frontiers | Frontiers in Blockchain

TYPE Original Research
PUBLISHED 12 December 2024
DOI 10.3389/fbloc.2024.1484894

OPEN ACCESS

EDITED BY
Alex Zarifis,
University of Southampton, United Kingdom

REVIEWED BY
Qasem Abu Al-Haija,
Jordan University of Science and Technology, Jordan
Abdur Rasool,
Chinese Academy of Sciences (CAS), China

*CORRESPONDENCE
Fouad Trad,
fat10@mail.aub.edu

RECEIVED 22 August 2024
ACCEPTED 28 November 2024
PUBLISHED 12 December 2024

CITATION
Trad F, Semaan-Nasr E and Chehab A (2024) MLPhishChain: a machine learning-based blockchain framework for reducing phishing threats.
Front. Blockchain 7:1484894.
doi: 10.3389/fbloc.2024.1484894

MLPhishChain: a machine learning-based blockchain framework for reducing phishing threats

Fouad Trad[1]*, Elie Semaan-Nasr[1,2] and Ali Chehab[1]

[1]Electrical and Computer Engineering Department, American University of Beirut, Beirut, Lebanon, [2]Computer Science Department, American University of Science and Technology, Beirut, Lebanon

Introduction: Phishing attacks pose a significant threat to online security by deceiving users into divulging sensitive information through fraudulent websites. Traditional anti-phishing approaches are centralized and reactive, exhibiting critical limitations such as delayed detection, poor adaptability to evolving threats, susceptibility to data tampering, and lack of transparency.

Methods: This paper presents MLPhishChain, a decentralized application (DApp) that integrates blockchain technology with machine learning (ML) to provide a proactive and transparent solution for URL verification. Users can submit URLs for automated phishing analysis via an ML model, with each URL's status securely recorded on an immutable blockchain ledger. To address the dynamic nature of phishing threats, MLPhishChain features a re-evaluation mechanism, enabling users to request updated assessments as URLs and website content evolve. Additionally, the system incorporates data from external security services (e.g., VirusTotal) to offer a multi-source validation of phishing risk, enhancing user confidence and decision-making.

Results: The system was built using Ganache and Truffle, and performance metrics were computed to evaluate its efficacy in terms of latency, scalability, and resource consumption. Results indicate that the proposed system achieves rapid URL verification with low latency, scales effectively to handle increasing user submissions, and optimizes resource usage.

Discussion: By leveraging the strengths of decentralized blockchain technology and intelligent ML algorithms, MLPhishChain addresses the shortcomings of traditional anti-phishing methods. It delivers a reliable and adaptable solution capable of addressing the evolving nature of phishing threats. This approach establishes a new standard in phishing detection, characterized by enhanced transparency, resilience, and adaptability.

KEYWORDS

blockchain, decentralized application (Dapp), machine learning, phishing URL detection, URL Re-evaluation

1 Introduction

In the digital age, the widespread use of online services for personal and financial transactions has brought convenience but also increased exposure to sophisticated cyber threats, particularly phishing attacks Aleroud and Zhou (2017). Phishing attacks deceive users by impersonating legitimate institutions to gain access to sensitive information, leading to financial losses, identity theft, and compromised data security Vijayalakshmi et al. (2020).

Traditional phishing defenses typically rely on centralized systems, which have significant limitations in adapting to the rapidly evolving tactics of cybercriminals Sun et al. (2022). These systems often suffer from delayed detection as updating databases struggles to keep pace with the constant emergence of new phishing sites Khonji et al. (2013). Additionally, centralized systems are vulnerable to breaches and manipulation, creating single points of failure where attackers can alter or delete critical data Oest et al. (2019); Apruzzese et al. (2022). Their opacity also reduces transparency, leaving users uninformed about how URLs are classified, which undermines trust in these systems' ability to accurately differentiate between legitimate and malicious websites Vidyakeerthi et al. (2022).

To address these challenges, this paper introduces MLPhishChain, a decentralized application (DApp) that combines blockchain's immutable ledger technology with machine learning (ML) to create a more proactive, transparent, and reliable solution for URL verification. MLPhishChain enables users to submit URLs for phishing risk assessment, using an ML model to evaluate each URL and record its status on an immutable blockchain ledger. This decentralized approach mitigates traditional vulnerabilities by ensuring that once a URL's risk status is recorded, it becomes tamper-proof and verifiable by anyone. The immutability of blockchain guards against unauthorized alterations, ensuring data integrity and fostering transparency that centralized systems cannot match.

MLPhishChain also introduces a re-evaluation mechanism to ensure that URL classifications remain accurate and relevant as websites evolve. Users can request a re-assessment of URLs, allowing MLPhishChain to perform fresh analyses and detect any changes in phishing risk. If a URL's status changes, the system flags it for further review, maintaining the balance between the blockchain's immutability and the need for up-to-date information. This feature ensures that users have access to reliable and current information on URL safety.

In addition, MLPhishChain integrates external evaluations through sources such as VirusTotal, providing a secondary layer of verification. This feature serves as a "second opinion," leveraging the extensive databases of established cybersecurity services to further validate URL safety Peng et al. (2019); Salem et al. (2021). By cross-referencing URLs with such services, MLPhishChain enhances the robustness of its phishing detection capabilities.

Finally, MLPhishChain leverages the distributed nature of blockchain to ensure system resilience. Unlike centralized systems susceptible to single points of failure, MLPhishChain's decentralized infrastructure enables continued operation even if parts of the network are compromised or offline. Through these combined features, MLPhishChain establishes a novel, decentralized approach to phishing detection that is both adaptable to changing threats and resilient against potential attacks.

In summary, the main contributions of this paper are:

- Integration of Blockchain and ML for phishing detection: Introducing MLPhishChain, a novel DApp that combines blockchain technology and ML for enhanced URL phishing detection.
- Adaptive re-evaluation of URLs: Introducing a mechanism that re-evaluates URLs to keep records up-to-date in response to website changes, which, to the best of our knowledge, no previous works have addressed.
- Incorporation of external sources: The Addition of an external evaluation feature, provides users with a second opinion and enhances the robustness and reliability of the phishing detection system.

The rest of the paper is organized as follows: Section 2 covers background concepts, including blockchain, machine learning fundamentals, and current phishing detection methods. Section 3 reviews related work, discussing the limitations of traditional systems. Section 4 details the MLPhishChain architecture, focusing on integrating ML with blockchain for improved security. Section 5 explains the technical simulation and testing of the proof-of-concept, from URL querying to blockchain recording and re-evaluation. Section 6 describes system validation, and Section 7 discusses system limitations. Section 8 concludes the study, and Section 9 outlines future directions for enhancing MLPhishChain.

2 Background and preliminaries

2.1 Blockchain technology

Blockchain is a decentralized digital ledger that records transactions securely and transparently across a network of nodes. Originally developed for Bitcoin Vranken (2017), blockchain applications now span finance Zhang et al. (2020); Varma (2019), supply chain management Moosavi et al. (2021); Cole et al. (2019); Queiroz et al. (2020), cybersecurity Maleh et al. (2020); Hasanova et al. (2019); Demirkan et al. (2020), and healthcare Agbo et al. (2019); Attaran (2022); Hölbl et al. (2018). Blockchain's decentralized nature eliminates the need for a central authority, improving resilience to tampering and fraud. Each block, cryptographically sealed and chronologically linked, creates a secure, immutable chain of records, verifiable by all network participants. Blockchain's decentralized and tamper-resistant structure not only secures transactions but also supports smart contracts—self-executing agreements embedded in code Chen J. et al. (2020)—and decentralized applications (DApps) that operate without a central authority.

