Developing an Intelligent Security Monitoring Platform for Internet Domains. A Practical Implementation Approach*

Mihail DUMITRACHE[1,3], Ioan Ștefan SACALĂ[2], Carmen Ionela ROTUNĂ[1,2]*, Alexandru GHEORGHIȚĂ[1], Ionut SANDU[1], Dragoș SMADA[1]

[1] National Institute for Research and Development in Informatics, 8-10 Mareșal Averescu Avenue, Bucharest, 011455, Romania
mihail.dumitrache@ici.ro, carmen.rotuna@ici.ro (**Corresponding author*),
alexandru.gheorghita@ici.ro, ionut.sandu@ici.ro,
dragos.smada@ici.ro

[2] National University of Science and Technology Politehnica Bucharest, 313 Splaiul Independenţei, Bucharest, 060042, Romania
ioan.sacala@upb.ro

[3] University of Bucharest, Faculty of Letters, 5-7 Edgar Quinet Street, Bucharest, 010017, Romania

Abstract: Due to the rise in the number of cyber threats, preserving the reputation of Internet domains has become a significant challenge. This study builds upon the prior research on the domain reputation analysis and proposes an application which was designed for evaluating and monitoring domain trustworthiness. As such, this study follows a structured approach involving an enterprise architecture framework (based on TOGAF) for implementing a scalable and adaptable monitoring system. The proposed solution integrates Django, PostgreSQL, Docker, and Machine Learning techniques for assessing domain reputation. This paper outlines the architecture, the evaluation mechanisms and the implementation of the domain reputation system for the proposed domain assessment platform.

Keywords: Domain name, Reputation, Cybersecurity, Big data, Machine learning, Domain security, ccTLD.

1. Introduction

The expansion of digital interactions has amplified the need for secure online environments. Malicious domains are a significant threat to users and organisations, therefore, efficient solutions are needed for domain name cybersecurity monitoring. (Vevera, Cîrnu & Rădulescu, 2022).

The domains registered with the intent of engaging in phishing, malware distribution, and fraud pose threats to individuals, businesses, and governments. The Domain Name Reputation System (TLDRep) project aims to establish an automated domain reputation monitoring system, ensuring that the domains under the .ro ccTLD (country code Top Level Domain) adhere to cybersecurity standards.

By integrating machine learning models, historical domain data, and a real-time analysis, TLDRep enables the proactive detection and mitigation of cybersecurity threats associated with domain registrations and operations. Similar approaches, such as the Notos reputation system, Kopis (Antonakakis et al., 2010), and Exposure (Bilge et al., 2011) demonstrate the importance of data-driven monitoring in domain security. However, these approaches lack the real-time scoring updates and the adaptability that TLDRep shall provide.

This study extends the findings from prior research (Smada et al., 2024; Rotună et al., 2023; Dumitrache et al., 2024) by introducing an implementation of a domain reputation system. It focuses on the design, development, and evaluation of an intelligent domain reputation monitoring platform leveraging the emerging technologies. The remainder of this paper is as follows. Section 2 presents prior research on domain reputation. Section 3 outlines a generic architecture of a domain name reputation system by using TOGAF, as a preliminary step to solution development. Section 4 highlights the technologies and tools that will be used for developing the application. Section 5 defines a scoring system for classifying domains into several categories, namely malicious, potentially malicious, normal, and trusted. The development process for the domain reputation system is highlighted in Section 6. Section 7 includes a comparative evaluation of the developed platform in relation to other existing domain reputation platforms. Finally, Section 8 outlines the conclusion of this study.

* This article represents an extension of the paper "A Domain Reputation System Architecture Description Using TOGAF", published in *Studies in Informatics and Control*, vol. 33(1), 2024, pp. 61-72.

2. State of the Art

Traditionally, blacklists and whitelists have been employed to determine the reputation of domain names. However, relying solely on blacklists proves to be ineffective in identifying both the known and newly generated malicious URLs. This approach often depends on human input and is time-consuming, especially in real-time environments. Vinayakumar et al. (2018) highlighted the inefficacy of using only blacklists for domain reputation assessment, emphasising the need for more dynamic and automated solutions. Furthermore, blacklisting can lead to false positives, in which case the legitimate domains are incorrectly flagged as malicious, causing unnecessary disruptions (Deri & Fusco 2023). Whitelisting, while more restrictive, can hinder legitimate activities if not properly managed, as it requires constant updates to include new trustworthy domains (InstaSafe Inc., 2024; Envescent Cybersecurity, 2024).

Machine learning (ML) has become integral to modern domain reputation systems. By analysing vast datasets of domain-related behaviors, ML models can identify patterns indicative of malicious activity (Banciu, Petre & Dumitrache, 2019). For instance, DNS reputation systems utilise ML to assess domains based on DNS behaviour, enhancing the detection of potentially malicious domains (Galloway et al., 2024). These systems analyse features such as query frequencies, response times, and traffic patterns to discern between legitimate and malicious domains. Advanced ML algorithms, including supervised and unsupervised learning techniques, have been employed to improve the detection accuracy for malicious activity and reduce the number of false positives. Recent studies have explored the use of deep learning approaches, such as graph neural networks, to capture complex relationships between domains and enhance cyber threat detection capabilities (Peng et al., 2019).

