



Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea

License Thesis

Absolvent

Alexandru Daniel Vid

Conducător

Your Supervisor's scientific title and name

Iulie 2018



TECHNICAL UNIVERSITY

OF CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

DEPARTAMENTUL CALCULATOARE

DECANUL FACULTĂȚII
Prof. dr. ing. Liviu MICLEA

DIRECTOR DEPARTAMENT
Prof. dr. ing. Rodica POTOLEA

Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea

Lucrare de licență

1. Absolvent: Alexandru Daniel Vid
2. Conducător: Your Supervisor's scientific title and name
3. Conținutul lucrării: Pagina de prezentare, aprecierile coordonatorului, titlul capitolului 1, titlul capitolului 2, ..., titlul capitolului n, bibliografie, anexe, CD.
4. Locul documentării: UTCN, Cluj-Napoca
5. Consultanți: Donald Knuth, Leslie Lamport, others ...
6. Data emiterii temei:
7. Data predării:

Semnătură Conducător
Your Supervisor's scientific title and name

Semnătură Absolvent
Alexandru Daniel Vid

Iulie 2018



Declarație pe proprie răspundere privind autenticitatea lucrării de licență

Subsemnatul *Alexandru Daniel Vid*, legitimat cu *CI* seria *XH* numărul *866549*, CNP *1950417055056*, autorul lucrării *Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea* elaborată în vederea susținerii examenului de finalizare a studiilor de masterat la Facultatea de Automatică și Calculatoare, Departamentul Calculatoare, Specializarea *Calculatoare* din cadrul Universității Tehnice din Cluj-Napoca, sesiunea *Iulie* a anului universitar *20XX/20XX*, declar pe proprie răspundere, că această lucrare este rezultatul propriei mele activități intelectuale, pe baza cercetărilor mele și pe baza informațiilor obținute din surse care au fost citate în textul lucrării și în bibliografie.

Declar că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în fața unei alte comisii de examen de licență sau disertație.

În cazul constatării ulterioare a unor declarații false, voi suporta sancțiunile administrative, respectiv, *anularea examenului de licență*.

Cluj-Napoca
data

Semnătură
Absolvent

Rezumat

Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea încapsulează trăsăturile normale ale unui reverse proxy oferind în plus protecție împotriva posibilelor atacuri la nivel de rețea. Produsul oferă protecție prin implementarea a două tipuri de prevenire a atacurilor: listă neagră pentru adresele IP și blocarea unor request-uri pe baza URL-urilor prin analiza conținutului lor. Pentru demonstrarea fiecărei metode în parte, s-a folosit: blocarea adreselor IP utilizate de rețeaua Tor pentru lista de IP-uri "negre" și detectarea atacurilor de SQL injection pentru protecția împotriva URL-urilor cu conținut malițios. Detectia atacurilor de tipul SQL injection se realizează prin analiza URI-urilor trimise de către clienți, în relație cu un model antrenat anterior folosind machine learning. Ambele implementări asigură adăugarea cu ușurință de noi detecții, lista de ip-uri blocate fiind accesibila utilizatorului atât pentru vizualizare cât și pentru editare.

Cuprins

Listă de figuri	iii
1 Introducere	1
1.1 Context	1
1.2 Motivație	2
1.3 Structura lucrării	3
2 Obiective și specificații	5
2.1 Obiective	5
2.2 Specificații	6
2.2.1 Specificații funcționale	7
2.2.2 Specificații non-funcționale	8
3 Studiu bibliografic	9
3.1 Abordări similare	9
3.2 Tehnici/Tehnologii/Surse folosite	14
4 Fundamente teoretice	17
4.1 Reverse proxy	17
4.2 Support vector machine	18
4.3 SQL injection	19
4.4 Adresele IP ale rețelei Tor	20
4.5 Sistem de prevenire a intruziunilor	21
5 Analiză și proiectare	25
5.1 Cerintele sistemului	25
5.2 Specificatiile cazurilor de utilizare	27
5.2.1 Actori, stakeholders si interese	27
5.2.2 Basic flow	27
5.2.3 Alternative flow	28
5.3 Arhitectura sistemului si modulele principale	29
5.4 Comportamentul sistemului. Diagrama de stare si de secventa	31
5.5 Dependintele sistemului	33

5.6	Algoritmi si metode	33
5.7	Diagrama de desfasurare	34
5.8	Justificarea design-ului	35
6	Detalii de implementare	37
6.1	Structura codului sursa	37
6.2	Algoritmi, metode si API-uri	42
6.2.1	Tor activity monitor	42
6.2.2	SQLi SVM	44
6.2.3	Interfata utilizator	46
7	Teste și rezultate experimentale	51
7.1	Teste de funcționalitate	51
7.2	Teste de performanță	53
8	Manual utilizator	55
8.1	Instalarea proiectului	55
8.2	Utilizare	55
9	Conluzii	59
9.1	Privire de ansamblu asupra sistemului	59
9.2	Dezvoltari ulterioare	59
	Bibliografie	61
A	Diverse anexe	65
B	Demonstrații matematice detaliate (dacă există)	67
C	Pseudo-cod sau cod (dacă există)	69
D	Articole publicate	71

Listă de figuri

1.1	Diagrama reverse proxy	1
2.1	Cutia neagra a sistemului	7
3.1	Administrarea securitatii unei aplicatii	10
3.2	Tipuri de trasaturi ale limbajului SQL	13
3.3	Arhitectura unui sistem de clasificare a request-urilor HTTP	15
4.1	Folosirea unui reverse proxy in arhitectura unei aplicatii.	17
4.2	Influentele aduse algoritmului de modifiacrea parametrului sigma in algoritmul de antrenare.	19
4.3	Exemplu de atac realizat prin SQL injection.	20
4.4	Exemplu de trafic realizat prin reseaua Tor.	21
4.5	Integrarea unui sistem de prevenire a intruziunilor intr-o retea.	22
5.1	Basic flow pentru evenimente	27
5.2	Principalele module ale sitemului propus	29
5.3	Diagrama de stare a sistemului propus.	31
5.4	Diagrama de secventa a sitemului propus.	32
5.5	Diagrama de desfasurare a sitemului propus.	34
6.1	Modulul pentru monitorizatea activitatii retelei Tor	37
6.2	Modulul pentru prevenirea atacurilor SQL injection	39
6.3	Interactiunea dintre interfata grafica si celelalte module	41
6.4	Meniul "Home" in interfata grafica	48
6.5	Meniul "Configure" in interfata grafica	48
6.6	Meniul "Monitor" in interfata grafica	49
6.7	Raspunsul pentru SQL injection URL	49
6.8	Principalele module ale sitemului propus	50
7.1	Raportul dintre numarul IP-urilor de pe Blacklist si Whitelist dintr-o luna	52
7.2	Raportul dintre uptime-ul IP-urile de pe Blacklist si Whitelist din aceasi luna	52
8.1	Meniul "Home" in interfata grafica	55

8.2	Meniul "Configure" in interfata grafica	56
8.3	Meniul "Monitor" in interfata grafica	57

Capitolul 1

Introducere

1.1 Context

În general, tentativele de exploatarea vulnerabilităților unei aplicații vin sub formă de input către aceasta, input generat de către un atacator care intenționează să întrerupă activitatea sau să preia controlul aplicației. Un sistem de prevenire a intruziunilor (IPS) are rolul de a sta între aplicație și clienții acesteia, și de a preveni exploatarea unor astfel de vulnerabilități.

Prin folosirea unui reverse proxy pentru accesarea resurselor unui server de către clienți, poate să se aducă numeroase beneficii procesului de administrare a serverului [1]. Spre deosebire de un forward proxy care e un intermediar pentru un o serie de clienți prestabiliți, pentru a accesa orice server, un reverse proxy e un intermediar pentru o serie de servere prestabilite pentru a fi accesate de orice client. Unul dintre avantajele folosirii unui reverse proxy este centralizarea întregului trafic al serverului/serverelor într-un singur punct de acces, aceasta fiind și principala caracteristică exploatată de acest proiect pentru filtrarea IP-urilor nedorite (în cazul nostru cele utilizate de rețeaua Tor) și verificarea conținutului URI-urilor pentru posibile atacuri (SQL injection).

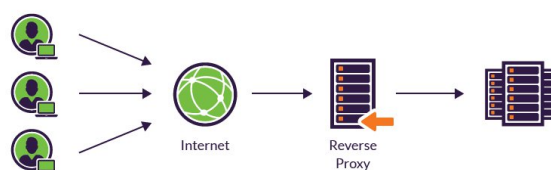


Figura 1.1: Diagrama reverse proxy

Figura 1.1 ilustrează modul de funcționare al unui reverse proxy în relație cu serverele aferente și posibili clienți.

Conform topului alcătuit de fundația OWASP cu cele mai mari 10 riscuri ale aplicațiilor în 2017 [2], SQL injection e considerat a fi cea mai mare vulnerabilitate a aplicațiilor web. Acest lucru se datorează faptului că mare parte din aceste aplicații se bazează pe procesarea conținutului furnizat de către utilizatori. Atacurile de tipul SQL injection constau în faptul că datele furnizate de către utilizator sunt introduse în interogări SQL, unde acestea sunt tratate ca și cod executabil [3]. Aplicațiile web vulnerabile la sql injection pot permite unor utilizatori neautorizați să facă interogări într-o bază de date asupra unor date la care nu ar avea acces în mod normal. Folosind acest tip de comportament neautorizat, un astfel de utilizator poate să obțină accesul la informații sensibile ale clienților, dar și a administratorilor aplicației, precum credentiale sau date personale. Această vulnerabilitate poate să ducă la furt de identitate sau fraudă.

În cazul rețelei Tor, aceasta le permite utilizatorilor să navigheze pe internet anonim. Anonimitatea online este importantă însă în multe cazuri aplicațiile web trebuie să știe cine se conectează la aceasta pentru a le putea determina intențiile. Numele de Tor vine de la "the onion router" care sugerează modul de operare al rețelei. Fiecare participant la rețea devine un nod de transfer, iar traficul rețelei traversează o serie de astfel noduri până să ajungă la nodul de ieșire ce crează conexiunea cu destinația dorită. Pachetele sunt criptate în mai multe "straturi", fiecare nod decriptând un singur strat de unde poate afla doar destinația nodului următor. Când pachetul ajunge la ultimul nod, acesta trimite conținutul la destinație fără să dezvăluie identitatea sursei. Această anonimitate usează desfășurarea atacurilor online. Conform datelor din rețeaua organizației CloudFlare 94% din traficul provenit din rețeaua Tor este alcătuit din request-uri automate de origine malițioasă [4].

1.2 Motivație

În piața actuală există multe sisteme de prevenire a intruziunilor ce oferă atât caracteristicile unui reverse proxy, cât și cele de securitate. Aceste caracteristici sunt oferite fie ca și produse individuale, fie ca și produse ce le încorporează pe ambele. Cu toate acestea, produsele de acest gen sunt în general scumpe, au o logică mascată de detectarea a posibilelor probleme de securitate și sunt greu de înțeles și de configurat de către utilizator după propriile nevoi.

Prin oferirea utilizatorilor posibilitatea de a înțelege și modifica modul de funcționare a unui astfel de sistem poate rezulta în sisteme mult mai eficiente și rapide, dedicate pentru preferințele și nevoile aplicației fiecărui utilizator în parte. Spre exemplu, un utilizator poate să decidă că nu are nevoie de funcționalitățile de detecție împotriva atacurilor de tip SQL injection pentru o anumită aplicație, întrucât aceasta nu prezintă vulnerabilități din acest punct de vedere, nefolosind o arhitectură bazată pe baze de date. În cazul acesta prin eliminarea unui astfel de modul, se elimină și verificările aferente asupra request-urilor clienților, îmbunătățind astfel performanțele sistemului.

Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea oferă

un sistem configurabil după preferințele utilizatorilor. Utilizatorul poate configura detecția bazată pe analiză request-ului primit de la client, acesta poate să aleagă care module să fie folosite pentru detecție, permițând și eventuala adăugare de noi module(cât timp acestea respectă anumite condiții de structură), meniurile din interfața utilizator fiind generate în mod dinamic în funcție de modelele prezente. Utilizator poate, de asemenea să configureze și lista ip-urilor blocate, permițându-i-se să șteargă din cele existente, respectiv să aduge unele noi, după bunul plac.

1.3 Structura lucrării

În această secțiune se prezintă structura lucrării pe capitole și o scurtă descriere a conținutului acestora:

Primul capitol 1, prezintă o scurtă introducere despre proiect, contextul acestuia și motivația pentru implementarea sistemului propus.

Capitolul 2 prezintă obiectivele lucrării, specificațiile sistemului, motivând deciziile luate în implementarea sistemului, cerințele funcționale și nonfunctionale necesare implementării sistemului.

Capitolul 3 descrie alte abordări similare ale problemelor tratate de proiectul propus, prin evidențierea asemănărilor și diferențelor dintre acestea și se explică tehnologiile și metodele folosite de proiect.

În capitolul 4 sunt evidențiate și explicate pe scurt aspectele teoretice pe care se bazează proiectul.

Capitolul 5 descrie design-ul proiectului și cuprinde: cerințele sistemului, specificațiile cazurilor de utilizare, arhitectura sistemului, comportamentul sistemului, datele utilizate de sistem, dependențele sistemului, algoritmi esențiali și metodele folosite. Descrierea acestora se realizează prin asocierea cu diagramele aferente.

Capitolul 6 prezintă în amănunt detaliile de implementarea a proiectului. În acest capitol sunt abordate: organizarea codului sursă, descrierea principalelor clase și funcționalități ale proiectului și posibilitățile de desfășurare a execuției programului.

Capitolul 7 reprezintă metodele de validare a soluțiilor/sistemului descris în capitolele anterioare, scenariile de testare a corectitudinii funcționale, a utilizabilității și performanței.

Capitolul 8 descrie pașii de instalare și rulare a aplicației.

Capitolul 9 cuprinde ce s-a realizat, relativ la ce s-a propus, în ce constă experiența acumulată, care au fost punctele dificile atinse și o descriere a posibilelor dezvoltări și îmbunătățiri ulterioare.

Capitolul 2

Obiective și specificații

Acest capitol prezintă obiectivele lucrării, motivând deciziile luate în implementarea sistemului, specificațiile sistemului, cerințele funcționale și nonfuncționale necesare implementării sistemului.

2.1 Obiective

Elaborarea unor măsuri de securitate împotriva anumitor tipuri de atacuri sau dezvoltarea unei logici de filtrare a clienților serviți de către aplicație este posibilă și în partea de implementare a serverului, cea ce ar putea oferi și o eventuală creștere în performanțele aplicației, eliminând astfel posibile interceptări suplimentare și validări ale traficului. Însă o astfel de implementare presupune o arhitectură mult mai complexă pentru partea de server și individual cunoștințe avansate despre securitate, posibilele vulnerabilități la care acesta poate să fie predispus, precum și modalități de combatere ale acestora.

O soluție mult mai simplă la această problemă este folosirea unui modul care să implementeze aceste funcționalități separat. Mare parte din produsele de pe piață, ce îndeplinesc aceste caracteristici sunt destul de scumpe și nu oferă foarte multă libertate din punctul de vedere al configurării securității dorite de către utilizator asupra produsului sau. În cazul IP-urilor blocate, multe aplicații nu oferă libertatea utilizatorilor de a edita listele de referință ale detecțiilor, acestea fiind actualizate automat conform unor date interne, iar eventualele abateri de la aceste reguli se realizează prin adăugarea de excepții. În cea ce privește validarea request-urilor primite de la clienți, mare parte din aceste sisteme nu oferă o protecție configurabilă. Cum sa precizat mai sus, acest tip de sisteme pot să introducă mici întârzieri datorate validărilor suplimentare asupra request-urilor primite de la clienții produsului, însă în unele cazuri anumite validări sunt făcute inutil întrucât produsele respective neputând să prezinte vulnerabilități de acel fel.

Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea are obiectivul de a oferi o componentă gratuită cât mai ușor de integrat și de configurat de

către utilizator după preferințele produsului sau, care să faciliteze o protecție cât mai eficientă cu suport pentru îmbunătățiri. Sistemul trebuie să fie ușor de instalat și de configurat, oferindu-i utilizatorului o interfață clară, sugestivă și ușor de folosit prin care să interacționeze cu acesta. Detecțiile sistemului trebuie să fie selectabile, utilizator putând alege în momentul pornirii sistemului ce vulnerabilități să fie tratate de acesta. Lista IP-urilor blocate trebuie să fie ușor de vizualizat și editabilă, permițând utilizatorului să își impună cu ușurință propriile reguli asupra modului de funcționare a sistemului. Pentru realizarea detecției împotriva atacurilor de tipul SQL injection se impune prelucrarea unor date reale pentru antrenarea modelului de machine learning. Prin folosirea unor date provenite din atacuri reușite sau tentative de atacuri reale, se poate crea o precizie mult mai bună pentru o clasificarea cât mai precisă a posibilelor atacuri. Sistemul trebuie de asemenea să fie construit modular pentru a permite realizarea de modificări cu ușurință, iar încărcarea detecțiilor realizată dinamic, permițând astfel că adăugarea de noi detecții să fie cât mai simplă.