2.1.1 Blockchain protocols

Various blockchain protocols offer distinct capabilities. For example, Ethereum is a decentralized platform enabling DApp development via smart contracts Aljofey et al. (2022). Unlike Bitcoin's peer-to-peer transaction focus, Ethereum supports

diverse applications Buterin (2014); Tikhomirov (2018). Hyperledger Fabric, by contrast, is an enterprise-grade platform for private, permissioned networks. This structure allows organizations to tightly control their blockchain environments, making it well-suited for applications demanding high confidentiality and performance, such as in supply chain management and healthcare Androulaki et al. (2018).

2.1.2 Ganache and Truffle

Ganache is a personal blockchain for Ethereum development, simulating a local blockchain network to facilitate the testing and development of smart contracts. It allows developers to control block mining, log events, and customize blockchain behavior. The Truffle framework complements Ganache by offering a development environment and testing tools for Ethereum, enabling automated smart contract compilation, deployment, and testing. Together, Ganache and Truffle streamline smart contract development.

2.1.3 Wallets

Digital wallets store cryptographic keys and sign transactions, enabling secure blockchain interactions. Wallets can be software- or hardware-based Thota et al. (2020); Khan et al. (2019). MetaMask, a widely-used Ethereum wallet, allows users to manage keys, interact with DApps, and sign transactions Lee and Lee (2019).

In MLPhishChain, blockchain's immutability enhances phishing detection by recording URL classifications securely, making them tamper-proof and verifiable. Through Ethereum, MLPhishChain uses smart contracts to automate URL re-evaluation, ensuring data remains current. Users interact with the MLPhishChain DApp via wallets like MetaMask to submit URLs and view results.

2.2 Machine learning

Machine learning (ML), a subset of artificial intelligence (AI), focuses on developing systems that learn from data and make decisions with minimal human input. ML algorithms improve as they process more data, creating predictive models for applications in language processing, fraud detection, and recommendation systems Studer et al. (2021). In cybersecurity, ML adapts to evolving threats, providing enhanced anomaly detection and threat prediction compared to rule-based methods Dasgupta et al. (2022). In MLPhishChain, ML is combined with blockchain to create a resilient phishing detection system, leveraging ML's adaptability and blockchain's transparency.

2.3 Phishing detection

Phishing detection aims to identify deceptive attempts to acquire sensitive information by impersonating legitimate entities Varshney et al. (2016). Traditional phishing databases use signature- and heuristic-based methods that compare URLs or messages to known phishing patterns. While effective for previously identified threats, these approaches struggle with rapidly evolving phishing tactics Alzahrani and Ghorbani (2015). To address these limitations, modern phishing systems incorporate ML techniques, such as traditional ML and data mining algorithms Al-Haija and Al Badawi (2021); Jibat et al. (2023); Odeh et al. (2023); Al-Fayoumi et al. (2024); Rao et al. (2020), deep learning models Do et al. (2022); Catal et al. (2022); Yang et al. (2019); Aslam et al. (2024), and more recently, large language models Trad and Chehab (2024a), Trad and Chehab (2024b), Trad and Chehab (2024c), to enhance detection capabilities. These ML models, trained on extensive datasets, can identify new phishing strategies and analyze URLs and websites in real-time, improving the adaptability to emerging threats Shahrivari et al. (2020). However, even with ML integration, phishing databases still face challenges like delayed updates and limited transparency. In this work, MLPhishChain utilizes an ML model to analyze URLs and records their status on the blockchain, ensuring secure, transparent, and immutable evaluations, and addressing key limitations of traditional phishing detection systems.

3 Related work

Crowdsourced phishing blacklists are instrumental in defending against phishing attacks Bell and Komisarczuk (2020). Among these, PhishTank stands out as one of the most longstanding and widely utilized platform, hosting a substantial repository of phishing URLs PhishTank (2024). Despite its significant contributions, PhishTank, like other centralized blacklisting systems, faces several challenges: it operates on a centralized architecture that has a single point of failure, it lacks transparency in decision-making, and is vulnerable to manipulations Sheng et al. (2009). For example, Vidyakeerthi et al. highlight notable discrepancies in PhishTank's labeling process, where URLs might be erroneously marked as phishing despite a majority of verifiers identifying them as legitimate, and *vice versa* Vidyakeerthi et al. (2022). Furthermore, there are instances where a URL is deemed legitimate without any verification from a verifier. These issues underscore concerns about the reliability and transparency of such systems.

In response to the escalating threat of phishing attacks and the inherent limitations of centralized blacklisting systems, there has been a growing interest in leveraging blockchain technology for decentralized phishing detection solutions Andryukhin (2019). While a number of studies have explored the potential of blockchain to identify phishing nodes and behaviors within blockchain networks Chen W. et al. (2020); Fu et al. (2022); Joshi et al. (2023); Zhang et al. (2021), efforts aimed specifically at detecting and blacklisting phishing URLs are still relatively uncommon. Identifying malicious nodes and behavioral patterns on blockchain networks plays a critical role in enhancing the overall security and integrity of these systems. Nevertheless, the specific challenge of URL phishing, which directly threatens end-users by leading to financial losses and data breaches, necessitates solutions that can accurately identify and address such threats in real-time.

One innovative initiative, PhishLedger, introduces a consortium blockchain-based mechanism specifically designed for the anti-tamper recording and multi-source reporting of phishing URLs. This approach aims to overcome the shortcomings of centralized systems by providing a platform for transparent, multi-party participation, thereby enhancing the efficiency and reliability of phishing detection mechanisms Liu et al. (2019). PhishLedger

employs a structured model comprising four types of nodes: reporting nodes, accounting nodes, servicing nodes, and supervising nodes. Each type of node plays a crucial role in the ecosystem, facilitating a streamlined process for reporting, verifying, and disseminating information on phishing URLs across a select consortium of organizations. By utilizing a consortium blockchain, PhishLedger ensures that the data within the network is immutable, preventing tampering and promoting a secure environment for data exchange. Despite its innovative design, PhishLedger faces certain limitations that restrict its broader applicability and impact. Primarily, the system's access is limited to specific organizations, thereby constraining the diversity and volume of phishing reports. This limitation potentially affects the comprehensiveness of the phishing URL database, as it excludes significant data that could be sourced from a broader community. Additionally, the lack of public accessibility raises questions about the credibility of its verifications, since external stakeholders cannot independently verify or contribute to the data. This restricts the system's transparency and limits its potential for collaborative enhancement.

PhishChain represents another significant advancement in the field of phishing detection, leveraging the power of blockchain technology to decentralize the process. By enabling open access and participation, PhishChain democratizes the task of phishing URL classification, thus addressing one of the limitations identified in PhishLedger's approach, which restricted access to specific organizations. This open participation model facilitated by PhishChain ensures that a wider base of users can contribute to and benefit from the phishing detection process, thus potentially increasing the volume and diversity of phishing URL data available for analysis and blacklisting Vidyakeerthi et al. (2022). While PhishChain's model of incentivizing participation through the assignment of skill points is a novel approach to encourage user engagement, it also presents challenges. The primary concern revolves around ensuring the quality and reliability of contributions. Since phishing detection expertise varies widely among participants, there is a risk that non-expert assessments could potentially dilute the accuracy of the phishing URL database. Additionally, the model's reliance on crowdsourced verification raises questions about the system's resilience to manipulation by malicious actors, who might seek to undermine the system's integrity for personal gain or to facilitate phishing attacks.

Building upon the insights gained from previous blockchain-based solutions like *PhishLedger* and *PhishChain*, this work introduces MLPhishChain, a novel transparent and decentralized framework. MLPhishChain distinctively leverages ML technologies to automate and refine the process of phishing URL detection. This approach significantly diverges from PhishLedger's reliance on a closed consortium model and PhishChain's dependency on crowdsourced user assessments. By integrating ML for the initial screening of URLs, MLPhishChain addresses a critical gap in existing systems: the variability in the quality of phishing detection stemming from the wide range of user expertise. This ML-driven method ensures a high level of accuracy and reliability in phishing detection, independent of user proficiency.

