



Departamentul Automatică și Informatică Industrială

**Facultatea Automatică și Calculatoare
Universitatea Națională de Știință și Tehnologie
POLITEHNICA București**



LUCRARE DE DIPLOMĂ

Sistem de detecție a intruziunii într-o rețea

Coordonator

Conf.dr.ing. Daniel-Marian MEREZEANU

Absolvent

Eduard DONEA

2025

Cuprins

1. Introducere.....	3
2. Prezentarea domeniului din care face parte lucrarea	4
3. Descrierea problemei și a soluției.....	7
3.1. Problema abordată	7
3.2. Analiza comportamentului utilizatorilor în fața riscurilor cibernetice	8
3.3. Arhitectura sistemului propus.....	10
3.4. Integrarea componentelor AI și XAI	11
3.5. Caz de utilizare – diagrama UML	11
3.6. Soluții existente pe piață pentru IDPS	12
4. Documentație tehnică	15
4.1. Echipamente utilizate (hardware).....	15
4.1.1. Topologia generală a sistemului	16
4.1.2. Descrierea echipamentelor hardware utilizate.....	17
4.1.3. Alegerea și utilizarea switch-ului	23
4.1.4. Conectica fizică și cabluri utilizate.....	24
4.1.5. Sursa de internet și integrarea în rețeaua căminului	25
4.2. Tehnologii utilizate (software)	25
4.2.1. Motorul de detecție	25
4.2.2. Serverele Flask pentru integrarea backend-ului	27
4.2.3. Bazele de date SQLite	31
4.2.4. Interfața <i>frontend</i> – structura vizuală și funcțională a sistemului.....	34
4.2.5. Componenta AI a sistemului – detaliere.....	37
4.3. Rezultate obținute	38
4.3.1. Evaluarea funcționalității aplicației în timp real.....	38
4.3.2. Performanța clasificatorului AI	40
4.3.3. Observații privind alertele imposibil de clasificat.....	42
4.3.4. Concluzii asupra rezultatelor	43
5. Concluzii și direcții de dezvoltare ulterioară.....	44
5.1. Avantajele soluției propuse	44
5.2. Direcții de dezvoltare viitoare	45
5.3. Concluzia finală.....	46
6. Bibliografie.....	47
7. Anexe.....	49

1. Introducere

În contextul accelerării digitalizării și al creșterii numărului de dispozitive conectate la rețea, securitatea cibernetică a devenit o componentă esențială a infrastructurii moderne, atât în mediile comerciale, cât și în cele personale. **Rețelele locale**ⁱ sunt expuse zilnic la riscuri precum atacuri de tip malware, tentative de acces neautorizat sau exploatarea de vulnerabilități de tip „ziua-zero”ⁱⁱ. Acestea din urmă sunt adesea dificil de detectat cu metode tradiționale, bazate exclusiv pe semnături.

Urmărind acest context, **sistemele de detecție și prevenire a intruziunilor** (cunoscute și ca *IDPS*), devin esențiale. Alegerea temei acestei lucrări pornește de la nevoia reală de a dezvolta un sistem de securitate cibernetică accesibil și eficient, destinat în special utilizatorilor fără experiență tehnică. Lucrarea de față propune un IDPS local, care integrează **inteligența artificială (AI)** pentru a crește performanța detecției și a reduce numărul de alerte fals-pozitive. În plus, pentru a crește transparența deciziilor automatizate, este inclusă o componentă de **inteligență artificială explicabilă** (sau *XAI*), care oferă justificări vizuale pentru alertele generate.

În structura lucrării, vor fi prezente următoarele capitole:

- **Capitolul 2** prezintă domeniul securității rețelelor și rolul sistemelor IDPS moderne, evidențiind limitele soluțiilor tradiționale și motivația utilizării inteligenței artificiale.
- **Capitolul 3** detaliază problema abordată, contextul rețelelor locale și arhitectura propusă pentru un IDPS inteligent și accesibil.
- **Capitolul 4** constituie documentația tehnică, descriind echipamentele, tehnologiile software utilizate, algoritmi, componentele aplicației dezvoltate și rezultatele obținute în urma testării.
- **Capitolul 5** sintetizează concluziile și propune direcții viitoare de dezvoltare, inclusiv componenta activă de prevenire (IPS), distribuția ca soluție *open-source* și posibile extinderi educaționale.
- **Capitolul 6** include bibliografia utilizată, iar **Capitolul 7** oferă anexe relevante, precum capturi de ecran, fragmente de cod, chestionarul aplicat și diagrama arhitecturii implementate.

Prin această lucrare, voi urmări oferirea unei soluții care să fie funcțională, scalabilă și prietenoasă, capabilă să aducă un plus de siguranță în rețelele locale, sprijinind utilizatorii mai puțin experimentați în fața pericolelor cibernetice moderne.

2. Prezentarea domeniului din care face parte lucrarea

Într-un context global în care interconectarea digitală crește exponențial, protejarea rețelelor informatice a devenit o componentă fundamentală a infrastructurii tehnologice. Fie că e vorbim despre instituții, companii sau utilizatori individuali, riscurile cibernetice sunt omniprezente, iar impactul unui atac reușit poate fi semnificativ atât din punct de vedere tehnic, cât și economic. Pe măsură ce tot mai multe dispozitive sunt conectate la rețea – de la stații de lucru și servere, la dispozitive inteligente de tip *IoT* – complexitatea amenințărilor cibernetice crește, iar sistemele tradiționale de apărare se dovedesc adesea insuficiente.

Una dintre cele mai importante componente din arsenalul de securitate al unei rețele, așa cum este descris și în [1], este reprezentată de **sistemele de detecție și prevenire a intruziunilor**, cunoscute sub denumirea de **IDPS** (*Intrusion Detection and Prevention Systems*). Acestea sunt concepute pentru a monitoriza traficul de rețea, a identifica activități suspecte și a furniza alerte sau acțiuni de blocare în fața unor comportamente malițioase. Sistemele IDPS evoluează din cele clasice de **detecție** (*IDS – Intrusion Detection Systems*), prin adăugarea unei componente pro-active – **prevenire** (*IPS – Intrusion Prevention Systems*) – care permite blocarea automată a unor atacuri în desfășurare și eliminarea ulterioară a lor.

Din punct de vedere funcțional, IDPS-urile pot fi clasificate în funcție de punctul în care intervin în monitorizarea rețelei. Sistemele bazate pe **gazdă** (*HIDPS*¹ sau *host-based IDPS*) sunt concepute pentru a analiza activitatea unui singur dispozitiv, incluzând jurnalizarea proceselor, fișierele de sistem și comportamentul aplicațiilor. În contrast, sisteme bazate pe **rețea** (*NIDPS*² sau *network-based IDPS*) monitorizează traficul care circulă între mai multe dispozitive, analizând pachetele de date direct la nivelul infrastructurii de rețea. Cele două abordări sunt combinate într-un sistem hibrid, în anumite implementări moderne (ceea ce am realizat și eu la rândul meu și voi dezvolta în următoarele capitole), care oferă atât vizibilitate locală, cât și contextuală asupra evenimentelor.

¹ Figura descriptivă pentru acest tip de sistem se găsește aici: https://www.researchgate.net/figure/Proposed-Host-based-Intrusion-Detection-and-Prevention-System-Model_fig3_271070098 [Accesat: 2025]

² Figura descriptivă pentru acest tip de sistem se găsește aici: https://www.researchgate.net/figure/Network-based-IDPS-Architecture_fig1_265248885 [Accesat: 2025]

Tradițional, sistemele IDPS se bazează pe un mecanism de detecție bazat pe semnături – adică identifică modele cunoscute de atacuri informatice stocate într-o bază de date actualizabilă. Această abordare este eficientă în cazul amenințărilor deja cunoscute, dar ridică probleme majore în fața atacurilor de tip „zero-day”, care exploatează vulnerabilități recent descoperite, pentru care nu există încă o semnătură sau o soluție disponibilă pentru a-l „dizolva”. De asemenea, un alt neajuns important al sistemelor bazate pe reguli îl reprezintă numărul ridicat de alerte fals-pozitive, care pot genera confuzie și consum de resurse în analiza incidentelor.

Pornind de la concluziile enunțate în [2], pentru a depăși aceste limite, în ultimii ani s-a conturat o direcție clară de cercetare și dezvoltare în integrarea metodelor de inteligență artificială în arhitectura sistemelor IDPS. Modelele de **învățare automată**ⁱⁱⁱ au capacitatea de a învăța din date istorice sau în timp real, identificând modele complexe în comportamentul rețelei. Aceste modele pot clasifica traficul în categorii precum normal sau malițios, chiar și în lipsa unor semnături prestabilite, adaptându-se constant la modificările mediului digital.

Învățarea automată poate fi implementată în mai multe moduri. **Algoritmii supervizați**, precum arborii de decizie, regresia logistică sau **rețelele neuronale**^{iv} de tipul celor cu multe clase, pot fi antrenați în seturi de date etichetate pentru a învăța diferențele între un trafic benign și atacuri. În același timp, **metodele nesupravegheate**, precum auto-codificatoarele^v sau algoritmii de grupare^{vi}, sunt utile în scenariile în care datele nu sunt clasificate, dar se dorește descoperirea anomaliilor pe baza distribuției statistice. Mai mult, *rețelele neuronale convoluționale*^{vii} au aplicații în identificarea tiparelor spațiale, în timp ce *rețelele neuronale recurente*^{viii} sunt preferate în analiza secvențelor temporale, cum ar fi fluxurile de trafic în rețea.

Totuși, odată cu adoptarea pe scară largă a modelelor AI, a apărut o nouă provocare: lipsa transparenței deciziilor. Sistemele de inteligență artificială, în special cele bazate pe **învățare profundă**^{ix}, sunt adesea percepute ca fiind opace („cutie-neagră” sau „black-box”), întrucât nu oferă explicații clare pentru predicțiile lor. Această lipsă de interpretabilitate este o problemă majoră în domeniul securității informatice, unde este esențial să se înțeleagă cauza unei alerte pentru a putea reacționa eficient.

Astfel, a apărut un subdomeniu complementar denumit **inteligență artificială explicabilă** (*XAI – Explainable AI*), descris și în [3], sau, pentru a conferi o traducere, o componentă de explicare a deciziei luate automat, bazată pe AI, care propune să aducă transparență și încredere. Printre cele mai utilizate tehnici XAI se numără *SHAP*³, care

³ Cunoscut și ca *SHapley Additive exPlanations* – resursă disponibilă la: https://shap.readthedocs.io/en/stable/example_notebooks/overviews/An%20introduction%20to%20explainable%20AI%20with%20Shapley%20values.html [Accesat: 2025]

calculează contribuția fiecărei caracteristici de intrare la rezultatul final, *LIME*⁴, care oferă explicații locale pentru decizii punctuale, și *LRP*⁵, care analizează influența pe starturile unei rețele neuronale.

Domeniul în care se încadrează această lucrare este definit, așadar, de convergența între securitatea cibernetică, analiza traficului de rețea și aplicarea metodelor moderne de inteligență artificială, precum și acelei de explicare a deciziilor. Lucrarea își propune să dezvolte un sistem IDPS dedicat rețelelor locale, care combină avantajele detecției automate bazate pe AI, cu claritatea deciziilor oferită de XAI, într-un mod accesibil și intuitiv pentru utilizatorii mai puțin tehnici.

⁴ Cunoscut și ca *Local Interpretable Model-agnostic Explanations* – resursă disponibilă la: <https://c3.ai/glossary/data-science/lime-local-interpretable-model-agnostic-explanations/> [Accesat: 2025]

⁵ Cunoscut și ca *Layer-wise Relevance Propagation* – resursă disponibilă la: <https://praveenkumar2909.medium.com/overview-of-explainable-ai-and-layer-wise-relevance-propagation-lrp-cb2d008fec57> [Accesat: 2025]

3. Descrierea problemei și a soluției

Privind actualitatea, rețelele locale rămân un pilon fundamental în infrastructura digitală a instituțiilor, organizațiilor sau chiar a locuințelor moderne. Deși atenția mediului de cercetare și a industriei se concentrează tot mai mult pe arhitecturi native în *cloud*⁶ și sisteme distribuite, multe puncte de acces vulnerabile rămân în rețelele LAN⁽ⁱ⁾, adesea prost configurate, nesecurizate sau complet lipsite de monitorizare. Această realitate este cu atât mai problematică în cazul utilizatorilor neexperimentați, care nu au cunoștințele sau instrumentele necesare pentru a detecta o potențială intruziune, cu atât mai puțin să o prevină sau să reacționeze corespunzător.

3.1. Problema abordată

Conform relatărilor din [4], riscurile cibernetice care vizează rețelele locale sunt diverse: atacuri de tip *spoofing*⁷, *sniffing pasiv*⁸, încercări de *brute-force*⁹ asupra porturilor deschise, propagarea de malware prin dispozitive compromise, dar și scenarii mai avansate, precum atacuri de tip *lateral movement*¹⁰ după compromiterea unui dispozitiv slab protejat. În multe cazuri, dispozitive precum imprimante inteligente, camere IP sau chiar laptopuri personale devin porți de acces către întreaga rețea, mai ales dacă nu sunt segmentate corespunzător sau nu au soluții de protecție active. Pornind de la concluziile din [5], literatura de specialitate subliniază că atacurile asupra rețelelor locale sunt adesea subestimate, deoarece nu implică infrastructuri mari sau vizibile, dar pot avea consecințe semnificative în propagarea ulterioară a amenințărilor.

O altă problemă esențială constă în lipsa accesibilității soluțiilor de securitate existente pentru utilizatorii fără experiență tehnică. Majoritatea sistemelor IDPS comerciale sau *open-source* presupun un grad ridicat de configurare, interpretare a alertelor sau chiar o expertiză în analiza de trafic de rețea. De exemplu, un utilizator obișnuit al unui laptop/PC, conectat printr-un simplu cablu de internet sau printr-o rețea *Wi-Fi*, nu va putea înțelege semnificația unei alerte de tip „*TCP NULL scan detected*” sau „*ICMP Ping detected*”, cu

⁶ Termen tehnic provenit din englezescul *cloud-native* – resursă disponibilă la:

<https://www.oracle.com/ro/cloud/cloud-native/what-is-cloud-native/> [Accesat: 2025]

⁷ Atac cibernetic – resursă disponibilă la: <https://www.cisco.com/site/us/en/learn/topics/security/what-is-spoofing.html> [Accesat: 2025]

⁸ Atac cibernetic – resursă disponibilă la: <https://www.geeksforgeeks.org/ethical-hacking/what-is-sniffing-attack-in-system-hacking/> [Accesat: 2025]

⁹ Atac cibernetic – resursă disponibilă la: <https://www.kaspersky.com/resource-center/definitions/brute-force-attack> [Accesat: 2025]

¹⁰ Atac cibernetic – resursă disponibilă la: <https://www.crowdstrike.com/en-us/cybersecurity-101/cyberattacks/lateral-movement/> [Accesat: 2025]

atât mai puțin să știe ce acțiune trebuie să întreprindă. În plus, de multe ori, interfețele acestor sisteme sunt aglomerate, tehnice și neprietenoase pentru publicul general.

Pornind de la această dublă problemă – expunerea rețelelor locale și inaccesibilitatea soluțiilor existente – această lucrare propune dezvoltarea unui sistem IDPS distribuit, orientat atât către detecție cât și către interpretabilitate, cu o componentă de interfață intuitivă, adaptată și pentru persoanele non-tehnice. Soluția propusă îmbină mai multe componente funcționale, fiecare cu un rol clar în monitorizarea, clasificarea și prezentarea riscurilor de securitate în rețea.

3.2. Analiza comportamentului utilizatorilor în fața riscurilor cibernetice

Pentru a înțelege mai bine nevoia unui sistem de detecție și prevenire a intruziunilor accesibil și prietenos cu utilizatorii neexperimentați, a fost realizată o cercetare exploratorie bazată pe un chestionar anonim. Acesta a vizat obiceiurile, percepțiile și nivelul de conștientizare al lor în ceea ce privește securitatea în mediul digital. Rezultatele au oferit o imagine relevantă asupra lacunelor actuale în educația cibernetică și asupra vulnerabilităților comportamentale ale publicului larg.

Chestionarul a fost completat de un eșantion diversificat de utilizatori, dintre care o mare parte nu dețin o pregătire tehnică avansată. Întrebările au abordat teme precum frecvența cu care utilizatorii își actualizează parolele, modul în care verifică sursa email-urilor sau link-uri suspecte, precum și încrederea pe care o acordă soluțiilor de securitate instalate pe propriul dispozitiv.