Graph-based methodologies have gained traction in computing domain reputations. By representing domains and their interactions as nodes and edges in a graph, these approaches can apply different algorithms to detect domains with a bad reputation (Mishsky, Gal-Oz & Gudes, 2014). One of these methods involves using flow algorithms on DNS-based graphs to compute domain reputation, demonstrating its effectiveness in identifying malicious domains. This approach leverages the interconnected nature of domains, analysing the relationships between domains and activity patterns to assess domain trustworthiness. Graph-based models can effectively capture the propagation of malicious activities across domains, aiding in the early detection of emerging threats. For example, the MalShoot system utilises graph embedding techniques to represent domains in a low-dimensional space, facilitating the identification of malicious domains through machine learning classifiers (Peng et al., 2019).

A significant challenge in domain reputation assessment is distinguishing between the compromised domains and those maliciously registered from the beginning. Addressing this, the COMAR (Maroofi et al., 2020) approach has been proposed to classify domains accordingly, complementing the existing reputation systems and enhancing detection accuracy in this context. COMAR employs a combination of behavioral analysis and historical data to differentiate between these two categories, enabling the implementation of more targeted and effective mitigation strategies. By understanding the origin and intent behind a domain's malicious activity, security systems can implement more precise countermeasures. This distinction is crucial, as compromised domains may belong to legitimate entities and require different remediation strategies in comparison with maliciously registered domains (Bilge et al., 2011).

The complexity of identifying and classifying malicious domains can be addressed by employing contemporary learning and adaptive methodologies. Unsupervised intelligent domain classification can be accomplished by using metaheuristic-based search algorithms such as Cuckoo Search. Sarkar et al. (2013) proposed the use of Cuckoo Search for unsupervised domain classification, suggesting that object-oriented engineering can be utilised to implement this model, with possibilities for future extensions. This approach leverages the optimisation capabilities of metaheuristic algorithms to explore the feature space effectively, facilitating the identification of malicious domains without having labeled data as a prerequisite. Further studies have demonstrated the efficacy of hybrid metaheuristic approaches, combining Cuckoo Search with other optimisation techniques to enhance the performance of algorithms in domain classification tasks (Yang & Deb, 2009).

The field has seen a shift towards developing reputation ontologies, with the aim of standardising and formalising the representation of domain reputation information. This evolution facilitates a better interoperability and understanding across different systems and domains. Various studies have investigated the existing ontologies in multi-agent systems, web services, and online communities, highlighting the benefits of this approach (Alnemr & Meinel, 2011).

Despite the current advancements, domain reputation systems face challenges such as evasion tactics by malicious actors, the dynamic nature of domain behaviors, and the need for a real-time analysis. Future research should focus on enhancing detection algorithms, improving the scalability of monitoring systems, and developing adaptive models that can respond to the evolving threats.

3. The TOGAF Based Architecture for TLDRep Development

TOGAF (The Open Group, 2023) provides a comprehensive framework for enterprise architecture, offering a structured methodology for the reputation system design and implementation. The proposed domain reputation platform follows the TOGAF principles, ensuring scalability, flexibility, and adaptability. The TOGAF Architecture Development Method (ADM) facilitates continuous evaluation and iterative enhancements. The key architectural components are *Business Architecture*, defining the reputation assessment workflows, *Information Systems and Data Architecture* (Figure 1), structuring domain reputation data for processing and organising system modules for an efficient processing, and *Technology Architecture*, which emphasises the appropriate technologies for system implementation.

While TOGAF provides an architectural framework for structuring the system, its role in TLDRep is limited to ensuring modularity and scalability. The TLDRep architecture follows a layered design, incorporating a data processing module, a machine learning engine, a storage and analytics component, and a user interface for security professionals and technical experts.

The platform is designed to monitor domain reputation by continuously analysing domain registration history, activity patterns, and security indicators by using Artificial Intelligence techniques. Unlike static reputation systems, it dynamically adjusts domain name scoring based on real-time cybersecurity developments. It integrates diverse data sources, including WHOIS records, DNS logs, SSL certificate status, and external security databases. An adaptive Scoring Engine may be developed in a future phase of the application development which will ensure that domains with suspicious behaviors like frequent ownership changes, DNS anomalies, or SSL certificate inconsistencies are flagged promptly.

The TLDRep platform follows a modular architecture to handle domain data efficiently. Its main components include a Data Acquisition Module that extracts domain data from various sources, a Storage Module for storing historical data, a Pre-Processing Module that sanitizes data, a Processing Module that uses machine learning to assess domain reputation, a Domain Scoring Module that applies classification techniques for reputation scoring, a Data Warehouse for complex analysis and statistics, and a Reporting Module, an interface that provides visual insights and reports to stakeholders (Dumitrache et al., 2024).