2.2 Specificații

Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea trebuie să fie capabil să servească ca și un reverse proxy pentru un server, să blocheze atacurile de tip SQL injection asupra lui și să nu permită conectarea clienților cu IP-uri continute în lista neagră.

Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea va procura utilizatorului o interfață grafică prietenoasă, ușor de folosit, prin intermediul căreia, acesta va putea să seteze mediul de rulare al sistemului. Interfața va permite setarea specificațiilor server-ului, adresa și portul pe care acesta acceptă conexiuni, dar și a interfețelor prin intermediul cărora se pot realiza conexiuni la server.

În interfața grafică se vor afișa și eventualele detecții realizate de produs. Într-o fereastră separată utilizatorul trebuie să aibă posibilitatea să vizualizeze toate deciziile sistemului și motivele din spatele deciziilor, permițând astfel acestuia să înțeleagă modul de funcționare, respectiv să raporteze sau să modifice sistemul (în cazurile în care i se oferă această posibilitate) când comportamentul acestuia nu se află în conformitate cu nevoile sau cerințele sale.

În momentul configurării modului de rulare al sistemului, utilizatorul trebuie să aibă și posibilitatea de a impune ce module de securitate să fie folosite de acesta în timpul rulării. Pentru a eficientiza cât mai mult sistemul, utilizatorul poate să aleagă care sunt modulele de interes pentru propria aplicare, evitând astfel validarea unor evenimente ce nu prezintă interes pentru acesta.

În timpul rulării sistemul va asculta pe interfețele setate de către utilizator pentru posibile cereri de conexiuni la server-ul setat. În funcție de modulele alese în momentul pornirii, acesta va verifica sau nu adresa clientului validând astfel conexiunea. În cazul în care adresa clientului se află pe lista neagră de adrese, conexiunea acestuia este refuzată,

iar aplicația înregistrează această decizie în fereastra de evenimente vizibilă utilizatorului. După realizarea conexiunii la server, fiecare request trimis de clienți către acesta va fi evaluat conform modulelor configurate. Dacă requesturile sunt considerate că fiind "curate" acestea sunt trimise mai departe la server. În caz contrar, clientului i se întoarce un cod de eroare, iar requestul nu va mai fi trimis mai departe către server, de asemenea înregistrând evenimentul în fereastra de evenimente vizibilă utilizatorului.

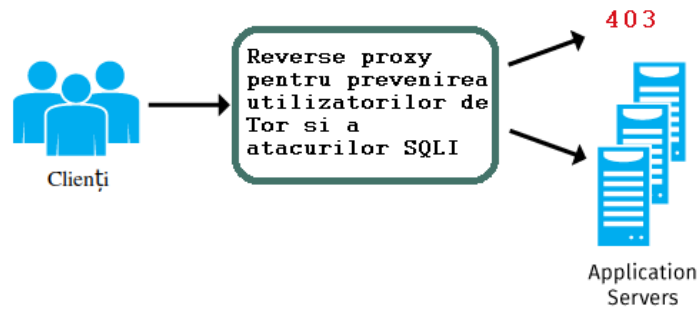


Figura 2.1: Cutia neagra a sistemului

Figura 2.1 prezinta cutia neagra a sistemului propus.

2.2.1 Specificații funcționale

Sistemul trebuie să prezinte o interfață grafică ușor de folosit de către utilizator și să fie capabil să redirectioneze traficul interceptat către un anumit server, clasificând și filtrând traficul malițios. Pentru a atinge obiectivele proiectului, următoarele cerințe funcționale trebuie îndeplinite:

- Să realizeze conexiunea la un server HTTP/HTTPS și să redirectioneze traficul primit către acesta.
- Să intercepteze traficul venit pe o anumită interfață și port prestabilit.
- Să prelucreze request-urile primite de la clienți într-un format specific clasificatorului de SQL injection.
- Să nu redirectioneze request-urile clasificate ca și SQL injection.
- Să blocheze conectarea clienților ce folosesc IP-uri clasificate ca IP-uri de Tor.
- Să permită utilizatorului să editeze și să vizualizeze lista IP-urilor de Tor.
- Să prezinte în interfața grafică toate intervențiile realizate asupra traficului (blocări de conexiuni sau de request-uri).

- Să permită utilizatorului să configureze modul de operare al sistemului.

2.2.2 Specificații non-funcționale

Sistemul trebuie, de asemenea, să aibă următoarele caracteristici non-funcționale pentru a realiza obiectivele specificate:

- Să fie ușor de instalat și de folosit pentru orice utilizator, oricât de neexperimentat .
- Să poată intercepta traficul de pe orice/oricâte interfețe disponibile.
- Să poată rula pe orice sistem de operare Windows cu Python2 instalat.
- Să aibă o rată de blocare de 100% a IP-urilor de pe lista neagră, iar în cazul detecției de SQL injection să nu aibă detecții false pozitive mai mari 2-3% și o acuratețe generală de peste 90%.

Capitolul 3

Studiu bibliografic

În acest capitol sunt prezentate alte abordări similare ale problemelor tratate de proiectul propus, prin evidențierea asemănărilor și diferențelor dintre acestea și se explică tehnologiile și metodele folosite de proiect.

3.1 Abordări similare

Precum Richard Bassett, Cesar Urrutia și Nick Ierace susțin în articolul **Intrusion prevention systems** [5] ”sistemele de prevenire a intruziunilor sunt o componentă importantă a sistemelor de protecție IT, iar fără această tehnologie, datele noastre și rețelele ar fi mult mai predispuse activităților malițioase”.

În general tentativele de exploatare a vulnerabilităților unei aplicații vin sub formă de input către o aplicația țintă. Acest input fiind generat de către un atacator ce intenționează o controleze sau să îi întrerupă activitatea. În cazul unui atac reușit, un astfel de atacator poate să dezactiveze temporar aplicația (atacuri de tipul denial of service) sau poate accesa, altera sau executa date/cod în interiorul aplicației. Un sistem de prevenire a intruziunilor are rolul de a examina traficul destinat unei aplicații și de intercepta și bloca astfel de tentative [6].

Un sistem de prevenire a intruziunilor este de regulă folosit alături de un sistem firewall respectiv alături de un sistem de detectare a intruziunilor. Deși au scopuri asemănătoare, aceste sisteme au funcționalități diferite și rezolvă diferite probleme de securitate. Un sistem de prevenție a instrucțiunilor este cel mai bine comprat cu sistemele de tip firewall. Un sistem firewall tipic este constituit dintr-o serie de reguli ce permit traficului să treacă. Aceste reguli sunt sub forma ”dacă traficul îndeplinește anumite condiții poate trece”, însă dacă nu există nici o regulă care să potrivească anumite pachete, acestea sunt blocate. Asemeni sistemelor firewall, sistemele de prevenire a intruziunilor prezintă un set de reguli de filtrare a pachetelor, request-urilor sau a clienților, însă aceste reguli sunt de regulă reguli de blocare. Astfel, dacă un anumit pachet nu potrivește nici o regulă sistemul de prevenire a intruziunilor îl lasă să treacă [7].

Spre deosebire de sistemele de tip firewall sau cele de prevenire a intruziunilor, care oferă control utilizatorului asupra traficului ce trece prin rețea, sistemele de detecție a intruziunilor permite acestuia să vizualizeze evenimentele din rețea. Precum și Joel Snyder susține în articolul **Do you need an IDS or IPS, or both?** [7] sistemele de detecție a intruziunilor oferă unui utilizator facilități asemănătoare unui analizator de pachete [8], însă din perspectiva de securitate a rețelei. Aceste informații furnizate de către sistem îi permit utilizatorului să decopere:

- Încălări ale politicilor de securitate, precum utilizatori sau sisteme ce desfășoară activități ce încalcă politicile prestabilite.
- Posibile sisteme infectate ce folosesc rețeaua pentru a se răspândi sau să atace alte sisteme.
- Scurgeri de informație cauzate de infectarea unor sisteme cu malwarei sau de utilizatori rău intenționați.
- Erori de configurare în sisteme sau aplicații cu setări de securitate incorecte sau configurări proaste ce consumă prea multă bandă de rețea.
- Detectarea unor clienți sau servere ce accesează sau sunt accesate în mod neautorizat, respectiv aplicații malițioase ce fac asta.

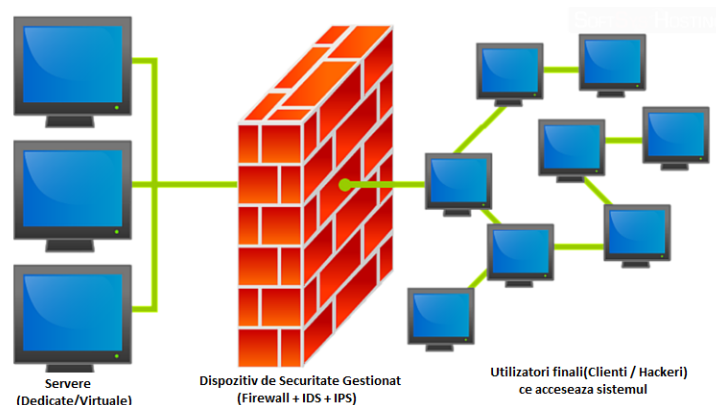


Figura 3.1: Administrarea securitatii unei aplicatii

Figura 3.1 prezintă administrarea securității unei aplicații folosind combinația dintre cele trei sisteme.

În comparație cu sistemele de detecție a intruziunilor care sunt sisteme pasive și scanează rețeaua fără să interfereze cu traficul, sistemele de prevenire a intruziunilor sunt plasate între server și cilenți, alterând în mod automat traficul în cazul în care acesta

declanșează una din regulile prezente în sistem. Precum sunt prezentate și în articolul **What is an intrusion prevention system?** [6], printre funcționalitățile unui sistem de prevenire a intruziunilor se număra:

- Notificarea unui administrator de rețea în cazul în care una sau mai multe reguli sunt declanșate.
- Oprirea pachetelor considerate malițioase pentru rețea.
- Blocarea unor utilizatori prin excluderea adreselor IP ale acestora.
- Resetarea unor conexiuni.

În ceea ce privește funcționalitățile oferite de un sistem de prevenire a intruziunilor, acestea sunt specifice tipului sistemului. Conform autorului articolului **Intrusion Prevention System (IPS): Definition & Types** [9], Beth Hendricks, există patru tipuri primare de astfel de sisteme:

- Network-based: Protejează întreaga rețea.
- Wireless: Protejează doar rețeaua wireless.
- Network behavior: Examinează traficul din rețea.
- Host-based: Software cu scopul de a proteja un singur calculator.

Sistemele de tipul Network-based (reprezentând și categoria în care se încadrează *Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea*) presupun implementarea unor senzori în rețea care capturează și analizează traficul ce trece prin acesta. Acești senzori sunt plasați în puncte cheie a rețelei pentru a putea captura în timp real traficul, iar în cazul interceptării unor activități malițioase să poată interveni imediat, fără să scadă performanța rețelei. Aceste sisteme oferă protecție rețelei indiferent de dimensiunile sau creșterea acesteia, adăugarea de noi device-uri fiind posibilă fără să necesite adăugarea de noi senzori. Adăugarea de noi senzori fiind nevoie doar în cazul în care traficul rețelei depășește capacitatea de procesare a senzorilor curenți, influențând astfel performanțele rețelei [10].

În funcție de nevoi, un sistem de prevenire a intruziunilor poate să ofere diferite opțiuni de protecție pentru diferite părți ale rețelei. Unele sunt capabile să oprească traficul malițios sau să limiteze lățimea de bandă către anumite părți ale rețelei. Conform [11] aceste sisteme pot oferi protecție împotriva următoarelor tipuri de atacuri:

- **ICMP Storms:** un volum mare de ecouri ICMP pot să indice activități malițioase precum cineva ce scanează rețeaua.
- **Ping to Death:** un utilizator poate să modifice comandă de ping, astfel încât să trimită un număr mare de pachete de dimensiune mare către o destinație țintă pentru a o scoate din uz.

- **SSL Evasion:** unele atacuri se pot folosi de criptarea SSL pentru a evita dispozitivele de securitate, întrucât în general acestea nu sunt decriptate.
- **IP Fragmentation:** constă în exploatarea faptului că pachetele sunt descompuse în fragmente pentru a satisface cerințele de dimensiune a rețelelor traversate, inundând o destinație țintă cu fragmente false pentru a îi consuma resursele.
- **SMTP mass mailing attacks:** un sistem infectat poate să se folosească de email-ul utilizatorului pentru a se răspândi, rezultând într-un trafic mare destinat serverului de mail.
- **DoS/DDoS attacks:** cu scopul de a face o resursa indisponibilă utilizatorilor, este realizată prin inundarea sistemului țintă cu un număr mare request-uri de la unul sau mai multe (în cazul DoS distribuit-DDoS) sisteme malițioase.
- **SYN Flood attacks:** atacatorul trimite un număr mare de pachete de inițiere a unei conexiuni fără să mai răspundă ulterior, epuizând astfel resursele de memorie.
- **Http obfuscation:** pentru a evita să fie detectate de anumite sisteme de protecție, unele atacuri folosesc tehnici de ofuscarea a request-urilor HTTP.
- **Port Scanning:** este folosit pentru descoperi ce porturi sunt deschise pe un sistem, ulterior permițându-i atacatorului să știe ce vulnerabilități ar putea prezenta sistemul.
- **ARP Spoofing:** un atacator trimite în rețea pachete false de ARP legându-și propria adresă MAC de adresa IP a unui alt sistem. Ca urmare, atacatorul va primi pachete destinate sistemului cu adresa IP folosită în pachetul de ARP.
- **CGI Attacks:** un atacator poate să trimită request-uri malițioase, determinând destinația să trateze request-ul primit ca și cod executabil, oferindu-i atacatorului acces pe sistem.
- **Buffer Overflow attacks:** presupune ca atacatorul să depășească limitele unui buffer de lungime fixă, excesul de date ajungând să suprascrie zone adiacente de memorie rezultând în căderea sistemului sau dându-i atacatorului oportunitatea să ruleze cod propriu.
- **OS Fingerprinting attacks:** presupune ca atacatorul să descopere ce sistem de operare rulează pe un sistem și folosindu-se de această informație să exploateze vulnerabilități specifice acelui sistem de operare.

Sistemul propus, *Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea* implementează un sistem de prevenire a intruziunilor folosindu-se de un reverse proxy pentru a intercepta tot traficul care intră și iese din rețea (reprezentând

senzorii ce au rol de a captura și analiza traficul) și oferind protecție împotriva a două categorii de atacuri: orpirea de URL-urilor malițioase destinate unei aplicații (protecție împotriva SQL injection) și blocarea adreselor IP cunoscute ca fiind folosite de utilizatori rău intenționați (protecție împotriva adreselor IP ale rețelei Tor).

Pentru prevenirea atacurilor de SQL injection, se folosește o metodă asemănătoare celei propuse de Eun Hong Cheon, Zhongyue Huang și Yon Sik Lee în lucrarea **Preventing SQL Injection Attack Based on Machine Learning** [12]. Pentru clasificarea request-urilor HTTP în SQL injection sau curate, se folosește un sistem bazat pe machine learning. Acest sistem este antrenat anterior cu date reale, ca și trăsături fundamentale în clasificare, folosindu-se cuvintele cheie și simbolurile specifice limbajului SQL (spre exemplu: SELECT, ADD, DELETE, ", ' etc.).

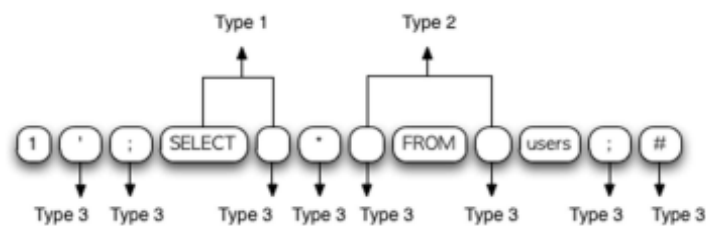


Figura 3.2: Tipuri de trăsături ale limbajului SQL

Figura 3.2 prezintă tipurile de trăsături luate în considerare în lucrarea **Preventing SQL Injection Attack Based on Machine Learning** în raport cu simbolurile sau cuvintele cheie folosite.