Una dintre cele mai revelatoare constatări a fost faptul că majoritatea respondenților nu folosesc parole diferite pentru fiecare cont și nu folosesc soluții de tip *manager de parole*. Această practică sporește semnificativ riscul de compromitere a conturilor în cazul unei breșe de securitate. De asemenea, o proporție importantă a utilizatorilor au declarat că nu sunt conștienți de conceptul de „*phishing*” – un tip de atac informatic în care un atacator încearcă să obțină date sensibile (parole, coduri bancare, etc.) prin imitarea unei surse de încredere. Lipsa familiarității cu terminologia și mecanismele uzuale ale amenințărilor informatice este un indicator clar al necesității unor soluții care să ofere nu doar protecție, ci și claritate și educație în timp real.

Vi s-a întâmplat vreodată să fiți victima unui atac online (ex: cont spart, link fals, viruși)? (alegere unică)

30 responses

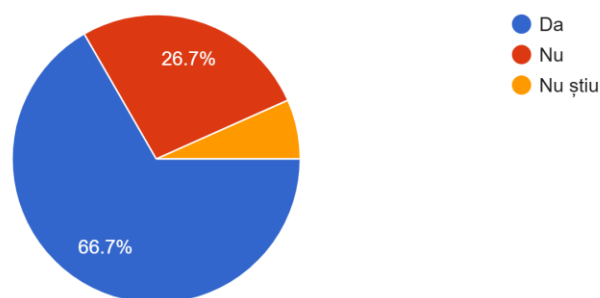


Fig. 3.1. Evidențierea răspunsurilor la întrebarea cu tematica „atacuri online”

Răspunsurile au mai evidențiat faptul că doar o minoritate dintre utilizatori au un comportament preventiv activ, așa cum este prezentat în [6], cum ar fi verificarea semnăturii digitale ale unui e-mail, consultarea certificatului SSL al unui site sau utilizarea unei rețele *VPN*¹¹ în conexiuni publice. Aceste rezultate conturează un profil de utilizator obișnuit care este expus în mod frecvent la riscuri cibernetice, dar care nu are nici instrumentele, nici cunoștințele necesare pentru a le gestiona eficient, conform [7].

Astfel, această analiză justifică necesitatea unui sistem IDPS care să fie nu doar tehnic robust, ci și accesibil din punct de vedere a cât de utilizabil este. Un sistem care explică alertele într-un mod clar și inteligibil, care educă în timp real și care oferă un sentiment de siguranță fără a necesita intervenție sau expertiză avansată din partea utilizatorului final.

Rezultatele acestui chestionar validează obiectivul general al lucrării: dezvoltarea unui sistem IDPS prietenos cu utilizatorii, dotat cu mecanisme intuitive ce explică alertele, care le permite acestora să fie mai conștienți și mai protejați în fața amenințărilor cibernetice moderne.

¹¹ Rețele virtuale private – resursă disponibilă la: <https://www.privateinternetaccess.com/ro/what-is-vpn> [Accesat: 2025]

3.3. Arhitectura sistemului propus

Sistemul IDPS propus se bazează pe o arhitectură de tip client-server, distribuită local, în care datele de trafic sunt monitorizate de un nod central (numit și *IDPS gateway* / „poartă de ieșire”), iar interfețele grafice sunt distribuite pe dispozitivele utilizatorilor (panouri vizuale și interactive, sau *dashboard-uri*). Monitorizarea se face prin *tehnica de oglindire a porturilor*¹², iar analiza traficului se realizează cu ajutorul *Suricata* – un motor performant de detecție care generează log-uri structurate în format JSON¹³.

Aceste log-uri sunt preluate și procesate de modul AI, care clasifică fiecare alertă în funcție de severitate și probabilitate de risc. În paralel, este integrată și o componentă de interpretabilitate prin utilizarea algoritmului *SHAP*⁽³⁾, care oferă o explicație vizuală pentru fiecare predicție a modelului, fapt relatat și în [3]. Astfel, chiar și în cazul unei alerte cu o severitate ridicată, utilizatorul poate înțelege de ce modelul a considerat acel eveniment ca fiind periculos, fără a avea cunoștințe de *machine learning*⁽ⁱⁱⁱ⁾.

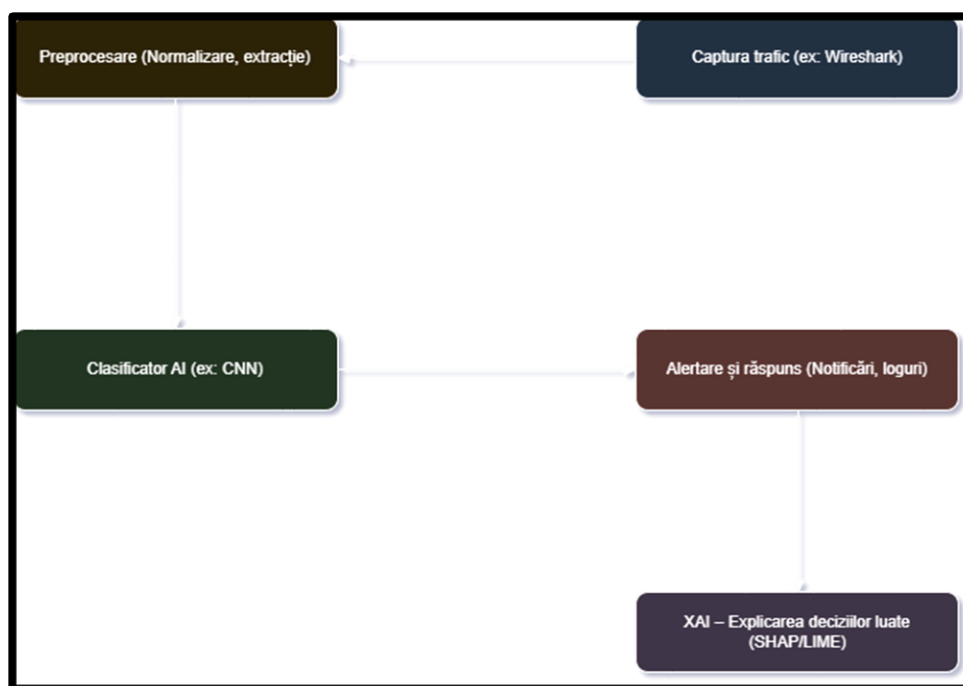


Fig. 3.2. Arhitectura sistemului IDPS propus (cu exemple intuitive)

¹² Termen tehnic provenit de la englezescul *port mirroring* – resursă disponibilă la: <https://www.cbtnuggets.com/blog/technology/networking/what-is-port-mirroring> [Accesat: 2025]

¹³ Notăție pentru *JavaScript Object Notation* – resursă disponibilă la: <https://www.json.org/json-en.html> [Accesat: 2025]



Fig. 3.3. Detalierea arhitecturii IDPS propuse

3.4. Integrarea componentelor AI și XAI

Modelul de clasificare utilizat a fost antrenat pe un set de date realist, cuprinzând atât trafic benign, cât și scenarii variate de atac. Alegerea acestui model de tip *Random Forest*¹⁴ a fost motivată de echilibrul între performanță și interpretabilitate. La fiecare alertă nouă generată de *Suricata*, modelul oferă o clasificare automată, iar componenta XAI aplică analiza SHAP pentru a explica, în timp real, factorii care au contribuit la acea clasificare.

Această abordare este dublată de o interfață de tip *dashboard*, unde utilizatorii pot vedea alertele proprii, statusul dispozitivului lor (**protejare activă** sau **atac în desfășurare**), dar și un istoric al incidentelor detectate. Interfața este concepută să fie ușor de vizualizat și înțeles, folosind culori, pictograme și mesaje explicative în loc de termeni tehnici abstracți. Astfel, se facilitează tranziția de la o protecție pasivă la o conștientizare activă din partea beneficiarilor sistemului.

3.5. Caz de utilizare – diagrama UML

Pentru a exemplifica funcționarea concretă a sistemului, am elaborat un *Use Case* în format *UML*, creat cu ajutorul aplicației *Microsoft Visio*, care descrie interacțiunea dintre trei tipuri de entități: utilizatorul obișnuit, administratorul rețelei și modul de detecție al IDPS-ului. Scenariul principal presupune conectarea unui utilizator la rețea, monitorizarea pasivă a traficului acestuia, identificarea unei activități suspecte (de exemplu, o scanare de

¹⁴ Algoritm de învățare automată bazat pe arbori de decizie – resursă disponibilă la: <https://www.geeksforgeeks.org/machine-learning/random-forest-algorithm-in-machine-learning/> [Accesat: 2025]

porturi din rețea), clasificarea comportamentului ca potențial malițios și notificarea în interfața grafică printr-un mesaj explicit și interpretabil.

Acest tip de interacțiune demonstrează beneficiul major al sistemului: transformarea unei infrastructuri de securitate sofisticate într-un instrument accesibil oricărui utilizator. Cazul de utilizare poate fi extins cu scenarii suplimentare, precum și alertarea administratorului în timp real, blocarea automată a traficului de la un anumit IP sau antrenarea continuă a modelului AI pe baza feedback-ului primit.

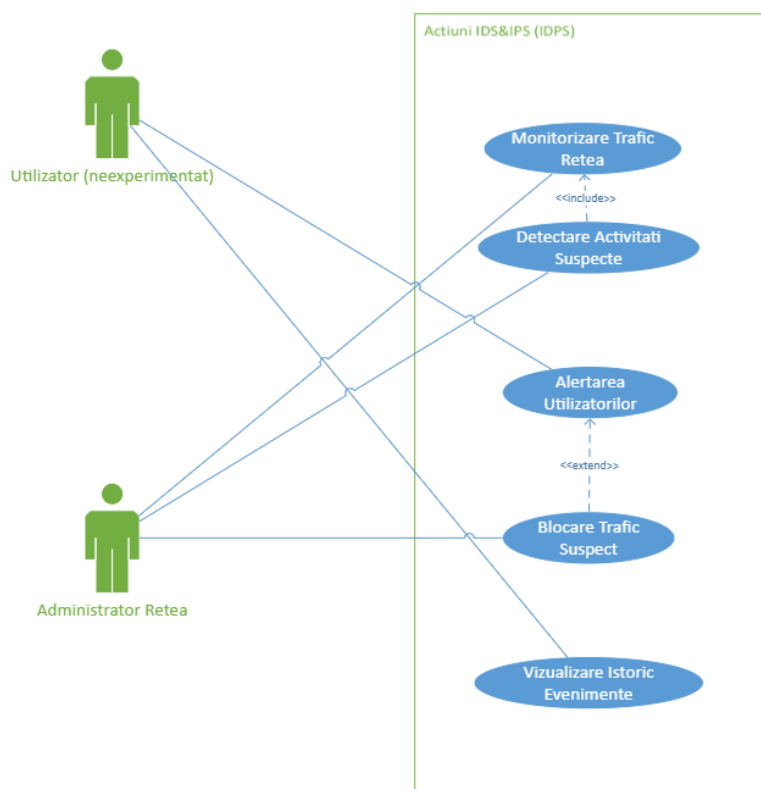


Fig. 3.4. Diagrama *Visio Use Case* IDPS

3.6. Soluții existente pe piață pentru IDPS

Înainte de a defini detaliile unei noi soluții tehnologice, este esențial să analizăm peisajul actual al produselor și platformelor existente în domeniul securității rețelelor, în special în ceea ce privește sistemele de detecție și prevenire a intruziunilor. Această analiză permite înțelegerea avantajelor și limitărilor celor mai utilizate instrumente, precum și

identificarea unor nevoi neacoperite, care pot constitui puncte forte pentru soluția propusă în cadrul acestei lucrări.

Pe piața actuală, o serie de platforme mature sunt utilizate pentru monitorizarea, analiza și reacția la comportamente suspecte din rețea. Printre cele mai cunoscute se numără *Suricata*, *Snort*, *Zeek* și *OSSEC*, fiecare oferind funcționalități distincte, adaptate diferitelor tipuri de rețele și scenarii de utilizare, potrivit și relatărilor din [8].

Suricata [9], dezvoltat de **Open Information Security Foundation (OISF)**, este una dintre cele mai populare soluții cu licență gratuită și cod la liber pentru detecția pachetelor de rețea și identificarea comportamentului periculos. El este capabil să inspecteze traficul la nivel de protocol, să detecteze anomalii și să ofere alerte în timp real. Un aspect important este suportul pentru *multithreading* (execuție paralelă pe mai multe nuclee de procesor), care îl face scalabil pentru rețele mari. De asemenea, generează alerte în format JSON în fișiere log, ceea ce permite integrarea ușoară cu alte sisteme sau interfețe grafice. Cu toate acestea, utilizarea lui presupune o bună înțelegere a regulilor și configurațiilor, ceea ce poate reprezenta un obstacol pentru utilizatorii mai puțin tehnici.

O altă soluție bine-cunoscută este *Snort* [10], dezvoltat de **Cisco Systems**. Deși este similar cu *Suricata* în ceea ce privește scopul principal, el utilizează un motor propriu de analiză a semnăturilor și este recunoscut pentru stabilitatea sa în medii corporative. Cu toate acestea, *Snort* are o interfață în linie de comandă și o curbă de învățare relativ abruptă, fiind mai potrivit pentru specialiști în securitate decât pentru utilizatori obișnuiți. Configurarea sa manuală și interpretarea log-urilor pot reprezenta provocări în absența unei interfețe vizuale intuitive.

Zeek (anterior cunoscut sub denumirea de *Bro*) [11] se distinge prin faptul că nu se bazează pe semnături statice, ci analizează comportamentul rețelei prin politici definite în limbaj propriu de *scripting*. Acest lucru îl face ideal pentru detectarea atacurilor de tip „zero-day”, deoarece nu se limitează la identificarea unor modele prestabilite, ci urmărește modificările contextuale din rețea. Totuși, flexibilitatea lui *Zeek* vine cu prețul unei complexități ridicate în configurare și întreținere, necesitând cunoștințe avansate pentru personalizare eficientă.

În ceea ce privește soluțiile orientate către protecția dispozitivelor individuale (de tip gazdă), *OSSEC* [12] este un exemplu notabil. El oferă monitorizare a integrității fișierelor, analiză a jurnalelor de sistem și detecție a activităților suspecte la nivelul gazdei. Deși este eficient în mediile bazate pe server, *OSSEC* nu este conceput pentru analiza traficului de rețea în ansamblu, ci pentru monitorizarea activității locale, ceea ce limitează aplicabilitatea sa în contexte distribuite sau în rețele complexe.

Pe lângă aceste soluții tehnice, există și o gamă variată de produse comerciale integrate, cum ar fi *Cisco Secure IPS*¹⁵, *Palo Alto Networks Threat Prevention*¹⁶, *Fortinet FortiGate IPS*¹⁷ sau *Trend Micro TippingPoint*¹⁸. Acestea oferă funcționalități extinse, cum ar fi actualizări automate ale semnăturilor, integrare cu sisteme de răspuns automat (*SOAR – Security Orchestration, Automation and Response*¹⁹) și interfețe grafice sofisticate. Totuși, costurile ridicate de licențiere, precum și necesitatea unei infrastructuri hardware dedicate, le fac inaccesibile pentru segmentul de utilizatori individuali sau pentru rețele educaționale de dimensiuni mici.

Un alt aspect important este faptul că majoritatea soluțiilor existente, chiar și cele comerciale, se axează preponderent pe furnizarea de alerte și log-uri tehnice, fără a oferi o interpretare clară a acestora. Astfel, cei non-tehnici se pot confrunța cu dificultăți în a înțelege natura unei amenințări, consecințele posibile sau pașii necesari pentru remediere. Lipsa unui mecanism explicativ accesibil amplifică sentimentul de incertitudine și descurajare, ceea ce conduce de multe ori la ignorarea alertelor sau la dezactivarea completă a sistemului de protecție.

De asemenea, în literatura de specialitate și în cercetările recente s-a evidențiat tendința tot mai accentuată de integrare a inteligenței artificiale în arhitectura acestor sisteme. Modelele de învățare automată pot îmbunătăți semnificativ performanța detecției, reducând numărul de alerte fals-pozitive și identificând modele complexe de atacuri cunoscute anterior. Cu toate acestea, doar câteva implementări comerciale sau cu licență liberă de folosire integrează și o componentă de inteligență artificială tip XAI, care să traducă deciziile automate într-un limbaj inteligibil pentru cei care folosesc sistemul, așa cum este descris și în [13]. Acest gol funcțional este exact zona în care soluția propusă de această lucrare intervine.