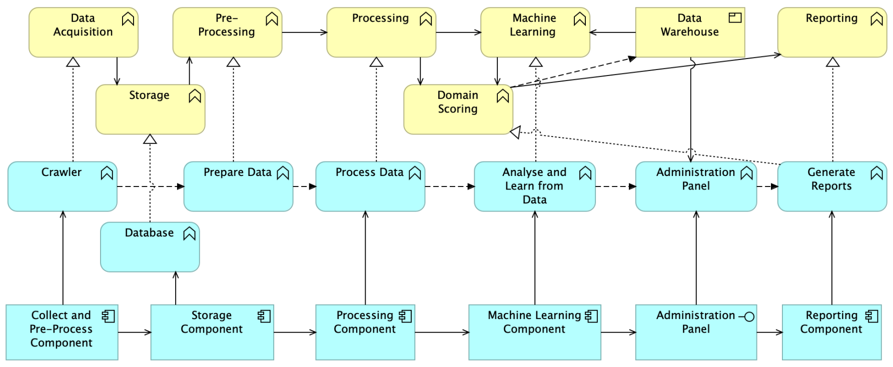


Figure 1. Information Systems and Data Architecture View (Dumitrache et al., 2024)

The TLDRep platform incorporates intelligent features to process vast amounts of security-related data, learn from patterns, and dynamically adjust to the emerging threats. It is designed to employ advanced machine learning models that continuously refine their detection capabilities based on the evolving domain activity. By integrating real-time anomaly detection, the system proactively identifies and mitigates domain-related cyber threats before they escalate. Its adaptive scoring mechanism updates domain reputation scores, ensuring that organisations receive timely and accurate security insights. The modular architecture, built on the TOGAF principles, supports the system's scalability and flexibility, allowing its seamless integration with external threat intelligence sources. The TLDRep system is not static, it evolves by incorporating new threat indicators, making it a proactive domain security tool rather than a reactive one. By leveraging data-driven insights, automated decision-making, and a continuous learning architecture, TLDRep employs intelligence in modern security monitoring.

4. Technologies and Tools Used

The TLDRep application is organised in a modular architecture that includes the following key components:

1. The frontend, developed in Django, provides a friendly and easy-to-use interface that allows the users to monitor and manage domain reputation. It includes interactive dashboards displaying reputation scores, graphs and customisable alerts. The users can also manually enter or import lists of domains for evaluation purposes and generate detailed reports;

2. The backend, developed in Django, handles all user requests, data processing and application logic. It enables machine learning algorithms` integration to analyse and evaluate domains, providing accurate and up-to-date information about their reputation. The integration with external sources, such as APIs for blacklists, whitelists and other external data sources is handled in the backend;

3. The PostgreSQL Database is used to store information about domains, including owner history, DNS activity, reputation scores and other relevant data. This database is optimised for fast queries and handles structured data to enable complex analytics and a detailed reporting;

4. The platform relies on machine learning libraries, such as scikit-learn, TensorFlow, or PyTorch, to build and train classification and anomaly detection models. The models are used to detect suspicious domains and assign reputation scores based on factors such as domain age, DNS activity, the existence of a SSL certificate, and a domain's presence on security blacklists;

5. All application components are containerised using Docker, which allows for their simple and uniform deployment across different environments (development, testing, or production). Docker Compose is used to orchestrate the containers, including the Django backend, PostgreSQL, and ancillary services, ensuring an efficient resource management and scalability. Containerisation simplifies the process of updating and maintaining the system.

The development phase focuses on transforming the conceptual model into a functional application.

5. Domain Classification and Scoring System

The scoring system aims to classify domains into the following categories: malicious, potentially malicious, normal, and trusted depending on the calculated score. A score will be calculated for each domain that will determine its classification into a particular category.

The specific ranges for each category are as follows:

- for less than -5, the domain is malicious;

- between -5 and 0 the domain is potentially malicious and interaction with it will require an increased attention;

- between 0 and 5 the domain will fall into the category of domains with a good reputation, it does not show indications that it would be used for illicit purposes, but neither does it feature the maturity or characteristics necessary to classify it as trusted;

- a domain with a score higher than 5 is a mature domain, with an increased security and it is considered trusted. Examples of trusted domains are rotld.ro, and google.ro.

To calculate the score for a certain domain, the criteria included in Table 1 are taken into account.

