International Journal of Emerging Multidisciplinary Research And Innovation (IJEMRI)

A Web-Based Threat Intelligence Platform for URL Risk Analysis

[1]T C Swetha Priya, [2]A Keerthana, [3]Karbuje Sruthika, [4]Palle Chandana

[1]Assistant Professor Department of Information Technology Stanley College of Engineering and Technology for Women.tcswethapriya@stanley.edu.in
[2]Student Department of Information Technology Stanley College of Engineering and Technology for Women aenugukeerthanareddy@g mail.com
[3]Student Department of Information Technology Stanley College of Engineering and Technology for Women karbujesruthika@gmail.com
[4]Student Department of Information Technology Stanley College of Engineering and Technology for Women pallechandana20@gmail.co m

ABSTRACT

The rapid growth of cybercrime activities on the darkweb has brought organizations, researchers and cyber security professionals into real troubles.These hidden market places, credential leak repositories, money fraud hubs and anonymous communication channels often run through technologies like The Onion Router (TOR) so pinning down threats and keeping continuous monitoring becomes hard. What comes out of these covert networks is often missed by standard security monitoring tools, largely due to limited access and the chaotic nature of threat intelligence collection in real-world situations. This paper introduces a DarkWeb Monitoring Simulation platform, a web based threat intelligence concept that can analyze suspicious URLs and then create risk assessments. The proposed system combines active web scanning with passive heuristic analysis techniques to discover indicators of phishing, financial fraud, credential theft, darknet marketplaces and other malicious activities. This application is built on Python Flask and SQLite and it has user authentication, OTP verification, scan history management, and an interactive dashboard for threat visualization.In experiments, the system can sort URLs into separate tiers of risk using a weighted scoring method, and the results show this. Essentially, the platform developed is a less expensive educational alternative to commercial threat intelligence solutions and still enables practical experience with cybersecurity monitoring concepts.

Keywords: *Dark Web Monitoring, Threat Intelligence, URL Analysis, Flask, SQLite, Cybersecurity Simulation, OTP Authentication, Web Scraping, Risk Scoring, Onion Network.*

DOI: https://doi.org/10.65180/ijemri.2026.2.3.03

Introduction

The internet is usually split into three parts, like the surface web , the deep web, and the dark web. The surface web is where websites are publicly reachable, and they get indexed by search engines so people can find them pretty easily. On the other hand, the dark web is made of concealed services that are hard to reach, and they typically need specialized software like TOR to enter [1]. Even

though these anonymous networks were originally made so privacy and free communication stay such as credential theft, ransomware distribution, illicit marketplaces, and various financial fraud schemes [2]. In response, organizations and security specialists use cyber threat intelligence solutions to track threats that can come out of these hidden environments and to stop cyberattacks before they fully land [8]. Still, a lot of commercial dark web monitoring products are costly, and they also demand specialized infrastructure, which ends up restricting who can really use them, especially for academic or research work [2]. That's why the proposed DarkWeb Monitoring Simulation platform was created, to offer a learning-oriented setting where students and researchers can understand threat intelligence workflows, URL analysis, and the cyber threat detection logic behind it all [1].

Dark web monitoring is now a key topic in cybersecurity research, mainly because criminals often rely on secluded online communities to swap stolen credentials, push malware into circulation, run financial scams, and even coordinate cyberattacks [3]. And because the amount of cybercrime keeps growing, there's a clearer need for automated systems that can spot odd or suspicious patterns and then produce something useful, like actionable threat intelligence [4]. Educational institutions often don't quite have access to enterprise-level threat intelligence platforms due to costs and also because their infrastructure is a bit limited. So, there is a rising need for learning tools that show how professional threat monitoring actually works, kind of in real operational terms, but still stay simple, transparent, and affordable [2].