Prin urmare, se constată existența unei lacune semnificative în peisajul actual al soluțiilor IDPS: lipsa unui sistem integrat care să combine detecția bazată pe AI cu o interfață prietenoasă și explicabilă, concepută special pentru utilizatorii fără pregătire tehnică avansată. Soluția propusă urmărește să umple acest gol, oferind o alternativă viabilă, scalabilă și accesibilă, care să răspundă atât nevoilor de securitate, cât și celor de interpretabilitate și educație în timp real.

¹⁵ Soluție IPS – resursă disponibilă la: https://www.cisco.com/c/en_uk/products/security/ngips/index.html [Accesat: 2025]

¹⁶ Soluție IPS – resursă disponibilă la: <https://www.paloaltonetworks.com/network-security/advanced-threat-prevention> [Accesat: 2025]

¹⁷ Soluție IPS – resursă disponibilă la: <https://www.fortinet.com/support/support-services/fortiguard-security-subscriptions/intrusion-prevention> [Accesat: 2025]

¹⁸ Soluție IPS – resursă disponibilă la: https://www.trendmicro.com/en_gb/business/products/network/intrusion-prevention/tipping-point-threat-protection-system.html [Accesat: 2025]

¹⁹ Sistem folosit în securitatea cibernetică – resursă disponibilă la: <https://www.techtarget.com/searchsecurity/definition/SOAR> [Accesat: 2025]

4. Documentație tehnică

Acest capitol prezintă în mod detaliat implementarea practică a sistemului distribuit de detecție și prevenire a intruziunilor, cu accent pe componentele hardware și software utilizate, configurațiile efectuate și rezultatele obținute în urma testărilor.

Sunt evidențiate toate echipamentele implicate în realizarea arhitecturii rețelei locale, inclusiv motivația alegerii acestora și modul în care au fost conectate și configurate. Se discută inclusiv despre realizarea fizică a rețelei, folosirea cablurilor sertizate manual de categorie superioară, precum și strategia de conectare prin switch pentru *port mirroring*.

De asemenea, sunt detaliate tehnologiile software care au stat la baza aplicației: de la *motorul de detecție* și *serverele de backend*, până la *interfața web*, distribuită pe două nivele – pentru administrator și pentru utilizator. Este explicat mecanismul de colectare și prelucrare a alertelor, integrarea unui model de inteligență artificială pentru clasificarea acestora, precum și componenta XAI de explicare a deciziilor.

Ultima secțiune oferă o prezentare a rezultatelor obținute în urma testării sistemului: interacțiunea cu utilizatorul, precizia clasificării, capturi de ecran ale interfețelor, alerte reale și simulate, dar și performanța sistemului în timp real.

Această documentație tehnică reflectă efortul integrator de îmbinare a componentelor fizice și logice într-un sistem coerent, funcțional și accesibil, adresând în mod direct nevoile utilizatorilor neexperimentați în fața amenințărilor cibernetice.

4.1. Echipamente utilizate (hardware)

În realizarea practică a sistemului distribuit IDPS, a fost construită o rețea locală funcțională, bazată pe echipamente fizice reale, care a permis testarea în condiții controlate a întregii soluții propuse. Infrastructura a fost concepută astfel încât să permită colectarea de trafic real a potențialelor atacuri și afișarea acestora într-o interfață destinată atât administratorilor, cât și utilizatorilor obișnuiți.

4.1.1. Topologia generală a sistemului

Rețeaua este compusă din patru laptopuri interconectate prin intermediul unui switch de management de tip *TP-Link Easy Smart TL-SG105E*, care oferă suport pentru funcționalități avansate de monitorizare a traficului, printre care și oglindirea porturilor. Toate dispozitivele sunt conectate fizic, iar două dintre ele prin cabluri de rețea tip *Cat6*, realizate manual prin sertizare, ceea ce asigură stabilitate mecanică și viteze ridicate de transmisie de până la *1 Gbps* (cu potențial teoretic de până la *10 Gbps* în configurații specifice).

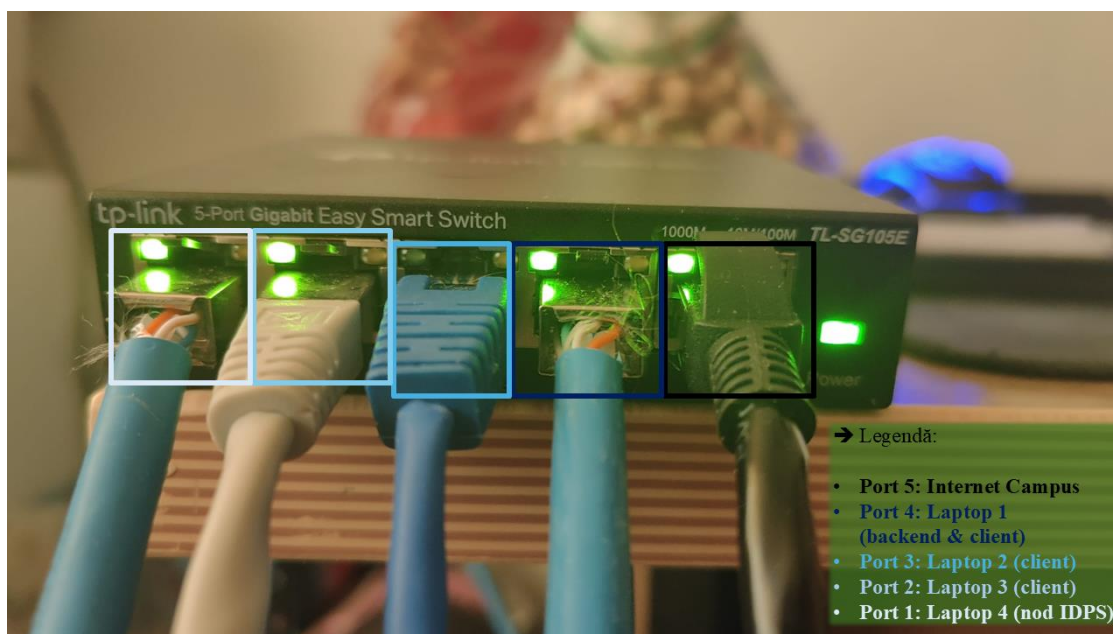


Fig. 4.1. Conexiunile realizate în switch

Internetul este furnizat printr-un port de rețea dedicat din căminul studentesc, conectat direct la *portul 5* al switch-ului. Alegerea portului 5 nu este întâmplătoare: acesta este separat logic și fizic de primele patru porturi, ceea ce permite o mai bună segregare a traficului extern față de cel intern generat în rețeaua locală de test.

Switch-ul permite direcționarea tuturor pachetelor de trafic din porturile 2-5 către portul 1, unde este conectat dispozitivul principal de monitorizare, adică sistemul IDPS. Această tehnică, care poartă numele de *port mirroring*, presupune redirectionarea pasivă a întregului trafic de pe una sau mai multe surse către un port special de analiză, fără a afecta fluxul original, conform detaliilor din [14].

4.1.2. Descrierea echipamentelor hardware utilizate

În cadrul acestei implementări, au fost utilizate patru laptopuri, fiecare cu un rol bine definit în arhitectura rețelei:

1. Laptop 1 – *HP RTL* (dispozitiv de monitorizare și captură trafic)

Acest echipament are rolul principal de **nod pasiv de monitorizare**, fiind responsabil de recepționarea întregului trafic din rețea (prin *ogłindire de porturi*) și de prelucrarea sa inițială. Funcționează exclusiv pe distribuția *Arch Linux*, aleasă datorită caracterului său minimalist, modular și orientat spre securitate, așa cum este descris și în [15] – oferind un control profund asupra serviciilor active, a configurării rețelei și a drepturilor de acces.



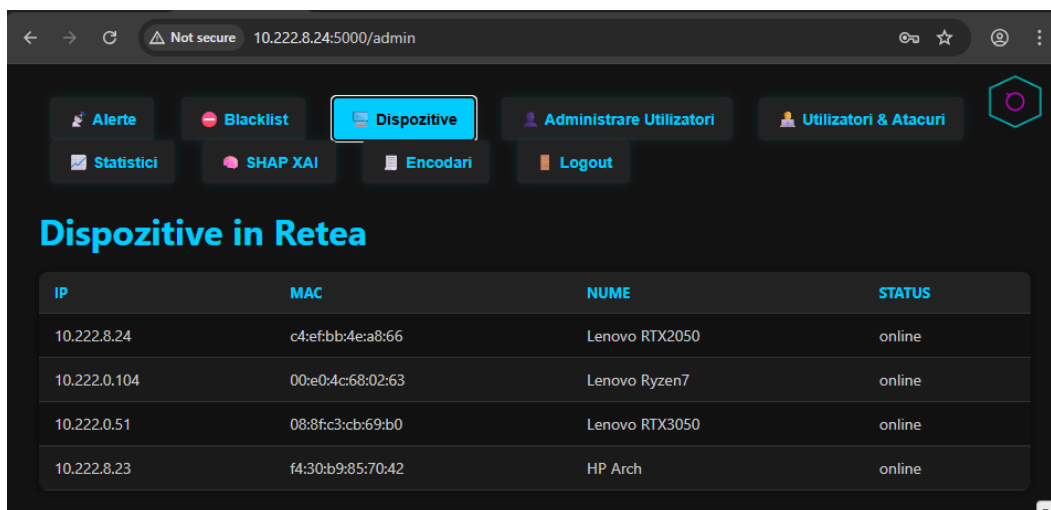
Fig. 4.2. Laptopul *HP RTL* utilizat în arhitectură

Pe acest laptop rulează două componente esențiale:

- **Motorul de detecție *Suricata*** [16], configurat să analizeze în timp real traficul de rețea și să genereze alerte în format JSON (standardul *eve.json*²⁰);
- **Un server *Flask*** [17] dedicat scanării rețelei locale, implementat printr-un scrip *Python* ce rulează în fundal sub formă de serviciu la pornire. Acest server identifică dispozitivele active folosind comenzi precum `ip neigh show` și `ping` în rețea, afișându-le în tab-ul *Dispozitive* din dashboard.

²⁰ Facilitatea de ieșire **EVE** (așa va fi notat de acum acest fișier) generează alerte, anomalii + altele, specifice protocolului prin JSON – resursă disponibilă la: <https://docs.suricata.io/en/latest/output/eve/eve-json-output.html> [Accesat: 2025]

Un alt element de securitate implementat aici este **filtrarea dispozitivelor necunoscute**, prin folosirea unei liste albe (sau *whitelist*) de IP-uri permise, Astfel, se evită includerea în analiza sistemului a dispozitivelor din afara echipamentelor deținute de administrator/ utilizator, având în vedere că rețeaua căminului este partajată.



IP	MAC	NUME	STATUS
10.222.8.24	c4:ef:bb:4e:a8:66	Lenovo RTX2050	online
10.222.0.104	00:e0:4c:68:02:63	Lenovo Ryzen7	online
10.222.0.51	08:8f:c3:cb:69:b0	Lenovo RTX3050	online
10.222.8.23	f4:30:b9:85:70:42	HP Arch	online

Fig. 4.3. Tab-ul „Dispozitive” din aplicație – vizualizare a nodurilor active din rețea

Totodată, conexiunea la acest laptop se realizează securizat prin SSH, cu serverul `sshd` activat manual și configurat pentru a avea acces și control, asigurând interacțiunea între componentele de *backend* care rulează pe celelalte dispozitive. În acest sens, laptopul HP acționează ca un „*senzor de rețea*” izolat, cu rol exclusiv de analiză și raportare, fără expunere la activități externe.

Componentă activă	Descriere	Tehnologii folosite
Detector trafic	Suricata	Filtrare + alerte JSON
Server scanare rețea	Flask + Python	Scanare ping + ip neigh
Acces securizat	OpenSSH	Acces remote controlat

Tabel 4.1. Descrierea componentelor utilizate pe laptopul HP

2. Laptop 2 – *Lenovo IdeaPad 3* (client și generator de trafic de test)

Acest dispozitiv are rol auxiliar, fiind utilizat în principal pentru generarea de trafic divers (inclusiv simulat malițios) în cadrul rețelei. Prin rularea de comenzi specifice precum `nmap`, sau prin accesarea intensivă a unor site-uri sau servicii, acest laptop permite testarea răspunsului sistemului IDPS în condiții variate. Totodată, ajută la validarea vizuală a alertelor, observând dacă anumite comportamente sunt corect detectate și etichetate.



Fig. 4.4. Laptopul *Lenovo IP 3* utilizat în arhitectură

Este un echipament cu o configurație decentă, dar care nu rulează componente critice ale aplicației – fiind utilizat exclusiv ca parte a scenariilor de test.

3. Laptop 3 – *Lenovo IdeaPad Gaming 3* (antrenare model AI)

Acesta are un rol central în **etapa de antrenare și optimizare a modelului AI**. Datorită plăcii sale dedicate *NVIDIA RTX 3050 Ti*, acesta oferă o putere de procesare paralelă considerabilă, esențială pentru manipularea unor volume mari de date și pentru accelerarea algoritmilor de învățare automată.

Modelul AI implementat este un clasificator *Random Forest*, ales din următoarele motive:

- Este un algoritm robust, capabil să gestioneze seturi de date dezechilibrate și cu multe caracteristici;
- Oferă performanțe ridicate în detecția anomaliilor fără a necesita o preprocesare excesivă;
- Este ușor de interpretat și integrabil într-un sistem XAI cu instrumente precum SHAP.



Fig. 4.5. Laptopul *Lenovo IPG 3* utilizat în arhitectură

Modelul antrenat pe acest dispozitiv este exportat în format *joblib*^x, împreună cu codificatorul etichetelor, și transferat ulterior pe laptopul descris mai jos (*Lenovo LOQ*) pentru inferență în timp real.

4. Laptop 4 – *Lenovo LOQ* (dispozitiv principal – backend, AI și UI)

Acesta este **nodul central activ** al întregii aplicații IDPS, având rolul de:

- **Rulare a interfeței *frontend*** (interfață grafică în stil *Bitdefender*²¹);
- **Procesare a alertelor venite de la *Suricata***, primite de la laptopul HP, prin intermediul unui *server Flask* ce transformă datele JSON în informații structurate pentru aplicație;
- **Aplicare a clasificării AI**, folosind modelul *Random Forest* menționat anterior, printr-un *server Flask* dedicat (API de inferență²²);
- **Generare a explicațiilor XAI**, folosind *biblioteca SHAP*, pentru a oferi transparență deciziilor luate de model.



Fig. 4.6. Laptopul *Lenovo LOQ* utilizat în arhitectură

²¹ Am încercat să mă apropiez de acel stil al aplicației lor dar nu și funcționalitățile complete, IDPS-ul meu fiind un produs care se axează doar pe traficul din rețeaua LAN de dimensiuni mici sau medii.

²² **Explicație:** *API-ul de inferență* permite utilizatorilor să folosească modele de învățare automată pre-antrenate pentru a face predicții pe baza datelor noi, fără a fi nevoie să implementeze sau să gestioneze modelul ei înșiși – resursă disponibilă la:
<https://www.elastic.co/docs/api/doc/elasticsearch/operation/operation-ilm-stop> [Accesat: 2025]

Toate cele trei componente rulează local, fiecare având propriul *endpoint* (nod terminal) Flask:

- **Serverul de backend** (*app.py*) – gestionează interfața, login-ul, afișarea alertelor și dispozitivelor;
- **Serverul Suricata** (*suricata_api.py*) – preia alertele din fișierul *EVE*⁽²⁰⁾ (fișier la care are acces mereu, fiind partajat de HP), la un interval de timp definit, le pune etichetă și le trimite într-o bază de date care se actualizează mereu cu noi alerte, fiind resursa principală de antrenare a modelului AI.
- **Serverul AI & XAI** (*ai_api.py*) – primește date brute de alertă de la serverul Suricata și returnează predicția (malicios/ normal / informațional); calculează importanța caracteristicilor folosind SHAP și returnează explicația alertelor generate după tipul de protocol și de semnătură.

Această distribuție modulară asigură că este scalabil și poate separa responsabilitățile în cadrul aplicației, făcând sistemul ușor de întreținut și extins.