The positive factors include the existence of a SSL certificate with a score of +2, DNSSEC

Table 1. Domain name scoring

Positive Criteria		**Negative Criteria**	
SSL CERTIFICATE PRESENT	+2	NO SSL CERTIFICATE	-3
DNSSEC PRESENT	+5	EXTENDED STATUSES AS “POLV” (Policy Violation)	-5
OK STATUS PRESENT	+1	AGE LESS THAN 6 MONTHS AND HAS NAMESERVERS	-1
AGE LONGER THAN 2 YEARS AND LESS THAN 2 TRADES	+1	THE CONTACT HAS BEEN UPDATED OR A TRADE (TRANSFER OF THE RIGHT OF USE) HAS BEEN PERFORMED	-3
AGE LONGER THAN 5 YEARS AND LESS THAN 2 TRADES	+2	STATUS INCLUDES PENDING DELETE	-1
AGE LONGER THAN 10 YEARS AND LESS THAN 2 TRADES	+3	MORE THAN 5 (TRADES)	-1
		REDIRECTS TO OTHER DOMAINS KNOWN TO BE MALICIOUS	-3
		THE HOSTS ASSOCIATED WITH THE DOMAIN ALSO HOSTED OTHER MALICIOUS DOMAINS	-1

implementation with a score of +5, and a domain longevity greater than 2 years with a score of +1, greater than 5 years with a score of +2 and greater than 10 years with a score of +3. The negative factors include no SSL certificate with a score of -3, policy violations with a score of -5, etc.

Domains are categorised as Malicious with a score below -5, Potentially Malicious with a score between -5 and 0, Neutral with a score between 0 and 5, and Trusted with a score above 5.

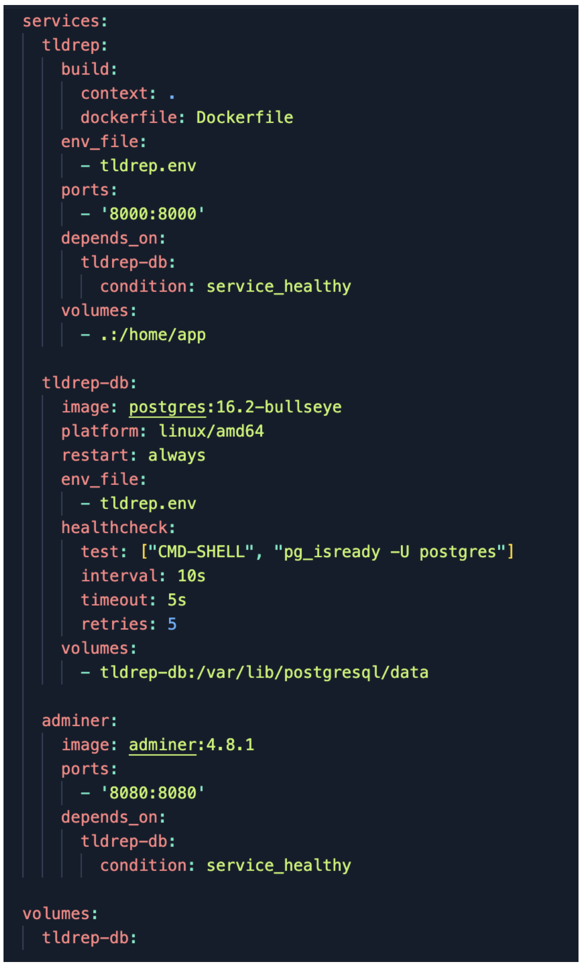
6. Domain Reputation Platform for Monitoring .ro Domain Names

6.1 Docker Configuration

Dockerfile and Docker Compose facilitate application deployment, ensuring consistent development and production environments. The use of containerised services minimises the compatibility issues and streamlines updates and maintenance.

Initially, a Dockerfile was created starting from a base image python:3.12.2-slim-bullseye. This contains a minimal version of the Debian operating system, as well as Python 3.12.2.

The services defined in the docker-compose.yml file are tldrep, tldrep-db, and adminer (Figure 2). The tldrep service starts from the image built using the Dockerfile described above, associates port 8000 in the container with port 8000 locally, and creates a volume that mounts the local directory where the docker-compose.yml file is located in /home/app on the container. This step is very important, as it ensures that changes made to the code in the container will be reflected in real time on the local machine and that these persist if the container is stopped. The credentials for authentication to the database will also be set among the system variables. The tldrep-db service starts a PostgreSQL database and creates a volume to ensure data persistence. The adminer service provides an interface to interact directly with the database.



```
services:
  tldrep:
    build:
      context: .
      dockerfile: Dockerfile
    env_file:
      - tldrep.env
    ports:
      - '8000:8000'
    depends_on:
      tldrep-db:
        condition: service_healthy
    volumes:
      - .:/home/app

  tldrep-db:
    image: postgres:16.2-bullseye
    platform: linux/amd64
    restart: always
    env_file:
      - tldrep.env
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -U postgres"]
      interval: 10s
      timeout: 5s
      retries: 5
    volumes:
      - tldrep-db:/var/lib/postgresql/data

  adminer:
    image: adminer:4.8.1
    ports:
      - '8080:8080'
    depends_on:
      tldrep-db:
        condition: service_healthy

volumes:
  tldrep-db:
```

Figure 2. TDLRep docker services

The entrypoint.sh script executes several commands. The set -e instruction will stop execution in case one of the commands gives an error. The start.sh file, responsible for starting the application, is made executable.