O altă abordare pentru prevenirea atacurilor SQLI este propusă de Fredrik Valeur, Darren Mutz și Giovanni Vigna în lucrarea **A Learning-Based Approach to the Detection of SQL Attacks** [13]. În această lucrare se prezintă folosirea unui sistem bazat pe detecția de anomalii pentru detectarea atacurilor ce exploatează o aplicație pentru a îi compromite baza de date. Asemeni abordării bazate pe machine learning, acest sistem presupune o fază anterioară de antrenare în care se învață comportamentul normal al utilizatorilor, alcătuind astfel niște profile specifice. Astfel în faza de detecție, comportamentul ce nu coincide cu profilele alcătuite în faza de antrenare, este considerat malițios.

Pentru prevenirea utilizatorilor de Tor, în general lista de IP-uri este alcătuită din toate IP-urile care au utilizat rețeaua într-un anumit interval de timp, aceasta fiind actualizată periodic. O astfel de abordare este folosită și în cazul marelui firewall al Chinei [14] care indentifică nodurile la prima accesare a rețelei de Tor. Însă precum precum se evidențiază și în articolul **Characterizing the Nature and Dynamics of Tor Exit Blocking** [15] o astfel de abordare nu este cinstită față de unii utilizatori de Tor, întrucât reputația acestora este împărțită între toți utilizatorii. Astfel un nod care este utilizat doar pentru câteva minute sau ore (probabil din motive de curiozitate) poate să ajungă să fie blocat, fiind tratat asemeni cu un nod ce functionază de câteva zile. O astfel de discriminare a încercat să fie evitată prin implementarea aleasă a sistemului propus. Pentru

realizarea listei de IP-uri blocate se folosește un algoritm ce stabilește o limită de timp minima de funcționare pentru un anumit nod în intervalul a 30 de zile.

3.2 Tehnici/Tehnologii/Surse folosite

Pentru realizarea sistemului propus s-au folosit două limbaje de programare: Python2/3(pentru partea de back end) și C#(pentru partea de front end). În partea de back end a proiectului se realizează implementarea unui reverse proxy pentru a intercepta traficul uneia sau mai multor interfețe, un modul pentru clasificarea request-urilor împotriva atacurilor SQL injection și un modul pentru generarea listei negre de IP-uri ce utilizează frecvent rețeaua Tor. Toate aceste componente sunt realizate prin utilizarea de librării open-source pentru a ușura și eficientiza munca precum: twisted [16], beautiful soup [17], libsvm [18].

Motivul utilizării atât limbajului Python3 cât și Python2 este datorat diferențelor de module și librării open-source disponibile pentru cele două limbaje dar și a faptului că suportul pentru Python2 se încheie în anul 2020. Conform documentațiilor oficiale [19] și [20], dar și articolului *Python 2 to python 3: why, and how hard can it be?* de Tim Grey [21], între cele două versiuni nu sunt modificări majore, însă în anumite cazuri pot exista librării care să ofere doar suport pentru una dintre acestea.

În realizarea modulului pentru clasificarea request-urilor împotriva atacurilor SQL injection s-a folosit o colecție de date reale atât de atacuri cât și de trafic curat. Pentru uniformizarea acestor date și pentru a trata tentativele de păcălire a clasificatorului prin codarea unor caractere în valoarea lor în cod hexadecimal (exemplu 'https://www.google.ro/search?q=a' echivalent cu 'https://www.google.ro/search?q=%61') datele au fost preprocesate și transformate în întregime în coduri hexadecimale [22]. În procesarea datelor, pentru indentificarea trăsăturilor relevant, s-au indentificat caracterele specifice limbajului [23] și cuvintele cheie rezervare [24]. Ulterior, pentru antrenarea modelului de support vector machine și pentru clasificarea noilor request-uri s-a folosit software-ul open-source libsvm [25].

Figura 3.3 prezintă arhitectura unui sistem de clasificare a request-urilor HTTP de un sistem bazat pe machine learning. Structura este prezentată în lucrarea prezentată și anterior **Preventing SQL Injection Attack Based on Machine Learning** [12]. Acesta structură a reprezentat un model de pornire în realizarea modulului de prevenire a atacurilor SQL injection, implementarea modulului încercând să aducă îmbunătățiri de performanță prin modificarea algoritmului folosit pentru antrenarea modelului de support vector machine, dar și prin filtrarea trăsăturilor propus în lucrare în conformitate cu raportul dintre obișnuința de apariție a acestora atât în request-urile ce intenționează să execute un atac cât și în cele curate.

Pentru blocarea IP-urilor utilizate de rețeaua Tor s-a folosit un script scris în Python3. Programul interoghează periodic(din 6 în 6 ore) informațiile oferite de *Tor Network Status* [26] indentificand astfel nodurile cu un "Uptime" mai mare de 7 zile în parcursul

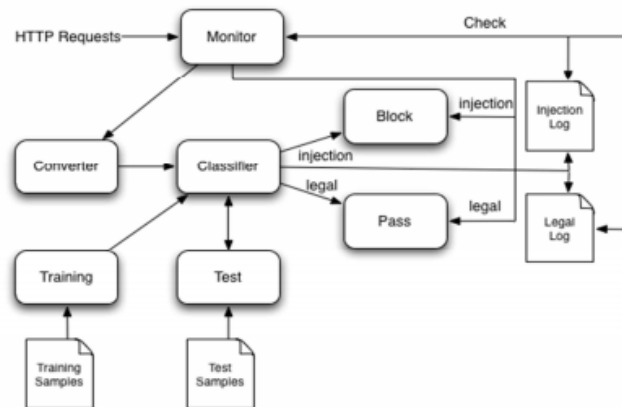


Figura 3.3: Arhitectura unui sistem de clasificare a request-urilor HTTP

unei luni. Blocarea IP-urilor se realizează prin compararea cu o astfel de listă generată lunar.

Componenta ce încorporează toate modulele de protecție, este cea de reverse proxy. Aici este monitorizat tot traficul ce vine de pe o anumită interfață (una sau mai multe, în funcție de configurația utilizatorului) și este trecut prin toate modulele disponibile pentru a verifica condițiile de securitate. Pentru testarea dacă o adresă IP este utilizată frecvent de rețeaua Tor, în momentul în care un client dorește să realizeze o conexiune la serverul protejat de sistem, adresa IP a acestuia este verificată să nu se afle pe lista IP-urilor blocate. Pentru actualitate, lista adreselor IP blocate este actualizată periodic cu adresele IP utilizate frecvent de rețeaua Tor în ultima luna. Modulul de prevenire a atacurilor SQL injection este integrat tot în componenta de reverse proxy, însă evaluarea request-urilor este făcută după realizarea conexiunii între client și server. Request-urile primite de către server sunt tratate asemănător celor folosite pentru antrenarea modului de support vector machine, însă pentru clasificarea acestora este folosit modulul antrenat în fază inițială și software-ul de prezicere oferit tot de libsvm [18].

Capitolul 4

Fundamente teoretice

În acest capitol sunt evidențiate și explicate pe scurt aspectele teoretice pe care se bazează proiectul.

4.1 Reverse proxy

Un reverse proxy este un server intermediar care trimite mai departe request-urile pentru conținut, de la mai mulți clienți nedefiniți, către unul sau mai multe servere. Un reverse proxy este un tip de proxy care în mod normal este situat în spatele unui firewall într-o rețea internă și redirectionează traficul clienților către serverele asociate. Acesta introduce un nivel în plus de abstractizare și control, asigurând controlul fluxului de trafic [27].

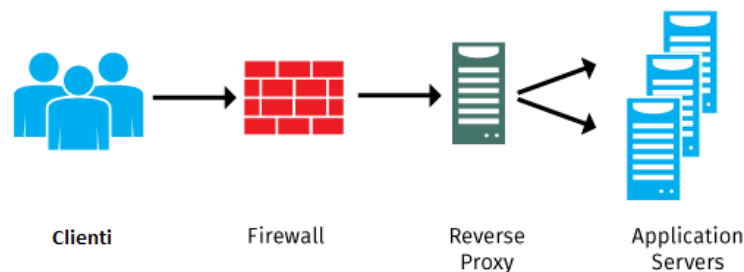


Figura 4.1: Folosirea unui reverse proxy în arhitectura unei aplicații.

Figura 4.2 prezintă modalitatea de integrare a unui reverse proxy în implementarea arhitecturii de back end a unei aplicații.

Cele mai obișnuite caracteristici ce pot fi oferite de utilizarea unui reverse proxy sunt:

- Load balancing - un reverse proxy poate să distribuie request-urile primite de la clienți, astfel încât nici un server să nu fie copleșit de requesturi. În cazul în care un

server este supraincarcat cu request-uri sau este cazut, acesta poate sa redirectioneze traficul catre alte servere functionale.

- Web acceleration - un reverse proxy poate sa realizeze compresia datelor sau sa memoreze in memoria cache continut ce este frecvent accesat sau poate sa realizeze operatiile de criptare SSL executate in mod normal de server, imbunatatind astfel in mod considerabil viteza de comunicare dintra client si serverul destinatie.
- Securitate si anonimitate - prin interceptarea request-urilor primite de catre server, acesta asigura anonimitatea serverului actionand ca un nivel extra de securitate. De asemenea se asigura ca mai multe servere pot fi accesate prin intermediul unui punct comun, indiferent de structura retelei interne.

4.2 Support vector machine

Algoritmul de machine learning, support vector machine reprezinta un model obtinut prin folosirea de diversi algoritmi pentru antrenarea acestuia, folosit pentru a clasifica date. Acest model intra in categoria de invatare supravegheata('supervised learning'), intrucat pentru obtinerea lui se foloseste un set de date ca si exemplu, date pe care modelul le va folosi ca referinta pentru clasificarea noilor evenimente.

Realizarea unui astfel de model se obtine in urma executarii unui proces elaborat ce implica mai multi pasi:

- Primul pas reprezinta indentificarea datelor relevante in cea ce priveste problema tratata(setul de antrenare). In conformitate cu scopul clasificarii unor evenimente/date, in doua(sau mai multe) categorii, initial trebuie indentificate o serie de astfel de evenimente si categorizate de catre utilizator in evenimente ce sigur apartin fiecarei dintre categoriile tinta.
- Dupa obtinerea datelor de antrenare, trebuie indentificate toate trasaturile relevante din aceste date, trasaturi care sa fie cat se poate de specifice fiecarei categorii in parte. Fiind recomandata evitarea trasaturilor ce sunt prezente in mare parte din date sub aceasi forma (ex:caracterul '=' sau '?' intr-un URI folosit pentru clasificarea atacurilor SQL injection), indiferent de categoria din care acestea fac parte.
- Dupa obtinerea trasaturilor specifice datelor de antrenare, se realizeaza antrenarea modelului folosind un algoritm specific. In cazul proiectului propus s-a folosit algoritmul gata implementat, furnizat de biblioteca open source LIBSVM [18]. Pentru obtinerea modelului, datele de antrenare au fost procesate folosind un kernel gaussian. Un kernel gaussian reprezinta modul in care modelul proceseaza datele de antrenare astfel incat clasificarea noilor date sa fie realizata prin calcularea similaritatilor dintre acestea si cele de antrenare. In calcularea similaritatii dintre aceste doua tipuri de date, un parametru foarte important este sigma. Acest parametru este ales pentru

intrg setul de date, iar valoarea lui este direct proportionala cu gradul de similaritate pe care algoritmul il va asocia la doua evenimente/date diferite.

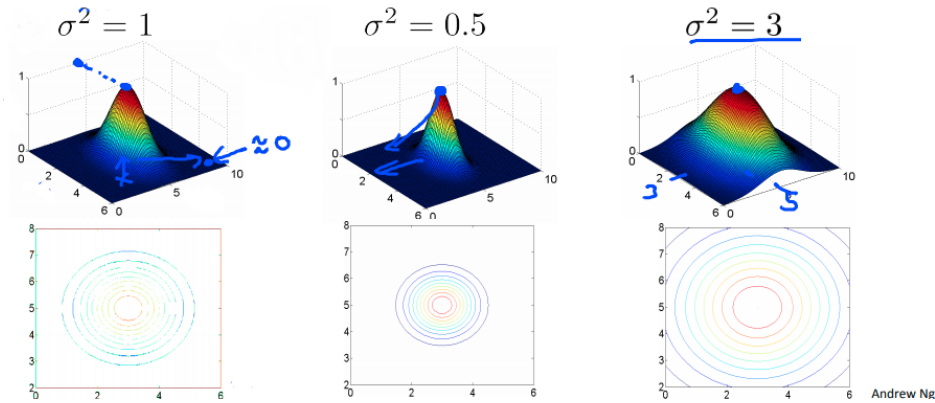


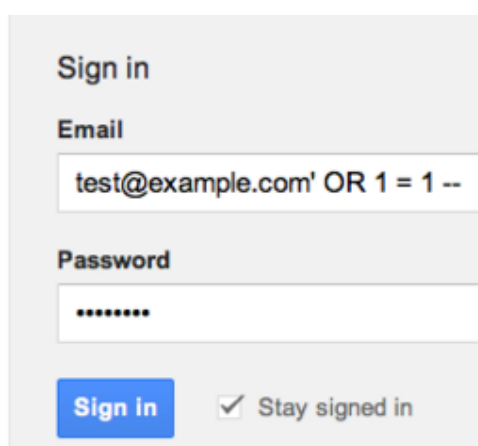
Figura 4.2: Influențele aduse algoritmului de modificare parametrului sigma in algoritmul de antrenare.

Figura 4.2 prezinta cum inflenteaza clasificarea unui nou eveniment valoarea parametrului sigma din formula algoritmului de support vector machine. Figura 4.2 a fost preluata din slide-urile cursului de machine learning sustinut de Andrew Ng [28]

4.3 SQL injection

Atacurile de tipul SQL injection sunt realizate prin injectarea de cod executabil intr-o baza de date.

Procesul de interactionare cu o baza de date presupune realizarea de interogari asupra acesteia. In formularea acestor interogari, utilizatorul trebuie sa prezinte interpretorului, sub forma de siruri de caractere, numele tabelelor interogate sau valorile unor capuri specifice din acestea. Aceste siruri de caractere sunt delimitate folosind caracterul " sau '. Atacurile de tipul SQL injection exploateaza folosirea acestor delimitatori de siruri de caractere, triminand siruri de caractere eronate intentionat catre baza de date. Un utilizator rau intentionat poate sa furnizeze astfel de siruri de caractere catre o baza de date prin intermediul oricarui procesator de continut disponibil unui client al unei aplicatii ce comunica cu o baza de date. Aceste siruri de caractere delimiteaza prematur valoarea care este folosita in interogare, introducand dupa aceasta o serie de caractere pe care interpretorul le va trata ca si cod executabil, oferindui astfel utilizatorului sa execute operatiuni asupra bazei de date la care nu ar avea accesul in mod normal. Aceste operatiuni pot sa reprezinte alterarea bazei de date sau obtinerea de date confidentiale.



The image shows a 'Sign in' form with two input fields: 'Email' and 'Password'. The 'Email' field contains the text 'test@example.com' OR 1 = 1 --'. The 'Password' field contains several dots, indicating it is masked. Below the fields are two buttons: a blue 'Sign in' button and a checkbox labeled 'Stay signed in'.

Figura 4.3: Exemplu de atac realizat prin SQL injection.

Figura 4.3 prezinta o tentativa de atac prin SQL injection in care in campul de validare a email-ului se incearca injectarea de cod ce va fi executat in interogarea de validare a credentialelor. Prin prezenta caracterului ' se escapeaza tot textul urmat dupa acesta ca fiind cod si nu un string ce face parte din campul de email. Operatia logica "OR 1=1" va determina interpretorul sa returneze adevarat(valid) pentru orice adresa de email introdusa inaintea caractrului '.

4.4 Adresele IP ale retelei Tor

Reteaua Tor reprezinta un softwrae gratis de anonimizare a traficului pe internet. Numele este de fapt un actonim pentru "The Onion Router"(router-ul ceapa) care sugereaza modul de functionarea al acestuia, ficare nod din retea adugand un strat extra de securitate celor precedente. Modul de functionare al retelei se bazeaza pe rutarea traficului prin cat mai multe noduri pentru a anonimiza si a face cat mai greu de urmarit traficul unei anumite persoane. Aceste noduri prin care traficul este directionat sunt sustinute gratis de catre voluntari/utilizatori de Tor din intreaga lume.

Pentru criptarea traficului reteaaua Tor foloseste criptarea la nivelul aplicatiei in cea ce priveste structura retelelor de calculatoare(modelul OSI sau TCP/IP). Datele tras-mise sunt criptate, incluzand destinatia, cu exceptia nodului urmator, astfel creandu-se structura de "ceapa" asupra unui pachet. Selectia nodurilor prin care se face rutarea pachetelor este aleasa random. Fiecare pachet decripteaza un "strat", afand nodul urmator pentru pachetul respectiv, nodul final decriptand datele initiale si realizand transmisia catre destinatie, fara sa ii comunice sursa pachetului. Intrucat in comunicarea pachetelor, pe parcursul rutelor parcurse, se cunoaste in permanenta doar nodul urmator, acest lucru impiedica monitorizarea traficului intre sursa si destinatie.