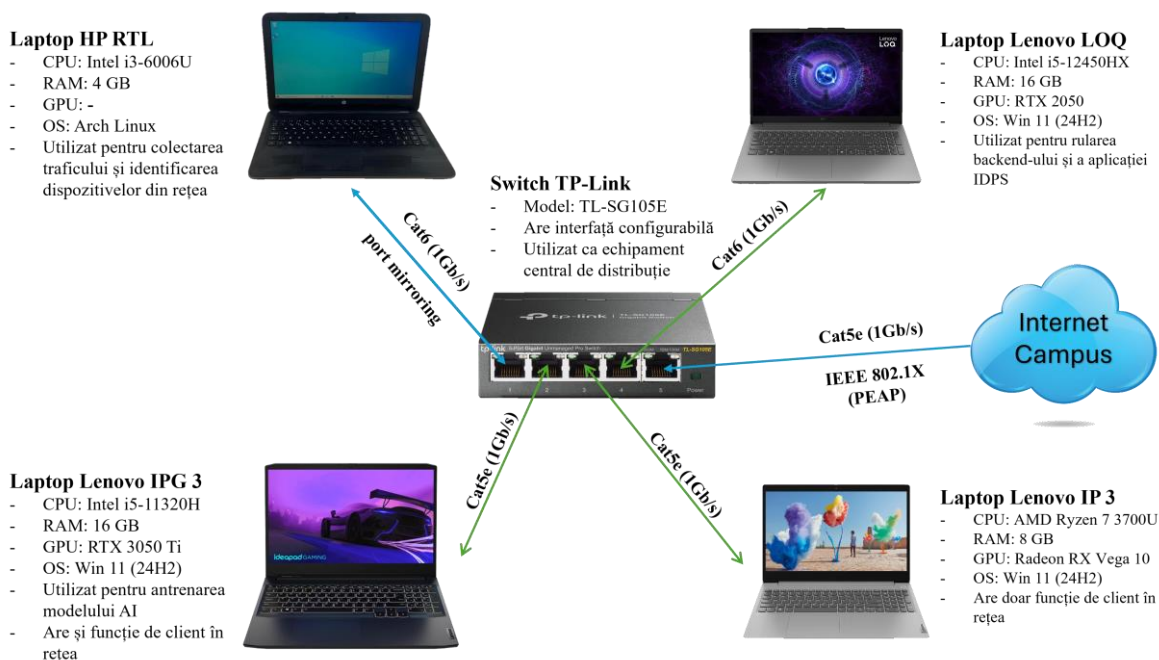


Fig. 4.7. Topologia IDPS – schema conexiunilor fizice și descrierea echipamentelor

Prin utilizarea a patru dispozitive distincte, fiecare cu o funcție bine definită, sistemul IDPS devine robust la sarcini variate, sigur din punct de vedere al accesului și filtrării, ușor de testat și monitorizat în condiții controlate și extensibil pentru viitoare îmbunătățiri (integrare în *cloud* sau adăugarea de noduri noi).

4.1.3. Alegerea și utilizarea switch-ului

Switch-ul *TP-Link TL-SG105E* reprezintă o soluție accesibilă dar performantă pentru rețelele locale de mici dimensiuni care necesită funcționalități semi-profesionale. Acesta oferă:

- 5 porturi *Gigabit Ethernet* (10/100/1000 Mbps);
- Funcție de *port mirror* configurabilă prin interfața de administrare;
- Suport pentru *VLAN*-uri (rețele virtuale locale);
- Diagnosticare de cablu și alte opțiuni utile pentru depanare.



Fig. 4.8. Switch *TP-Link TL-SG105E*

A fost ales tocmai pentru că permite configurarea rapidă a direcționării traficului în scopuri de analiză, fără a fi nevoie de echipamente costisitoare. În acest context, porturile 2-5 au fost configurate ca surse de trafic, iar portul 1 ca destinație pentru monitorizare.

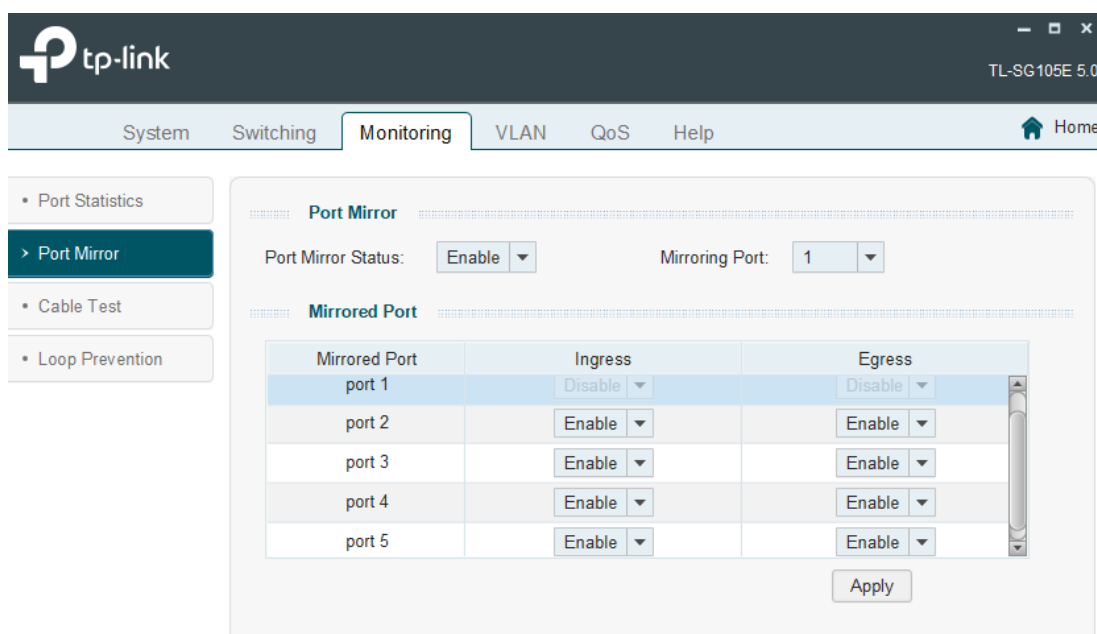


Fig. 4.9. Configurarea funcției de *port mirror* pe switch

4.1.4. Conectica fizică și cabluri utilizate

Toate conexiunile dintre dispozitive au fost realizate cu cabluri de internet *Cat5e*, în afară de două dispozitive, *HP* și *Lenovo LOQ*, pentru ca ele stau la baza aplicației IDPS. Așadar, pentru ultimele menționate, am sertizat manual cabluri tip Cat6. Potrivit explicațiilor de la [18], acestea sunt proiectate să susțină viteze ridicate de transfer, fiind utilizate frecvent în infrastructuri moderne care necesită stabilitate și performanță ridicată.

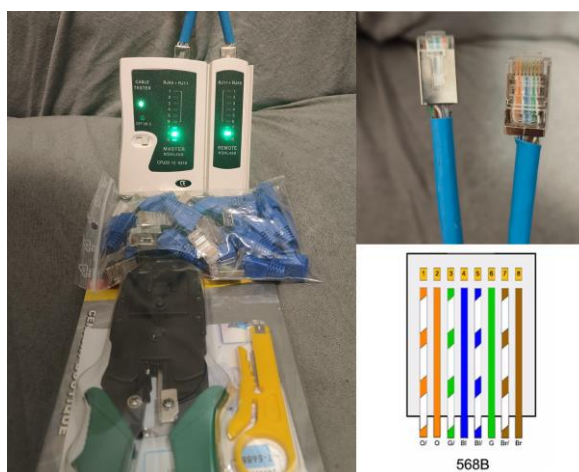


Fig. 4.10. Echipamentele utilizate (stânga), cablul sertizat *Cat6* (dreapta-sus) și schema de sertizare T-568B folosită (dreapta-jos)

Sertizarea cablurilor reprezintă procesul de fixare mecanică a capetelor (mufe *RJ45*) pe un cablu de rețea prin utilizarea unui clește special, după care se introduce cablul într-un tester care verifică ordinea firelor. Dacă ledurile aparatului se aprind sincron la ambele capete, atunci sertizarea a avut succes, altfel se reia procesul. Acest proces permite personalizarea lungimii cablurilor și evitarea pierderilor de semnal datorate conectorilor de slabă calitate.

4.1.5. Sursa de internet și integrarea în rețeaua căminului

Accesul la internet este realizat prin intermediul unui port de rețea din peretele camerei de cămin, furnizat de rețeaua administrativă a universității. Acest port oferă conexiune directă la rețeaua administrativă a universității. Acest port oferă conexiune directă la rețeaua campusului și este conectat fizic în *portul 5 al switch-ului*, alegere justificată de separarea logică a acestuia față de restul echipamentelor interne.

Abordarea prezentată permite izolarea traficului extern de cel intern, oferind un model simplificat de segmentare, în care datele provenite din internet pot fi ușor monitorizate separat față de conexiunile locale dintre laptopurile personale.

4.2. Tehnologii utilizate (software)

Sistemul IDPS, pe lângă implementarea sa hardware, are nevoie și de o parte software, unde a fost necesară integrarea unei game variate de tehnologii care să asigure detecția în timp real a traficului periculos, clasificarea alertelor folosind modele de învățare automată, precum și prezentarea informațiilor într-o formă accesibilă și prietenoasă. În această secțiune, voi detalia fiecare componentă software utilizată, justificând alegerile făcute și explicând rolul fiecăreia în arhitectura sistemului.

4.2.1. Motorul de detecție

Pornind de la concluziile enunțate în [19], **Suricata** este un sistem *open-source* de detecție din categoria *IDS*, capabil să monitorizeze în timp real pachetele care tranzitează o interfață de rețea și să genereze alerte în funcție de semnături predefinite sau comportamente suspecte. Este implementat în **C**, are suport pentru *multithreading* și poate procesa volume mari de trafic fără a introduce latențe semnificative (*descriere detaliată și la secțiunea [3.6.](#)*).

- a) **Rolul Suricata în proiect** – Acționează ca un senzor de rețea, instalat pe laptopul HP (cel cu *Arch Linux*). Motorul rulează cu drepturi de administrator și este configurat să analizeze, în timp real, tot traficul primit, atât capul de pachet (*header*), cât și conținutul transmis (*payload*).
- b) **Configurare YAML²³** – Configurarea motorului se face printr-un fișier de tip *YAML* (*Yet Another Markup Language*) [20] [21], care definește:
- **Interfața de rețea** ce trebuie monitorizată (în cazul meu, interfața este *eno1* – portul de internet al *laptopului HP*);
 - **Log-urile generate** (tipul și locația acestora);
 - **Activarea formatului *EVE*** (pentru integrare cu restul aplicației);
 - **Tipul de detecție** (bazat pe semnături sau detectarea de anomalii).
- c) **Reguli și semnături** – *Suricata* folosește fișiere de tip *set de reguli* pentru a determina ce tipuri de trafic să alerteze. Fișierele sunt grupuri de semnături, similare cu antivirusul, care specifică modele de pachete asociate atacurilor comune (*descrie în detaliu la secțiunea 3.1.*).

```

GNU nano 8.4 /etc/suricata/rules/emerging-phishing.rules
# This Ruleset is EmergingThreats Open optimized for suricata-5.0-enhanced.
#alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ET PHISHING Paypal Phishing victim POSTing data"; flow:established)
#alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"ET PHISHING Potential Paypal Phishing Form Attachment"; flow:est)
#alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"ET PHISHING Potential ACH Transaction Phishing Attachment"; flow)
#alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET PHISHING Spam Campaign JPG CnC Link"; flow:established,to_client)
#alert smtp $EXTERNAL_NET any -> $SMTP_SERVERS any (msg:"ET PHISHING Possible Phishing E-ZPass Email Toll Notification")
#alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET PHISHING Potential Sofacy Phishing Redirect"; flow:established)
#alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"ET PHISHING Operation Huyao Landing Page Nov 07 2014"; flow:established)
#alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET PHISHING Possible Tsukuba Banker Edwards Packed proxy.pac"; flow)
#alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET PHISHING Successful Bank of America Phish 2015-10-02"; flow:to_client)
#alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET PHISHING Chase Account Phish Landing Oct 22"; flow:established)
#alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET PHISHING Outlook WebApp Phish Landing 2015-11-05"; flow:established)
#alert http $EXTERNAL_NET any -> $HOME_NET any (msg:"ET PHISHING Mailbox Renewal Phish_Landing Nov 13"; flow:established)
^G Help      ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark
^X Exit      ^R Read File  ^E Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-R Redo      M-G Copy

```

Fig. 4.11. Exemplu de fișier cu reguli active pentru detectarea încercării unui atac de tip *phishing* (furt de date personale)

²³ O secțiune din acest fișier este disponibilă la [Anexe](#) – Anexa 1.

d) **Formatul *EVE*** – Toate alertele sunt exportate în acest fișier care respectă formatul standardizat JSON, conținând câmpuri esențiale precum:

- **timestamp**: momentul producerii evenimentului;
- **src_ip, dest_ip**: adresele IP sursă și destinație;
- **alert.signature**: semnătura alertei;
- **proto**: protocolul implicat (exemple: TCP, UDP, ICMP);
- **severity**: nivelul de severitate.

4.2.2. Serverele Flask pentru integrarea backend-ului

Flask^(Error! Bookmark not defined.) este un „*micro-framework*” web scris în limbaj *Python*, utilizat pentru a crea aplicații rapide, simple și scalabile . În cadrul acestui proiect, el este utilizat pentru a construi **trei servicii backend** care intermediază comunicația dintre diferitele componente ale sistemului.

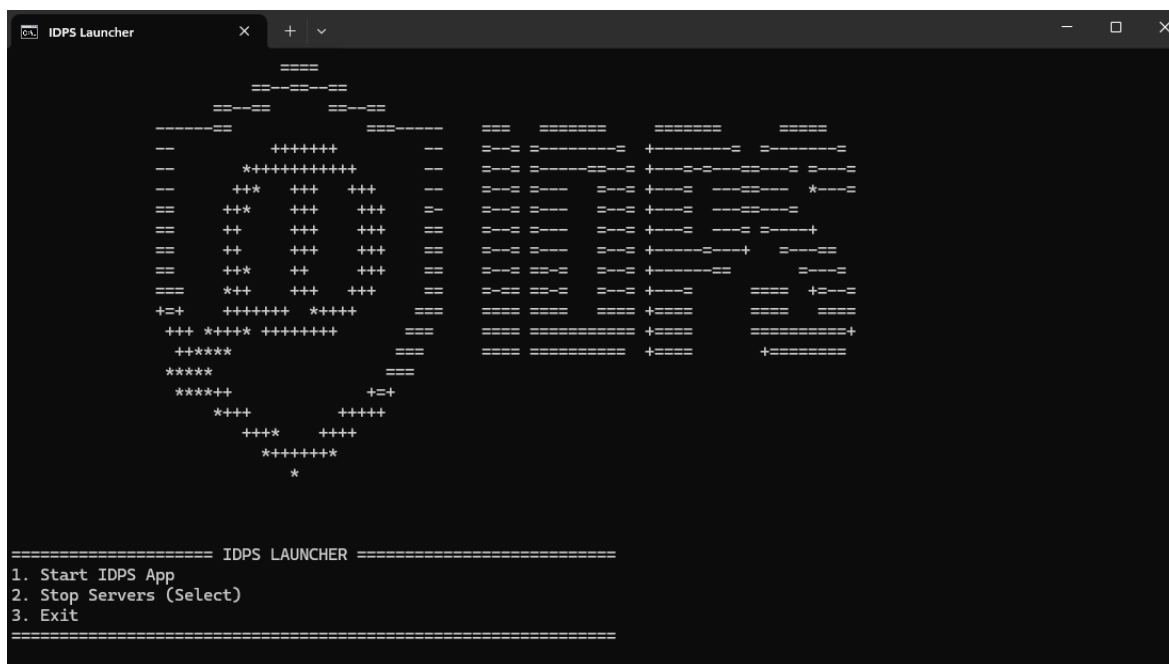


Fig. 4.12. Vizualizarea interfeței de control a *serverelor Flask* utilizate în *aplicația IDPS*

a) **Serverul *suricata_api.py*** – Un prim server, care rulează pe *Lenovo LOQ*, și este responsabil de:

- **monitorizarea și citirea** în mod continuu din *EVE* printr-un proces asincron;
- **prelucrarea conținutului *JSON*** într-un format utilizabil de aplicație, informații precum adresa IP (sursă / destinație), semnătura și altele;
- **inserarea în baza de date locală SQLite (*alert_history.db*)** a fiecărei alerte, care servește atât interfața de admin, cât și modul AI/XAI (vezi secțiunea [4.2.3.1](#));
- **transmiterea în timp real** către interfața aplicației a alertelor noi de la endpointul */alerts* expus (exemplu: *10.222.8.24:5003/alerts*).