Literature survey

protected, over time they've turned into places where cybercriminals do more and more things, Lately, a number of studies have shown how artificial intelligence, machine learning, and threat intelligence methods can work effectively for dark web monitoring. Jin, for example, introduced DarkBERT, which is a transformer-based language model made especially for dark web text, and it reports better classification results than older or more traditional language models [1]. Alquwayzani also looked at Open Source Intelligence (OSINT) frameworks together with machine learning, aiming to gather and make sense of cyber threat intelligence coming from concealed online communities. It is a pretty targeted workflow [2]. In a broader way, Madouh carried out a study of dark web ecosystems and described how resilient and intricate underground cybercriminal communities can be, not at all simple or linear [3].

Whitman proposed adversarial machine learning strategies, which are meant to strengthen cyber threat detection systems and also help them stay robust when evasion attacks try to bypass the models [4]. Rekowski investigated how healthcare cybersecurity incidents connect to dark web activity, and he pointed out that credential theft, plus data breaches, are increasingly influencing critical sectors [5]. Rawat and colleagues built predictive models to anticipate dark web events tied to organized cybercrime and illicit product distribution, basically trying to get ahead of the pattern [6]. And then, Sotirios and his co-researchers used social network analysis techniques to trace the links between threat actors that work inside hidden communities; the interconnections matter a lot [7]. Fernández put together a cyber threat intelligence framework for digging into leaks and the cybercriminal crews that work on the dark

web [8]. Rajawat suggested neural-network-based classification methods for handling huge dark web talked about how generative artificial intelligence is playing a bigger part in cybersecurity and what it might enable in threat intelligence systems [10]. Overall, these works sort of agree that automated monitoring, machine learning, threat intelligence, and behavioral analysis really do form core pieces of today's cybersecurity operations.

For the proposed system to work, it needs a medium-level hardware and software setup, you know enough to do web application development along with cybersecurity analysis tasks. In general, the platform can run on a personal computer as long as it has a recent processor, enough memory, and stable internet connectivity. Python was picked as the main programming language, mainly because it's simple, it has broad library support, and it fits well with cybersecurity use cases. The web application itself was built with Flask , since this framework is lightweight and pretty flexible for dealing with user requests and those routing operations. For storing data, SQLite was selected because it offers dependable storage and still keeps things simple and easy to deploy. On top of that, HTML, CSS, and JavaScript were also incorporated to shape the interface and improve user interaction throughout the platform.

Existing system

Existing dark web monitoring systems are basically built to pull information from hidden forums , underground marketplaces, breach repositories, and also anonymous messaging channels so they can spot malicious activity and other security threats. Typically, these setups use web scraping, machine learning algorithms, threat intelligence feeds, and natural language processing methods to collect and datasets and doing automated threat labeling, basically to help the process scale up [9]. Curran then reason about what's going on inside these obscured online spaces [2]. There are commercial options like Recorded Future, Flashpoint, and DarkOwl that often cover things such as credential leak detection, ransomware monitoring, malware tracking, and cybercrime intelligence gathering [8]. Even though these services can be quite useful , they are usually tied to expensive subscription fees, and they may need specialized infrastructure to run , deploy, and keep steady over time. On top of that, a lot of commercial tools rely on proprietary threat scoring mechanisms, so for students and researchers, it becomes hard to follow the actual reasoning steps, or at least to interpret them clearly [4]. All these constraints lower accessibility and make it harder for educational institutions to get hands-on experience with cyber threat intelligence ideas.

Also, a lot of existing systems need access to the TOR network to monitor onion services and hidden marketplaces. That kind of dependence raises deployment complexity, and it can bring legal plus operational worries, especially in academic settings [15]. Many platforms don't show clear educational workflows, and they provide limited room for experimentation. So in general, there is a clear need for a lighter , more transparent educational platform, one that can demonstrate the main principles of dark web monitoring without forcing dedicated infrastructure or demanding access to real dark web networks.