Moreover, MLPhishChain integrates with external sources such as VirusTotal for the additional assessment of URLs, allowing users to seek opinions other than those generated by the ML model. This integration enables MLPhishChain to utilize an extensive database of malware signatures and phishing heuristics, thereby significantly enhancing the depth and breadth of phishing URL analysis. Such an approach contrasts sharply with PhishChain's model, which, while inclusive and participatory, may lack the technical depth in its assessment capabilities due to its reliance on non-expert contributions.

Furthermore, MLPhishChain is the first to introduce a dynamic re-assessment mechanism for URLs, a feature not present in prior works like PhishChain and PhishLedger. This innovative approach enables continuous updates to URL classifications as content changes, ensuring that the detection system remains responsive to evolving phishing tactics.

This work aims to bridge the expertise gap in blockchain-based phishing detection systems by employing an ML model for initial assessments, supplemented by the option to re-evaluate URLs as their content changes, and further enhanced by the comprehensive external analysis for additional verification. This approach promises to significantly enhance the reliability and effectiveness of phishing blacklists while maintaining the benefits of decentralization and community involvement.

4 Methodology

MLPhishChain encompasses communication between six components as shown in Figure 1.

1. The user, who submits URLs for phishing assessment and requests re-evaluations.
2. The DApp, which provides an interface for user interactions and displays phishing assessment results.
3. The smart contract, which handles requests, queries the ML service and the external validation system, records and updates URL statuses on the blockchain, and flags URLs that need verification by an admin.
4. The admin, an expert who reviews flagged URLs and makes final decisions on uncertain classifications through the DApp;
5. The ML-based phishing detection service, which analyzes URLs to classify their phishing risk.
6. The external validation system, which provides an independent assessment of URLs for additional user verification.

The MLPhishChain methodology is designed as a user-centric process that begins when a user inputs a URL into the DApp. Initially, the smart contract verifies whether the URL's phishing status is already recorded on the ledger. If a status is found, it is immediately made available to the user. For URLs without a pre-recorded status, the smart contract queries the ML-based phishing detection service to determine the phishing status of the URL. This status is then securely recorded on the blockchain via the smart contract, ensuring immutability and transparency for future queries. As websites may evolve over time, MLPhishChain incorporates a mechanism for re-evaluation, reinforcing the platform's commitment to accuracy and adaptability. When the user initiates a re-evaluation request, the smart contract re-queries the ML model to obtain an updated classification for the URL, regardless of any previous status recorded on the ledger. If the new classification differs from the recorded status, the URL is

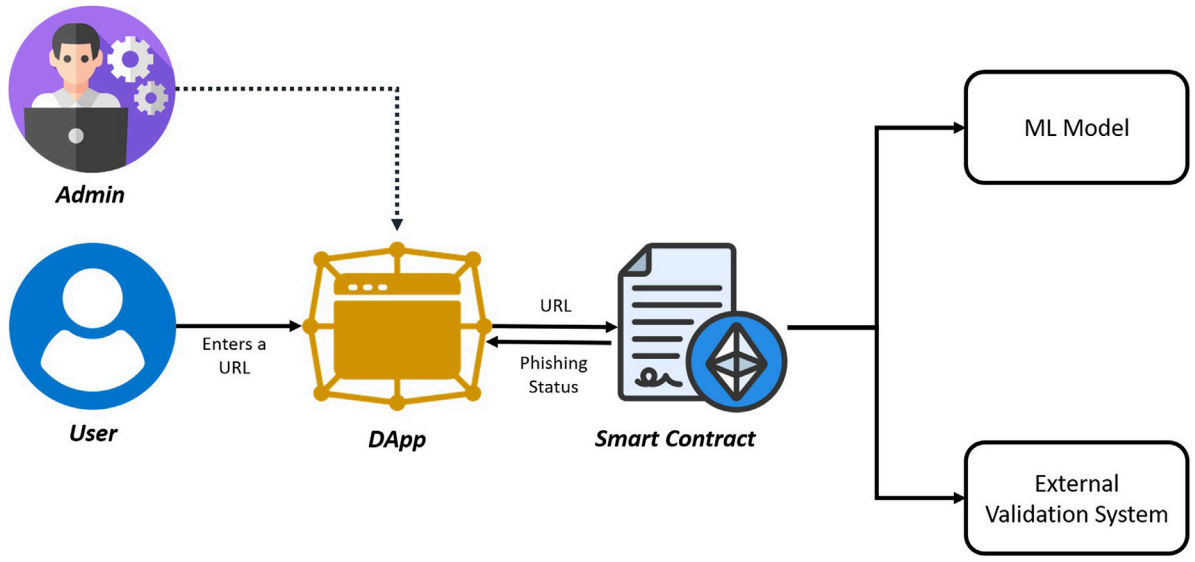

FIGURE 1
The six components of MLPhishChain: 1) User submits URLs and requests re-evaluations, 2) DApp displays results and interacts with the user, 3) Smart Contract processes requests, queries services, and records data on the blockchain, 4) Admin reviews flagged URLs for final classification, 5) ML-based Phishing Detection Service classifies URLs based on phishing risk, and 6) External Validation System provides independent URL assessments.

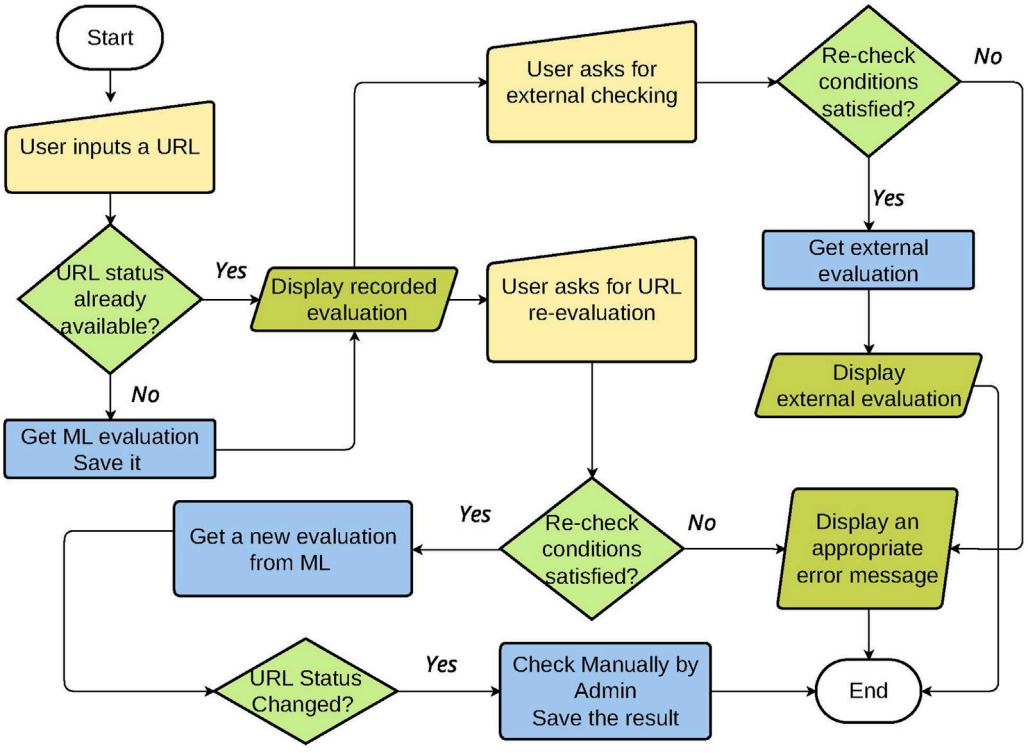

FIGURE 2
MLPhishChain FlowChart. The user submits a URL via the DApp, the smart contract checks if the URL's phishing status is recorded on the blockchain. If not, it queries the ML model to classify the URL. The status is then recorded. The user can request a re-evaluation, triggering the smart contract to re-query the ML model. If there's a discrepancy, the URL is flagged for admin review. The system also supports external validation via a third-party service.

flagged for expert review by the admin to determine a final decision for the ledger. This process ensures the blockchain reflects the most current and accurate status of URLs. Additionally, even if the classification changes, the full history of status changes is permanently recorded on the ledger, preserving its immutability.