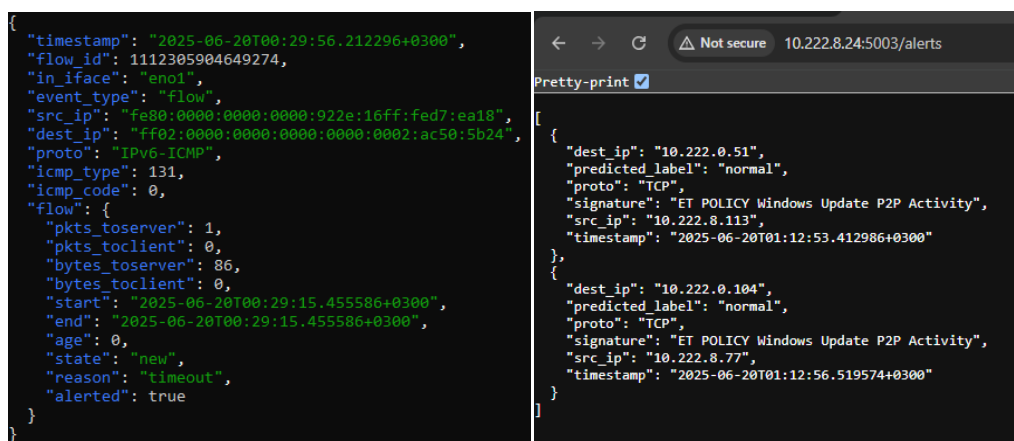
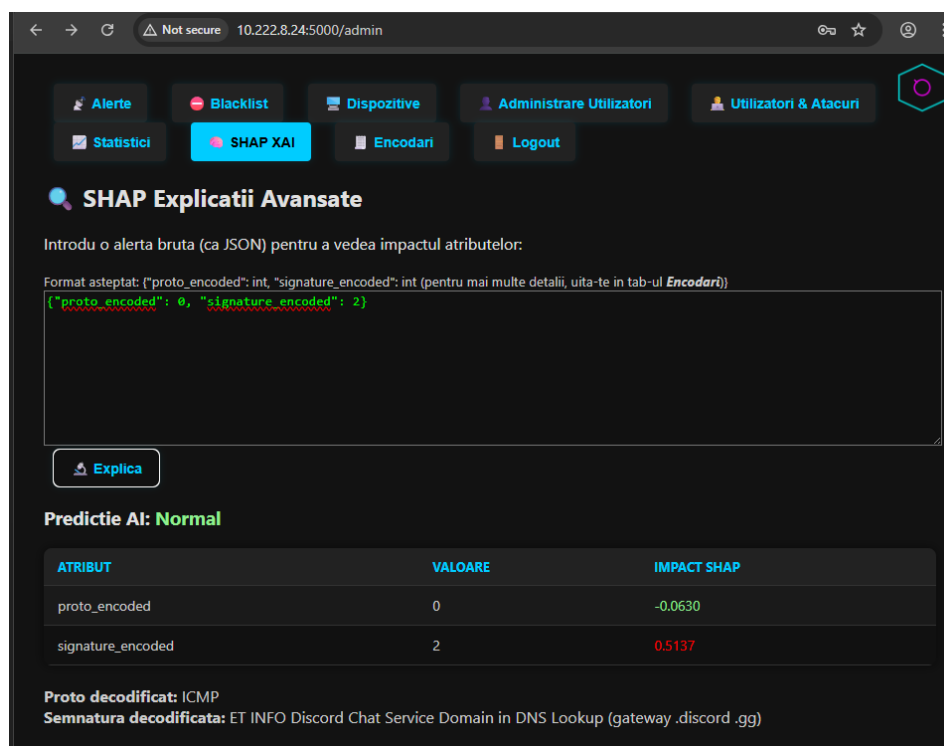


Fig. 4.13. Exemplu de alertă identificată și formatată în stil JSON (stânga – extras din EVE, dreapta – vizualizare în browser)

b) **Serverul *ai_api.py*** – Acest server *Flask* are un rol dublu și esențial: pe de o parte, realizează clasificarea alertelor cu ajutorul unui model de învățare automată; pe de altă parte, oferă explicații interpretabile pentru aceste decizii, în cadrul componentei XAI (vezi [4.2.5](#) – *detaalierea componentei AI*). El oferă următoarele caracteristici:

- **primește alertele** în format JSON, iar modelul AI (antrenat în prealabil și salvat sub forma unui fișier *joblib* – exemplu: *classifier_model.joblib*) este încărcat și utilizat pentru a clasifica alerta (*malicious* – potențial atac; *normal* – activitate legitimă; *information* – trafic inofensiv, dar înregistrat).
- **generează explicații vizuale** pentru fiecare clasificare, folosind biblioteca SHAP;
- **SHAP** calculează importanța fiecărui atribut (exemplu: protocol și semnătură) în decizia modelului;
- **serverul returnează un fișier imagine** (diagrama tip „*waterfall*”), ce este afișată doar în interfața administratorului (sunt două diagrame, una pentru numărul de alerte de-a lungul timpului și una pentru trend-ul a 7 zile de alerte).



SHAP Explicatii Avansate

Introdu o alerta bruta (ca JSON) pentru a vedea impactul atributelor:

Format asteptat: {"proto_encoded": int, "signature_encoded": int (pentru mai multe detalii, uita-te in tab-ul *Encodari*)}

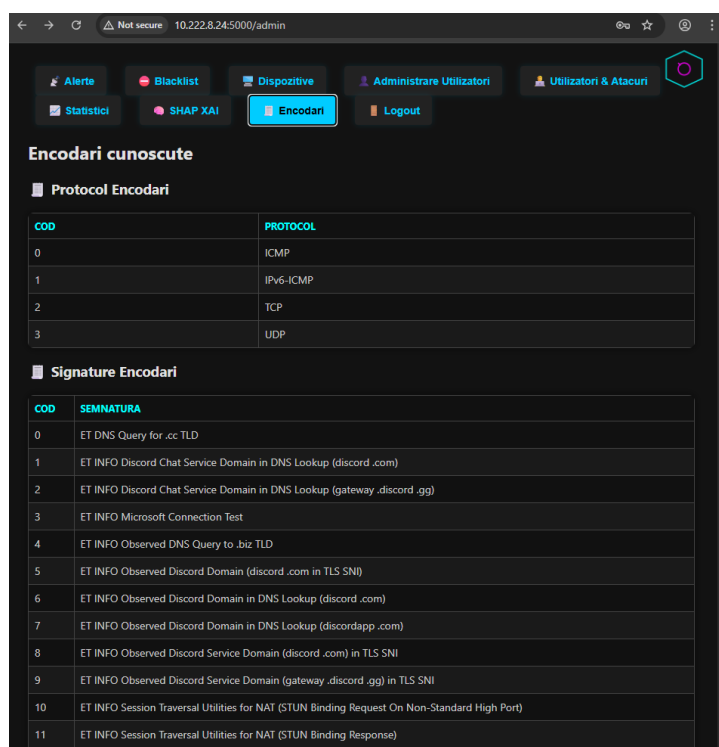
```
{"proto_encoded": 0, "signature_encoded": 2}
```

Predictie AI: Normal

ATRIBUT	VALOARE	IMPACT SHAP
proto_encoded	0	-0.0630
signature_encoded	2	0.5137

Proto decodificat: ICMP
Semnatura decodificata: ET INFO Discord Chat Service Domain in DNS Lookup (gateway.discord.gg)

Fig. 4.14. Tab-ul *SHAP XAI* cu explicația vizuală a unei alerte, după tipul de semnătură și protocol (codificările lor se găsesc în tab-ul aplicației *Encodari*)



Encodari cunoscute

Protocol Encodari

COD	PROTOCOL
0	ICMP
1	IPv6-ICMP
2	TCP
3	UDP

Signature Encodari

COD	SEMNATURA
0	ET DNS Query for .cc TLD
1	ET INFO Discord Chat Service Domain in DNS Lookup (discord.com)
2	ET INFO Discord Chat Service Domain in DNS Lookup (gateway.discord.gg)
3	ET INFO Microsoft Connection Test
4	ET INFO Observed DNS Query to .biz TLD
5	ET INFO Observed Discord Domain (discord.com in TLS SNI)
6	ET INFO Observed Discord Domain in DNS Lookup (discord.com)
7	ET INFO Observed Discord Domain in DNS Lookup (discordapp.com)
8	ET INFO Observed Discord Service Domain (discord.com) in TLS SNI
9	ET INFO Observed Discord Service Domain (gateway.discord.gg) in TLS SNI
10	ET INFO Session Traversal Utilities for NAT (STUN Binding Request On Non-Standard High Port)
11	ET INFO Session Traversal Utilities for NAT (STUN Binding Response)

Fig. 4.15. Tab-ul *Encodari* cu protocoalele și semnăturile cunoscute decodificate

6/16/2025	2025-06-16T01:47:31.210666+0300	10.222.8.113	10.222.8.24	TCP	ET POLICY Windows Update P2P Activity	normal
<p>🔴 Explicatie AI: Activitatea pare a fi legitima si in concordanta cu traficul normal.</p> <p>✅ Recomandare: Nu sunt necesare actiuni in acest moment.</p>						

6/16/2025	2025-06-16T02:03:56.874983+0300	10.222.0.1	10.222.8.24	ICMP	Ping detected	malicious
<p>🔴 Explicatie AI: Aceasta activitate este clasificata ca malitioasa pe baza semnaturii, comportamentului si altor factori relevanti.</p> <p>✅ Recomandare: Verifica aplicatiile sau procesele care initiaza conexiuni suspecte, la in considerare izolarea acestui dispozitiv.</p>						

Fig. 4.16. Exemplu de alertă normală și malițioasă identificată și explicată de algoritmul AI în interfața utilizatorului normal

Status dispozitiv
Istoric alerte
Logout
⚠️ Activitate suspecta detectata!

Statusul Dispozitivului

IP-ul dispozitivului: 10.222.8.24

Nume dispozitiv: Lenovo RTX2050

Stare curenta: In atac!

Alerte recente

TIMESTAMP	IP SURSA	IP DESTINATIE	PROTOCOL	SEMNATURA	ETICHETA AI
2025-06-20T01:45:23.224047+0300	10.222.0.1	10.222.8.24	ICMP	Ping detected	malicious
2025-06-20T01:45:21.504406+0300	10.222.0.104	10.222.8.24	TCP	ET POLICY Windows Update P2P Activity	normal

Fig. 4.17. Exemplu de afișare a interfeței utilizatorului în cazul unui potențial atac malițios

- c) **Serverul *app.py*** – Acesta este *serverul Flask principal*, responsabil cu servirea aplicației web propriu-zise. El gestionează logica aplicației, controlul utilizatorilor și redirecționarea către interfețele dedicate în funcție de rolul utilizatorului, conform [22]. Funcționalitățile acestuia sunt următoarele:
- **Servește fișierele frontend** (*HTML, CSS, JavaScript*), care sunt organizate în directorul *frontend/public* (vezi secțiunea 4.2.4. pentru detalii extinse).
 - **Gestione a autentificării** prin verificarea credențialelor în baza de date *users.db*, criptarea parolelor folosind algoritmul *bcrypt* [23] și alocarea unui *token de sesiune* simplificat.

- **Redirecționare automată:**
 - Utilizatorii cu rol de *administrator* sunt direcționați către interfața de monitorizare completă (**/admin**);
 - Utilizatorii cu rol *standard* accesează o interfață simplificată (**/user**) care le afișează doar alertele proprii și istoricul lor.
- **Gestiune IP-uri asociate și permisiuni de vizualizare** (admin-ul poate asocia/dezasocia IP-uri de utilizatori normali).
- **Expune endpoint-uri dedicate** (`/login`, `/user_devices`, `/admin/add_user`, etc.) pentru funcționalitățile *frontend*-ului.

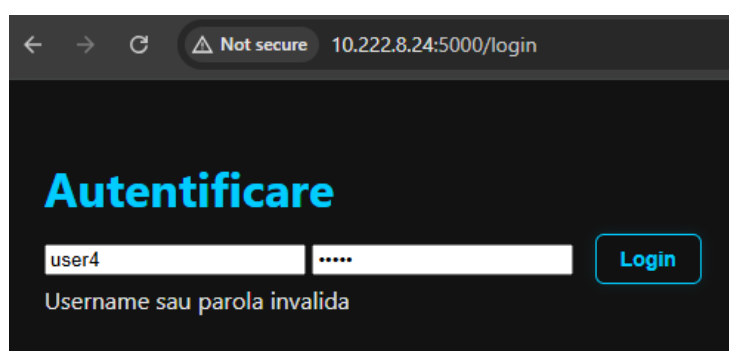


Fig. 4.18. Afișarea paginii de logare de la endpoint-ul `/login`, cu exemplu de utilizator sau parolă introdusă greșit

4.2.3. Bazele de date SQLite

Persistența datelor este un aspect esențial într-un sistem de detecție a intruziunilor, mai ales în cazul în care interacțiunea cu utilizatorul implică istoricul alertelor, autentificarea pe roluri și corelarea evenimentelor pe termen lung. Așadar, am optat, în cadrul acestui proiect, la folosirea sistemului de gestiune a bazelor de date **SQLite**, datorită avantajelor sale legate de portabilitate, ușurință în utilizare și integrarea perfectă într-o aplicație locală tip desktop. Conform descrierii din [24], *SQLite* este o bază de date relațională, care nu necesită server dedicat, oferind în același timp funcționalități *SQL* complete și fișiere ușor de transportat și replicat.

În arhitectura implementată, sunt utilizate două baze de date distincte, fiecare cu un rol bine definit: una pentru gestiunea conturilor de utilizator și a rolurilor acestora, iar cealaltă pentru înregistrarea și conservarea istoricului alertelor generate de sistemul *Suricata*, împreună cu meta-informațiile aferente procesării lor de către componenta AI.

4.2.3.1. Baza de date *users.db*

Această bază de date este responsabilă de administrarea celor autentificați în sistem, oferind suport pentru funcționalitatea de login și pentru controlul accesului diferențiat în funcție de rolul atribuit. Structura acestei baze de date este minimalistă, dar suficient de expresivă pentru a acoperi cerințele de securitate și personalizare a interfeței.

Fiecare utilizator este identificat printr-un nume unic (*username*), iar parola acestuia este salvată în format criptat, utilizând algoritmul *bcrypt*^(Error! Bookmark not defined.), o metodă robustă și consacrată pentru protejarea parolelor împotriva atacurilor de tip *dicționar* sau *brute-force*. În loc să fie stocate în clar, parola este transformată într-un *hash criptografic* care nu poate fi inversat în mod direct. În acest fel, nici chiar în cazul unui acces neautorizat la fișierul bazei de date, parolele reale nu pot fi compromise.

Pentru fiecare utilizator se atribuie un rol logic, care poate fi fie „*admin*”, fie „*user*”. Aceste roluri determină automat accesul la una dintre cele două interfețe: cea administrativă, cu acces complet la toate alertele, dispozitivele din rețea și funcționalitățile avansate (AI/XAI), respectiv cea destinată utilizatorilor obișnuiți, care se concentrează pe afișarea stării propriului dispozitiv și a alertelor relevante.

Pe lângă aceste câmpuri, baza *users.db* conține o funcționalitate esențială pentru filtrarea personalizată a alertelor: asocierea unui user cu una sau mai multe adrese IP. Această asociere este folosită pentru a filtra alertele din interfața utilizatorului și pentru a-i furniza o experiență personalizată, în care sunt vizibile doar evenimentele care îl privesc în mod direct. Astfel, sistemul îndeplinește și o funcție de izolare logică între utilizatori.

Există tot în aceeași bază de date o tabelă cu denumirea *user_devices*, care ajută la asignarea unui device/IP sau mai multe unui utilizator, ceea ce se rezolvă din interfața administrativă, în tab-ul *Administrare Utilizatori*²⁴.

4.2.3.2. Baza de date *alert_history.db*

A doua componentă de stocare persistentă este reprezentată de fișierul *alert_history.db*, care joacă un rol crucial în înregistrarea evenimentelor detectate în rețea. Aceasta este populată în timp real de către serverul *suricata_api.py* (vezi [4.2.2. – subpunctul a\)](#)).

Fiecare alertă detectată de sistem este salvată într-o tabelă numită, convențional, *alerts*. Ea cuprinde un număr semnificativ de attribute, menite să captureze toate detaliile

²⁴ Pentru o prezentare completă a structurii tabelor utilizate în *users.db*, inclusiv tipurile de date, și constrângerile de integritate, se face referire la [Anexe – Anexa 2](#) a lucrării.

relevante ale fiecărui eveniment de securitate. Printre acestea se numără: adresa IP sursă și destinație, porturile implicate, protocolul de transport, nivelul de severitate, semnătura care a declanșat alerta, momentul temporal exact (*timestamp*) și eventuale *flag*-uri specifice. În plus, fiecare înregistrare este completată cu eticheta generată de componenta de inteligență artificială, care poate clasifica alerta ca fiind benignă, malițioasă sau informativă.