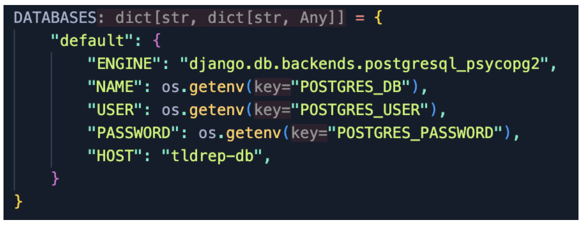
Then, a Python script is run that waits until it can connect to the database. Once the connection has been established, the migrations are run, using the manage.py file. If the database is empty, it will be populated with the domains from the domain_ input.txt list.

6.2 Django Implementation

Django's MVT (Model-View-Template) paradigm structures application logic, providing modularity and maintainability. The application integrates user authentication, a RESTful API for domain reputation queries, and an administrative dashboard for managing system configurations.

The manage.py file is created by Django after the command "django-admin startproject tldrep" is run. This command initialises the Django project with the name "tldrep" in the current directory. An application named "core" was created using **the command "python manage.py startapp core".** Within this application, the models and views have been defined.

The "core" application is added to the list of applications. Django provides modules for authentication, data management and sessions.



```
DATABASES: dict[str, dict[str, Any]] = {
    "default": {
        "ENGINE": "django.db.backends.postgresql_psycopg2",
        "NAME": os.getenv(key="POSTGRES_DB"),
        "USER": os.getenv(key="POSTGRES_USER"),
        "PASSWORD": os.getenv(key="POSTGRES_PASSWORD"),
        "HOST": "tldrep-db",
    }
}
```

Figure 3. PostgreSQL connection

The adapter used for interacting with the database is psycopg2 (Figure 3). The database and user names, as well as the password used for connection, are taken from the system variables (set in docker-compose.yml), and the application will connect to the container on which the database is running.

Django provides a session management module. File-based sessions are used, which means that the data corresponding to a user's session will be saved in a file created in the /tmp directory, and a session will expire after one hour.

6.3 Database Models

The Database model, illustrated in Figure 4 contains the following tables: Domain, for storing domain names and metadata, Evaluation for maintaining domain reputation scores, and PriorityDomain listing the domains which require immediate assessment. PostgreSQL's indexing and query optimisation ensure an efficient data retrieval.

The Domain table is meant for storing the domains whose data will be extracted for evaluation. For each domain, the fqdn ("fully qualified domain name" i.e. the full name of a domain) is stored together with its "status" and "last_processed" information. The "Status" can be "Not processed", "Processed" or "Processing" and "last_processed" contains the date of the last processing or "NULL" if it has never been processed.

The Evaluation table is developed for storing the result of an evaluation. It is necessary to store the *id* of the domain for which the evaluation has been carried out. The properties field is of the type JSON ("JavaScript Object Notation") and its role is to store the properties collected with regard to a certain domain, which will be used for evaluating it. The "Rating" field stores the grade obtained for the respective set of properties. The "Date" field will retain the date on which an evaluation was made. This field is relevant

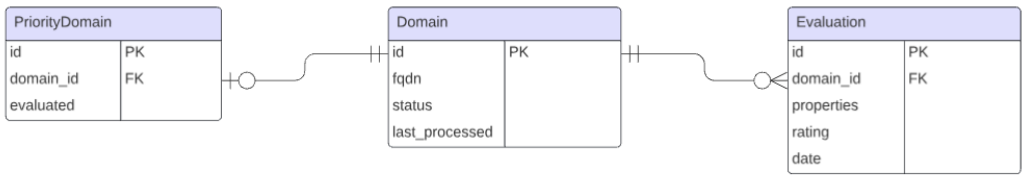


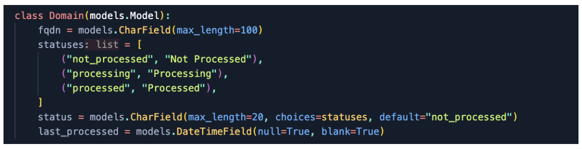
Figure 4. Database Model

because storing all evaluations together with the timestamps will allow observing the evolution of the domains over time.

PriorityDomain is a table that will allow a priority evaluation of the domains.

The Domain class (Figure 5) includes the following fields:

- *id*, the identifier of the domain;
- *fqdn*, a character string of a maximum length of 100 characters;
- *status*, a character string that can strictly take the values defined within the .ro Registry domain statuses;
- last_processed, a date field.



