškvietimai vykdomi teisingai | **Praėjo** |

4 lentelė Sistemos efektyvumo testavimo planas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testo ID**  **(multiprocess queue)** | **Scenarijus** | **Abstraktūs žingsniai** | **Tikėtinas rezultatas** | **Gautas rezultatas** | **Praėjo/nepraėjo** |
| **TEM01** | Iš NIC requestai turi būti kaupiami teisingai bei neperkraunant atminties pagal paskaičiuotus dabartinius atakos requestų kiekių maksimumus (max 600 req./s). | 1. Paleisti traffic scriptą su norimais parametrais 2. Stebėti serverio resursų naudojimą kol vykdoma ataka 3. Stebėti request‘ų pasiskirstymą pagal *Wireshark* 4. Stebėti sistemos apkrautumą *Resource, Objgraph pagalba* | Serverio resursai (ypač RAM) neturi būti perkrauti atakos atveju, request‘ų sąrašai turi atitikti *Wireshark* | Dabartinėmis atakos sąlygomis, multiprocesinės eilės apdorojimas yra teisingas, bei našus | **Praėjo** |

5 lentelė Kodo įvertinimo planas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testo ID**  **(bendras)** | **Scenarijus** | **Abstraktūs žingsniai** | **Tikėtinas rezultatas** | **Gautas rezultatas** | **Praėjo/nepraėjo** |
| **TEB01** | Kodas testuojamas statinės analizės įrankiu | 1. Paleisti/įdiegti įrankį 2. Stebėti pranešimus | Kodas neturi kritinių klaidų, įspėjimų, refactorinimo siūlymų, kodo kartojimosi bendras kiekis (eilutėmis) turi neviršyti 10 % viso kodo | Įrankis rodo įspėjimus kodo refactorinimui, tačiau riba neviršijama | **Praėjo** |

### Išvados

Trečiuoju sprintu komanda baigė projektinį darbą. Projektas laikomas įgyvendintu, nes pavyko teisingai suprogramuoti pagrindinio komponento funkciją: programa geba atskirti tarp normalaus traffic’o ir atakos traffic’o būsenos.

Neįgyvendinti reikalavimai liko GUI dalyje: negalima pasirinkti platesnių intervalų traffic istorijos peržiūrėjimui, nėra *logout* funkcijos, *login* atliekama tik programiniame lygmenyje. Taip pat programa iki galo neištestuota dėl patirties trūkumo, laikas skirtas pagrindinių komponentų funkcionalumui įgyvendinti.

Matavimą, jog programos sudarytas ir naudojamas *Support vector machine* algoritmo taikymas testavimo metu atpažino ~93 % atakos traffic’o parametrų laikome projekto sėkmės rodikliu.

# Dokumentacija naudotojui

## Įvadas

Šis programinės įrangos produktas yra viena dalių, turinčių funkcionuoti bendroje L7 ugniasienės platformoje, tačiau jį galima naudoti ir kaip atskirą programą. Tai nėra plačiajai vartotojų auditorijai skirta programinė įranga, todėl daroma prielaida, jog šio dokumento skaitytojas – sistemos administratorius, testuotojas ar kitos srities IT profesionalas.

### Papildoma informacija

Neatlikus papildomų, šiame dokumente neaprašytų pasirengimo darbų ir sistemų sinchronizacijos, analizatorių galima naudoti vienos (lokalios) tinklo sąsajos plokštės stebėjimui ir pranešimams apie galimą ataką gauti. Daroma prielaida, jog sistemoje jau įdiegta *Python 3.6* versija ir jo modulių tvarkyklė *pip3*.

Programa konfigūruojama per *yaml* formato failus.

Kadangi tam tikri programos naudojami paketai funkcionuoja tik linux platformoje, dokumente bus pateikti įdiegimo šablonai būtent šio tipo operacinei sistemai (konkrečiai Ubuntu 17.10).

Naudojami sutrumpinimai:

* IP– internet protocol address
* ML– machine learning
* HRPI– HTTP GET requests per second
* GUI– graphical user interface

## Programos funkcijos ir diegimas

### Pagrindinės funkcijos

Programa atlieka veiksmus, leidžiančius:

* įspėti apie ataką paleidžiant nurodytą *skriptą*
* stebėti tinklo būseną grafinėje sąsajoje