Această bază de date nu doar că păstrează un jurnal cronologic al activității de securitate din rețea, dar servește și ca sursă de antrenament și validare pentru modelele AI. Având în vedere că fiecare înregistrare este structurată și etichetată, ea poate fi reutilizată în mod eficient pentru recalibrarea modelelor predictive sau pentru dezvoltarea de metode noi XAI, în fazele ulterioare ale proiectului²⁵.

id (PK)	timestamp	src_ip	dest_ip	proto	signature	label
55105	2025-06-16T03:58:57.061264+0300	10.222.0.51	141.85.216.244	TCP	ET INFO Discord Chat Service Domain in DNS Lookup (discord.com)	information

Tabel 4.2. Structura tabelii *alerts* din baza de date *alert_history.db*, cu un exemplu de alertă informațională

4.2.3.3. Integrarea bazelor de date în sistem

Ambele baze de date sunt accesate local de către serverele Flask corespunzătoare (*app.py* pentru *users.db*, respectiv *suricata_api.py* pentru *alert_history.db*), folosind biblioteca standard `sqlite3` din Python. Interacțiunea este eficientă și securizată, întrucât nu implică transmisia rețelei, iar fișierele pot fi protejate la nivel de sistem de operare. Această decizie arhitecturală contribuie la stabilitatea și la cât de scalabil este sistemul IDPS, reducând în același timp dependențele externe și resursele necesare pentru rulare.

²⁵ Pentru o prezentare completă a structurii tabelului utilizat în *alert_history.db*, inclusiv tipurile de date, și constrângerile de integritate, se face referire la [Anexe – Anexa 2](#) a lucrării.

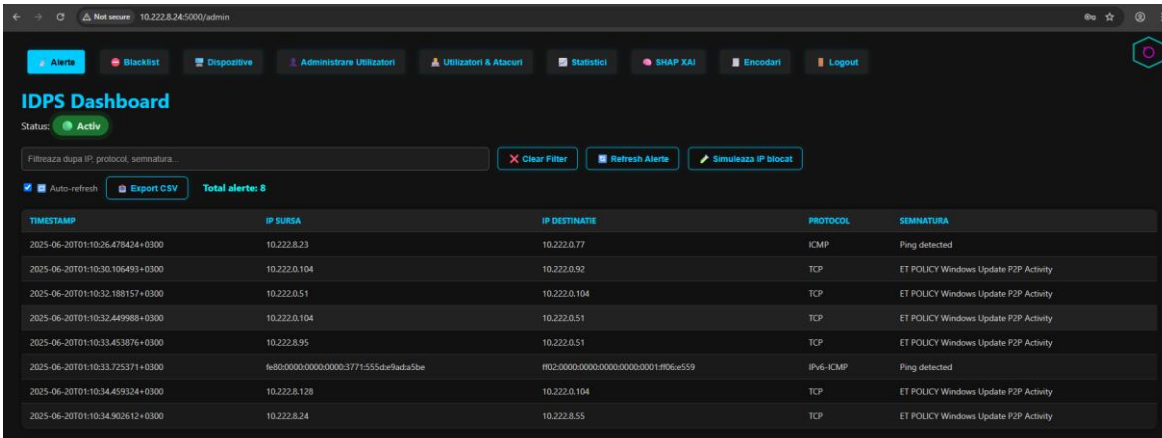
Folosirea *SQLite* în acest context oferă echilibru foarte bun între simplitate, performanță și controlul complet asupra datelor. Alegerea făcută este justificată nu doar de constrângerile hardware ale platformei, ci și de cerințele de portabilitate, mentenanță redusă și integrare ușoară cu aplicația scrisă în *Python*.

4.2.4. Interfața *frontend* – structura vizuală și funcțională a sistemului

Unul dintre elementele definitorii ale unui sistem modern de securitate informatică este capacitatea sa de a comunica eficient cu user-ul final. Interfața grafică reprezintă, în această privință, puntea esențială între complexitatea proceselor interne și nevoia utilizatorului de acces simplificat la informații relevante. În cazul aplicației IDPS dezvoltate, interfața frontend a fost concepută pentru a răspunde atât nevoilor unui administrator de sistem, cât și unui client obișnuit, cu sau fără pregătire tehnică.

Interfața a fost construită folosind tehnologii web consacrate: *HTML* pentru structurarea conținutului, *CSS* pentru aspectul vizual și *JavaScript* pentru funcționalitatea dinamică. Împreună, acestea oferă o experiență coerentă, receptivă și intuitivă, inspirată de soluții comerciale, consacrate din domeniul securității cibernetice, precum și [25].

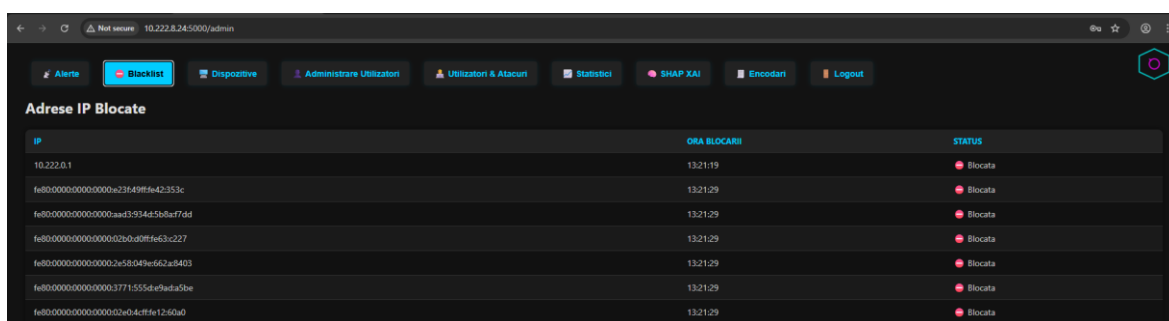
Pentru administrator, interfața include o serie de componente esențiale. Printre acestea se numără o zonă dedicată vizualizării alertelor generate de sistemul Suricata, cu posibilitatea de filtrare în timp real, resetare a criteriilor de căutare și afișarea unor explicații detaliate generate de componenta XAI (observată în tab-urile *Administrare Utilizatori*, *Utilizatori & Atacuri* și *SHAP XAI*).



TIMESTAMP	IP SURSA	IP DESTINATIE	PROTOCOL	SEMNATURA
2025-06-20T01:10:26.478424+0300	10.222.8.23	10.222.0.77	ICMP	Ping detected
2025-06-20T01:10:30.106493+0300	10.222.0.104	10.222.0.92	TCP	ET POLICY Windows Update P2P Activity
2025-06-20T01:10:32.188157+0300	10.222.0.51	10.222.0.104	TCP	ET POLICY Windows Update P2P Activity
2025-06-20T01:10:32.449988+0300	10.222.0.104	10.222.0.51	TCP	ET POLICY Windows Update P2P Activity
2025-06-20T01:10:33.453876+0300	10.222.8.95	10.222.0.51	TCP	ET POLICY Windows Update P2P Activity
2025-06-20T01:10:33.725371+0300	fe80:0000:0000:0000:3771:5555:fe8a:0bae	ff02::0000:0000:0000:0001:ff0e:ec59	IPv6-ICMP	Ping detected
2025-06-20T01:10:34.459324+0300	10.222.8.128	10.222.0.104	TCP	ET POLICY Windows Update P2P Activity
2025-06-20T01:10:34.902612+0300	10.222.8.24	10.222.8.55	TCP	ET POLICY Windows Update P2P Activity

Fig. 4.19. Interfața administratorului, în tab-ul *Alerte*

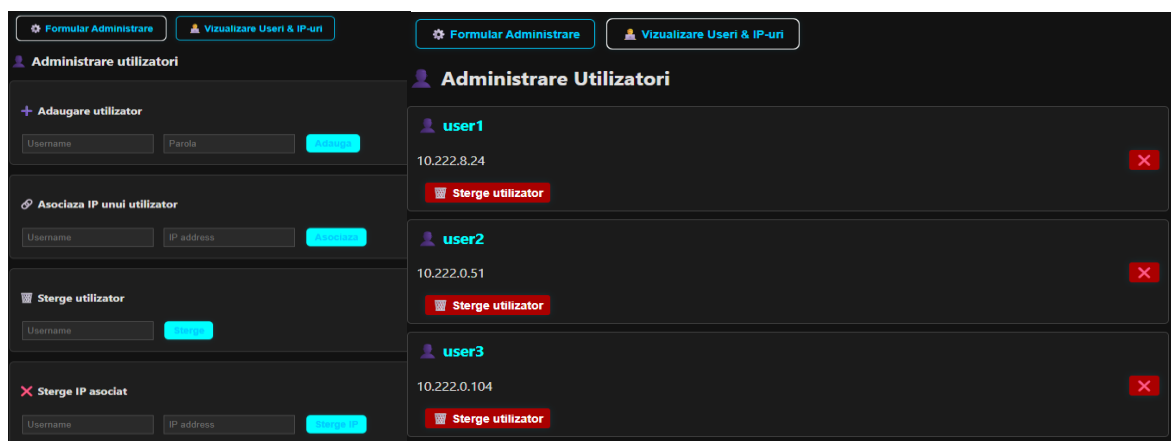
Imediat după acest tab de *Alerte*, se află tab-ul *Blacklist* unde serverul cu motorul de monitorizare, nu doar că preia alertele, dar și blochează acele IP-uri găsite în cele malițioase, activând astfel componenta importantă din IDPS – *IPS*. Într-o aplicație de acest tip, trebuie să existe concomitent cele două elemente, IDS pentru **deteție** și IPS pentru **prevenire**. Chiar dacă această componentă este realizată în mod simulat, ea verifică dacă acea alertă este de un potențial malițios și preia IP-ul acesteia care este blocat pentru o perioadă determinată de tip cu ajutorul unui script scris în `bash`, existent pe *laptopul HP* – `block_ip.sh`.



IP	ORA BLOCARII	STATUS
10.222.0.1	13:21:19	Blocata
fe80:0000:0000:0000:c234:9ff:fe42:353c	13:21:29	Blocata
fe80:0000:0000:0000:aad3:934d:5b8a:f7dd	13:21:29	Blocata
fe80:0000:0000:0000:02b0:adff:fe63:c227	13:21:29	Blocata
fe80:0000:0000:0000:2e58:04fe:662a:8403	13:21:29	Blocata
fe80:0000:0000:0000:3771:555d:efada:5be	13:21:29	Blocata
fe80:0000:0000:0000:02a0:4cffe:12:60a0	13:21:29	Blocata

Fig. 4.20. Interfața administratorului, în tab-ul *Blacklist*

O altă componentă importantă o reprezintă tab-ul *Dispozitive* pentru monitorizarea dispozitivelor conectate la rețea (vezi [4.1.2.](#) la *figura 4.3.*), unde sunt afișate informații precum numele asociat, adresa IP, adresa fizică (MAC) și statusul de conectivitate (online/offline). Mai mult, admin-ul poate asigna manual adrese IP unor utilizatori din baza de date, facilitând astfel o asociere clară între activitate și responsabilitate.



Administrare utilizatori

+ Adaugare utilizator

Username: Parola: Adauga

Asociaza IP unui utilizator

Username: IP address: Asociaza

Sterge utilizator

Username: Sterge

Sterge IP asociat

Username: IP address: Sterge

Administrare Utilizatori

user1	10.222.8.24	<input type="button" value="Sterge utilizator"/>
user2	10.222.0.51	<input type="button" value="Sterge utilizator"/>
user3	10.222.0.104	<input type="button" value="Sterge utilizator"/>

Fig. 4.21. Interfața administratorului, în tab-ul *Administrare Utilizatori* → *Formular Administrare* (stânga) și *Vizualizare Useri & IP-uri* (dreapta)

Tot în această interfață, mai găsim tab-ul *Statistici* unde se poate observa un grafic generat pentru toate alertele colectate de-a lungul timpului, tratate ca alerte pe oră, și trend-ul a șapte zile de alerte (se mai poate implementa o funcționalitate de selectare a unei anumite zile în care să observi specific pentru ea, există doar ca o listă dar nefuncțională momentan – posibil în dezvoltări ulterioare ale acestui proiect să aibă utilitate practică).

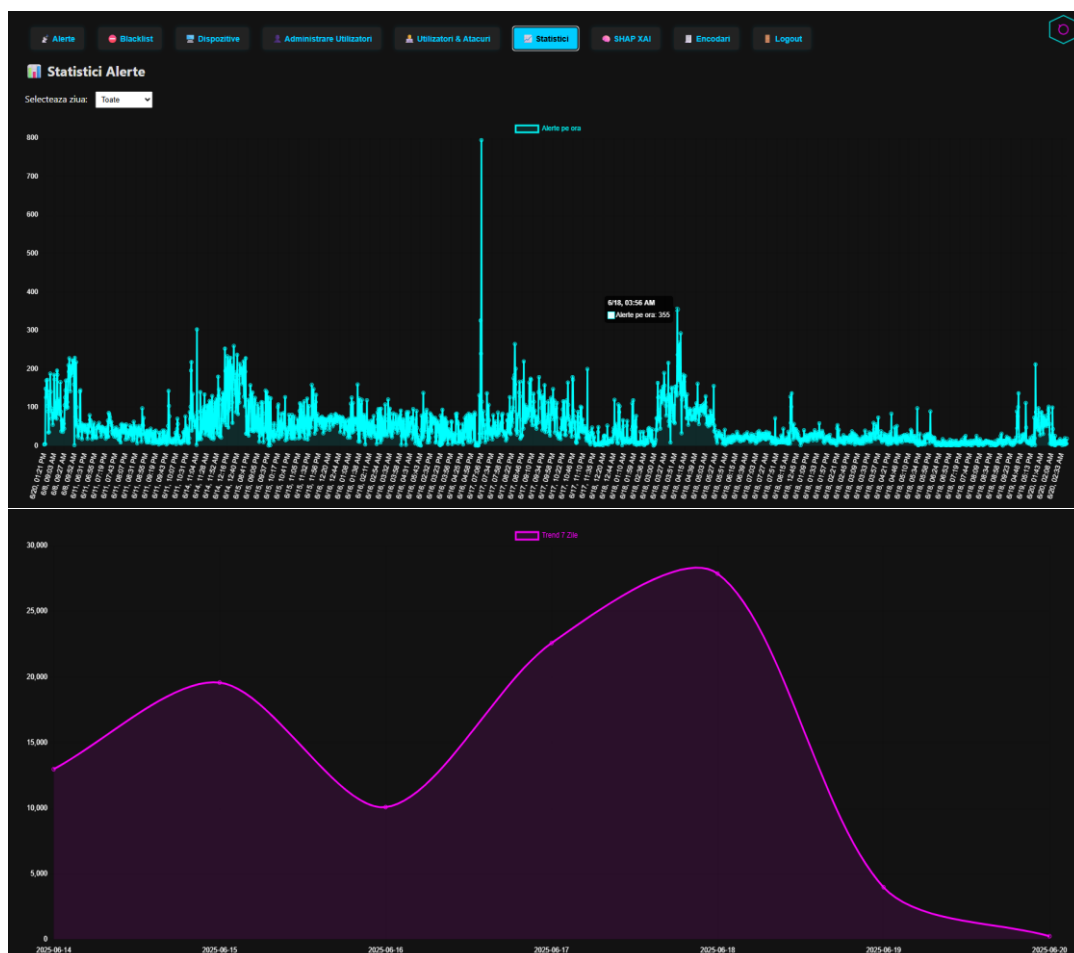


Fig. 4.22. Interfața administratorului, în tab-ul *Statistici*

Tab-urile *SHAP XAI* și *Encodari* au fost prezentate anterior în 4.2.2. la subpunctul b), în primele două figuri. Sunt foarte multe modele pe piață de reprezentarea a dashboard-ului administrativ, printre care stilul prezentat și în [26].

Pentru utilizatorul obișnuit, interfața a fost simplificată considerabil, punând accent pe claritate și relevanță. Fiecare utilizator se autentifică și vizualizează doar informațiile referitoare la propriul dispozitiv: starea de protecție, alertele asociate și un istoric detaliat al evenimentelor detectate. În plus, este inclusă o secțiune rezervată explicațiilor oferite de sistemul XAI, care va permite user-ului să înțeleagă deciziile luate de algoritmul AI.

DATA	TIMESTAMP	IP SURSA	IP DESTINATIE	PROTOCOL	SEMNATURA	ETICHETA AI
6/20/2025	2025-06-20T01:10:34.902612+0300	10.222.8.24	10.222.8.55	TCP	ET POLICY Windows Update P2P Activity	normal

Fig. 4.23. Interfața utilizatorului, în tab-ul *Istoric alerte*

4.2.5. Componenta AI a sistemului – detaliere

Integrarea unei componente de inteligență artificială în cadrul IDPS, în special IDS, adaugă un strat suplimentar sofisticat, permițând identificarea unor tipare de atac mai subtile sau necunoscute anterior. Modelul AI, din această lucrare, are rolul de a analiza alertele generate de Suricata și de a emite o etichetă de clasificare pentru fiecare alertă detectată. Acest proces automatizat contribuie nu doar la eficiență, ci și la reducerea erorilor umane în analiza traficului de rețea.