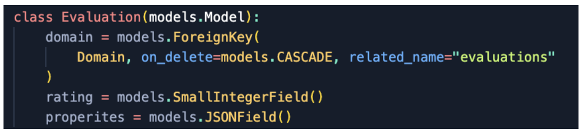
```
class Domain(models.Model):
    fqdn = models.CharField(max_length=100)
    statuses: list = [
        ("not_processed", "Not Processed"),
        ("processing", "Processing"),
        ("processed", "Processed"),
    ]
    status = models.CharField(max_length=20, choices=statuses, default="not_processed")
    last_processed = models.DateTimeField(null=True, blank=True)
```

Figure 5. Database Domain table

It is noted that the explicit definition of an id is not necessary, as Django automatically adds a primary key field. However, it is also possible to explicitly define a primary key field if primary_key=True is set.

The Evaluation class (Figure 6) comprises the following fields:

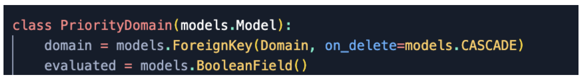
- *id*;
- *domain*, used for modeling the 1 to N relationship between Domain and Evaluation;
- *evaluated*, where the grade obtained is retained.



```
class Evaluation(models.Model):
    domain = models.ForeignKey(
        Domain, on_delete=models.CASCADE, related_name="evaluations"
    )
    rating = models.SmallIntegerField()
    properites = models.JSONField()
```

Figure 6. Database Evaluation table

PriorityDomain (Figure 7) is the table in which the domains that require a priority evaluation will be entered. *evaluated* is a field that marks whether a domain has been evaluated or not.



```
class PriorityDomain(models.Model):
    domain = models.ForeignKey(Domain, on_delete=models.CASCADE)
    evaluated = models.BooleanField()
```

Figure 7. Database PriorityDomain table

6.4 Application Interface

The user interface is developed using Bootstrap, one of the most influential and widely used front-end frameworks in web development. Bootstrap's grid system allows developers to align content in a structured manner, using rows and columns. Bootstrap facilitates the development of cross-platform applications by adapting the grid system to any device. Bootstrap is developed using a combination of essential web technologies to provide a rich functionality and responsive interfaces. The most relevant technologies include:

- HTML (HyperText Markup Language), the fundamental markup language of any website, used for structuring the content;
- CSS (Cascading Style Sheets), used for styling components or web pages. Bootstrap includes an extensive set of CSS files that provide predefined styles for various components. This allows developers to create cohesive and complex interfaces;
- JavaScript is a programming language used for adding interactivity to Bootstrap components. Vanilla JavaScript has replaced the dependency on jQuery since Bootstrap version 5 emerged to keep up with the web development standards.

Visual Studio Code (VS Code) is a source code editor developed by Microsoft, available for Windows, Linux, and macOS operating systems. VS Code is one of the most popular tools in the IT industry due to its versatility and the multitude of functionalities integrated into it.

Git is an open-source version control system, designed to manage various projects with speed and efficiency. Git tracks changes in any set of computer files, being used as a coordination report between programmers during collective software development.

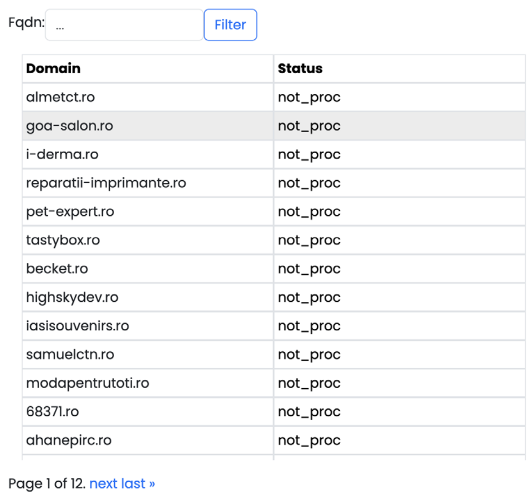
The user can register and authenticate himself in the application through the pages dedicated to the respective actions. Once authenticated, he is redirected to the main page, representing the dashboard. The user can navigate within the application using a "sidebar" (sidebar), located on the left side of each page. The page navigation options and the application logout button are within the sidebar. The "Domains" page contains the table with domains and the status of each domain, which represents its reputation level. Also, the interface includes a page displaying the relevant information about different domains.

The application works only locally, on a device with the resources necessary for commissioning. Once the application is functional, the user will be greeted, in the first phase, by the authentication page, where he will enter the user data: username and password.

The TLDRep application interface is designed to be user-friendly, visually appealing, and functional. The main components of the interface include the Dashboard, which provides an overview of reputation insights with visual charts and key performance indicators. Users can access the Search Functionality, which allows them to input specific domains and retrieve detailed reputation reports. The Report Generation feature enables users to download customised reports in various formats, providing historical data and risk assessments. The Notification System is integrated to alert users about suspicious activities, domain risk updates, and system-generated recommendations. Additional features include advanced filtering options, intuitive navigation menus, and the real-time data refresh feature for an interactive user experience.

The menu includes the "Dashboard" pages, as well as the "Domains" page. At the bottom of the page, there's a user logout button.