To avoid reliance on a single source, MLPhishChain enables users to seek an additional assessment of a given URL. In such cases, the smart contract queries a system for external validation to retrieve an independent phishing risk score. This score is then displayed to the user via the DApp but is not recorded on the blockchain. The purpose of this step is to provide users with a second opinion on the URL's safety within the same platform, helping them decide whether or not to visit the site.

To optimize the service's efficiency and discourage unnecessary re-evaluations, MLPhishChain enforces specific usage guidelines. Each user is allotted a limited number of re-evaluation and external validation requests within a specific timeframe, which cannot be exceeded. This policy manages the rate of API calls to external services, preventing system overload. These measures ensure fair distribution of system resources among users and encourage thoughtful submission of URLs for re-evaluation.

A detailed breakdown of the process is illustrated in the flowchart provided in Figure 2.

Through these structured processes and the integration of trusted external data, MLPhishChain not only improves the transparency and reliability of its operations but also ensures that its database of phishing URLs is continuously updated and reflective of the changing nature of web threats. The combination of Blockchain with advanced ML techniques and the comprehensive cybersecurity knowledge from external sources provides MLPhishChain with a powerful mechanism to protect users from phishing, ensuring a secure and trustworthy Internet environment.

5 Simulation

This section details the steps followed to implement a proof of concept for the presented methodology, allowing for both validation and replication of the system with the same or different design choices.

5.1 Experimental setup and design choices

First, we detail the key design choices behind MLPhishChain, including the selection of the ML model, the external validation system, re-evaluation/external validation criteria, and admin evaluation.

5.1.1 Selection of ML model for phishing detection

A variety of ML systems can fulfill the requirements of MLPhishChain; however, the primary criteria for model selection include high accuracy, accessibility, and the ability to assess website content rather than relying solely on URL patterns. This approach ensures that re-evaluations can reflect content changes accurately, as the model does not rely on precomputed results but performs a fresh analysis for each evaluation. CheckPhish by Bolster AI was chosen as it meets these criteria effectively. Known for its high accuracy in detecting phishing patterns, CheckPhish employs sophisticated ML techniques trained on diverse phishing data to analyze complex indicators, such as visual and contextual elements, making it well-suited for identifying evolving phishing tactics. By leveraging CheckPhish, MLPhishChain gains robustness in detecting both known and novel phishing URLs.

5.1.2 External validation system

To provide an independent secondary assessment of URL classifications, MLPhishChain integrates an external validation system. The selected system must offer a reliable, comprehensive phishing threat database and an extensive reputation-scoring mechanism to reinforce user confidence in URL classifications. VirusTotal was chosen for this role due to its well-regarded, extensive database and its ability to draw on multiple threat detection sources to assess phishing risks. By using VirusTotal, MLPhishChain provides users with an additional layer of validation, enhancing the reliability of phishing detection and offering an extra level of confidence in the system's URL classifications.

5.1.3 URL Re-evaluation and external validation criteria

To ensure that URL classifications remain up-to-date, MLPhishChain allows users to request re-evaluations. The system can determine re-evaluation intervals based on specific criteria, including the time elapsed since the last check, user re-evaluation requests, and significant changes in the URL's content or structure. For the initial setup presented in this paper, users are allowed a limited number of re-evaluation or external validation requests per minute (e.g., 3 requests per minute). Once this limit is reached, further requests are blocked until the next time interval, preventing excessive usage and ensuring fair resource distribution. As the system scales, these parameters and conditions can be adjusted based on usage patterns and resource constraints to maintain optimal performance.

5.1.4 Admin evaluation

In the initial deployment, the admin is expected to be a professional security analyst who manually reviews URLs flagged by the system to confirm or override phishing classifications, providing a critical check for edge cases and ambiguous results. To ensure objectivity and reduce potential bias, MLPhishChain records all admin decisions on the blockchain, preserving a transparent history. In future iterations, we aim to replace admin oversight with an automated review system based on advanced consensus algorithms or multi-model voting to handle increased scale. This transition would further enhance the system's objectivity and resilience, as well as reduce potential human errors or biases.

5.2 Smart contract design

The smart contract is the backbone of any blockchain-based application, as it encapsulates all the logic and rules governing user interactions with the DApp. The smart contract of MLPhishChain primarily involves three elements: data assets, modifiers, and functions.

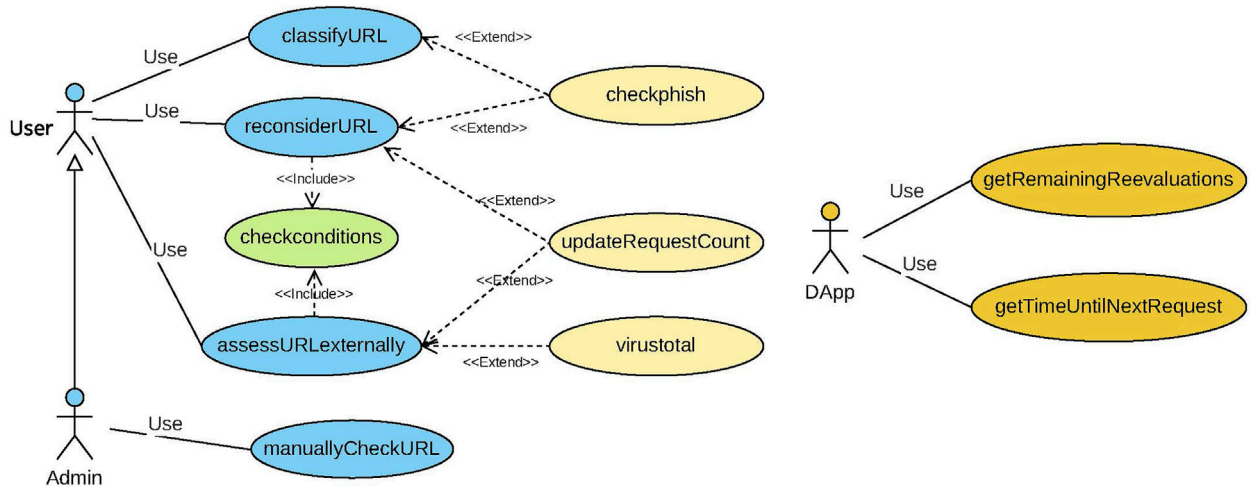

FIGURE 3
MLPhishChain use case diagram showing the interactions between users, the DApp, and the smart contract through various functions. Users can classify URLs, request re-evaluations, or seek external validation. The system checks conditions to prevent excessive requests, interacts with external services like CheckPhish and VirusTotal, and updates request counts. The admin has the ability to manually check and assign classifications to URLs after re-evaluation. The DApp can get the remaining evaluations in a given time frame, and the time until the next request could be sent.