Modelul AI ales este unul bazat pe algoritmul *Random Forest*⁽¹⁴⁾, un ansamblu de arbori de decizie care funcționează prin combinarea mai multor modele simple pentru a obține un rezultat final robust și precis. Algoritmul este cunoscut pentru acuratețea sa ridicată, capacitatea de a lucra bine cu date eterogene și toleranța la **supra-potrivire**^{xi}. Antrenamentul modelului a fost realizat folosind un set de date obținut din capturi reale și simulate de trafic, extrase din baza de date *alert_history.db*. Fiecare instanță conține caracteristici esențiale precum adresele IP, protocolul utilizat, severitatea alertei, semnătura declanșată și timpul evenimentului.

El este integrat în aplicație printr-un server Flask dedicat, denumit *ai_api.py* (vezi și [4.2.2.](#) la *subpunctul b*), care primește alerte în format JSON, aplică clasificatorul și returnează una dintre etichetele predefinite: „malițios” (*malicious*), „benign” (*normal*) sau „informațional” (*information*). Clasificarea permite o triere automată a alertelor, reducând timpul de răspuns și focalizând atenția operatorului uman pe incidente critice.

Un aspect distinctiv al acestui proiect îl reprezintă integrarea inteligenței artificiale de explicare a deciziilor automate, sau XAI. Spre deosebire de abordările „*black-box*” care oferă doar rezultate fără justificări, XAI aduce un grad sporit de transparență. Astfel, clienții aplicației pot înțelege **de ce** un anumit pachet de date a fost etichetat ca fiind malițios.

Pentru a susține această funcționalitate, am utilizat **SHAP**, bazată pe teoria valorilor *Shapley* din jocurile cooperative. Potrivit [27], el evaluează contribuția fiecărei caracteristici la decizia finală a modelului, generând un răspuns în funcție de combinația protocol-semnătură oferită.

4.3. Rezultate obținute

Această secțiune prezintă în mod sistematic rezultatele obținute în urma implementării și testării sistemului IDPS, atât din punct de vedere funcțional – prin intermediul interfeței utilizator – cât și din perspectiva performanței componentelor de inteligență artificială. Evaluările includ comportamentul aplicației în fața unor atacuri simulate, analiza acurateții clasificării, fiabilitatea componentei XAI, precum și timpul de reacție și nivelul de transparență oferit utilizatorului.

4.3.1. Evaluarea funcționalității aplicației în timp real

După stabilirea conexiunilor, pornirea serviciilor, configurarea și integrarea componentelor backend și frontend, sistemul a fost testat într-un scenariu de rețea locală cu mai multe dispozitive conectate printr-un switch de management dedicat. În urma generării controlate a unor fluxuri de trafic, sistemul a reușit să capteze și să clasifice alerte, oferind feedback imediat atât administratorului cât user-ului.

De exemplu, în figura de mai jos, se observă interfața unui utilizator logat (cu numele dispozitivului *Lenovo RTX 2050*), care a fost notificat în timp real despre o activitate detectată de *Suricata* pe *portul TCP*. Sistemul a semnalat existența unei semnături de tip „*ET POLICY Windows Update P2P^{xii} Activity*”, alertă frecvent întâlnită în traficul obișnuit al sistemelor Windows. După ce alerta a fost preluată de serverul AI, aceasta a fost clasificată drept „normală”, decizie reflectată în *dashboard* sub forma unei etichete vizuale.



Fig. 4.24. Afișarea unei alerte etichetate ca trafic normal, în interfața utilizatorului²⁶

Pe lângă această clasificare, sistemul a oferit și o explicație generată prin intermediul XAI, care a indicat că activitatea observată este în concordanță cu comportamentul standard al sistemului și nu necesită acțiuni suplimentare. Explicația este vizibilă în *istoricul* complet al *alertelor*.

DATA	TIMESTAMP	IP SURSA	IP DESTINATIE	PROTOCOL	SEMNATURA	ETICHETA AI
6/20/2025	2025-06-20T01:10:34.902612+0300	10.222.8.24	10.222.8.55	TCP	ET POLICY Windows Update P2P Activity	normal
<p>● Explicatie AI: Activitatea pare a fi legitima si in concordanta cu traficul normal.</p> <p>✓ Recomandare: Nu sunt necesare actiuni in acest moment.</p>						

Fig. 4.25. Exemplu de explicație oferită de XAI pentru o alertă detectată – „Activitatea pare a fi legitimă și în concordanță cu traficul normal.”

²⁶ **Observație:** Sistemul actual are doar două statusuri, „Protejată” sau „In atac!”. Starea dispozitivului care apare în imagine este prezentă indiferent dacă a fost identificat ca pachet *normal/informational/malicious*. Implementarea actuală avertizează utilizatorul de orice trafic îi vizează dispozitivul, care apoi este înregistrat într-un istoric de alerte. În viitor, se poate implementa o funcționalitate îmbunătățită care să apară ca fiind „In atac!” doar dacă atacul identificat este periculos.

4.3.2. Performanța clasificatorului AI

Pentru a evalua componenta AI, am realizat mai multe sesiuni de antrenament pe seturi de date exportate automat din baza de date *alert_history.db*. Aceste date au fost etichetate fie automat (prin asocierea semnăturilor cu etichete predefinite), fie manual, acolo unde semnătura nu avea corespondent. Etichetele disponibile sunt „malicious”, „normal” și „information”.

Prima sesiune de antrenament a folosit un eșantion de aproximativ 10.000 de alerte. Rezultatele au fost încurajatoare, dar imperfecte – în special pentru clasa „information”, unde *recall*²⁷-ul a fost 0.10, semnalând o dificultate a modelului în identificarea corectă a acestei categorii. Matricea de confuzie aferentă confirmă acest dezechilibru.

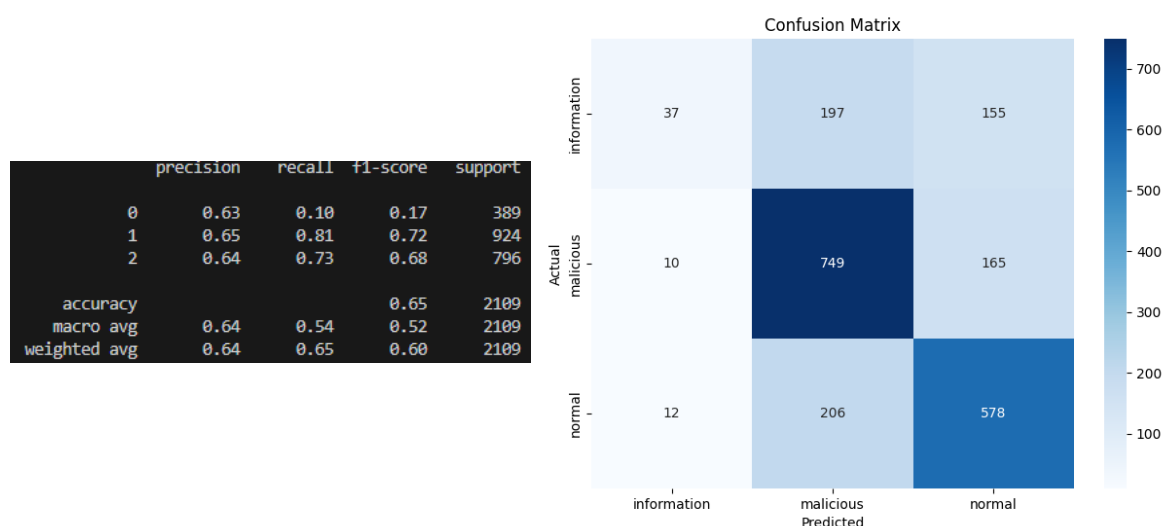


Fig. 4.26. Măsurători de performanță pentru modelul antrenat inițial – acuratețe totală: 65% (stânga) și matricea de confuzie aferentă (dreapta)

Într-o a doua sesiune, am dublat volumul de date utilizate, antrenând modelul pe un total de 20.000 de alerte distincte. Pentru a evita supra-învățarea și a simula date realiste, 10% din aceste instanțe au fost modificate printr-o metodă de corupere aleatorie a etichetelor²⁸. Această tehnică adaugă variabilitate și oferă modelului șansa de a învăța diferențe mai subtile între instanțe similare.

²⁷ Termen tehnic englezesc cu specific în domeniul ML care se referă la capacitatea unui model de a identifica **toate instanțele pozitive** dintr-un set de date (**raportul** dintre **numărul de predicții corecte** și **numărul total de instanțe pozitive** din setul de date).

²⁸ O secțiune din fișierul de antrenare este disponibilă la [Anexe](#) – Anexa 3 a acestei lucrări.

După reantrenare, acuratețea a mai crescut cu un procent semnificativ la 87%, iar scorurile pentru toate cele trei clase au crescut. În special, scorul F1, pentru clasa „*information*”, a crescut de la 0.17 la 0.36, iar media macro s-a ridicat de la 0.52 la 0.72, semnificând o distribuție mai echilibrată a performanței de clase.

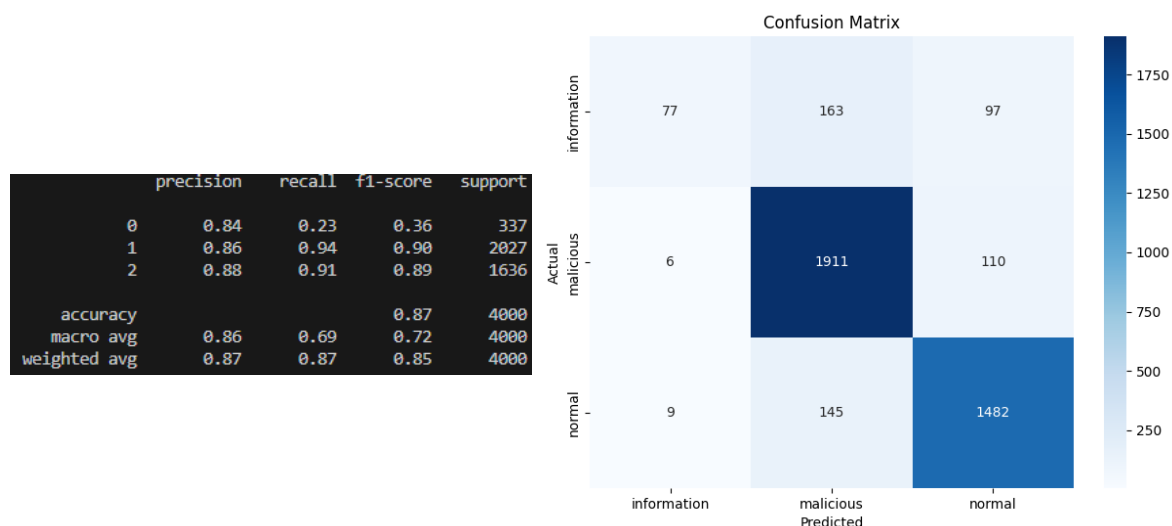


Fig. 4.27. Măsurători de performanță pentru modelul antrenat în faza a doua – acuratețe totală: 87% (stânga) și matricea de confuzie aferentă (dreapta)

După mai multe sesiuni de prelucrare și de antrenare, am ajuns la o acuratețe de 93%, cu scoruri îmbunătățite, mai ales pentru categoria „*information*” care avea foarte puține date de antrenament în trecut, dar colectând în timp, suportul lui a crescut semnificativ, de asemenea și scorul F1 ajunge la 0.55, cu un *recall* mai bun decât modelele anterioare și mediile mult mai mari. Ultimul set de date pe care a fost antrenat conține 100.000 de date brute prelucrate cu o distribuție mai bună a claselor, însă cu timpul, acest model poate fi îmbunătățit, putând să se ajungă la precizii mai mari de cea actuală.

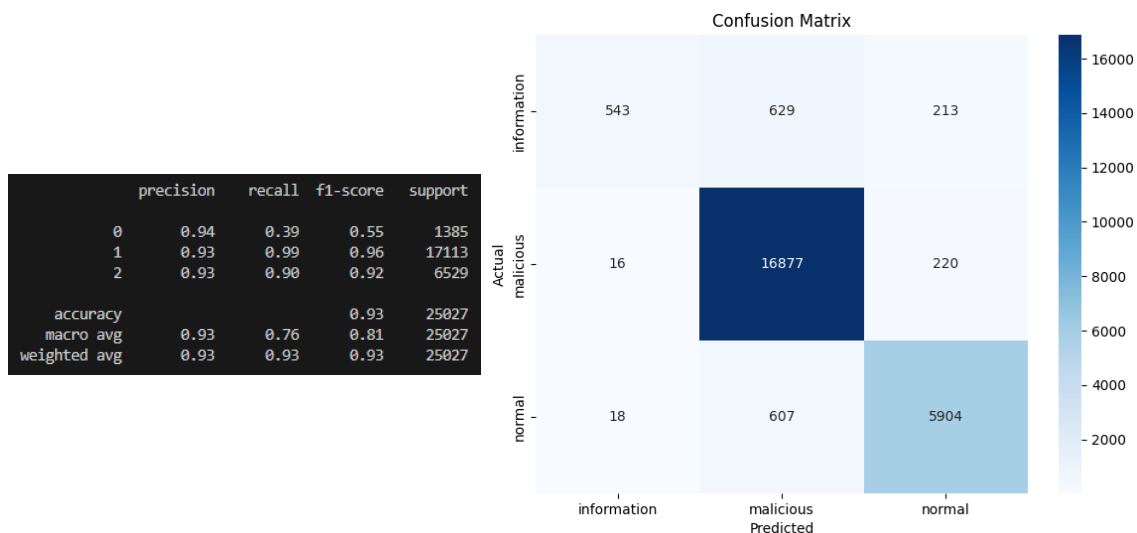


Fig. 4.28. Măsurători de performanță pentru modelul antrenat în faza finală – acuratețe totală: 93% (stânga) și matricea de confuzie aferentă (dreapta)

Aceste rezultate validează alegerea unui model *Random Forest* ca soluție eficientă pentru clasificarea alertelor în timp real. De asemenea, demonstrează importanța unui volum mare de date de antrenare unor modele robuste.

4.3.3. Observații privind alertele imposibil de clasificat

În etapa incipientă a dezvoltării, modelul nu dispunea de suficiente date pentru a învăța tiparele fiecărei semnături. Drept urmare, unele alerte generate de Suricata nu puteau fi clasificate, iar eticheta AI era înregistrată ca indisponibilă. Un exemplu concret este prezentat în imaginea următoare, unde semnătura „*ET POLICY Windows Update P2P activity*” nu fusese încă etichetată în sistemul de învățare automată, iar modelul nu a returnat o predicție.

6/11/2025		6/8/2025				
DATA	TIMESTAMP	IP SURSA	IP DESTINATIE	PROTOCOL	SEMNATURA	ETICHETA AI
6/11/2025	2025-06-11T18:10:13.119299+0300	10.222.8.24	10.222.0.92	TCP	ET POLICY Windows Update P2P Activity	n/a

Fig. 4.29. Alertă fără clasificare AI în absența unei etichete în setul de antrenament

Situația curentă evidențiază importanța mentenanței continue a bazei de date pentru a învăța tiparele fiecărei semnături.

4.3.4. Concluzii asupra rezultatelor

Modelul antrenat în cadrul aplicației mele IDPS a demonstrat o capacitate consistentă de a clasifica și explica alerte în rețea. Precizia generală a fost ridicată (peste 93%), iar metoda de explicare a deciziilor AI a fost implementată eficient prin SHAP. Mai mult, integrarea acestora într-o interfață modernă, accesibilă atât pentru utilizatori cu experiență sau fără experiență tehnică, validează obiectivul principal al proiectului – de a oferi un sistem de protecție inteligent, transparent și ușor de utilizat.

5. Concluzii și direcții de dezvoltare ulterioară

Lucrarea de față a avut ca obiectiv **proiectarea și implementarea unui sistem distribuit de detecție și prevenire a intruziunilor într-o rețea**, sau **IDPS**, adaptat nevoilor utilizatorilor fără experiență în domeniul securității cibernetice. Prin îmbinarea componentelor clasice de monitorizare a traficului cu tehnologii moderne de **inteligență artificială** și **explicații automatizate (XAI)**, soluția propusă reușește să transforme un domeniu tehnic, adesea neclar, într-o interfață accesibilă, vizuală și educativă.