Cu toate ca retea tor ofera anonimitate de partea clientului, acest lucru nu se

realizeaza si fata de ea. Reteaua nu se ascunde fata de serviciile accesate prin intermediul ei. Astfel un site anume poate sa detecteze daca un anumit client il acceseaza folosind reteaua Tor sau nu.

Intrucat reteaua Tor nu isi ascunde adresele IP folosite de catre aceasta, indentificarea lor si procurarea de date despre acestea este destul de usoara. In proiectul propus s-a folosit un serviciu care furnizeaza gratuit astfel de date [26](adresa IP, uptime etc.) si s-au folosit algoritmi proprii pentru procesarea acestor date, eliminand astfel necorectitudinea dintre utilizatorii retelei.

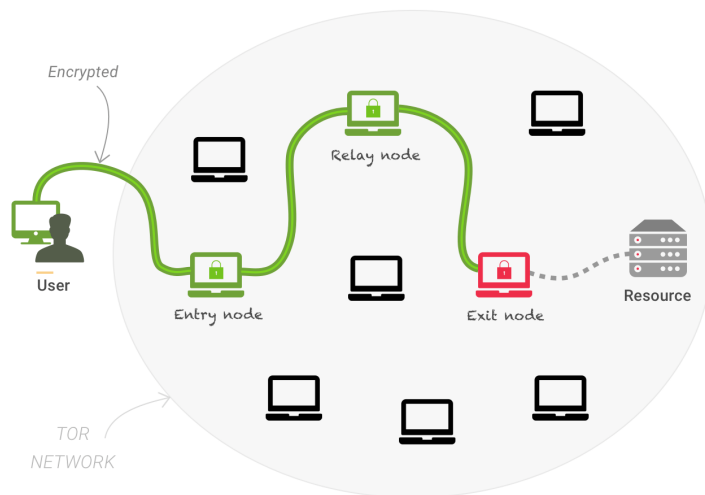


Figura 4.4: Exemplu de trafic realizat prin reteaua Tor.

Figura 4.4 prezinta principalele elemente folosite la rutarea treficului de la client la destinatie prin intermediul retelei Tor.

4.5 Sistem de prevenire a intruziunilor

Conform scurtei descrieri prezentate in capitolul anterior, un sistem de prevenire a intruziunilor are rolul de a filtra traficul dintre clientii unui server si serverul propiu zis.

Acest sistem functioneaza liniar, adica este plasat direct intre server si clienti acestuia. In cazul proiectului propus, componenta de baza pentru interceptarea traficului este realizata prin implementarea unui reverse proxy, oferind astfel caracteristica de interceptare si decriptare a traficului, ce permite analiza acestuia, dar si avantajele specifice utilizarii unui reverse proxy.

Pentru filtrarea traficului, un sistem de prevenire a intruziunilor implementeaza anumiti senzori care au rolul sa inspecteze tot traficul ce trece prin sistem, realizand aceasta inspectie in timp real. Datorita acestei verificari, orice pachet considerat malitios

este oprit din a ajunge la serverul destinatie. In proiectul propus, implementarea acestor senzori este realizata in doua moduri. In cazul validarii adreselor IP impotriva utilizatorilor de Tor se foloseste o lista de IP-uri ce contine adrese frecvent utilizate de reseaua Tor. In interiorul reverse proxy-ului, in momentul crearii unei noi conexiuni, acesta verifica ca adresa IP ce solicita conectarea la server sa nu fie continuta de lista mentionata. In cazul detectiei impotriva atacurilor de SQL injection, senzorul este implementat utilizand un model de support vector machine. In interiorul reverse proxy-ului in momentul interceptarii unui request venit din exterior catre reseaua interna, acesta verifica daca requestul poate fi clasificat ca si tentativa de atac. In caz afirmativ blocand trecerea acestuia mai departe catre server.

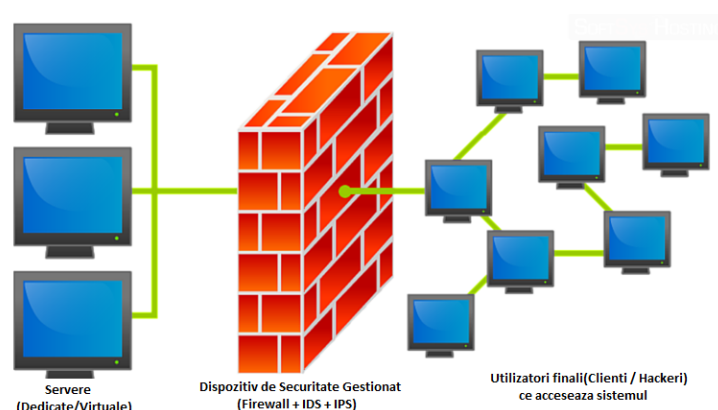


Figura 4.5: Integrarea unui sistem de prevenire a intruziunilor intr-o retea.

Figura 4.5 prezinta arhitectura unei retele interne ce integreaza un sistem de prevenire a intruziunilor pentru protejarea acesteia.

Un sistem de prevenire a intruziunilor poate sa efectueze oricare din urmatoarele actiuni in momentul detectarii unui eveniment malitios [29]:

- Sa intrerupa sesiunea dintre client si server, in cazul in care clientul desfasoara sau incearca sa desfasoare activitati malitioase in reseaua protejata de sistem. Acest lucru se poate realiza prin blocarea anumitor credentiale asociate cu utilizatorul respectiv sau prin blocarea adresei IP a acestuia.
- In conditiile in care un sistem de prevenire a intruziunilor detecteaza/clasifica o activitate ca fiind malitioasa, acesta poate sa ia masuri automat pentru a preveni un astfel de atac pe viitor(ex: in momentul detectarii unei tentative de atac prin SQL injection, sistemul de prevenire a intruziunilor poate sa blocheze in mod automat adresa IP a utilizatorului ce incerca sa faca atacul, nepermitandu-i acestuia sa se mai conecteze la serverul destinatie pentru un anumit interval de timp sau pana la interventia unui administrator).

- O alata abordare posibila in momentul declansarii unui eveniment malitios este alterarea traficului astfel incat sa elimine continutul malitios din acesta. Pentru realizarea acestui lucru, un sistem de prevenire a intruziunilor poate sa stearga atasamente infectate din interiorul unui mail, sa altereze continutul unui pachet sau sa omita transmiterea mai departe a unor pachete.

Capitolul 5

Analiză și proiectare

În acest capitol este descris design-ul proiectului și cuprinde: cerințele sistemului, specificațiile cazurilor de utilizare, arhitectura sistemului, comportamentul sistemului, datele utilizate de sistem, dependențele sistemului și algoritmi esențiali și metodele folosite. Descrierea acestora se realizează prin asocierea cu diagramelor aferente.

5.1 Cerințele sistemului

Sistemul propus *Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea* reprezintă un software gratis care oferă clientului atât caracteristici specifice unui reverse proxy, cât și modalități de protecție asemănătoare unui sistem de prevenire a intruziunilor. Sistemul este ușor de instalat și de utilizat, chiar și de către utilizatorii neexperimentați, oferindu-le acestora o interfață clară și sugestivă, ce maschează logica complicată din spate. În spate, sistemul oferă un reverse proxy ce interceptează tot traficul destinat către un anumit server, de pe una sau mai multe interfețe. În procesul de interceptare, acesta implementează și câteva modalități de prevenire a unor tentative de intruziune. Sistemul blochează toate ip-urile utilizate frecvent de rețeaua Tor în ultima lună și atacurile de SQL injection cu o acuratețe de 90%.

Sistemul trebuie să îndeplinească următoarele cerințe **functionale**:

1. Să realizeze conexiunea la un server HTTP/HTTPS și să redirectioneze traficul primit către acesta.
2. Să intercepteze traficul venit pe o anumită interfață și port prestabilit.
3. Să prelucereze request-urile primite de la clienți într-un format specific clasificatorului de SQL injection.
4. Să nu redirectioneze requesturile clasificate ca și SQL injection.
5. Să blocheze conectarea clientilor ce folosesc ip-uri clasificate ca ip-uri de Tor.

6. Sa permita utilizatorului sa editez si sa vizualizeze lista ip-urilor de Tor.
7. Sa prezinte in interfata grafica toate interventiile rezlizate asupra traficului(blocari de conexiuni sau de request-uri).
8. Sa permita utilizatorului sa configureze modul de operare al sistemului.

Sistemul trebuie, de asemenea, să aibă următoarele caracteristici **non-funcționale**:

1. Sa fie usor de instalat si de folosit pentru orice utilizator, oricat de neexperimentat.
2. Sa poata intercepta traficul de pe orice/oricate interfete disponibile.
3. Sa poata rula pe orice sistem de operare Windows cu Python2 instalat.
4. Sa aiba o rata de blocare de 100% a ip-urilor de pe lista neagra, iar in cazul detectiei de SQL injection sa nu aiba detectii false pozitive mai mari 2-3% si o acuratete generala de peste 90%

5.2 Specificatiile cazurilor de utilizare

5.2.1 Actori, stakeholders si interese

Principalii actori si stakeholder-i sunt administratorii de servere, respectiv de baze de date. In implementarea sistemului propus se urmareste satisfacerea nevoilor acestor persoane, oferindule un plus de securitate asupra datelor ce sunt accesate de catre clienti, respectiv impotriva clientilor rau intentionati. Interesele acestor comunitati de utilizatori sunt urmarite pentru a livra un produs care sa ofere aceste protectii intr-o maniera cat de prietenoasa pentru utilizator si cat mai eficienta.

5.2.2 Basic flow

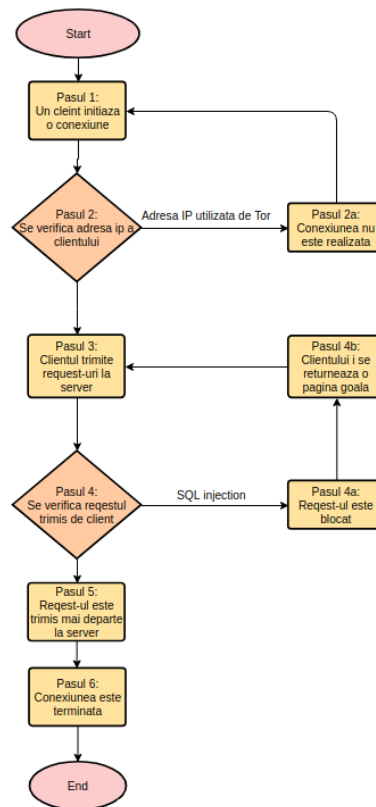


Figura 5.1: Basic flow pentru evenimente

Figura 5.1 prezinta cum arata basic flow-ul pentru evenimentele din sistemul propus.

Evenimentele sistemului:

1. **Start:** aceste cazuri de utilizare sunt initiate in momentul in care sistemul este plasat intre un server si utilizatorii acestuia.
2. **Pasul 1:** un client doreste si initializeze o conexiune la server.
3. **Pasul 2:** adresa ip a clientului este verificata ca acesta sa nu fie un utilizator de Tor.
4. **Pasul 3:** clientul comunica cu serverul prin request-uri individuale.
5. **Pasul 4:** request-urile sunt verificate ca acestea sa nu fie atacuri de SQL injection.
6. **Pasul 5:** request-urile sunt trimise mai departe catre server.
7. **Pasul 6:** conexiunea este terminata(de catre client sau server).
8. **End:** la sfarsitul unui sir de evenimente, utilizatorul sistemului poate sa vada daca clientul ce a initializat evenimentele a realizat actiuni considerate ca malitioase.

5.2.3 Alternative flow

Evenimentele alternative in cazurile de utilizare sunt urmatoarele:

1. **Pasul 2a:** in cazul in care un utilizator cu adresa ip de Tor incearca sa realizeze conexiunea la server aceasta este refuzata.
2. **Pasul 4a:** in cazul in care un utilizator incearca sa trimita la server un request ce reprezinta o tentativa de atac SQL injection, acesta este blocat.
3. **Pasul 4b:** pentru request-urile blocate ca si SQL injection clientului i se returneaza o pagina goala.

5.3 Arhitectura sistemului si modulele principale

În conformitate cu cerințele sistemului și specificațiile cazurilor de utilizare, sistemul propus este alcătuit din module specifice care să trateze în mod individual problemele majore ridicate de sistem, printr-o implementare modulară, oferind astfel ușurința implementării de noi funcționalități.

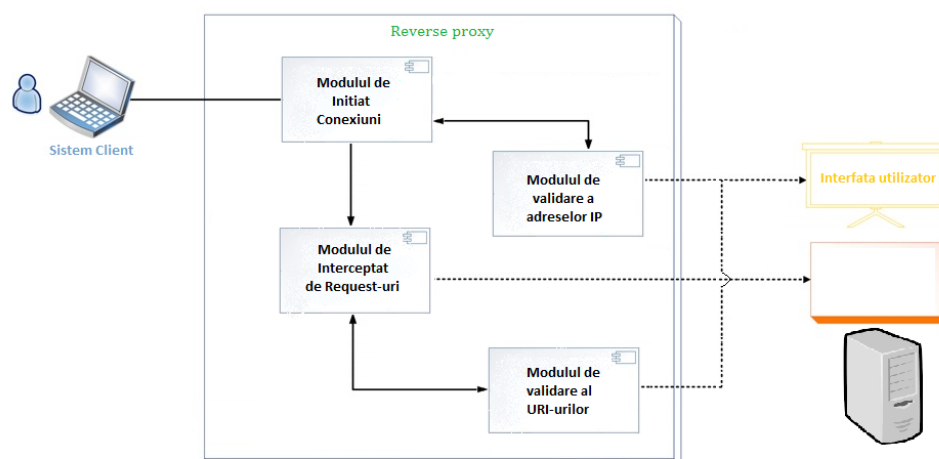


Figura 5.2: Principalele module ale sistemului propus

Figura 5.2 prezintă care sunt principalele module ale sistemului propus, precum și interacțiunea dintre acestea.

Principală și teoretic singura componentă a sistemului este reprezentată de cea de **reverse proxy**, însă pentru a obține rezultatele dorite și pentru a încorpora funcționalitățile de protecție în această, logica ei a fost împărțită în cele 4 mari submodule: **connections acceptance module** și **request interceptor module**, ce reprezintă componentele ce suportă logica unui reverse proxy și **ip address validation module** și **SQL injection prevention module**, ce reprezintă modulele care interacționează cu cele ce implementează structura de reverse proxy pentru a oferi funcționalitățile de securitate.

Client system și application server.

Aceste două componente reprezintă componentele clasice între care vin plasat sistemul propus. **Client system** este reprezentat de orice client dorește să acceseze baza de date/parte de server a unei aplicații. **Application server** reprezintă serverul aplicație la care se pot conecta clienții pentru a avea acces la un anumit conținut. Sistemul propus are rolul de intermediere între cele două tipuri de componente, prevenind astfel eventuale tentative de exploatare a unor vulnerabilități din partea clientului către server.

Connections acceptance module.

Acest modul este constituit din componentele oferite de biblioteca open source **twisted**, fiind modificate ulterior pentru a permite integrarea modului **Ip address validation module**. Modulul are rolul de a crea un socket pe sistemul pe care ruleaza acesta, care sa asculte pentru posibile conexiuni. In cazul in care un client incearca sa initieze o noua conexiune acesta transmite adresa ip a clietului catre modului **Ip address validation module**, iar in functie de raspunsul acestuia conexiunea acestuia va fi realizata sau refuzata.

Request interceptor module.

Asemeni modului anterior, **Connections acceptance module**, acest modul este construit peste codul oferit de biblioteca open source **twisted**. Biblioteca ofera suport pentru interceptarea request-urilor dintre un client si server, implementandu-se o logica suplimentarea ce incorporeaza modulul de detectie a atacurilor SQLI(**SQL injection prevention module**). In acest modul se interpreteaza natura URL-urilor trimise de catre client catre server. Interpretarea constand in analiza URI-ului unui request de catre modulul de prevenire a atacurilor SQLI, iar in functie de rezultatul acestuia returnand raspunsuri corespunzatoare atat clientului cat si serverului(in caz de tentativa de atac, la server nu ca ajunge nimic).

Ip address validation module.

Rolul acestui modul este de a valida o adresa ip impotriva adreselor folosite frecvent de reseaua Tor. La pornirea sistemului modulul incarca in memorie o lista cu toate adresele ip care trebuie sa fie blocate. Aceasta lista este construita la pornirea sistemului pe baza unor date salvate local si reactualizate periodic. Pentru validarea unei adrese, in cazul in care aceasta se afla in lista cu ip-uri blocate modulul returneaza valoarea "false" codului apelant, iar in caz contrar valoarea "true".