5.2.1 Data assets

In terms of data assets, the smart contract primarily tracks URLs and their classifications. Additionally, it records the last time a user requested an external validation or re-evaluation, the number of requests made by a user within a specified time frame, the duration of this time frame, and the maximum number of requests permitted within it. The contract also monitors URLs flagged for manual review by the admin and maintains the addresses of both CheckPhish and VirusTotal to be able to communicate with them (more on that in next sections). It is important to note that MLPhishChain does not require any personal data from the user. The system exclusively processes URLs and their classifications, which are inherently public and do not include any personally identifiable information (PII) that would be subject to GDPR or other data privacy regulations.

5.2.2 Modifiers

One modifier is declared to identify the admin, who is the only entity authorized to check the URLs flagged for manual review. These URLs have ML classifications that have changed following re-evaluation.

5.2.3 Functions

The functions enable users to interact with the smart contract. There are nine functions, as depicted in the use case diagram in Figure 3.

1. ClassifyURL: This is the main function called when a user wishes to evaluate a URL. It takes the URL and returns a classification (phishing or legitimate). This function first checks if the URL's classification exists on the blockchain. If it does, the classification is returned. If not, the classification is determined using the checkphish function.
2. ReconsiderURL: This function sends a URL for reconsideration by checkphish, after verifying that the conditions for re-checking are met. If so, it returns a classification.
3. AssessURLexternally: This function is used to check the classification of a URL by Virus Total. It also ensures the conditions for this request are met before obtaining Virus Total's classification via the virustotal function.
4. Checkconditions: This internal function is used when re-evaluating a URL with checkphish or with Virus Total. Its primary aim is to minimize API requests by verifying whether the user has exceeded the maximum number of requests per time frame.
5. Checkphish: An internal function that interacts with CheckPhish to obtain a URL's classification.
6. Virustotal: An internal function that communicates with Virus Total to obtain a URL's classification.
7. UpdateRequestCount: An internal function that updates the count of requests within the time frame whenever a user requests re-evaluation or external verification.
8. GetRemainingReevaluations: A public function that the DApp automatically calls to determine the number of remaining re-evaluations or external verifications a user can perform within the current time frame.
9. GetTimeUntilNextRequest: A public function that the DApp automatically calls to retrieve the remaining time until a user can submit a new request for re-evaluation or external assessment within the current time frame.
10. ManuallyCheckURL: A public function that only the admin can execute. This function is designed to manually assign a phishing status to a URL whose classification has changed following re-evaluation.

The smart contract diagram is illustrated in Figure 4.

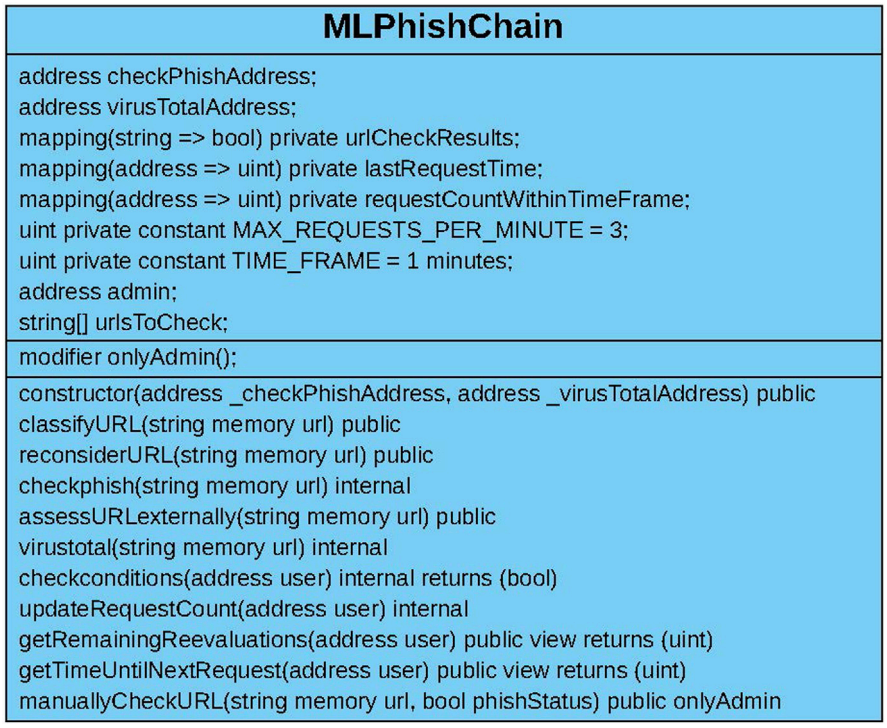

FIGURE 4
MLPhishChain contract diagram showing the data assets, modifiers, and available functions.

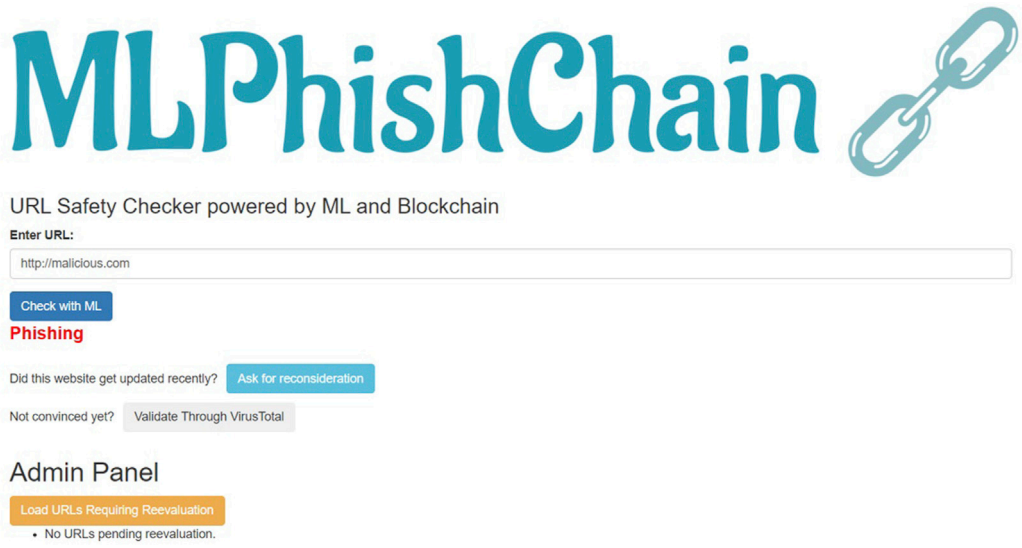

FIGURE 5
MLPhishChain DApp showing where a user can check a given URL, ask for reconsideration, or externally validate. Also, the admin panel is shown only for admins and lists URLs requiring evaluation.