The "Domains" page (Figure 8) features a table with two columns that lists the domains in a database. The domain names appear in the first column, while the second column displays each domain's score (rating), indicating its level of trustworthiness. Initially, a "not_proc" status is displayed for all domains, as the process of establishing the reputation score by using Machine Learning techniques for each domain is still in the development phase. The table has a search bar that allows users to find any domain in the list, and it also includes a pagination component to handle a large volume of data, showing 100 entries per table page.



Domain	Status
almetct.ro	not_proc
goa-salon.ro	not_proc
i-derma.ro	not_proc
reparatii-imprimante.ro	not_proc
pet-expert.ro	not_proc
tastybox.ro	not_proc
becket.ro	not_proc
highskydev.ro	not_proc
iasisouvenirs.ro	not_proc
samuelctn.ro	not_proc
modapentrutoti.ro	not_proc
68371.ro	not_proc
ahanepirc.ro	not_proc

Page 1 of 12. next last »

Figure 8. Domains page

7. Security Monitoring System Evaluation and Comparison with Other Platforms

To measure the effectiveness of the proposed security monitoring system, a comparative evaluation was conducted against the existing domain reputation platforms. This section presents the security enhancements introduced by the system, along with a comparison of the key features specific to the evaluated platforms.

Several similar applications were assessed (traditional blacklists, and other similar systems such as COMAR and Cuckoo Search) as illustrated in Table 2**.** These approaches featured several limitations. The blacklists took time to reflect the newly registered malicious domains. As such, some legitimate domains were erroneously

Table 2. Comparative Analysis involving Other Reputation Systems

The Analysed Feature	Traditional Blacklists	COMAR/Cuckoo Search	The proposed TLDRep System
Real-time Scoring	No	Partial	Yes
Machine Learning-Based Detection	No	Yes	Yes (Advanced ML + Graph-Based Methods)
Anomaly Detection	No	Limited	Yes
Security Behaviour Analysis	No	Limited	Yes (Dynamic Threat Adaptation)
Modularity & Scalability	No	No	Yes (TOGAF-Based Architecture)

classified as malicious, while new harmful domains were overlooked. To that, the traditional methods lacked the ability to analyse real-time behavioural patterns.

The results of the Comparative Analysis also involving Other Reputation Systems demonstrate that the proposed platform enhances security by leveraging machine learning, real-time monitoring, and adaptive scoring, which were absent in other systems.

8. Conclusion

Before implementing the proposed platform, a series of preparatory steps was undertaken, including Requirement Analysis to identify the essential system functionalities, Technology Selection to evaluate Django, PostgreSQL, and Docker for application deployment, and Data Structuring to define the key reputation indicators and assessment criteria.

TLDRep follows a Model-View-Controller (MVC) structure, ensuring an efficient request handling and data processing. The application integrates a user authentication module, role-based access control, and API functionalities.

The domain reputation assessment process includes the following steps: data is ingested from domain sources, extraction and preprocessing are performed, the Machine learning-based component provides a reputation scoring, the visualization of insights and statistics is provided through the Reporting dashboards and the alerting system for suspicious activities is enabled.

This study proposes a practical solution for intelligent domain reputation monitoring by leveraging the TOGAF architecture and integrating advanced technologies, through which the TLDRep system ensures an accurate and scalable reputation assessment. Future work could focus on refining machine learning models and enhancing system automation in order to improve the real-time cyber threat detection.

The enhancements to the TLDRep system significantly improve domain security monitoring by addressing the shortcomings of the traditional methods. Through advanced analytics, real-time evaluation, and an adaptive ML-driven approach, this system provides a robust and intelligent mechanism for ensuring domain trustworthiness.

In the future, the research could continue by integrating Machine Learning algorithms within the TLDRep platform that will enable domain reputation assessment.

Acknowledgements

This research work was supported by a grant of the Romanian Ministry of Innovation and Research, the Nucleu Programme, project code: PN 2338 02 01, project name: “Architecture - platform of an intelligent system for monitoring Internet domains by developing a dynamic reputation establishment system (TLDRep)”.

REFERENCES

Alnemr, R. & Meinel, C. (2011) From Reputation Models and Systems to Reputation Ontologies. In: Wakeman, I., Gudes, E., Jensen, C.D. & Crampton, J. (eds.) *Trust Management V (IFIPTM 2011,* part of the book series *IFIP Advances in Information and Communication Technology*, vol 358). Berlin, Heidelberg, Springer, pp. 98-116.

Antonakakis, M., Perdisci, R., Dagon, D., Lee, W. & Feamster, N. (2010) Building a Dynamic Reputation System for DNS. In: Proceedings of the 19th USENIX Security Symposium, SEC`10, 11-13 August 2010, Washington, DC, USA. Berkeley, CA, USA, USENIX Association. pp. 273-290.

Banciu, D., Petre, I. & Dumitrache M. (2019) Electronic system for assessing and analysing digital competences in the context of Knowledge Society. In: *The 11th International Conference on Electronics, Computers and Artificial Intelligence (ECAI 2019), 7-29 June 2019, Pitesti, Romania*. https://doi.org/10.1109/ECAI46879.2019.9042151.