SQL injection prevention module.

Logica modului de prevenire a atacurilor SQL injection este impartita in doua componente. In prima parte acesta indentifica trasaturile specifice unui atac SQL injection intr-un URI. Aceste trasaturi sunt reprezentate de caractere si cuvinte cheie ce pot fi gasite in acesta. In cazul indentificarii unor astfel de trasaturi, URI-ul este pasat mai departe celei de a doua componenta. Aceasta din urma este reprezentata de un model de support vector machine antrenat anterior, care va decide daca URI-ul primit se incadreaza in cele specifice atacurilor de SQL injection sau nu. Asemeni modului anterior **Ip address validation module**, in functie de rezultatul obtinut acesta va returna "true" sau "false" codului apelant.

5.4 Comportamentul sistemului. Diagrama de stare si de secventa

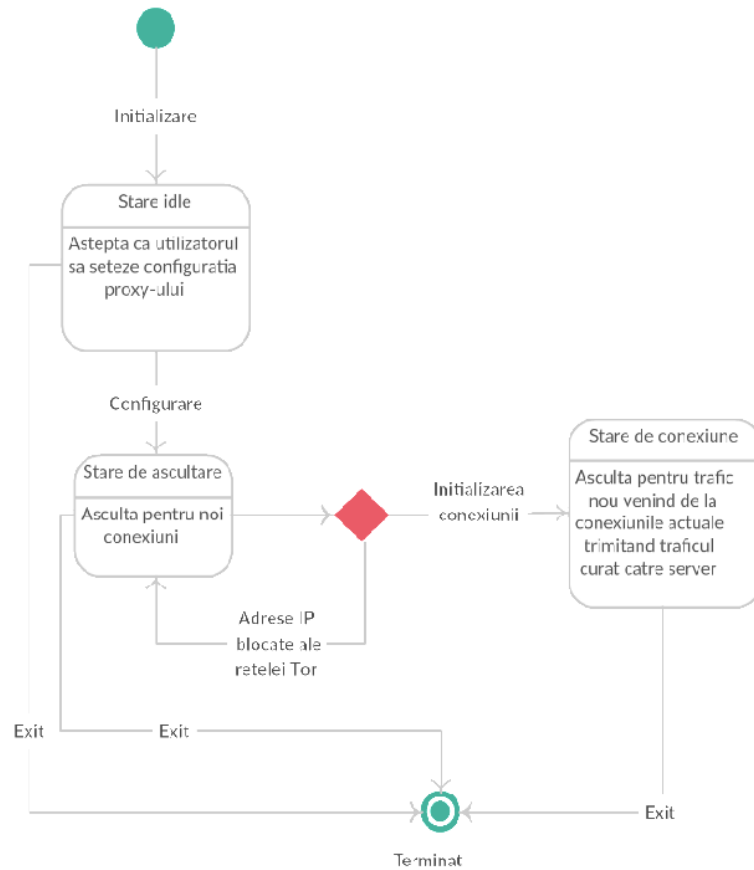


Figura 5.3: Diagrama de stare a sistemului propus.

Figura 5.3 prezintă diagrama de stare a sistemului propus.

După pornirea aplicației reprezentată prin initializare, utilizatorului i se pune la dispoziție interfața de utilizator în care sistemul așteaptă să fie configurat și pornit ("Idle state"). După configurarea aplicației în meniul "Configure", prin setarea specificațiilor serverului de reverse proxy, prin apăsarea butonului "start", sistemul trece în următoarea stare "Listening state". În această stare sistemul accepta noi conexiuni de la diferiți clienți ce doresc să acceseze serverele protejate de acesta. Tot aici are loc și verificarea adreselor IP ale clienților pentru protejarea împotriva utilizatorilor Tor. După stabilirea unei noi conexiuni, sistemul intră în starea "Connected state", în care acesta ascultă pentru noi request-uri între client și server și le transmite mai departe către destinația dorită. Tot aici are loc și validarea request-urilor trimise de clienți către server pentru prevenirea atacurilor

SQL injection. În această stare utilizatorul poate să urmărească activitatea sistemului în meniul "Monitor" din interfața de utilizator, unde sunt logate toate evenimentele generate de sistem. Pentru oprirea aplicației, utilizatorul poate pur și simplu să o închidă de la butonul din dreapta sus, aceasta terminându-și toată activitatea indiferent de starea în care se afla.

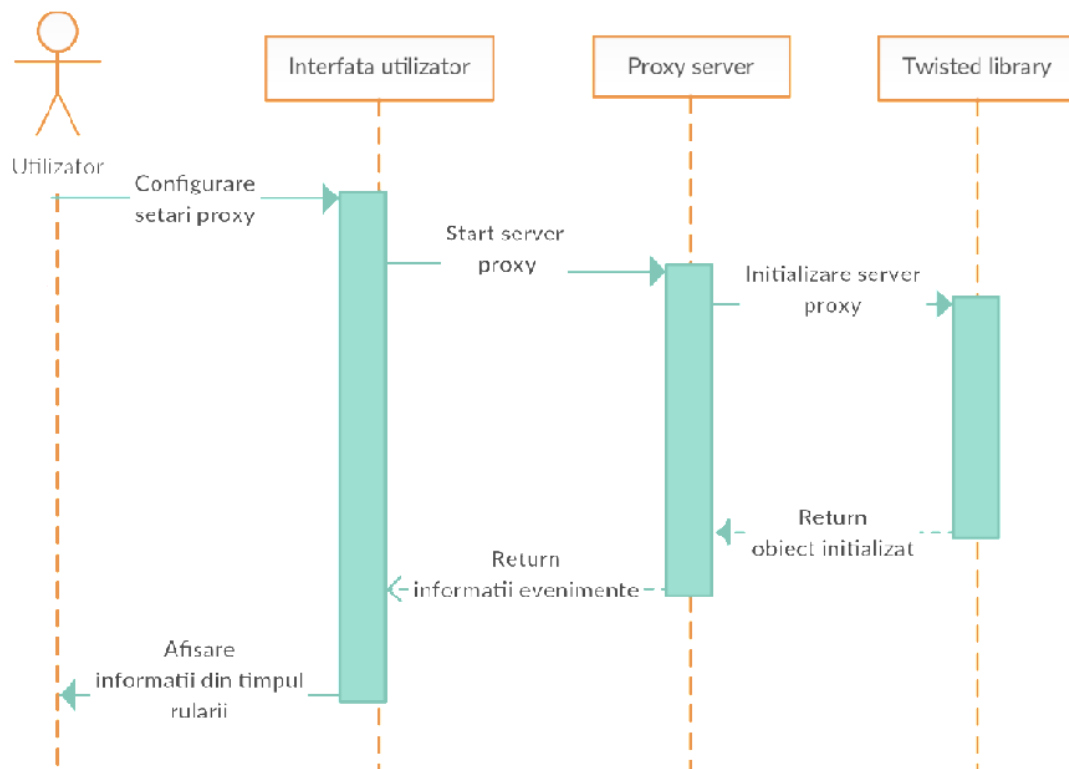


Figura 5.4: Diagrama de secvență a sistemului propus.

Figura 5.4 prezintă diagrama de secvență a sistemului propus. După pornirea sistemului, utilizatorul configurează setările pentru server-ul de reverse proxy din interfața de utilizator (în meniul "Configure"). După configurarea sistemului, prin apăsarea butonului de start, este lansată în execuție server-ul de reverse proxy cu caracteristicile specificate de utilizator. Scriptul de pornire a server-ului de reverse proxy face apel la fișierele bibliotecii Twisted, care vor returna către acesta o instanță a server-ului inițializat. În interiorul server-ului de reverse proxy, apariția de noi evenimente este semnalată către interfața grafică. Aceste evenimente sunt puse la dispoziția utilizatorului pentru vizualizare.

5.5 Dependintele sistemului

Pentru functionarea corecta a sistemului propus, acesta are nevoie de urmatoarele dependinte:

1. Sistem de operare Windows cu Python2 instalat pe sistem.
2. Conexiune stabila la internet pentru rularea script-urilor ce obtin adresele ip utilizate de Tor.
3. Sistemul sa aiba acces direct la interfetele prin care clientii se pot conecta la server-ul protejat
4. Sistemul sa aiba o conexiune directa la server-ul ce trebuie protejat.

5.6 Algoritmi si metode

Pentru prevenirea atacurilor de SQL injection s-a folosit un algoritm de machine learning, support vector machine. Acest algoritm presupune antrenarea anterioara a unui model cu un set de date de referinta etichetate in prealabil corect de catre utilizator. Aceste date sunt folosite de algoritm pentru a raporta noi seturi de date, natura acestora fiind necunoscuta, incercand sa gaseasca similitudin intre cele noi si cele de referinta, determinandu-le astfel natura celor noi. Algoritmul de support vector machine rezulta intr-un model obtinut prin folosire a diversi algoritmi pentru antrenarea acestuia, folosit pentru a clasifica date.

Realizarea unui astfel de model sa realizat in urma executarii unui proces elaborat ce implica mai multi pasi:

- Primul pas reprezinta indentificarea datelor relevante in cea ce priveste problema tratata(prevenirea atacurilor SQL injection). In conformitate cu scopul clasificarii evenimentelor/datelor in doua categorii (tentativa de atac si request-uri clean) initial au fost indentificate o serie de astfel de evenimente si categorizate in evenimente ce sigur apartin fiecarei dintre categoriile tinta.
- Dupa obtinerea datelor de antrenare, au fost indentificate toate trasaturile relevante din aceste date, trasaturi care sa fie cat se poate de specifice fiecarei categorii in parte. Aceste trasaturi au reprezentat cuvintele cheie a limbajului SQL dar si caractere specifice folosite frecvent de acesta.
- Dupa obtinerea trasaturilor specifice datelor de antrenare, sa realizat antrenarea modelului folosind un algoritm specific. In cazul proiectului propus s-a folsoit algoritmul gata implementat, furnizat de biblioteca open source LIBSVM [18]. Pentru obtinerea modelului datele de antrenare au fost procesate folosind un kernel gaussian. Un kernel gaussian reprezinta modul in care modelul proceseaza datele de antrenare astfel incat

clasificarea noilor date sa fie realizata prin calcularea similaritatilor dintre acestea si cele de antrenare. In calcularea similaritatii dintre aceste doua tipuri de date, un parametru foarte important este sigma. Acest parametru este ales pentru intrg setul de date, iar valoarea lui este direct proportionala cu gradul de similaritate pe care algoritmul il va asocia la doua evenimente/date diferite.

Pentru blocarea ip-urilor utilizate de rețea Tor s-a folosit un script scris in Python3. Programul interogheaza periodic(din 6 in 6 ore) informatiile oferite de *Tor Network Status* [26], salvand uptime-ul fiecărei adrese din ultimele 6 ore. Indentificarea adreselor malitioase se face prin insumarea timpului total din ultima luna de uptime, iar cele cu o valoare mai mare de 7 zile sunt considerate malitioase. Blocarea ip-urilor se realizeaza prin compoarea cu o astfel de lista generata lunar.

5.7 Diagrama de desfasurare

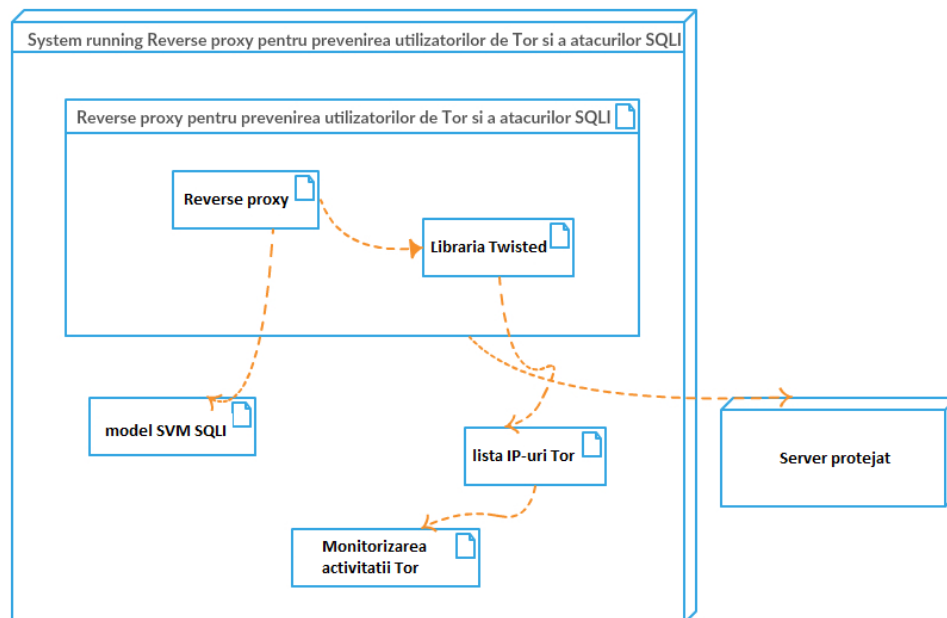


Figura 5.5: Diagrama de desfasurare a sistemului propus.

Figura 5.5 prezinta diagrama de desfasurare a sistemului propus. Pentru a putea rula sistemul *Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea* pe un sistem, acesta trebuie sa aiba acces direct la server-ul ce se doreste a fi protejat. Sistemul trebuie sa ii fie furnizate modelul de SVM folosit pentru prevenirea atacurilor de SQL injection. De asemenea scriptul pentru monitorizarea activitatii rețelei Tor si cel pentru generarea listei de adrese IP cu un uptime mai mare de 7 zile, trebuie sa se afle pe sistem, pentru ca acesta sa utilizeze date "up-to-date".

5.8 Justificarea design-ului

Totate deciziile de design au fost luate in vederea obtinerii unor functionalitati cat mai importante pentru utilizatorii tinta, dar si pentru a obtine o eficienta cat mai mare a sistemului, atat din punct de vedere al corectitudinii cat si a timpului de executie.

Pentru o acuratete cat mai mare in prevenirea atacurilor de SQL injection s-a ales folosirea unui abordari bazate pe machine learning, intrucat detectiile statice prezinta o logica foarte complicata, pot omite multe cazuri esentiale si sunt limitate de capacitatea de observare a programatorului.

Pentru indentificarea adreselor utilizate de reseaua Tor se putea aborda o implementare cu lista fixa, intrucat aceste liste prezinta un set mic de adrese ce raman neschimbate in timp, insa pentru o acuratete cat mai mare, s-a ales folosirea celor doua scripturi suplimentare ce vor obtine periodic toate adresele utilizate de retea perioada respectiva de timp.

De asemenea dezvoltarea modulara a sistemului asigura usurinta de adaugare de noi module sau functionalitati fara a necesita modificari majore sistemului actual.

Capitolul 6

Detalii de implementare

6.1 Structura codului sursa

Codul folosit pentru obtinerea sistemului *Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea* este impartit in 3 proiecte separate:

- **Tor activity monitor** ce constituie logica necesara obtinerii listei de adrese IP blocate de sistem.
- **SQLi SVM** scopul acestuia fiind obtinerea modelului de SVM folosit pentru prevenirea atacurilor de SQL injection.
- **Implementarea unui reverse proxy pentru prevenirea atacurilor la nivel de rețea** reprezentand sistemul propus si incorporeaza rezultatele obtinute de celelalte doua proiecte.

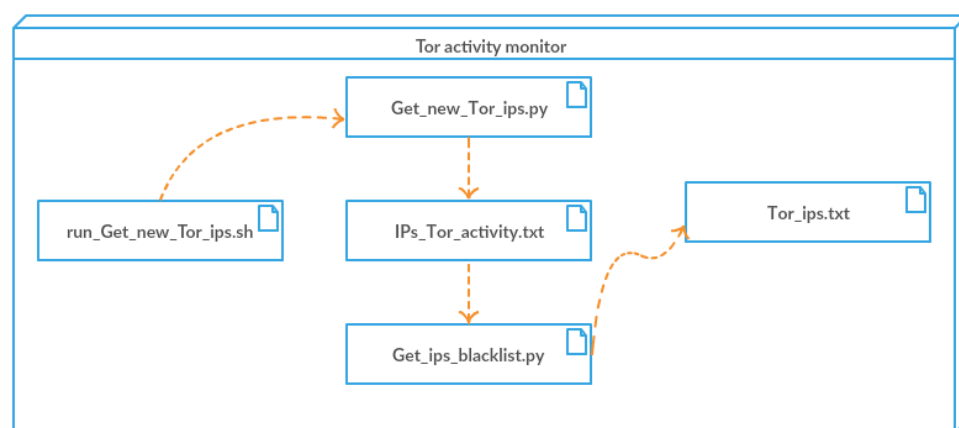


Figura 6.1: Modulul pentru monitorizarea activității rețelei Tor

Figura 6.1 prezinta care sunt fisierele folosite pentru monitorizarea activitatii retelei Tor si pentru obtinerea listei cu adresele IP ce trebuie blocate de catre sistem si relatiile dintre acestea.

run_Get_new_Tor_ips.sh este un fisier de bash ce are rolul de a rula **Get_new_Tor_ips.py**. Fisierul este programat sa lanseze in executie **Get_new_Tor_ips.py** la ore fixe, acesta rula incontinuu pe un sistem cu acces la internet neintrerupt. Lnsarile in executie au loc o data la 6 ore, respectiv la ora 12 am si pm si 6 am si pm.