5.3 DApp

A screenshot of the DApp is shown in Figure 5. The DApp is constructed as a user interface using HTML/CSS and JavaScript, with web3.js for connecting to the smart contract. Users enter a URL and click on the "Check with ML" button to receive a classification. Once the result is obtained, users can request a re-evaluation or an external check by clicking the corresponding

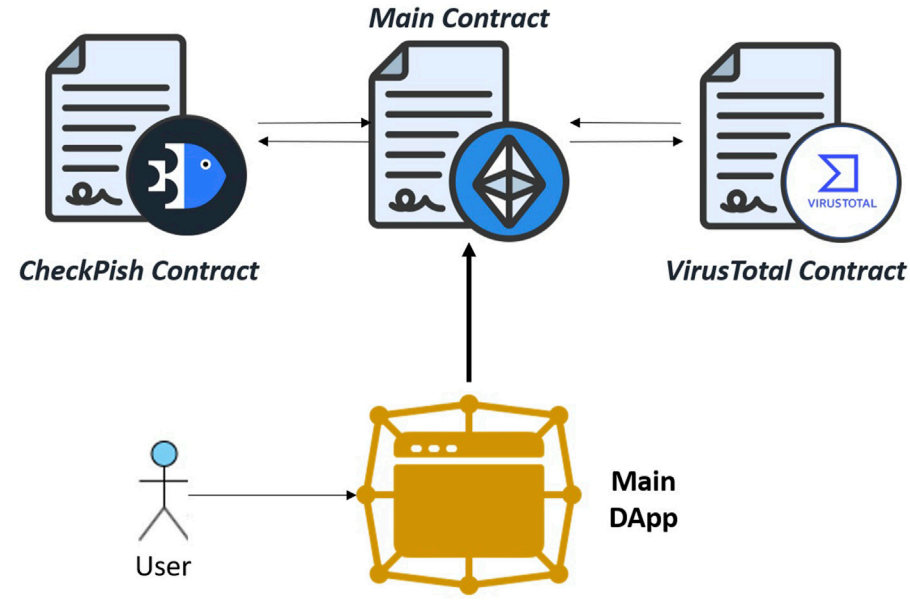

FIGURE 6
Emulating communication between the main contract and the services via two additional smart contracts; one for CheckPhish and the other for VirusTotal.

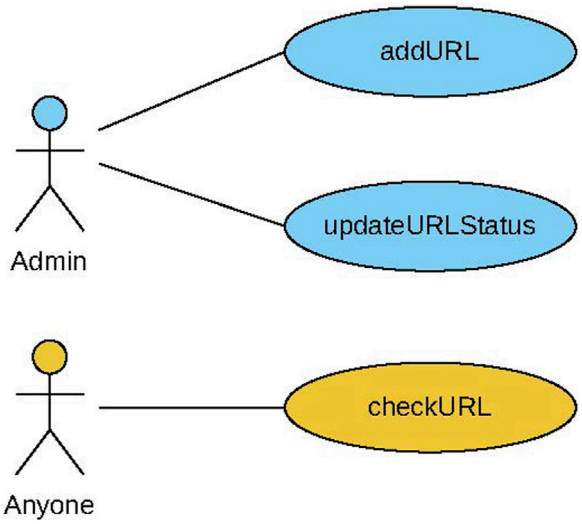

FIGURE 7
CheckPhish use case diagram showing how a CheckPhish admin can add URLs and update their statuses. Any user can check the status of a given URL.

buttons. All transactions must be confirmed via MetaMask before execution. The DApp also has an admin section that displays the URLs submitted for re-evaluation, and this only happens if the entity logged in is the admin, which is identified by the address of the MetaMask account in use. The admin has to manually inspect each of the flagged URLs and assign a suitable class for it.

5.4 Connection to CheckPhish and VirusTotal

To access the necessary information from CheckPhish and VirusTotal, oracles are typically employed. However, to simulate this functionality without real Ether, and with more flexibility to illustrate how a URL class might change over time, two additional contracts were created: one for CheckPhish and one for VirusTotal to emulate how the main contract can communicate with and retrieve information from these services, as shown in Figure 6.

5.4.1 CheckPhish contract

The CheckPhish contract includes a list of URLs with their classifications. It features a function that the main smart contract can use to retrieve a URL classification (checkURL). Additional functions are included for adding a URL with its classification to the list (addURL) and for updating the status of a URL (updateURLStatus), which can only be performed by the administrator. The use case diagram for this contract is shown in Figure 7, and the contract diagram is shown in Figure 8. The VirusTotal contract features similar functionalities.

5.4.2 CheckPhish DApp

To simulate scenarios where a URL's classification might change, a CheckPhish DApp was developed. This allows CheckPhish's admin to change the status of a URL. In such a case, when a user from the MLPhishChain DApp requests reconsideration, the new class is retrieved from CheckPhish. If the new class is the same as the one recorded on the ledger, nothing happens. However, if the new class is different, we should wait for the admin to check the URL manually. Once this

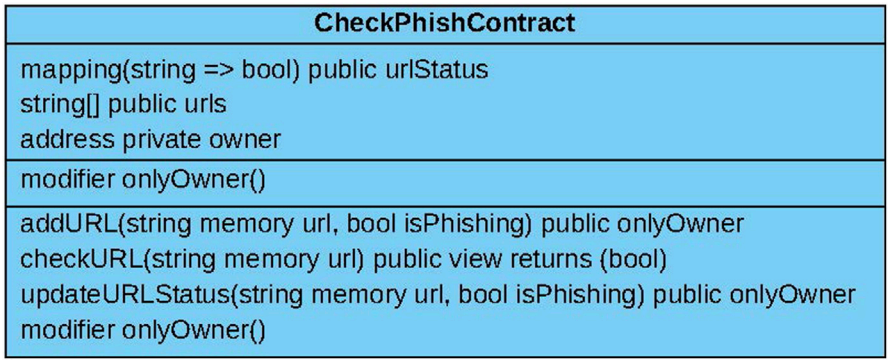

FIGURE 8
CheckPhish contract diagram showing the data assets, modifiers, and available functions.

FIGURE 9
CheckPhish DApp showing how an admin can change the status of a given URL.

is done, the new status is updated on the ledger, and historical classifications are always saved and cannot be altered. A screenshot of the admin panel in the CheckPhish DApp is presented in Figure 9.

5.5 Deployment

To deploy the full MLPhishChain system, we utilized Ganache and the Truffle framework. Ganache is a personal blockchain that allows us to simulate a local Ethereum network, enabling us to deploy and test multiple smart contracts in a controlled environment. This setup includes the core contract responsible for URL classification, alongside contracts for external validation and re-evaluation. It allows testing of user interactions, including URL submissions, re-evaluations, external validation requests via services like CheckPhish and VirusTotal, and admin reviews. For the test, we set up a simulated environment with 10 sample users, each holding an amount of 100 Ether to interact with the contract and request URL classifications or re-evaluations. One of the users is designated as an admin, to make sure they have exclusive access to perform admin-related tasks. Users could request up to 3 re-evaluations per minute, and their request counts were monitored and recorded. The setup allowed us to ensure that the system functions correctly across different types of interactions, from user requests to admin manual reviews of flagged URLs. This approach is valuable for validating the system's core functionalities before deployment on a live network, as it ensures that the entire interaction chain—from user actions to external validations and admin oversight—is working as intended.

However, while Ganache and Truffle provide an effective testing environment, they do not fully simulate a live Ethereum network, where factors such as gas fees, network latency, and the decentralized nature of transactions can affect system performance. These limitations are further discussed in Section 7 of the paper.

5.6 Demo

A demo of the DApp is available on YouTube and can be accessed through the link below: https://www.youtube.com/watch?v=xc49i3phlcU.

6 Validation

The proposed blockchain system adheres to the major cybersecurity principles required, as it ensures integrity, reliability, and immutability. To validate the effectiveness of MLPhishChain, a series of tests were conducted to demonstrate the system's capabilities in detecting and managing phishing threats.