Proposed system

The proposed DarkWeb Monitoring Simulation platform is meant to be a web-based threat

intelligence system that helps people analyze suspicious URLs and then get a more detailed, risk-web-based threat intelligence system that helps people analyze suspicious URLs and then get a more detailed, risk-oriented assessment. It uses active scanning procedures, passive heuristic analysis, and identify phishing attempts, credential theft, financial fraud, darknet markets, and illegal content distribution [1]. Various checks are made after entering the link including the content check, domain check, keywords, metadata, and more. A score is given according to the weighted list of threat indicators and the risk category is assigned [4].Compared to the options available on the market described in the previous section, the proposed design seems more transparent and educational foritspotential users. Namely, the inner workings of detections are explained at greater detail, thus providing a clearer perspective of the whole CIC process involved. Furthermore, it is stated that the proposed software will include a comprehensive set of the most common account management features, such as (but not limited to) authentication, OTP email confirmation, account recovery, scan history, and dashboards [11], [14]. Indeed, such an option would serve as an affordable dark web monitoring service, which would not require users to be technology experts in any capacity in order to use its most basic functions.

System architecture

oriented assessment. The proposed DarkWeb Monitoring Simulation platform is meant to be a The DarkWeb Monitoring Simulation tool is implemented using the Model View Controller (MVC) pattern and Python programming language. As demonstrated in the Figure below, the application consists of three major layers or tiers including the Application layer, Presentation layer, and Data layer. The top layer of the MVC is known as the Presentation layer, which is horizontally aligned. This layer was built using HTML, CSS, and JavaScript in a Flask framework. An example of a cybersecurity dashboard interface is given below. It demonstrates the features on the dashboard, including a field for entering the target URL, history scan field, and other sections, for example, by clicking in the History Scan box will open a new window with an overview of previous scans. The second layer is the Application layer, which is also referred to as the Business Logic layer and is located in the middle of the diagram. That is, the Application layer essentially drives the model view controller structure, which means that it holds the business logic required for the application. This layer requires several procedures, functions, and operations to carry out processes such as scanning and reporting on a database. Finally, the lowest layer is the Data layer, which is responsible for ensuring data is banked and retrieved properly from a database, according to the MVC structure.

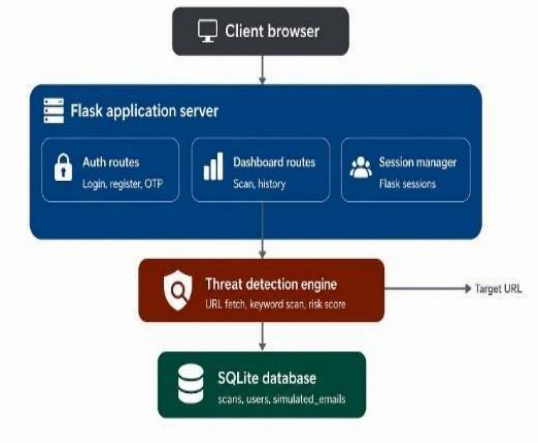

Fig 1: Overall System Architecture Diagram

Figure below represents the system architecture, and the user interface, application server, threat detection engine, and database components are connected. The focus is placed on the data flow within the context of the system. Then the Data Layer was designed with an SQLite backend database manager as the storage media for the user data, authentication table, scans, reports, and simulated email messages [13]. It was implemented in a modular, easily maintained, and scalable manner, while also designing it in a communications-friendly way for the components mentioned.