Get_new_Tor_ips.py are rolul de a documenta modificarile de uptime din ultimele 6 ore ale nodurilor retelei Tor. Datele sunt furnizate de pe pagina "Tor Network Status" [26] si pentru fiecare adresa IP prezenta in date, se verifica care este uptime ul din ultimele 6 ore, informatiile acestea fiind stocate in **IPs_Tor_activity.txt**.

IPs_Tor_activity.txt are rolul de a stoca informatiile despre toate adresele IP utilizate de reseaua tor din ultima luna. Acestea sunt salvate in liste, pentru fiecare adresa IP in parte o lista.

Get_ips_blacklist.py are rolul de genera fisierul **Tor_ips.txt** folosit de catre sistemul propus pentru blocarea adreselor IP. Acesta foloseste datele din interiorul fisierului **IPs_Tor_activity.txt**, pentru a genera o lista cu toate adresele IP ce au un uptime total mai mare de 7 zile in ultimele 30 de zile.

Tor_ips.txt reprezinta rezultatul proiectului. Acesta este alcatuit dintr-o lista formata din toate adresele IP ce vor fi blocate de sistemul propus in timpul rularii.

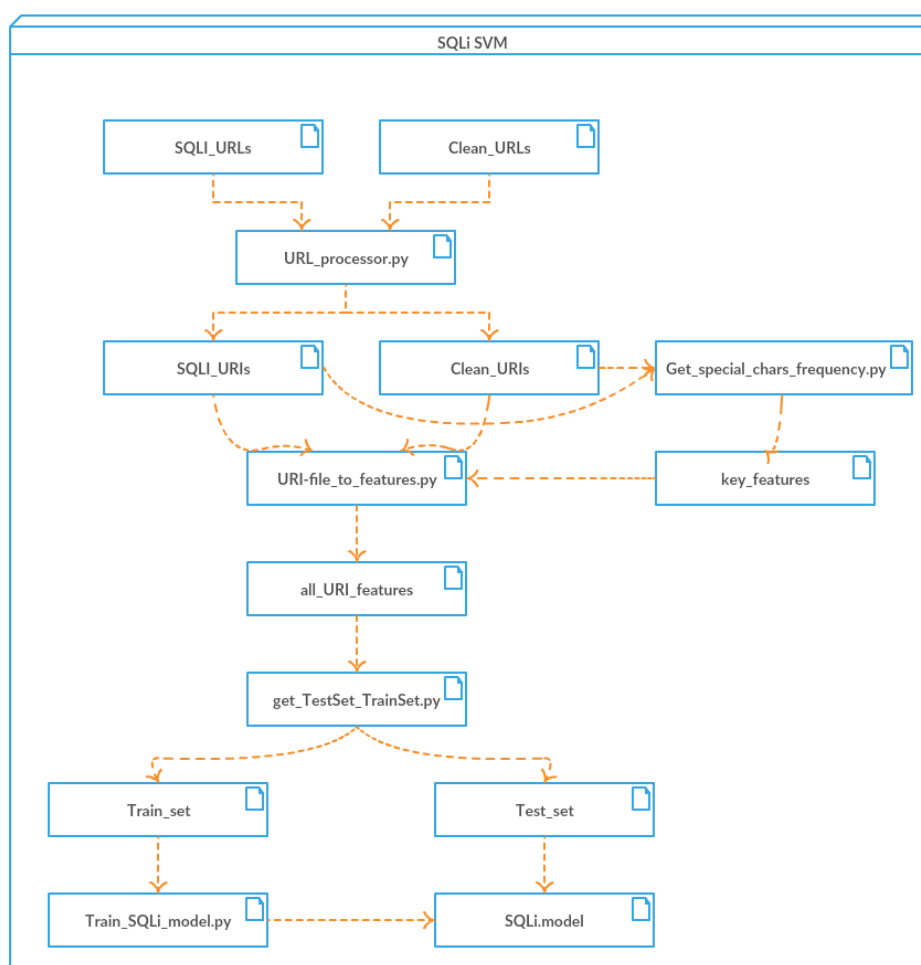


Figura 6.2: Modulul pentru prevenirea atacurilor SQL injection

Figura 6.2 prezinta care sunt fisierele utilizate pentru procesarea datelor necesare in procesul de antrenare a modelului de SVM, pana la obtinerea modelului propriu-zis si care sunt realtiile dintre acestea.

SQLI_URLs reprezinta setul initial de date "infected" folosite pentru indentificarea atacurilor SQL injection. Acest set este constituit din URL-uri ce au fost indentificate de un produs autorizat ca fiind tentative de SQL injection.

Clean_URLs reprezinta setul initial de date "clean" folosite pentru indentificarea atacurilor SQL injection. Acest set este constituit din URL-uri ce au fost indentificate de un produs autorizat ca fiind URL-uri curate.

URL_processor.py primeste ca input un fisier sau string si are rolul de a procesa URL-uri intr-un format uniform(se elimina encodările) si specific pentru pasii urmasori.

SQLI_URLs constituie noua lista rezultata din procesarea fisierului SQLI_URLs de catre URL_processor.py.

Clean_URLs constituie noua lista rezultata din procesarea fisierului Clean_URLs de catre URL_processor.py.

Get_social_chars_frequency.py are rolul de a caldula frecventa de aparite a unor caractere speciale in cele doua seturi de date si de a decide in functie de frecventa lor de aparite, care din acestea sunt relevante in vederea alegerii trasaturilor de clasificare.

key_features este o lista alcatuita din toate cuvintele cheie a limbajului SQL, dar si din caracterele speciale utilizate in acesta si considerate ca fiind relevante in urma executiei scriptului Get_social_chars_frequency.py.

URI-file.to.features.py are rolul de a procesa cele doua fisiere de date si pe baza trasaturilor din key_features sa constituie un nou fisier ce contine pentru fiecare URL din cele doua fisiere tipul acestuia si trasaturile gasite in el, precum si frecventa lor.

all_URI_features este rezultatul rularii scriptului URI-file.to.features.py si contine pentru fiecare URL din cele doua fisiere de date, tipul acestuia si trasaturile gasite in el, precum si frecventa lor, acestea fiind folosite pentru antrenarea si testarea modelului de SVM.

get_TestSet_TrainSet.py are rolul de a imparti datele prezente in all_URI_features in doua seturi de proportie 70-30. Aceste doua seturi fiind folosite pentru antrenarea si testarea modelului.

Train_set constituie 70% din totalul de exemple acumulate pentru antrenarea modelului, doar acestea fiind de fapt folosite pentru antrenarea sa.

Test_set constituie 30% din exemplele acumulate, aceste date fiind folosite pentru testarea acuratetii modelului dupa antrenare.

Train_SQLi_model.py realizeaza obtinerea modelului de SVM folosit de sistem pentru prevenirea atacurilor SQL injection. Pentru antrenare modelului este folosit setul de date din fisierul Test_set si algoritmi pusi la dispozitie de biblioteca open source libsvm [18].

SQLi.model reprezinta rezultatul proiectului. Acesta este testat cu ajutorul setului de date din fisierul Test_set si ulterior integrat in sistemul propus pentru a fi folosit pentru prevenirea atacurilor SQL injection.

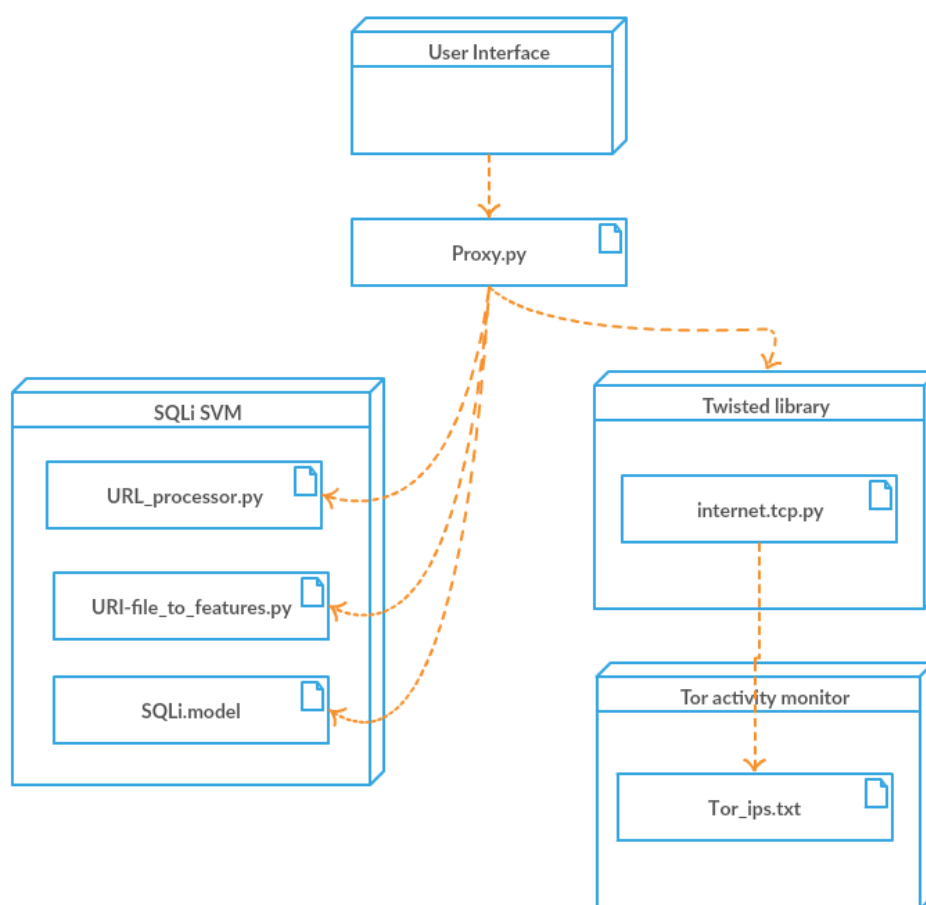


Figura 6.3: Interactiunea dintre interfata grafica si celelalte module

Figura 6.3 prezinta care sunt fisierele, specific fiecarui modul, care sunt accesate in mod direct de catre interfata sau indirect de alte fisiere, in timpul rularii si relatiile dintre acestea.

User Interface reprezinta intreaga componenta ce realizeaza interfata de utilizator cu toate fisierele necesare realizarii ei, incorporate in compozitia sa. Aceasta a fost realizata de un proiect dezvoltat in .Net implicand multe fisiere cu scopul realizarii elementelor grafice. Aceste elemente nu vor fi tratate in aceasta sectiune.

Proxy.py reprezinta fisierul ce incorporeaza, respectiv leaga, tot codul ce constituie partea de "back end" a proiectului. In acest script de python se realizeaza instantierea elementului de reverse proxy cu parametri de adrese IP si porturi aferente, precum si furnizarea metodelor de detectie componentei de reverse proxy.

Twisted library este o biblioteca open source scrisa in Python, ce ofera suport pentru diferite protocoale(TCP, UDP, SSL/TLS). Biblioteca a fost folosita pentru partea de cod ce ofera implementarea unui reverse proxy. Mare parte din fisierele oferite de

acesta biblioteca au fost folosite fara a fi suprascrisa sau a li se aduce modificari ulterioare, insa in vederea atingeri scopului propus, asupra unor fisiere sursa au fost aduse mici modificari(internet.tcp.py).

internet.tcp.py este scriptul din biblioteca twisted ce ofera suportul pentru protocolul TCP. Acest fisier a fost modificat pentru introducerea detectiei impotriva utilizatorilor de Tor. Script-ul integreaza lista realizata de proiectul "Tor activity monitor" pentru verificarea adresei utilizatorilor ce doresc sa stabileasca o conexiune TCP.

6.2 Algoritmi, metode si API-uri

In aceasta sectiune se urmareste descrierea codului sursa folosit la realizarea sistemului propus si explicarea amanuntita a codului/metodelor considerate mai relevante, precum si a principalelor api-uri utilizate in implementarea acestuia. Abordarea codului sursa se realizeaza conform sub-proiectelor prezentate in sectiunea anterioara(Tor activity monitor, AQLi SVM, Interfata utilizator).

6.2.1 Tor activity monitor

Conform figurii 6.1, acest proiect este realizat din 5 fisiere, acestea avand o relatie liniara intre ele.

Fisierul ce incepe ciclul de executie, run_Get_new_Tor_ips.sh, este un fisier de bash ce ruleaza in bucla infinita. Fisierul trebuie sa ruleze pe un sistem ce este functional non-stop si cu acces nelimitat la internet. La ore fixe acesta (12 am si pm si 6 am si pm), acesta lanseaza in executie scriptul de python Get_new_Tor_ips.py.

Fisierul principal din acest proiect il reprezinta Get_new_Tor_ips.py. Acesta descarca pagina "Tor Network Status" [26] si proceseaza datele de pe acestea, introducand in fisierul IPs_Tor_activity.txt informatii referitoare la adresele IP gasite pe pagina si uptime-ul lor din ultimele 6 ore. Procesarea adreselor IP si extragerea valorii lor de uptime se poate observa in urmatoarele doua secvente de cod.

```
time_up = row.findAll('td')[4].contents[0]
ip = row.findAll('td', attrs={'class': 'iT'})[0].findAll('a',
    attrs={'class': 'who'})[0].contents[0]
```

Pentru prelucrarea continutului paginii, acesta a fost downloadat in memoria programului, iar codul HTML rezultat a fost prelucrat cu ajutorul bibliotecii de python open source, BeautifulSoup. Codul de mai sus reprezinta extragerea valori de timp(uptime) si adresa IP careia aceasta corespunde. Variabila "row" fiind un elemnet din obiectul iterabil rezultat din initializarea bibliotecii BeautifulSoup cu codul HTML al paginii.

```
days = time_up.split()[1]
```

```
if days == 'd':
    hours = 6
else:
    hours = int(time_up.split()[0])
    if hours > 6:
        hours = 6
```

In secventa de mai sus de cod, este indentificata valoarea corecta de uptime din ultimele 6 ore. Pentru cazul in care valoare de uptime este sub forma de zile sau aceasta este mai mare de 6 ore, ea se seteaza pe 6 ore, intrucat nu ne intereseaza decat activitatea din ultimele 6 ore.

```
from bs4 import BeautifulSoup
from urllib.request import urlopen

pagesource = urlopen(page)
soup = BeautifulSoup(pagesource.read())
table = soup.findAll('table', attrs={'class': 'displayTable'})
```

Pentru procesarea continutului unei pagini web("Tor Network Status" [26]) s-au folosit bibliotecile de python, open source, urllib si BeautifulSoup [17]. Biblioteca urllib ofera functii si clase ce pot fi folosite pentru deschiderea de URL-uri(urlopen). Functia folosita in codul de mai sus primeste ca parametru "page" adresa sub forma de string a paginii ce se doreste a fi citita. Biblioteca BeautifulSoup este o biblioteca de python folosita pentru extragerea de date din fisiere de HTML sau XML. Acesta este initializata cu un document HTML/XML si ca rezultat ofera un obiect ce permite navigarea prin codul sursa asemeni unei structuri de date imbricate. Astfel, spre exemplu, pentru indentificarea structurii corespunzatoare tabelii de adrese IP din codul sursa al paginii, s-a folosit o singura linie de cod, in care s-au specificat numele si attributele specifice acesteia.

Rezultatele rularii fisierului Get_new_Tor_ips.py sunt actualizate in IPs_Tor_activity.txt. Acest fisier este de fapt un json in care se realizeaza dump la noul dictionar obtinut de Get_new_Tor_ips.py. Dictionarul este constituit din adresa IP ca si cheie si o liste. Structura acestor liste este realizata dintr-o serie de numere intre 0 si 6 ce reprezinta timpul total de uptime corespunzator sfertului respectiv de zi. Dimensiunea acestei liste este fixata la 30(zile)*4(sferturi de zi), pe masura ce un element nou este adaugat, primul element din lista fiind scos.

Urmatorul pas este filtrarea tuturor adreselor IP ce au un uptime mai mare de 7 zile. acest lucru este realizat de scriptul Get_ips_blacklist.py, iar rezultatele sunt stocate in Tor_ips.txt, o adresa IP pe linie.