Implementarea a acoperit întregul lanț funcțional, de la colectarea traficului prin opțiunea de *port mirror* pe un switch de management, până la clasificarea alertelor folosind modele *Random Forest* și afișarea acestora în timp real într-un *dashboard* personalizat. Infrastructura a fost distribuită între mai multe dispozitive, în scopul simulării unui mediu real, cu diverse roluri: de la **senzori de rețea** la **servere AI** și **interfețe grafice**.

Sistemul a fost testat cu succes în condiții reale de rețea, fiind capabil să detecteze și să clasifice o varietate de alerte, de la activitate legitimă – precum actualizări de sistem sau conexiuni *Discord/ Spotify* – până la tentative simulate de atac (*scanări de porturi, ping excesiv, trafic P2P^(xii) necunoscut*). Rata de detecție a fost una ridicată, iar integrarea cu componenta de învățare automată a permis extinderea capabilităților dincolo de semnăturile statice.

5.1. Avantajele soluției propuse

Una dintre cele mai importante contribuții asupra sistemului implementat este adaptarea sa la utilizatorul obișnuit. Interfața simplificată oferă informații clare despre starea dispozitivului propriu, alertele relevante și recomandările aferente, fără a solicita cunoștințe de specialitate. În același timp, administratorii beneficiază de o imagine detaliată a întregii rețele, cu opțiuni avansate de filtrare, analiză și asociere a alertelor.

Prin utilizarea algoritmilor AI și XAI, fiecare decizie de clasificare a fost însoțită de o justificare clară, exprimată atât în text, cât și grafic. Astfel, sistemul nu doar că detectează, ci și educă, explicând de ce un anumit comportament este considerat legitim sau periculos.

Nu în ultimul rând, modul în care aplicația este structurată, cu roluri diferențiate (*admin* și *user normal*), baze de date locale, rulare distribuită, asigură atât securitate cât și o scalare care poate fi mărită, fără a necesita infrastructură suplimentară sau resurse costisitoare.

5.2. Direcții de dezvoltare viitoare

Deși sistemul este funcțional și complet în forma sa actuală, există numeroase direcții de îmbunătățire și extindere, fie pentru uz profesional, fie pentru adaptarea sa ca **instrument educativ** sau **open-source**. Voi menționa pe cele mai relevante mai jos.

1. Extinderea componentei de prevenire (IPS)

Sistemul actual funcționează predominant în mod pasiv, detectând și semnalând evenimente suspecte și simulând **componenta IPS**. O posibilă dezvoltare, cum este descris și în [28], ar consta în transformarea sa într-un sistem activ de prevenire, care să poată bloca automat traficul considerat malițios, pe baza unei *liste albe* (*whitelist*) îmbunătățit. Astfel, orice adresă IP sau semnătură necunoscută ar putea fi automat respinsă la nivel de rețea sau aplicație.

2. Lansarea unei versiuni open-source și crearea unei aplicații desktop complete

Publicarea proiectului sub o licență deschisă (exemplu: MIT sau GPL) ar putea permite comunității să contribuie la dezvoltare, să adauge noi funcționalități, semnături personalizate sau metode alternative de antrenare AI. Ar încuraja, astfel, auditarea codului și testarea sistemului în contexte diverse, ceea ce ar conduce la un produs mai sigur și mai performant, subliniind și concluziile din [29].

În forma actuală, interfața este disponibilă doar prin browser, cu pornirea automată a serviciilor de monitorizare și detecție de pe un laptop și de pe altul pornirea *serverelor Flask* dedicate (vezi *figura 4.12.* de la *secțiunea 4.2.2.*). Totuși, ea poate fi împachetată într-o aplicație de sine stătătoare, folosind tehnologii precum *Electron.js*. Aceasta ar permite rularea sistemului într-un mod izolat, offline, fără dependențe externe și ar îmbunătăți experiența userilor.

3. Posibilă utilizare educațională

Având în vedere nivelul ridicat de transparență și modularitatea arhitecturii, sistemul IDPS poate fi adaptat ca platformă de predare pentru cursuri de securitate cibernetică, învățare automată aplicată sau rețelistică, cum este și [30]. Studenții ar putea interacționa direct cu alertele, modelul AI și explicațiile SHAP, învățând prin practică cum funcționează un sistem real de apărare.

5.3. Concluzia finală

Prin complexitatea sa internă, dar și prin simplitatea și claritatea interfeței vizuale, sistemul realizat reușește să atingă un echilibru între funcționalitate, securitate și accesibilitate. El demonstrează că soluțiile moderne de securitate pot fi adaptate nu doar pentru experți, ci și pentru un om obișnuit, fără a compromite eficiența. În același timp, prin modul în care este construit, acest sistem deschide numeroase posibilități de extindere, adaptare și aplicare în contexte mai largi, fie în mediul academic, fie în cel industrial. Este, așadar, un exemplu de inițiativă practică cu aplicabilitate reală și potențial de impact pozitiv în societate.

6. Bibliografie

- [1] N. & B. T. & M. S. & A. A. & v. d. V. C. & A. R. Azeez, "Intrusion Detection and Prevention Systems: An Updated Review," *Advances in Intelligent Systems and Computing*, vol. 1042, pp. 685-696, Jan 2020. [Accesat 2025].
- [2] B. J. Mary, „AI Techniques in Intrusion Detection and Prevention,” ResearchGate, New York, 2025. [Accesat 2025].
- [3] S. & A. J. & A. W. & M. S. & R. S. & B. I. & S. M. Neupane, „Explainable Intrusion Detection Systems (X-IDS): A Survey of Current Methods, Challenges, and Opportunities,” *IEEE Access*, vol. 10, nr. 7, pp. 112392-112415, 2022. [Accesat 2025].
- [4] R. Baloch, *Ethical hacking and penetration testing guide*, vol. xxvii, NW: Boca Raton : CRC Press, Taylor & Francis Group, 2015. [Accesat 2025].
- [5] P. European, „Securitate cibernetică: principalele amenințări,” EuroParl, EU, 2023. [Accesat 2025].
- [6] European Commission, „Attitudes on Data Protection and Electronic Identity in the European Union,” EU Institution, EU, 2011. [Accesat 2025].
- [7] D. Eduard, „Analiza sistemelor IDPS bazate pe AI,” SCSS - Secțiunea SIAIV [Moodle], București, 2025. [Accesat 2025].
- [8] J. D.-V. G. M.-F. E. V. P. García-Teodoro, „Anomaly-based network intrusion detection: Techniques, systems and challenges,” *Computers & Security*, vol. 28, nr. 1-2, pp. 18-28, February-March 2009. [Accesat 2025].
- [9] J. & G. H. & Z. Z. Guo, „Research on High Performance Intrusion Prevention System Based on Suricata,” *Highlights in Science, Engineering and Technology*, vol. 7, nr. 2022, pp. 238-245, 2022. [Accesat 2025].
- [10] A. & K. A. & S. S. Tasneem, "Intrusion Detection Prevention System using SNORT," *International Journal of Computer Applications*, vol. 181, no. 32, pp. 21-24, December 2018. [Accesat 2025].
- [11] A. a. S. S. a. D. U. a. P. S. Tiwari, „Refinements In Zeek Intrusion Detection System,” *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1, pp. 974-979, 2022. [Accesat 2025].
- [12] M. Salmon, „Suricata and OSSEC IDPS Systems Review,” ResearchGate, Auckland, New Zealand, 2022. [Accesat 2025].
- [13] H. K. M. O. A.-K. A. P. G. M. a. M. C. Sampath Rajapaksha, „AI-Based Intrusion Detection Systems for In-Vehicle Networks: A Survey,” *ACM Computing Surveys*, vol. 55, nr. 11, pp. 1-40, 09 February 2023. [Accesat 2025].
- [14] „How to configure Port Mirror on TP-Link Easy Smart Switches,” TP-Link, 29 06 2022. [Interactiv]. Available: <https://www.tp-link.com/us/support/faq/527/>. [Accesat 2025].
- [15] „Security best practices (newest version),” Arch Linux, 15 06 2025. [Interactiv]. Available: <https://wiki.archlinux.org/title/Security>. [Accesat 2025].
- [16] „Suricata Documentation - Network Traffic Analysis,” OISF, 2024. [Interactiv]. Available: <https://docs.suricata.io/en/latest/>. [Accesat 2025].

- [17] M. Grinberg, „Flask Web Development: Developing Web Applications with Python,” Flask Book, 2018. [Interactiv]. Available: <https://flaskbook.com/>. [Accesat 2025].
- [18] „Ethernet Cable Categories Explained: A Brief History,” Fluke Networks, 24 02 2022. [Interactiv]. Available: <https://www.flukenetworks.com/blog/cabling-chronicles/ethernet-cable-history>. [Accesat 2025].
- [19] E. a. R. N. C. Albin, „A Realistic Experimental Comparison of the Suricata and Snort Intrusion-Detection Systems,” in *2012 26th International Conference on Advanced Information Networking and Applications Workshops*, Fukuoka, Japan, 2012. [Accesat 2025].
- [20] E. Francis, „YAML Tutorial: A Complete Guide to Language, Format, and Syntax,” 2025. [Interactiv]. Available: <https://www.cloudbees.com/blog/yaml-tutorial-everything-you-need-get-started>. [Accesat 2025].
- [21] „Balancing Flexibility and Security in YAML Configurations,” Vasile Crudu & MoldStud Research Team, 2024. [Interactiv]. Available: <https://moldstud.com/articles/p-balancing-flexibility-and-security-in-yaml-configurations>. [Accesat 2025].
- [22] „Securing REST APIs with Flask: Authentication and Authorization,” YouTube - Code And Course, 2023. [Interactiv]. Available: <https://www.youtube.com/watch?v=HQbX4mwCZl4>. [Accesat 2025].
- [23] G. Reed, „Hashing Passwords with Bcrypt in Flask: How to Keep User Data Safe,” Medium, 19 04 2025. [Interactiv]. [Accesat 2025].
- [24] D. R. Hipp, „Appropriate Uses For SQLite,” SQLite, 2007-2025. [Interactiv]. Available: <https://www.sqlite.org/whentouse.html>. [Accesat 2025].
- [25] „W3.CSS Website Templates,” W3schools, 2024. [Interactiv]. Available: https://www.w3schools.com/w3css/w3css_templates.asp. [Accesat 2025].
- [26] „Responsive Sales Dashboard Design Using HTML And CSS | Admin Dashboard,” YouTube - Programmer Cloud, 2023. [Interactiv]. Available: <https://www.youtube.com/watch?v=G1II-xhJUIE>. [Accesat 2025].
- [27] S. a. L. S.-I. Lundberg, „A Unified Approach to Interpreting Model Predictions,” in *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach Convention & Entertainment Center, 2017. [Accesat 2025].
- [28] R. a. H. H. Altaie, „An Intrusion Detection System using a Hybrid Lightweight Deep Learning Algorithm,” *Engineering, Technology and Applied Science Research*, vol. 14, nr. 5, pp. 16740-16743, 10 2024. [Accesat 2025].
- [29] J. a. K. M. a. S. S. Reza, „Proactive Cyber Threat Detection Using AI and Open-Source Intelligence,” *Journal of Computer Science and Technology Studies*, vol. 7, nr. 5, pp. 558-576, 06 2025. [Accesat 2025].
- [30] „Learn Cyber Security | TryHackMe Cyber Training,” TryHackMe, 2024. [Interactiv]. Available: <https://tryhackme.com/>. [Accesat 2025].

7. Anexe

Anexa 1 – Prezintă o secțiune a fișierului *suricata.yaml* axată pe alegerea regulilor:

```
default-rule-path: /etc/suricata/rules

rule-files:

  - suricata.rules

  - local.rules

##

## Auxiliary configuration files.

##

classification-file: /etc/suricata/classification.config

reference-config-file: /etc/suricata/reference.config

threshold-file: /etc/suricata/threshold.config
```

Anexa 2 – Prezintă tabelele din bazele de date *users.db* și *alert_history.db* și codul SQL de inițializare a lor:

a) *users.db*

	id	username	password_hash	role
1	1	admin	2432622431322471543451514a55414a764a34456...	admin
2	2	user1	243262243132247a384971626273564a414d714b4...	user
3	3	user2	243262243132247a4c577057636734435078697a5...	user
4	5	user3	24326224313224517166333246694162574d574c4...	user

Fig. 7.2.a.1. Tabela *users*

- Interogarea SQL pentru crearea tabelii *users*:

```
CREATE TABLE IF NOT EXISTS users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    username TEXT UNIQUE NOT NULL,
    password_hash TEXT NOT NULL,
    role TEXT NOT NULL CHECK(role IN ('admin', 'user'))
)
```

	id	user_id	ip_address
1	2	2	10.222.8.24
2	3	3	10.222.0.51
3	4	5	10.222.0.104

Fig. 7.2.a.2. Tabela *user_devices*

- Interogarea SQL pentru crearea tabelii *user_devices*:

```
CREATE TABLE IF NOT EXISTS user_devices (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    user_id INTEGER NOT NULL,
    ip_address TEXT NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(id)
)
```

b) *alert_history.db*

id	timestamp	src_ip	dest_ip	proto	signature	label
1	2025-06-08T08:42:25.077300+0300	10.222.0.116	141.85.216.244	UDP	ET INFO Observed Discord Domain in DNS Lookup (disc...	NULL
2	2025-06-08T08:42:25.077300+0300	10.222.0.116	141.85.216.244	UDP	ET INFO Discord Chat Service Domain in DNS Lookup (d...	NULL
3	2025-06-08T08:42:25.077595+0300	10.222.0.116	141.85.216.244	UDP	ET INFO Observed Discord Domain in DNS Lookup (disc...	NULL
4	2025-06-08T08:42:25.077595+0300	10.222.0.116	141.85.216.244	UDP	ET INFO Discord Chat Service Domain in DNS Lookup (d...	NULL

Fig. 7.2.b. Tabela *alerts*

- Interogarea SQL pentru crearea tabelii *alerts*:

```
CREATE TABLE IF NOT EXISTS alerts (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    timestamp TEXT,
    src_ip TEXT,
    dest_ip TEXT,
    proto TEXT,
    signature TEXT,
    label TEXT,
)
```

Anexa 3 – Prezintă partea de antrenare a algoritmului *Random Forest*, realizat în *training_model.py*:

- Încărcarea datelor: `df = pd.read_csv('db/alerts_for_labeling.csv')`
- Eliminarea rândurilor fără label: `df = df.dropna(subset=['label'])`
- Introducem variație în date prin perturbarea etichetelor: `label_corruptie = df.sample(frac=0.1)`
`df.loc[label_corruptie.index, 'label'] = np.random.choice(df['label'].unique(), size=len(label_corruptie))`
- Codificare etichete: `label_encoder = LabelEncoder()`
`df['label_encoded'] = label_encoder.fit_transform(df['label'])`
- Codificare consistentă a feature-urilor:
`proto_encoder = LabelEncoder()`
`sig_encoder = LabelEncoder()`
`df['proto_encoded'] = proto_encoder.fit_transform(df['proto'])`
`df['signature_encoded'] = sig_encoder.fit_transform(df['signature'])`
`X = df[['proto_encoded', 'signature_encoded']].copy()`
`y = df['label_encoded']`
- Aplicăm tehnica de split a setului de date:
`X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)`
- Modelul:
`model = RandomForestClassifier(n_estimators=100, random_state=42)`
`model.fit(X_train, y_train)`
- Predicțiile: `y_pred = model.predict(X_test)`
- Clasificarea: `print(classification_report(y_test, y_pred))`
- Matricea de confuzie: `cm = confusion_matrix(y_test, y_pred)`

- *Notații speciale din această lucrare:*

Termen^(x), unde (x) face referire la un *footnote/endnote* deja existent.

ⁱ Termen tehnic preluat din englezescul *local area networks (LAN)*

ⁱⁱ Termen tehnic preluat din englezescul „zero-day” attack

ⁱⁱⁱ Termen tehnic preluat din englezescul *machine learning (ML)*

^{iv} Termen tehnic preluat din englezescul *neural network (NN)*

^v Termen tehnic preluat din englezescul *autoencoder (AE)*

^{vi} Termen tehnic preluat din englezescul *clustering*

^{vii} Termen tehnic preluat din englezescul *convolutional neural network (CNN)*

^{viii} Termen tehnic preluat din englezescul *recurrent neural network (RNN)*

^{ix} Termen tehnic preluat din englezescul *deep learning (DL)*

^x Termen tehnic cu specific în domeniul de *ML* (aplicat și în această lucrare)

^{xi} Termen tehnic preluat din englezescul *overfitting*

^{xii} Abreviere de la termenul englezesc *peer-to-peer*