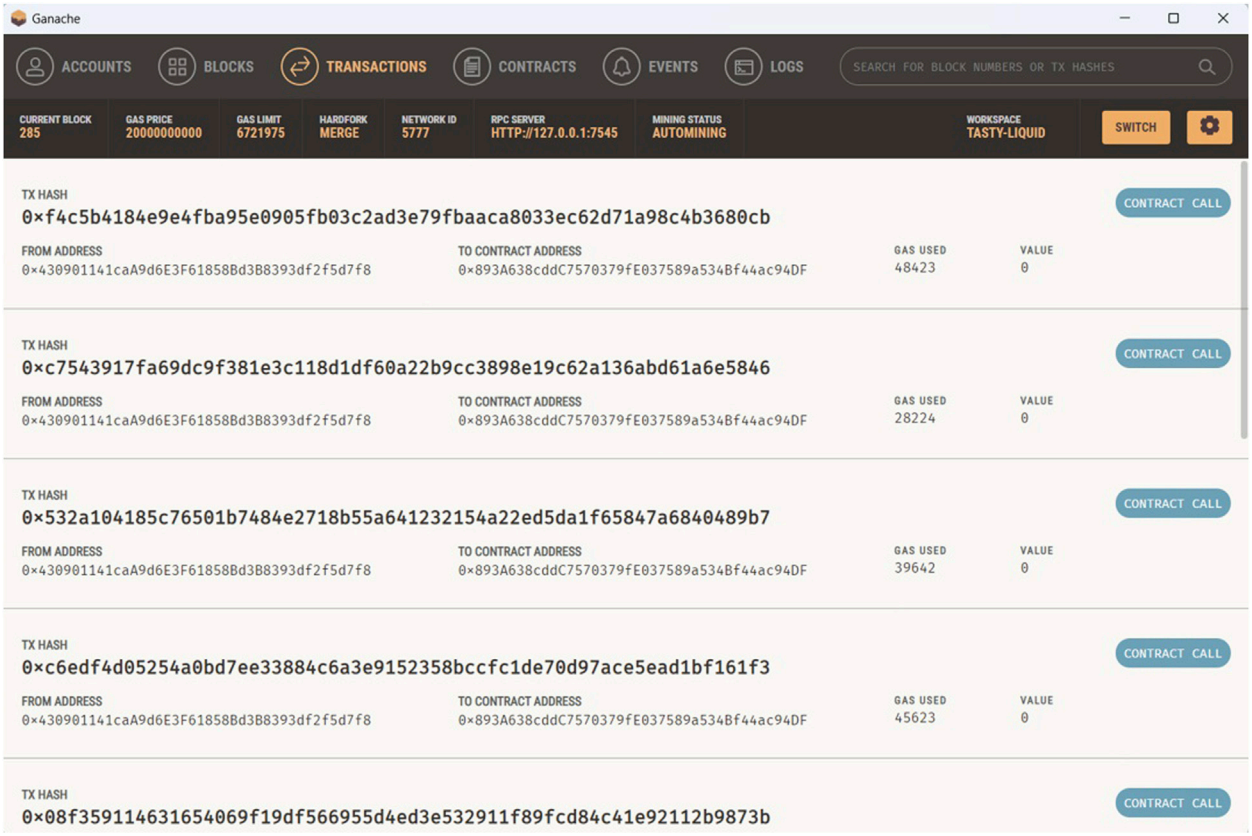

FIGURE 10
Sample transactions recorded by Ganache.

6.1 Integrity and immutability

First, the integrity and immutability of the blockchain-based records were validated. Each transaction on the blockchain—whether adding a new URL's classification or updating an existing one—creates a permanent, unalterable record. A list of sample transactions is shown in Figure 10 where each transaction is identified by a specific hash, and any attempt to alter the content of the transaction will change this hash. This is crucial because it ensures that the data remains tamper-proof and verifiable, thereby maintaining trust in the system.

In addition, the immutability feature provides a transparent audit trail, allowing stakeholders to trace the history of any URL's classification. Such transparency is essential for building user confidence in the platform's reliability and accuracy. Moreover, it prevents malicious actors from manipulating past records to hide phishing activities, as any unauthorized changes would be immediately detectable through the altered hashes.

6.2 Reliability

To assess reliability, the system's uptime and responsiveness under various conditions were measured. MLPhishChain was subjected to continuous operation over a prolonged period, during which it processed URL classifications without significant downtime. Additionally, high traffic was simulated with multiple simultaneous requests. The results showed that MLPhishChain maintained functionality with minimal latency, thus confirming its capability to handle real-world operational demands.

6.3 Performance metrics

To evaluate the practical performance of MLPhishChain, key metrics such as transaction latency, scalability, and resource consumption were measured using a simulated blockchain environment with Ganache. This setup allowed us to obtain preliminary insights into the system's performance under controlled conditions.

6.3.1 Latency

The average transaction confirmation time for operations such as URL classification or re-evaluation was observed to be 0.1489 s when running on a Ganache simulated blockchain. To obtain this metric, a script was created to perform a single transaction—specifically, classifying a random URL. The script was executed 10 times as shown in Table 1, and the average time across these runs was calculated to be 0.1489 s. This value reflects the fast processing capabilities of a local blockchain environment like Ganache. Also, the standard deviation is 0.0071, reflecting the low variability of the system. However, it is important to note that on a production-level network such as Ethereum Mainnet, transaction confirmation times can vary significantly due to factors such as network congestion, fluctuations in gas prices, and miner

TABLE 1 Transaction times for 10 runs.

Run	1	2	3	4	5	6	7	8	9	10	AVG
Time (sec)	0.154	0.154	0.147	0.164	0.143	0.144	0.141	0.144	0.155	0.143	**0.1489**

Bold value represents the average runtime of a single transaction.

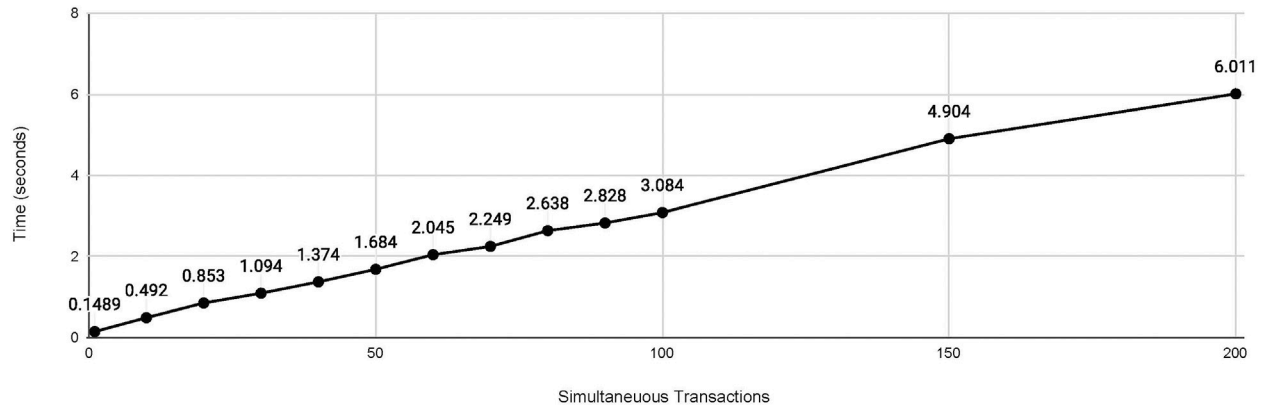

FIGURE 11
Number of Simultaneous transactions and their required times.

prioritization. As a result, transaction times in a live environment are likely to be higher and less predictable.

6.3.2 Scalability

Scalability was tested by simulating high traffic volumes with multiple simultaneous requests. The system demonstrated consistent performance, with no significant increase in latency, indicating that the underlying smart contract design and logic can efficiently handle high transaction volumes. The number of simultaneous transactions and the time to complete them is reported in Figure 11. Specifically, the time taken to complete 10, 100, and 200 simultaneous transactions was 0.492, 3.084, and 6.011 s, respectively. While these results are promising, scalability on real-world blockchains may vary due to factors such as block size limitations and fluctuating transaction throughput capacities.

6.3.3 Resource consumption

Gas consumption per transaction was measured for different functions. URL classification operations consumed less than 30,000 gas units, while more complex operations, such as re-evaluation or external validation requests, or updates required 30,000–50,000 gas units, as seen in Figure 10. These values indicate that the smart contract was designed to minimize unnecessary computations, optimizing cost efficiency in environments where gas fees are a significant consideration. However, in a real-world deployment, the gas fees could differ.