VI. Methodology

The proposed method is based on the hybrid threat detection model, which combines Active Scanning with Passive Heuristic Analysis. First, within the scope of Active Scanning, the submitted URL is normalized and validated before any active scanning techniques are applied, as an HTTP request is initiated by Python's urllib library [12]. Then, the page content is extracted, with its titles, metadata, visible data, and server headers included. HTML pages are processed by various parsing means to remove all irrelevant data, such as scripts or cascading style sheets, thus extracting information that allows generating meaningful text snippets. Next, in parallel, the detected content is compared to various threat groups, including Drugs and Controlled Substances, Data Breaches and Credentials, Financial Fraud and Carding, and Darknet Forums and Markets [6], within the scope of the risk classification algorithm that produces a score varying from 0 (Low Risk) to 100. In particular, for each URL, a risk category is selected according to the score: Low Risk (0-39), Medium Risk (40-74), or Critical Risk (75-100) [6]. As a result, users can estimate the severity level of each detected threat quickly and efficiently, thus facilitating the threat classification process.On top of that, the system examines the webpage for obfuscation behaviors, cryptojacking scripts, and odd code arrangements, which could suggest a malicious purpose.

Every threat it spots adds a weight to the overall risk score, more or less. If the active side doesn't work because of connection issues , unreachable domains, or onion addresses, the platform flips over by itself to Passive Heuristic Analysis. In that second phase , it looks at suspicious top-level domains, phishing-oriented keywords, homograph attack shapes, matches against breach databases, and also onion routing clues [2].

Finally, results from both rounds are merged using a weighted scoring method , and URLs end up labeled as Low Risk, Medium Risk, or Critical Risk [4].

The proposed DarkWeb Monitoring Simulation platform is kind of split into several functional modules that cooperate to do threat detection and risk assessment, in a sort of connected way. The Authentication Module handles things like user registration, login, OTP verification, and password recovery too. To keep user credentials safe, secure password hashing techniques are applied, which should ensure access is stable and protected [14]. Then the URL Analysis Module is basically the core piece of the whole system, where it does active scanning along with passive heuristic analysis on whatever URLs are submitted. It looks at webpage content, questionable terms, domain attributes, and various threat indicators to come up with a security assessment [8]. After that, the Risk Scoring Module takes the identified threat vectors and produces a weighted score, which is used to set the severity level tied to that specific URL [4]. Another small suggestion would be to equip the software with a scan history module, so it can store details from the earlier scans. In that way the user can follow the scan outcomes and also notice how the identified threats end up evolving over time. Thinking about these proposed modules, and yeah the benefits properly considered, it feels like designing a more efficient solution for cybersecurity monitoring and analysis is really possible.

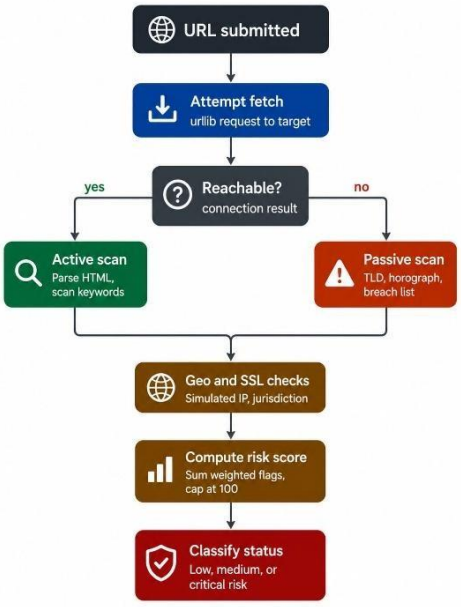

Fig 2: Threat Detection Engine Flow

The figure showcases the framework of a Flask-based system crafted for spotting phishing URLs, detailing the flow of data between its main components.

Overall, the proposed system runs in a kind of step-by-step rhythm, starting at user authentication and finishing at threat report generation. First, the user goes to the platform and goes through authentication, either via registration or login, and yeah, that part matters. After the user is authenticated, the user submits a URL for review. Then the system carries out

active scanning by pulling webpage content and comparing it with established threat indicators. If for any reason active scanning cannot be completed, the system shifts to passive heuristic analysis, aiming to surface questionable attributes tied to the URL. Those gathered indicators are routed into the risk scoring engine, where a numeric risk value is produced, along with the matching threat classification. After that, the whole outcome is shown on the dashboard and also stored in the scan history database so it can be reused later.