Bilge, L., Kirda, E., Krügel, C. & Balduzzi, M. (2011) *EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis*. [Lecture] The Dana on Mission Bay, San Diego, CA, USA, 7th February.

Deri, L. & Fusco, F. (2023) Evaluating IP Blacklists Effectiveness. https://arxiv.org/abs/2308.08356 [Accessed: 3rd February 2025].

Dumitrache, M., Rotună, C.I., Gheorghiță, A., Vevera, A.V., Sandu, I. & Smada, D. (2024) A Domain Reputation System Architecture Description Using TOGAF. *Studies in Informatics and Control*. 33(1), 61-72. https://doi.org/10.24846/v33i1y202406.

Envescent Cybersecurity. (2024) *The Value of DNS Blacklisting*. https://envescent.com/insights/the-value-of-dns-blacklisting/ [Accessed: 21st March 2025].

Galloway, T., Karakolios, K., Ma, Z., Perdisci, R., Keromytis, A. & Antonakakis, M. (2024) Practical Attacks Against DNS Reputation Systems. In: *2022 IEEE Symposium on Security and Privacy (SP), 19-23 May 2024, San Francisco, CA, USA*. Piscataway, New Jersey, USA, IEEE. pp. 4516–4534.

InstaSafe Inc. (2024) *Whitelisting vs Blacklisting: What's the Difference?* https://instasafe.com/blog/whitelisting-vs-blacklisting-whats-the-difference/#:~:text=Whitelisting%20only%20allows%20access%20to,unknown%2Funlisted%20entities%20by%20default./ [Accessed: 3rd February 2025].

Maroofi, S., Korczyński, M., Hesselman, C., Ampeau, B. & Duda, A. (2020) COMAR: Classification of Compromised versus Maliciously Registered Domains. In: *2020 IEEE European Symposium on Security and Privacy (EuroS&P), 7-11 September 2020, Genoa, Italy*. Piscataway, New Jersey, USA, IEEE. pp. 607-623.

Mishsky, I., Gal-Oz, N. & Gudes, E. (2014) Computing domains reputation using flow. In: *The 9th International Conference for Internet Technology and Secured Transactions (ICITST-2014), 8-10 December 2014, London, UK*. Piscataway, New Jersey, USA, IEEE. pp. 426–431.

Peng, C., Yun, X., Zhang, Y. & Li, S. (2019) MalShoot: Shooting Malicious Domains Through Graph Embedding on Passive DNS Data. In: Gao, H., Wang, X., Yin, Y. & Iqbal, M. (eds.) *Collaborative Computing: Networking, Applications and Worksharing* (*Proceedings of the 14th EAI International Conference, CollaborateCom 2018, 1-3 December 2018, Shanghai, China,* part of the book series *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, vol 268). Cham, Switzerland, Springer, pp. 488–503. https://doi.org/10.1007/978-3-030-12981-1_34.

Rotună C.I., Gheorghiță, A., Sandu, I., Dumitrache, M., Udroiu, M. & Smada, D. (2023) A Generic Architecture for Building a Domain Name Reputation System. *Studies in Informatics and Control*. 32(2), 39-49. https://doi.org/10.24846/v32i2y202304.

Sarkar, M., Banerjee, S. and Hassanieen, A.E., 2013. Searching DNS for malicious domain registration: Identification through hybrid cuckoo search metaphor and object–oriented implementation. *International Journal of Reasoning-based Intelligent Systems*. 5(4), 280-289. https://doi.org/10.1504/IJRIS.2013.058773.

Smada, D., Dumitrache, M., Rotună, C. I. & Gheorghiță, A. (2024) The impact of internet domain name status on their reputation. *Romanian Journal of Information Technology and Automatic Control*. 34(1), 31-44. https://doi.org/10.33436/v34i1y202404.

The Open Group (2023) *An Introduction to the TOGAF® Standard, 10th Edition*. https://pubs.opengroup.org/togaf-standard [Accessed: 3rd February 2025].

Vevera, A.-V., Cîrnu, C.E. & Rădulescu, C.Z. (2022) O abordare multi-criterială pentru calculul unui indicator complex de Securitate Cibernetică și Dezvoltare Digitală (A multi-criteria approach for the calculation of a complex indicator of Cyber Security and Digital Development). *Romanian Journal of Information Technology and Automatic Control*. 32(4), 19–32. https://doi.org/10.33436/v32i4y202202.

Vinayakumar, R., Soman, K. P. & Poornachandran, P. (2018) Detecting malicious domain names using deep learning approaches at scale. *Journal of Intelligent & Fuzzy Systems*. 34(3), 1355-1367. https://doi.org/10.3233/JIFS-169431.

Yang, X.-S. & Deb, S. (2009) Cuckoo Search via Lévy flights. In: *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), 9-11 December 2009, Coimbatore, India*. Piscataway, New Jersey, USA, IEEE. pp. 210-214.