6.2.2 SQLi SVM

Desfasurarea proiectului incepe de la cele doua fisiere SQLi_URLs si Clean_URLs. In aceste doua fisiere se afla URL-uri complete din categoria conforma cu numele fiecarui fisier. Aceste fisiere sunt procesate de scriptul URI_processor.py care are rolul de a uniformiza datele in aceasi encodare. Urmatoarea bucata de cod prezinta secventa ce transforma valorile hexa dintr-un URI in caractere si modul in care erorile de encodare sunt tratate(sunt printate pentru a fi tratate manual de catre programator):

```
for index, sub_uri in enumerate(uri.split('%')):
    if sub_uri:
        if index == 0:
            new_uri = sub_uri
            continue
        try:
            hex_val = bytearray.fromhex(sub_uri[:2]).decode()
        except UnicodeDecodeError:
            print(uri + ' --- ' + sub_uri[:2])
            return ''
        new_uri += hex_val + sub_uri[2:]
```

Intrucat intr-un URL caracterul '%' nu poate sa apara decat daca acesta este encodat('%25'- valoarea pentru encodarea caracterului '%'), indentificarea tuturor caracterelor encodate a fost facuta prin indentificarea tuturor caracterelor de tipul '%' in URI. In cazul in care un astfel de caracter este gasit, se incearca conversia urmatoarelor doua caractere(asteptandu-se, conform conventiei, sa fie cifre), din valoarea in hexa corespunzatoare unui anumit caracter in caracterul in sine.

In urma procesarii datelor, rezulta cele doua fisiere SQLi_URLs si Clean_URLs, acestea avand acelasi continut cele anterioare insa in aceasi encodare. In urma obtinerii acestor doua fisiere, a fost realizata completarea listei de trasaturi(lista initiala este constituita din toate cuvintele cheie a limbajului SQL) cu caracterele speciale intalnite in acest limbaj. Scriptul Get_social_chars_frequency.py are rolul de a determina frecventa de aparitie a fiecarui caracter special in cele doua seturi de date, iar pe baza unei observatii umane, a fost realizata determinarea caracterelor ce constituie trasaturi rentabile. Urmatoarea bucata de cod este din scriptul Get_social_chars_frequency.py si realizeaza numararea URI-urilor(pentru comparare) din setul de date si frecventa caracterelor speciale("dict" este un dictionar cu cheile fiind caracterele speciale utilizate in limbaj):

```
with open(args.f, 'r') as fd:
    for lines in fd.readlines():
        lines_nr += 1
        line = lines.strip()
        for keys in dict:
```

```

if keys in line:
    dict[keys] += 1

```

Pentru indentificarea caracterelor relevante din limbajul SQL in comparatie cu cele folosite intr-un URL obijnuit, s-a ales numararea frecventei de aparitie a acestora in URL-uri normale, dar si in URL-uri ce contin atacuri de SQL injection. Numararea se face alternativ, rezultatele fiind furnizate ca doua seturi de date separate, bucata anterioara de cod reprezentand procesul doar pentru una dintre cele doua categorii. Dictionarul referit in cod este alcatuit dintr-un dictionar ce are ca si chei valoarea caracterelor speciale cautate, iar ca valoarea acestea sunt initializate pe zero pentru a fi incrementate o data cu numarul de aparitii ale caracterelor cautate. Pentru uniformizarea setului de date, intrucat distributia acestor caractere nu este tot timpul uniforma, se tine cont si de numarul de linii(pe fiecare linie se afla un URI distinct) in care au fost indentificate frecventele lor de aparitie.

Dupa determinarea caracterelor relevante, acestea au fost completate manual in fisierul `key_features`, fisier ce insumeaza toate trasaturile ce sunt folosite in antrenarea modelului(fiecare linie contine o trasatura, numarul liniei fiind si indicele de referinta a trasaturii).

Pentru obtinerea datelor intr-un format ce poate fi procesat de biblioteca `libsvm` [18] a fost necesara transformarea acestora intr-un anumit format:

```

+1 23:1 37:4 103:2
-1 54:1 77:1

```

Convertirea URI-urilor in formatul de mai sus este realizata de scriptul `URI-file_to_features.py`. Formatul de mai sus este reprezentarea fiecarui URL in functie de tipul acestuia si indicele trasaturii gasite in interiorul sau precum si frecventa de aparite a acestora(tip trasatura:frecventa trasatura:frecventa). Urmatoarele bucati de cod sunt extrasa din fisierul `URI-file_to_features.py` si realizeaza conversia unui URI in echivalentul sau in trasaturi:

```

if keys in uri:
    if keywords_list.index(keys) > 184:
        keywords[keys] = uri.count(keys)
    if not uri.count(keys) == 0:
        ok = True

```

In lista cu trasaturi, primele 184 de pozitii corespund cuvintelor cheie ale limbajului SQL, urmatoarele 8 pozitii apartinand trasaturilor de tipul caracter special(determinate la pasul anterior de scriptul `"Get_special_chars_frequency.py"`). In cazul in care o astfel de trasatura este gasita intr-un URI, pur si simplu se gaseste numarul total de aparitii a caracterului respectiv in URI.

```
sub_uri = uri.split(keys)
for index, ele in enumerate(sub_uri):
    if index + 1 < len(sub_uri):
        if ele and sub_uri[index + 1] and not ele[-1].isalpha()
        and not sub_uri[index+1][0].isalpha():
            keywords_aux[keys] += 1
    elif not ele and sub_uri[index - 1]:
        keywords_aux[keys] += 1
```

Pentru trasaturile de pe primele 184 de pozitii, adica cuvintele cheie ale limbajului SQL, s-a abordat o logica mai complexa de procesare. O data ce un cuvant este gasit intr-un URI, pentru fiecare aparitie a acestui cuvant se verifica ca primul caracter anterior cuvantului si primul urmator cuvantului sa nu fie litera, astfel eliminandu-se cazurile in care unele cuvinte sunt incluse in altele(ex: **all** si **deallocate**).

In urma procesarii fisierelor de date, rezulta un fisier(all_URI_features) ce insumeaza toate datele utilizate in antrenarea modelului(exemple atat pozitive cat si negative), insa sub forma prezentata anterior(tip trasatura:frecventa trasatura:frecventa). Acest fisier este impartit in doua fisiere cu proportia de 70% pentru Train_set si 30% Tes_set de scriptul get_Trainset_Testset.py. cu scopul de a obtine un grup de date atat pentru antrenarea modelului cat si pentru testarea ulterioara a acestuia.

Pentru antrenarea modelului, s-a folosit biblioteca open source libsvm. Aceasta biblioteca furnizeaza fisierele necesare antrenarii modelului de svm si de prezicere a unui rezultat pe un model existent. Apelul catre executabilul "svm-train.exe" se face prin intermediul scriptului de python Train_SQLi_model.py. Executabilul de windows furnizat de biblioteca libsvm poate fi apelat conform exemplului urmator:

```
svm-train.exe [options] training_set_file [model_file]
```

6.2.3 Interfata utilizator

Componenta de interfata reprezinta atat proiectul in .net in care a fost realizata interfata sistemului, dar si fisierele/script-urile ce au rolul de a interconecta rezultatele celor doua proiecte anterioare cu acesta.

Scriptul Proxy.py incapsuleaza toate elemntele, din partea de back end, a proiectului propus. Acesta de asemenea furnizeaza ca output intr-un format specific(format ce este cunoscut si interpretat de interfata grafica) toate evenimentele ce apar in timpul rularii sistemului. In acest script se realizeaza instantierea elementului de reverse proxy cu parametri de adrese IP si porturi aferente, precum si furnizarea metodelor de detectie componentei de reverse proxy. Initializarea elementului de reverse proxy se realizeaza cu ajutorul codului oferit de biblioteca twisted. Acest cod este prezentat in "Anexa 1" a acestei lucrari.

In urmatoarele imagini sunt prezentate posibilele pagini ale interfatei utilizator precum si o scurta descriere ce ilustreaza functionalitatile oferite de paginile respective pentru utilizator.

]

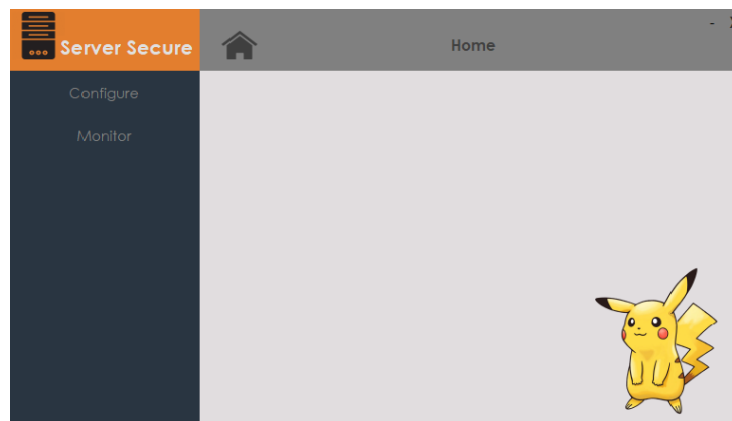


Figura 6.4: Meniul "Home" in interfata grafica

Figura 8.1 prezinta design-ul paginii de "Home" din interfata grafica pusa la dispozitie utilizatorilor sistemului. Acesta pagina este afisata la lansarea in executie a aplicatiei, dar poate fi acesata de catre utilizator si prin apasarea butonului in forma de "casuta" din dreapta logoului aplicatiei.



Figura 6.5: Meniul "Configure" in interfata grafica

Figura 8.2 prezinta design-ul paginii de "Configure" din interfata grafica, in care utilizatorul sistemului poate sa seteze interfetele si porturile care vor fi tratate in timpul rularii. Partea superioara a meniului reprezinta setarile aferente partii de server, precum sugereaza si eticheta din stanga sus "Server Settings". In partea inferioara se pot seta detaliile interfetelor ce se pot conecta la server-ul setat mai sus. (aceste interfete pot fi multiple).

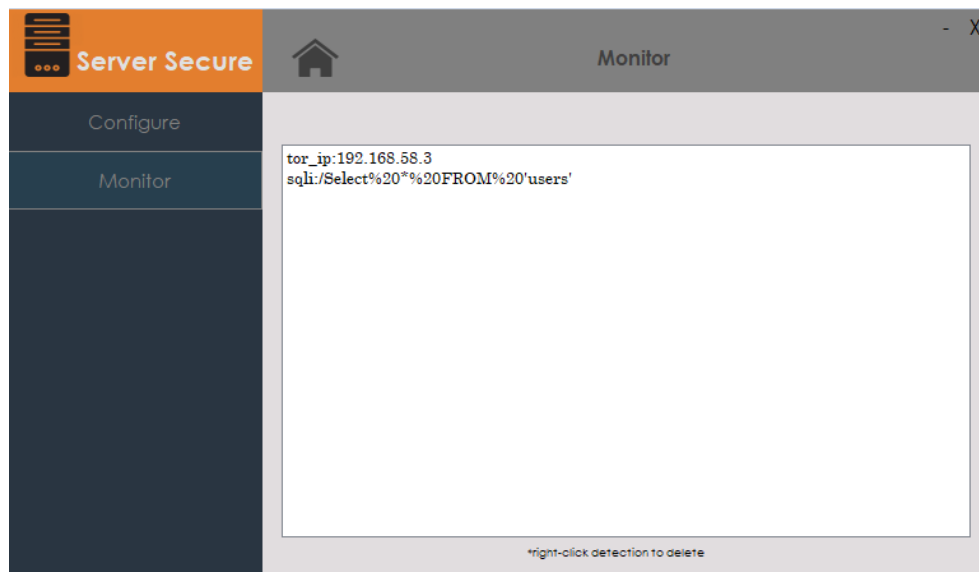


Figura 6.6: Meniul "Monitor" in interfata grafica

Figura 8.3 prezinta design-ul paginii de "Monitor" din interfata grafica, in care utilizatorul poate sa urmareasca activitatea sistemului in timpul rularii. In aceasta fereastră apar evenimentele aparute in timpul rularii aplicatiei, evenimente ce sunt sub forma de detectii.

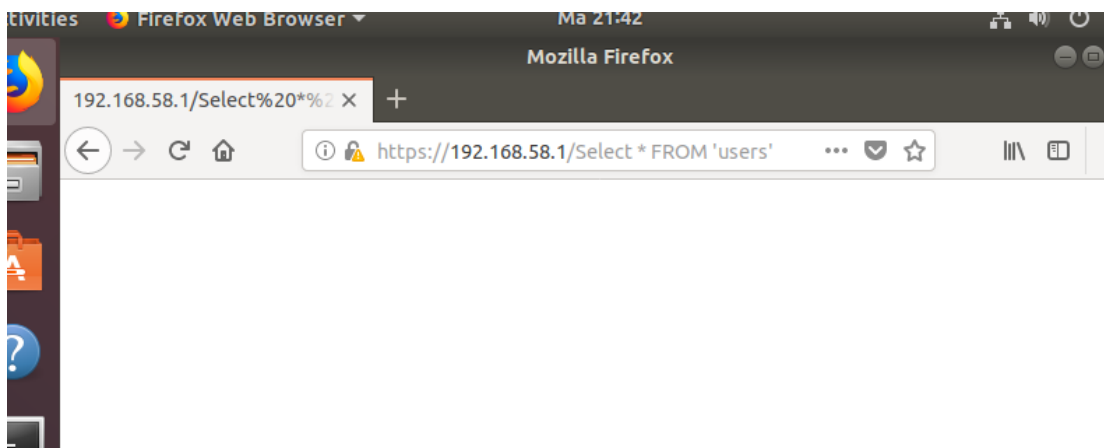


Figura 6.7: Raspunsul pentru SQL injection URL

Figura 6.7 prezinta cum arata raspunsul primit de la server de catre un utilizator dupa trimiterea unui request catre server, cu intentia de a realiza un atac de tipul SQL injection. Utilizatorului i se returneaza o pagina goala.

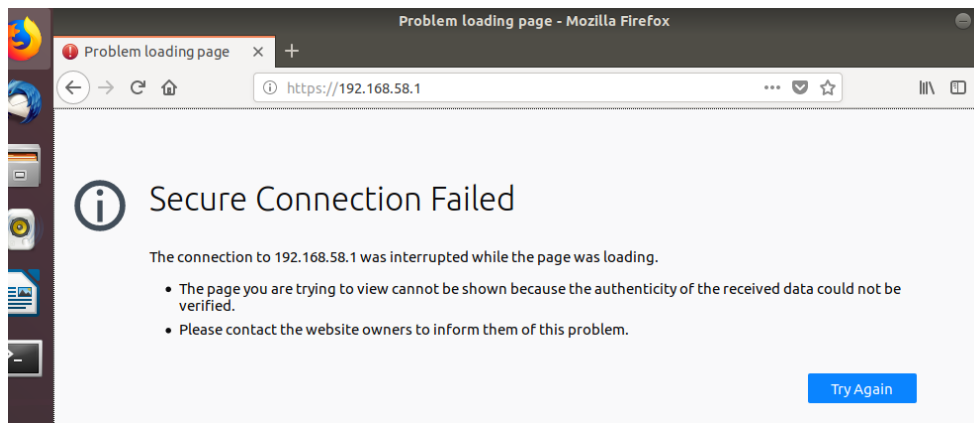


Figura 6.8: Principalele module ale sistemului propus

Figura 6.8 prezinta cum arata raspunsul primit de la server de catre un utilizator al retelei Tor ce intentioneaza sa se conecteze la acesta. Acestor utilizatori li se refuza conexiunea.

Capitolul 7

Teste și rezultate experimentale

7.1 Teste de funcționalitate

În testarea sistemului s-a pus accentul pe testare celor două funcționalități de protecție împotriva atacurilor de SQL injection și a utilizatorilor de Tor.

Pentru testarea modelului folosit în prevenirea atacurilor de SQL injection s-a folosit setul de date de test specificat și în capitolul 6. Acest set a fost obținut din setul inițial de date de antrenare, acesta fiind împărțit în 2 seturi separate cu proporția de 70%(set antrenare) și respectiv 30%(set testare).

Tip set de testare	Preziceri corecte	Dimensiune set	Acuratete(%)
Clean & infected	187596	189278	99.11
Clean	5466	6258	87.34
Infected	182130	193020	99.51

Pe baza tabelului de mai sus se pot observa diferențe majore de performanță între detectia pe setul "Clean" și pe "Infected". Aceste diferențe, respectiv scăderi de performanță în cazul setului Clean, se datorează dimensiunii mult mai mici a setului de Clean folosit și în antrenarea modelului.

Pentru testarea eficienței sistemului de blocare a adreselor IP ale rețelei Tor, s-a realizat o referință între datele colectate de modulul de monitorizare a activității rețelei Tor.