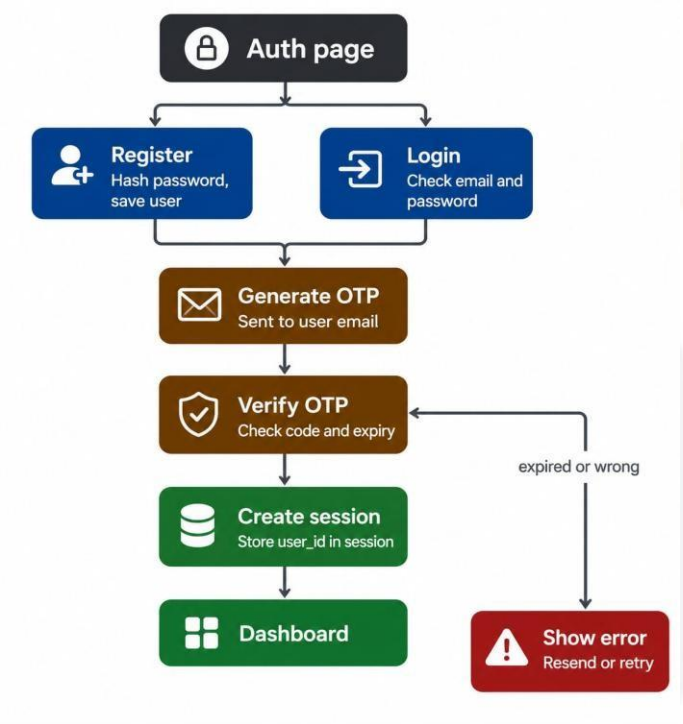

Fig 3: User Authentication Flow Diagram

VII. Results and discussion

The tool was tested using different URLs including phishing, suspicious, credential theft, and onion sites. The tool managed to differentiate between the safe and unsafe URLs by identifying the risks associated with each site. URLs containing phishing terms, suspicious domains, and references to the dark web were reported with high risk levels, whereas tools with onion extension were majorly flagged as critical. The tool also had an effective authentication system that only allowed users with OTP access to the dashboard where all the scan results, risk levels, and threat intelligence were reported. In most cases, the active scan took a few seconds to complete, whereas the passive heuristic scan was instant. Therefore, the tool is effective since it can be used to scan threats and assess the risk level of a particular URL. The figure below shows the sign-up page for Dark Web Monitoring Simulation. The page collects personal and login details from the user before they are granted access to the dashboard.

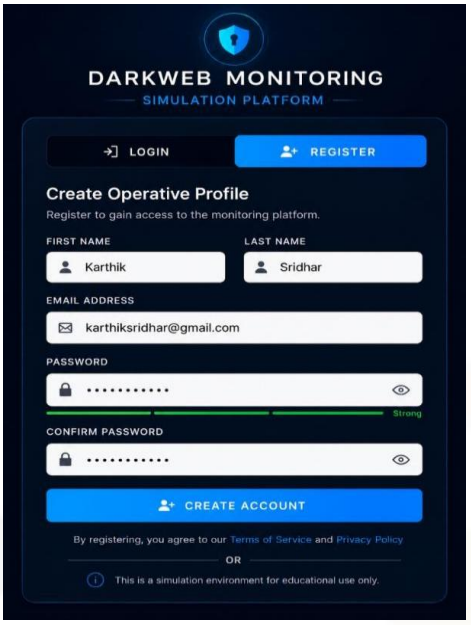

Fig 4: Authentication Portal

The figure above presents the registration window which is the first step before setting up an account on the Dark Web Monitoring Simulation portal. On this page, users can provide personal information and set their login details.