Programa pagal užklausų ir IP adresų skaičiaus, sužinomų išanalizavus į tinklo plokštę ateinančius paketus, pasisirstymo entropiją išgauna tam tikrus parametrus, kurie ML algoritmu kas 0.1 sekunės klasifikuoja tinklo būseną kaip ataką arba ne. Tinklo būsenos charakteristikos ir jų įvertinimas saugojami duomenų bazėje, istorinį ir realaus laiko grafikus galima matyti grafinėje vartotojo sąsajoje (naršyklėse Chrome (versija 66) bei Mozilla naršyklės (versija 60))

### Diegimas

Pasirengimui programinės įrangos vartojimui etapai yra šie:

* programinio kodo atsisiuntimas
* *dependencies* instaliavimas
* programos paleidimas

Programą galima atsisiųsti iš repozitorijos adresu: <https://github.com/sarand2/ELITNET1>

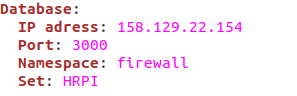
Atsisiuntus programinį kodą į norimą aplanką, reikia instaliuoti jai reikalingas bibliotekas ir modulius. Ubuntu platformoje tą galima padaryti per pip modulių tvarkyklę. Visi reikalavimai programai surašyti faile *requirements.txt*. Pavyzdys:



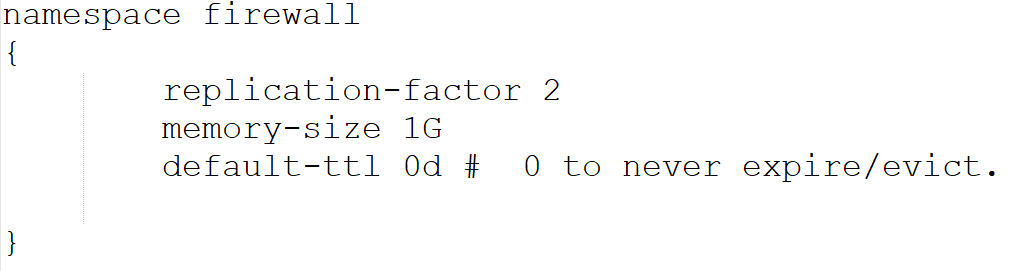
pav. Paketų diegimas

Jeigu visi paketai instaliuojasi sėkmingai, galima tęsti diegimą, jei ne – programa gali funkciuonuoti neteisingai arba iš viso neveikti.

Kadangi analizatorius naudoja *Aerospike* duomenų bazę charakteristikų saugojimui, reikia lokaliai įsidiegti arba naudoti turimą duomenų bazės serverį. Informaciją apie instaliaciją Ubuntu galite rasti čia: <https://www.aerospike.com/docs/operations/install/linux/ubuntu>

Jeigu serveris veikia kitame kompiuteryje, programos *configuration* aplanko *scriptConfig.yaml* faile galima nurodyti prisijungimo prie duomenų bazės informaciją:

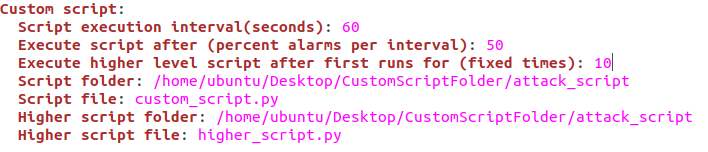
pav. Prieigos prie duomenų bazės konfigūracija

Analizatorius naudoja įvardintą *Aerospike* duomenų bazės *namespace* su tam tikra konfigūracija. Aukščiau parodyta informacija yra iš */etc/aerospike/aerospike.conf* aplanko aplinkoje, naudotoje programos testavimui. Čia nurodyta bazinė *namespace* informacija, ją galima keisti/papildyti pritaikant atitinkamai platformai.

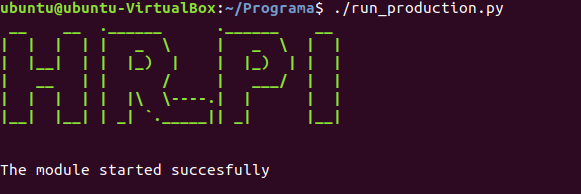
pav. Aerospike duomenų bazės *namespace* konfigūracija

Programinis kodas tinklo būsenai nustatyti *“sniffina”* paketus naudodamas *scapy* biblioteką, kuri įdiegiama kartu su kitais reikalingais paketais iš *requirements.txt* failo. Kad analizatorius veiktų teisingai, reikia nurodyti virtualią arba fizinę sąsają, iš kur bus imami paketai. Tai galima padaryti *scriptConfig.yaml* faile, nurodant atitinkamos sąsajos pavadinimą:

pav. Sąsajos konfigūracija

Analizatorius suteikia galimybę paleisti pasirinktą *skriptą,* kai tinklo būsena tam tikrą laiką signalizuoja ataką. Nustatymus, kada tiksliai vykdyti, kur *skriptai* yra kompiuteryje galite rasti *scriptConfig.yaml.*

pav. Skripto konfigūracija

Programos paleidimas vykdomas iš katalogo, kur yra programinis kodas, arba nurodant absoliutų kelią. Pavyzdys:

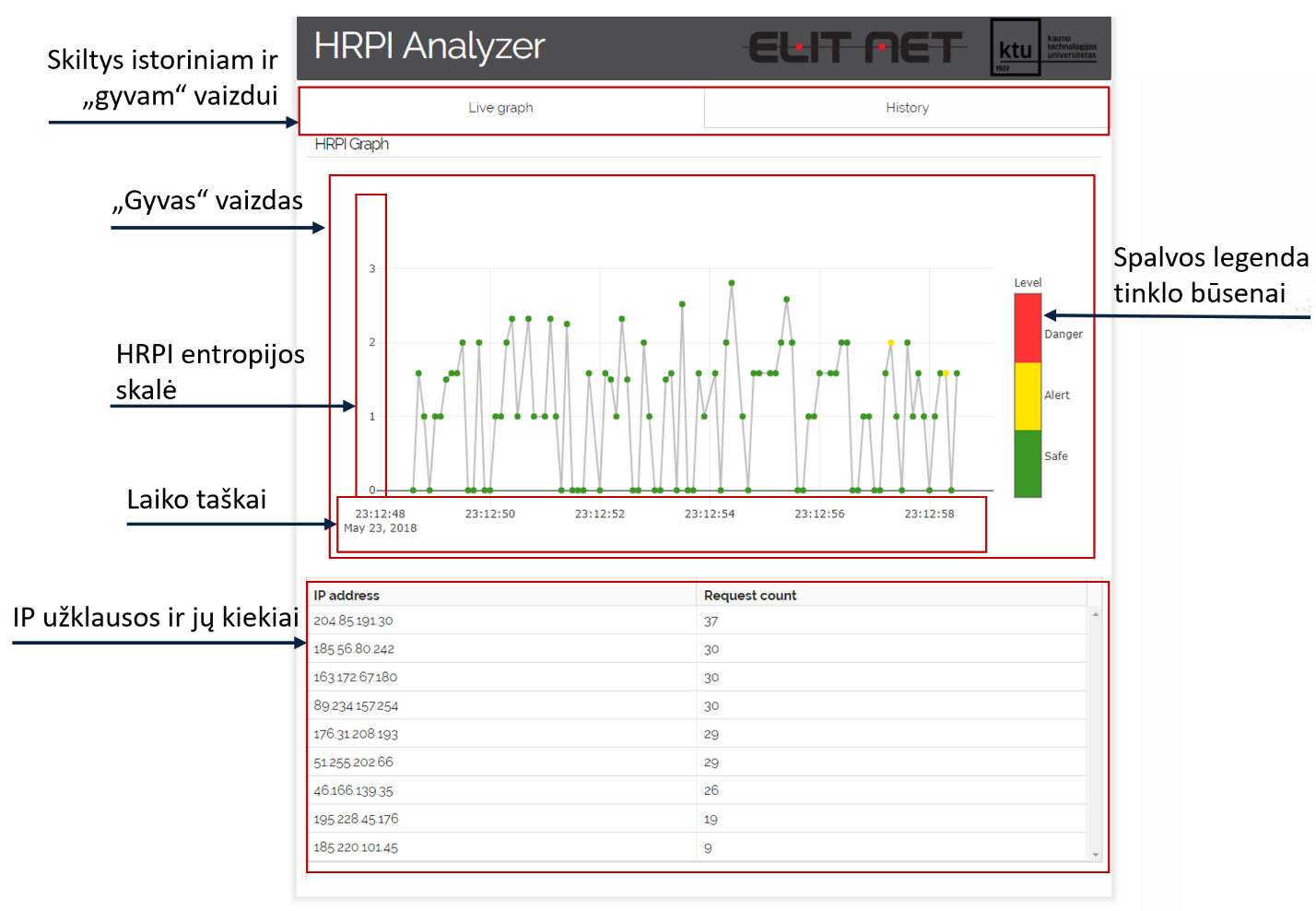
pav. Programos paleidimas

Informacija apie programos veikimą galima stebėti *info.log* bei *error.log* failuose, esančiuose programos pagrindinio scripto *run\_production.py* aplanke.