În prima diagramă este prezentată diferența procentuală dintre adresele IP cu un uptime mai mare de 7 zile în ultima lună și cele cu un uptime mai mic. În cea de a doua diagramă este evidențiată diferența dintre uptime-ul total al acestor adrese IP din ultima lună. Printr-o analiză simplă în paralel a datelor din cele două diagrame, se poate observa că deși doar 20.31% din IP-urile folosite de rețeaua Tor sunt blocate, din timpul total de uptime din ultima lună, 75.6% aparține acestor adrese IP.

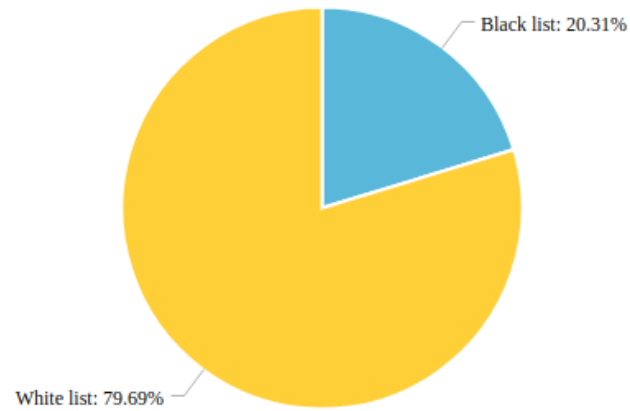


Figura 7.1: Raportul dintre numarul IP-urilor de pe Blacklist si Whitelist dintr-o luna

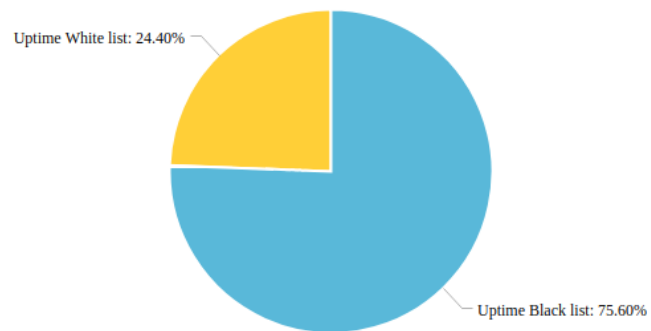


Figura 7.2: Raportul dintre uptime-ul IP-urile de pe Blacklist si Whitelist din aceasi luna

7.2 Teste de performanță

Sistemul propus a fost testat pe două configurații diferite pentru PC-ul gazda: Intel Core i7-6600U CPU cu 4 nuclee și o frecvență de 2.6 GHz, cu 16 GB RAM DDR4 și i7-4790k CPU cu 4 nuclee și o frecvență maximă de 4.0 GHz, cu 16 GB RAM DDR4. Ambele sisteme furnizând mult mai multe resurse decât cele necesare unei funcționări optime. În ceea ce privește resursele minime necesare, acestea trebuie să fie cele necesare rularii unui sistem de operare Windows 10: un procesor cu o frecvență mai mare de 1.0 GHz și o memorie mai mare sau egală cu 2 GB de RAM. În ceea ce privește capacitățile sistemului de a suporta conexiuni exterioare, acesta nu a fost testat decât manual, prin intermediul unor mașini virtuale.

Capitolul 8

Manual utilizator

8.1 Instalarea proiectului

Pentru instalarea sistemului, atat timp cat toate dependintele acestuia sunt implinite, utilizatorul nu mai trebuie sa faca nimic. Toate fisierele necesare sistemului au fost impachetate in modulul principal, singurul pas ce poate fi executat de utilizator este crearea unui shortcut catre executabilul ce lanseaza in executie sistemul.

8.2 Utilizare

Dupa deschiderea aplicatiei, utilizatorul este intampinat de meniul de Home al acesteia. In urmatoarea fereastră se poate observa designul acestuia:

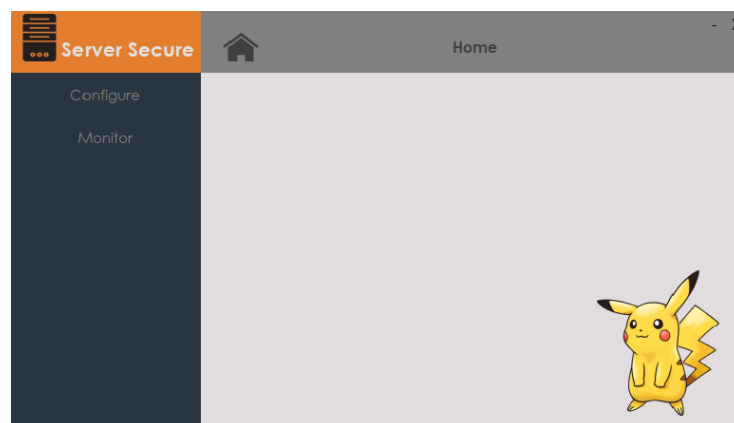


Figura 8.1: Meniul "Home" in interfata grafica

In partea superioara a ferestrei se afla in permanenta numele ferestrei curente(in cazul de fata fereastră Home). In meniul prezent in stanga ferestrei sunt situate celelate

doua ferestre disponibile utilizatorului: Configure si Monitor. Utilizatorul poate sa navigheze intre aceste ferestre dand un click pe numele ferestrelor, respectiv pe pictograma in forma de casuta pentru meniul Home.

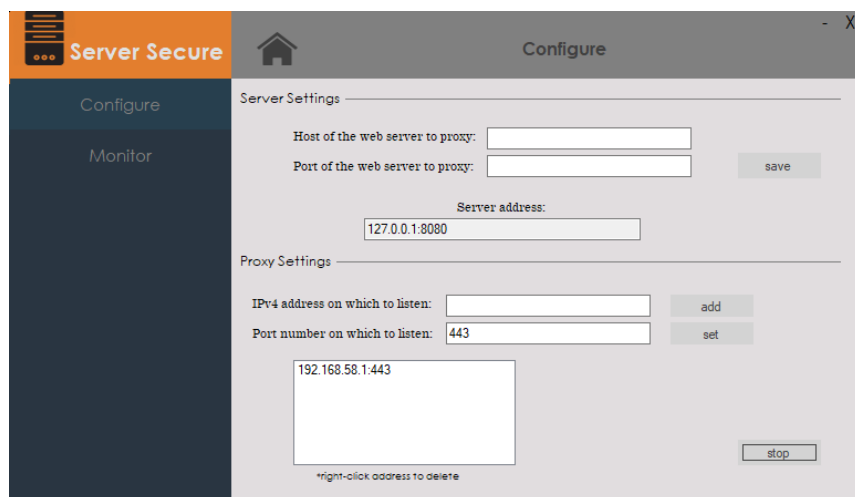


Figura 8.2: Meniul "Configure" in interfata grafica

In fereasra Configure, utilizatorul poate sa seteze parametrii de rulare a sistemului. Acestuia i se prezinta doua rubrici: Server Settings si Proxy Settings. In partea de Server Settings, utilizatorul seteaza datele server-ului: adresa IP a acestuia si portul aferent pe care acesta accepta conexiunii. Salavarea sau suprascrierea acestor date se realizeaza prin apasarea butonului "save" din rubrica respectiva. In partea de Proxy Settings, utilizatorul poate sa introduca mai multe adrese IP si un port, pe care sistemul sa accepte conexiuni si sa le redirectioneze catre server. Dupa setarea parametrilor de mai sus, sistemul se poate porni/opri prin apasarea butonului din dreapta jos(cu textul "start"/"stop" dupa caz).

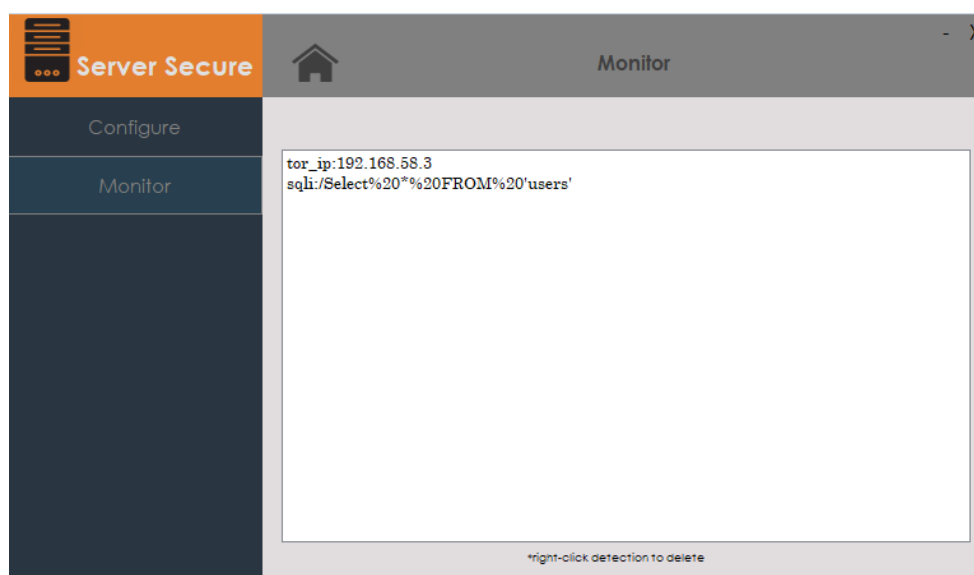


Figura 8.3: Meniul "Monitor" in interfata grafica

In fereastra de Monitor, utilizatorul poate sa urmareasca activitatea sistemului. In cazul in care sistemul detecteaza un eveniment, acesta este afisat in interfata grafica in aceasta fereastre(conform exemplului din imagine). Daca utilizatorul doreste stergerea evenimentelor antrioare, aceasta se poate realiza prin click dreapta pe eveniment.

Capitolul 9

Conluzii

9.1 Privire de ansamblu asupra sistemului

În urma eforturilor depuse, s-a realizat un sistem de prevenirea a intruziunilor care îndeplinește aproape în întregime obiectivele inițial propuse, mici inconveniente fiind la partea de performanță în detecție a sistemului. În ceea ce privește performanța sistemului, în cazul detecției atacurilor de SQL injection (precum se poate vedea și în capitolul 7) procentul de fals pozitiv este mult mai mare decât se intenționa inițial (12.66% în loc de 3-4%), însă acest lucru datorându-se în principal setului mic de date de antrenare avute la dispoziție.

Prin implementarea detecției atacurilor SQL injection folosind tehnici de machine learning, s-au dobândit cunoștințe valoroase și experiența ce pot fi folosite în viitoare proiecte. Întrucât în practică, abordarea acestui subiect a reprezentat un lucru nou, o mare parte din timpul investit în dezvoltarea sistemului a reprezentat documentarea și acumularea de noi cunoștințe și experiența pentru rezolvarea unor probleme cu tehnici de machine learning.

9.2 Dezvoltări ulterioare

Datorită unei abordări modulare a dezvoltării sistemului, aceștia îi pot fi adăugate cu ușurință noi funcționalități.

Sistemul prezintă două posibilități majore de dezvoltări ulterioare, acestea fiind aferente celor două tipuri de protecție implementate. În cazul protecției împotriva adreselor IP malicioase, lista folosită momentan și poate extinde cu ușurință sau prin mici modificări în cod se pot adăuga noi liste.

Protecția bazată pe detecția unui anumit conținut în URL poate fi cu ușurință extinsă, prin adăugarea de noi modele de machine learning sau noi logici de detecție, ambele cazuri necesitând mici modificări în codul sursă.

Bibliografie

- [1] J. C. Villanueva, “Top 8 Benefits of a Reverse Proxy,” <https://www.jscape.com/blog/bid/87841/Top-8-Benefits-of-a-Reverse-Proxy>, Aug 2012.
- [2] “OWASP Top 10 Application Security Risks - 2017,” https://www.owasp.org/index.php/Top_10-2017_Top_10, Last accessed: September 30st, 2018.
- [3] J. V. William G.J. Halfond and A. Orso, “A Classification of SQL Injection Attacks and Countermeasures,” 2006, College of Computing Georgia Institute of Technology, Proceedings of the IEEE International Symposium on Secure Software Engineering.
- [4] M. Prince, “The Trouble with Tor,” <https://blog.cloudflare.com/the-trouble-with-tor/>, Mar 2016.
- [5] R. Bassett, C. Urrutia, and N. Ierace, “Intrusion prevention systems,” <https://dl.acm.org/citation.cfm?id=1071927>, June 2005, ACM New York, NY, USA .
- [6] “What is an intrusion prevention system?” <https://www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-prevention-system-ips>.
- [7] J. Snyder, “Information Security,” <https://searchsecurity.techtarget.com/Do-you-need-an-IDS-or-IPS-or-both>.
- [8] M. Rouse, “Network management and monitoring: The evolution of network control,” <https://searchnetworking.techtarget.com/definition/network-analyzer>.
- [9] B. Hendricks, “Intrusion Prevention System (IPS): Definition & Types,” <https://study.com/academy/lesson/intrusion-prevention-system-ips-definition-types.html>.
- [10] C. Paquet, “Implementing cisco ios network security (iins): (ccna security exam 640-553) (authorized self-study guide),” Apr 2009, by Cisco Press.
- [11] K. Rajesh, “An overview of ips – intrusion prevention system and types of network threats,” <https://www.excitingip.com/626/an-overview-of-ips-intrusion-prevention-system-and-types-of-network-threats/>, October 2009.

- [12] E. H. Cheon, Z. Huang, and Y. S. Lee, "Preventing SQL Injection Attack Based on Machine Learning," 2013, Department of Computer Engineering, Woosuk University and Department of Computer Information Engineering, Kunsan National University .
- [13] F. Valeur, D. Mutz, and G. Vigna, "A Learning-Based Approach to the Detection of SQL Attacks," 2005, Reliable Software Group, Department of Computer Science University of California Santa Barbara .
- [14] P. Winter and S. Lindskog, "How the Great Firewall of China is Blocking Tor," 2012, Karlstad University .
- [15] R. Singh, R. Nithyanand², S. Afroz, P. Pearce, M. C. Tschantz, P. Gill¹, and V. Paxson, "Characterizing the Nature and Dynamics of Tor Exit Blocking," 2017, University of Massachusetts – Amherst, Stony Brook University, University of California – Berkeley, International Computer Science Institute .
- [16] "Framework pentru programarea retelelor scris in python." [Online]. Available: <https://twistedmatrix.com/documents/8.1.0/api/twisted.html>
- [17] "Librarie pentru procesarea de cod html/xml scris in python." [Online]. Available: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
- [18] "Software integrat cu suport pentru svm." [Online]. Available: <https://www.csie.ntu.edu.tw/~cjlin/libsvm/>
- [19] "Documentation for python 3.7.1rc1." [Online]. Available: <https://docs.python.org/3/>
- [20] "Documentation for python 2.7.15." [Online]. Available: <https://docs.python.org/2.7/>
- [21] Tim Gray, "Python 2 to python 3: why, and how hard can it be?" [Online]. Available: <https://optimalbi.com/blog/2018/01/19/python-2-to-python-3-why-and-how-hard-can-it-be/>
- [22] "Tabela cu toate caracterele ascii si codurile lor specifice in diferite baze." [Online]. Available: <https://www.asciitable.com/>
- [23] "Caracterele speciale ce pot fi folosite intr-o interogare sql." [Online]. Available: https://docs.oracle.com/cd/A97630_01/text.920/a96518/cqspcl.htm
- [24] "Cuvinte rezervate specifice limbajului sql." [Online]. Available: <https://docs.microsoft.com/en-us/sql/t-sql/language-elements/reserved-keywords-transact-sql?view=sql-server-2017>
- [25] "A practical guide to support vector classification." [Online]. Available: <https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>

- [26] “Statusul nodurilor utilizate de tor.” [Online]. Available: <https://torstatus.blutmagie.de/index.php>
- [27] by NGINX Page, “What is a reverse proxy server?” [Online]. Available: <https://www.nginx.com/resources/glossary/reverse-proxy-server/>
- [28] Andrew Yan-Tak Ng, “Machine learning by stanford university.” [Online]. Available: <https://www.coursera.org/learn/machine-learning/home/welcome>
- [29] by Firewalls.com Page, “How intrusion prevention systems (ips) work in firewall.” [Online]. Available: <https://community.spiceworks.com/topic/362007-how-intrusion-prevention-systems-ips-work-in-firewall>

Anexa A

Diverse anexe

Anexa B

Demonstrații matematice detaliate (dacă există)

Anexa C

Pseudo-cod sau cod (dacă există)

```
/** Maps are easy to use in Scala. */
object Maps {
  val colors = Map("red" -> 0xFF0000,
                   "turquoise" -> 0x00FFFF,
                   "black" -> 0x000000,
                   "orange" -> 0xFF8040,
                   "brown" -> 0x804000)

  def main(args: Array[String]) {
    for (name <- args) println(
      colors.get(name) match {
        case Some(code) =>
          name + " has code: " + code
        case None =>
          "Unknown color: " + name
      }
    )
  }
}
```


Anexa D

Articole publicate