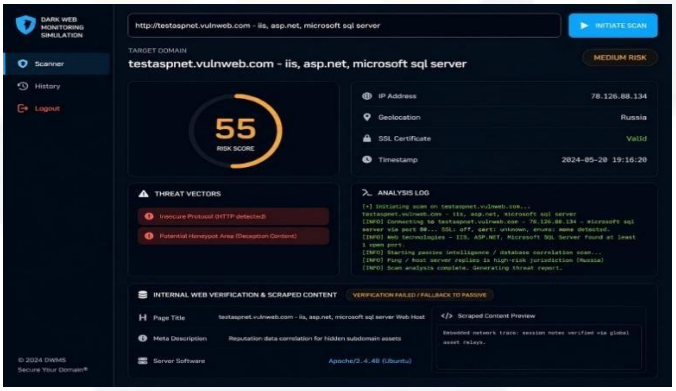

Fig 5: Risk Gauge - Critical Risk URL Sample

The picture below represents the dashboard of the Dark Web Monitoring Simulation. By means of this dashboard, users can analyze the results of domain scans, assess the risk score, examine the threat analysis and check the security verification data.

VIII. Conclusion & future scope

By examining the results of the conducted simulation of the dark web monitoring, it is possible to determine the benefits of the discussed threat intelligence, based on the obtained results. Primarily, one should mention that the software under the consideration was capable of quickly scanning the suspicious links and providing the user with the appropriate security reports. Secondly, the application was able to protect the user from potential threats through the links rating, the identification of users, and storing the obtained data in databases.

Altogether, the described opportunities allow for supposing that the usage of the discussed application could help in determining the actual state of the security and detecting the risky or safe elements on web pages. Hence, it is possible to state that the examined software possesses a number of advantages and opportunities that could be beneficial for the user in terms of cybersecurity.

Nevertheless, it should be noted that the considered project possesses a number of opportunities that will ensure the application's competitiveness and its ability to attract users' attention. In the first place, it should be mentioned that the software under consideration could be improved through the introduction of the artificial intelligence technologies that allow for getting better results in terms of detecting threats. Secondly, it would be interesting to see how the examined application would perform while using the opportunity of real-time monitoring and alerts. Lastly, from the perspective of cloud computing, the addition of such features as Docker, multiple users, report generation, and many others could become an attraction for the users.

References

1. Jin, Y. (2023). DarkBERT: A Language Model for the Dark Side of the Internet. Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL2023).Association for Computational Linguistics.
2. Alquwayzani,A.(2023).How Dark Web Monitoring Can Be Used for OSINT and Investigations. International Journal ofCybersecurity Intelligence& Cybercrime,6(2), 18–35.
3. Madouh,A.(2025).The Dark Web Ecology:ANetno graphic Studyof HiddenCommunitiesontheDark Web. Journal of Cyberspace Studies, 9(1),44–67.
4. Whitman,J.(2024).Adversarial Machine Learning in Dark Web Threat Detection.IEEE Transactions on Information Forensics and Security, 19, 1120–1135.
5. Rekowski,J.R.(2025).Cyber Attack sand the Dark Web: Primerfor Physicians. Journal of the American Medical Informatics Association (JAMIA), 32(3),210–218.
6. Rawat,R.,Mahor,V.,Chirgaiya,S.,&Garg,B.(2023).T echniquesfor Predicting Dark Web Events Focusedonthe DeliveryofIllicit Productsand Ordered Crime. International Journal of Information Security and Privacy, 17(1),1–22.
7. Sotirios, N., Papadopoulos, S.,& Kompatsiaris, Y.(2022). Employing Social Network Analysis on Dark Web Communities—IPASS System.Computers & Security, 118, 118,102729.
8. Fernández,C.A.(2024).CyberThreatIntelligenceToolf orLeaksandCybercriminalGroupInvestigation in the Dark Web. FutureInternet, 16(4),112. https://doi.org/10.3390/fi16040112
9. Rajawat,A.S.,Rawat,R.,Barhanpurkar,K.,Shaw,R.N., &Ghosh,A.(2022).DarkWebData Classification Using Neural Network.In Advanced Computing and Intelligent Technologies (Lecture Notes in Networks and Systems).Springer.