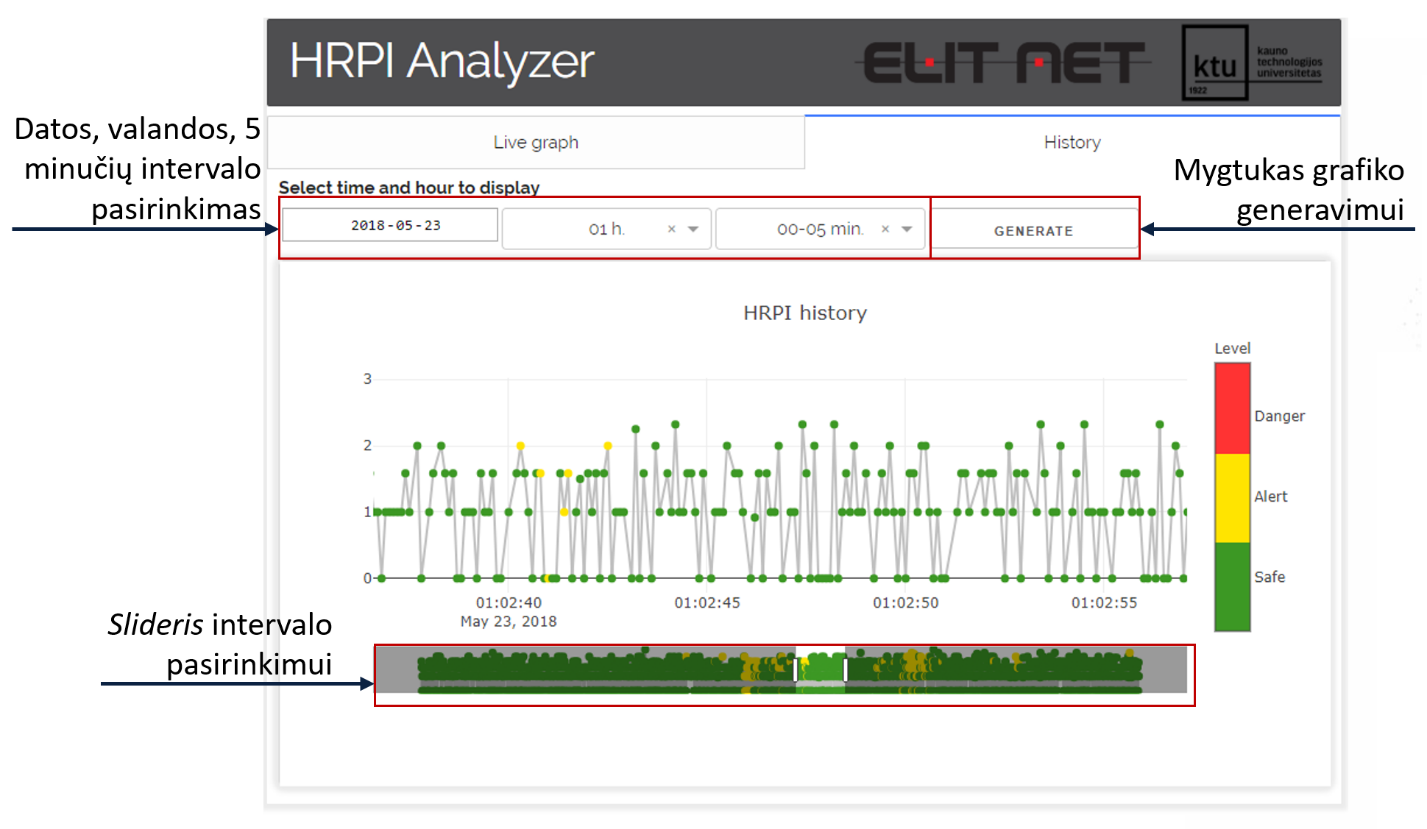
Vartotojo sąsaja diegimo nereikalauja, programa paleidžia GUI *web app* ant *localhost:3000*, ji gali būti pasiekiama per internetą atlikus *port forward*.

### Vartotojo sąsaja

Šio produkto vartotojo sąsaja yra tik papildomas programinės įrangos kūrimo proceso produktas, ji skirta patogiau stebėti, ar programa teisingai funkcionuoja realiu laiku. Vartotojo sąsają sudaro 2 sekcijos: realaus laiko grafikas, rodantis dabartines tinklo *traffic* charakteristikas, ir istorijos sekcija, rodanti pasirinkto intervalo informaciją.



17 pav. Programos GUI stebėjimo realiu laiku langas



pav. Programos GUI istorinių duomenų langas

# Rezultatų apibendrinimas ir išvados

Bendradarbiaujančios įmonės „Elitnet“ pasiūlytas ugniasienės komponento kūrimas iš pat pradžių pasirodė nelengva užduotis, o kūrimo stadijoje ši tendencija tik ryškėjo. Nepaisant to, metodas, aprašytas moksliniame straipsnyje, buvo įgyvendintas, pasiekti panašūs efektyvumo rezultatai.

Priklausomai nuo to, ką kūrė, komandos nariai susipažino su statistinės analizės metodais (*Kalman filter*), *machine learning* algoritmo *Support vector machine* specifika, nereliacine duomenų baze *Aerospike,* taip pat *dashboard* kūrimo karkasu *plot.ly*. Projekte buvo naudojama išskirtinai *Python* programavimo kalba dėl jos lankstumo bei galimybės patogiai dirbti su *machine* *learning* moduliais *linux* aplinkoje, todėl kiekvienas komandos narys stipriai patobulino minėtos kalbos žinias, geriau susipažino su Ubuntu OS. Bendra išvada tokia: kuriant matematiniais modeliais grįstas programas daugiausia laiko sugaištama bandant teisingai implementuoti aprašytus algoritmus. Taip atsitinka todėl, jog mokslininkai pateikia tik pseudokodus ir aprašymus, kurie reikalauja gilesnių matematikos ir statistikos žinių. Visi kiti reikalavimai buvo įgyvendinti įsisavinant karkaso, duomenų bazės projektavimo ir panaudojimo specifiką, tai buvo labiau esančių metodų pritaikymo programos specifikai uždavinys.

Programos inžinerijos procesų prasme, projektas ėjosi sklandžiai dėl pakankamai gerai suplanuotų sprintų bei darbo pasiskirstymo. Agile metodologija leido daryti kompromisus tarp iteracijų, nespėjant atlikti ar papildant užduotis bei reikalavimus projekto vykdymo metu. Dėl to, jog komandos nariai iš karto tiksliai identifikavo savo stipriąsias puses ir darbų pasiskirstymas nesukėlė didelių problemų, dauguma iteracijų praėjo sklandžiai, dauguma reikalavimų buvo įvykdyti.

Tai, jog įmonės programuotojai perėmė kai kuriuos sudarytus algoritmus, bei tai, kad tinklo būsenos (DDoS ir normalaus traffic) klasifikavimo teisingas pataikymų skaičius nepriklausomuose eksperimentuose buvo virš 90 %, laikome projekto sėkme.

# Literatūros sąrašas

|  |  |
| --- | --- |
| [1] | T. Ni, G. Xiaoqing, H. Wang ir L. Yu, „Real-Time Detection of Application-Layer DDoS Attack Using Time Series Analysis,“ *Journal of Control Science and Engineering,* t. vol. 2013, nr. Article ID 821315, p. 6, 2013. |

# Paveikslų sąrašas

1 pav. Abstrakti sistemos architektūra 5

2 pav. Komponentų schema 6

3 pav. Vieno laiko intervalo paketų rinkinio įvertinimo vykdymo sekos diagrama 7

4 pav. Įdiegimo rodinio schema 8

5 pav. Projekto valdymo įrankio iškarpa 10

6 pav. Sistemos architektūra 11

7 pav. Pirmo sprinto atliktų darbų sąrašas 12

8 pav. Antro sprinto atliktų darbų sąrašas 13

9 pav. „Burn-Down“ grafikas 14

10 pav. UML panaudos atvejų diagrama 15

11 pav. Paketų diegimas 20

12 pav. Prieigos prie duomenų bazės konfigūracija 21

13 pav. Aerospike duomenų bazės *namespace* konfigūracija 21

14 pav. Sąsajos konfigūracija 21

15 pav. Skripto konfigūracija 21

16 pav. Programos paleidimas 22

17 pav. Programos GUI stebėjimo realiu laiku langas 22

18 pav. Programos GUI istorinių duomenų langas 23

# Lentelių Sąrašas

[1 lentelė. Grafinės vartotojo sąsajos testavimo planas 18](#_Toc515406173)

[2 lentelė. Duomenų srauto testavimo planas 19](#_Toc515406174)

[3 lentelė. HRPI įvertinimo testavimo planas 19](#_Toc515406175)

[4 lentelė Sistemos efektyvumo testavimo planas 20](#_Toc515406176)

[5 lentelė Kodo įvertinimo planas 20](#_Toc515406177)

1. https://www.hindawi.com/journals/jcse/2013/821315/, Žiūrėta 2018 03 15 [↑](#footnote-ref-1)