7 Limitations

In a simulated testing environment, Ganache and Truffle offer several advantages for rapid development, including fast transaction processing, easy blockchain resets, and customizable mining speeds. These features provide a controlled environment to test smart contract functionality, enabling developers to identify and address logic errors and conduct initial performance testing without the costs associated with gas fees. While this simulated testing is sufficient for the proof-of-concept presented in this paper, it also presents some limitations that need to be addressed before moving to the Ethereum Mainnet or other production-level networks:

- Network Conditions and Latency: Ganache and Truffle do not replicate the network latency, variability in node response times, or potential congestion encountered on the Mainnet. This can lead to discrepancies between simulated and real-world transaction times and network behavior, affecting how certain timing-sensitive functions or concurrent transactions may perform.
- Gas Costs and Optimization: In Ganache, gas costs are minimal or customizable, which means that developers do not experience the actual economic constraints of gas optimization as they would on the Mainnet. This difference can result in smart contracts that function well in a test environment but encounter issues with gas costs and efficiency when deployed on a production network.
- Security and Robustness Testing: Testing in Ganache may not reveal security vulnerabilities that could be exploited in a more distributed, public network environment. For instance, attack vectors like front-running, denial-of-service attacks, or spam transactions that might occur on Mainnet cannot be fully simulated on Ganache, limiting the assessment of the contract's robustness against such threats.

- Decentralization and Consensus: Ganache operates with a single instance simulating a blockchain network, lacking the true distributed nature of Ethereum's consensus mechanism. As a result, testing on Ganache does not account for the potential variability introduced by decentralized consensus processes, which could affect how transactions are ordered and confirmed on the Mainnet.
- Immutable Data and Irreversibility: On Ganache, developers can easily reset or alter blockchain states, which is not possible on the Mainnet where transactions are irreversible. This difference impacts how data integrity and error-handling procedures are tested, as issues that are easily corrected in a test environment could have significant, irreversible consequences on the Mainnet.

To address the limitations of Ganache and ensure a more realistic evaluation, future work will involve deploying a beta version of the system on a public test network, such as Ethereum Testnet, to gather real-world performance metrics. This will help assess the system's behavior under actual network conditions, including transaction latency, gas costs, and network congestion. Additionally, stress testing will be conducted to simulate high transaction volumes and peak usage, providing insights into the system's scalability and identifying any potential bottlenecks. By integrating real-time performance monitoring and feedback from external users during the beta phase, we can continuously fine-tune the system for improved efficiency and robustness before full deployment on the Mainnet.

8 Conclusion

In this paper, we have introduced MLPhishChain, a DApp that leverages the strengths of blockchain technology and ML to address the limitations of traditional anti-phishing methods. By combining the analytical capabilities of CheckPhish AI with the immutable and transparent nature of blockchain, MLPhishChain provides a more secure and trustworthy solution for URL verification and phishing detection.

The integration of blockchain ensures that URL risk statuses are recorded on an immutable ledger, making the data tamper-proof and verifiable by all users. This addresses critical gaps in data integrity and transparency that plague centralized systems. Additionally, the re-evaluation mechanism allows for continuous updating of URL classifications, ensuring the database remains accurate and relevant as websites evolve. This feature maintains a balance between the immutability of blockchain and the need for up-to-date information.

Moreover, the inclusion of VirusTotal as an external evaluation source provides users with a second opinion, further enhancing the system's robustness and reliability. By leveraging comprehensive external databases and expertise, MLPhishChain ensures that URL reassessments benefit from extensive cybersecurity knowledge.

Finally, the decentralized nature of MLPhishChain enhances system redundancy and resilience, preventing single points of failure and enabling continuous operation even if parts of the network are compromised. This distributed approach democratizes the verification process and fosters greater trust among users.

Overall, MLPhishChain represents a significant advancement in the fight against phishing attacks, providing a secure, transparent, and resilient solution that addresses the inherent weaknesses of traditional centralized systems.

9 Future work

While MLPhishChain has demonstrated significant potential in the detection and management of phishing threats, there are several avenues for future development that could enhance its capabilities and extend its application further:

- Expanding Machine Learning Models: Developing and integrating additional machine learning models tailored to detect emerging phishing techniques. This would improve detection accuracy and adaptability to new phishing strategies.
- Integrating Additional Cybersecurity Tools: Integrating more external services for a comprehensive security assessment of each URL. In addition to VirusTotal, incorporating tools such as Hybrid Analysis, Jotti's Malware Scan, MetaDefender, Any.Run, and Malwr could provide a more thorough evaluation of potential threats. This multi-faceted approach would improve the overall robustness of the platform by leveraging diverse sources of threat intelligence and malware analysis.
- User Feedback Mechanism: Implementing a user feedback system to allow users to report suspicious URLs and verify classifications. This crowdsourced data, combined with ML data, can enhance the system's ability to learn from real-world phishing attempts and continually improve its detection capabilities.
- Automated Re-Checking Mechanisms: Currently, re-checking URL classifications is manually initiated by the user, ensuring minimal impact on system performance. Future work will explore the introduction of automated re-checking mechanisms to improve system efficiency and reliability. Experiments will be conducted to determine the optimal balance between re-check frequency and performance, particularly under varying system loads, ensuring scalability and seamless operation in large-scale deployments.
- Addressing Blockchain Scalability and Storage Optimization: To address potential blockchain bloat and rising storage costs from accumulating URL data, we propose a hybrid storage model. In this approach, essential data such as recent classifications and cryptographic proofs would be stored on-chain, while full historical data would be maintained off-chain using decentralized storage solutions like IPFS or Arweave. To manage storage efficiently, pruning could be applied, which involves removing or archiving outdated records after a set retention period, keeping only the most critical data on-chain. This helps prevent excessive data buildup while maintaining operational integrity. Additionally, by utilizing Layer 2 scaling solutions like rollups, which process transactions off the main blockchain and batch them together before finalizing them on-chain, we can reduce transaction costs and improve overall scalability, avoiding congestion on the primary blockchain.
- Integration with Web Browsers: Developing browser extensions for popular web browsers to seamlessly integrate

MLPhishChain's URL verification process. This would provide users with instant phishing detection and protection while browsing the web.

- Collaborative Threat Intelligence Sharing: Establishing partnerships with other cybersecurity organizations and platforms to share threat intelligence data. This collaborative approach would enhance the overall effectiveness of phishing detection and contribute to a safer online ecosystem.
- Deeper Security Analysis: Future work will focus on enhancing MLPhishChain's resilience to threats, such as tampering with transactions, bypassing re-evaluation, and spoofing API interactions. This could include integrating Merkle trees for tamper-proof verification, exploring decentralized decision-making to reduce centralized reviews, and improving security with real-time monitoring and logging of API interactions.
- Real-World Deployment and Testing: To validate MLPhishChain's performance in real-world conditions, future work will involve deploying a beta version on a public test network like Ethereum Testnet. This will allow us to gather real-world performance metrics, assess network behavior under varying conditions, and stress test the system's scalability and transaction handling capabilities.

These enhancements not only aim to enhance MLPhishChain's technical capabilities but also aim to develop a safer internet environment through innovative and collaborative approaches to cybersecurity.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

FT: Conceptualization, Writing–original draft, Formal Analysis, Investigation, Methodology, Software, Validation, Visualization. ES-N: Conceptualization, Project administration, Supervision, Validation, Writing–review and editing. AC: Project administration, Supervision, Validation, Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. The authors would like to acknowledge that this work has been supported by the Maroun Semaan Faculty of Engineering and Architecture (MSFEA) at the American University of Beirut (AUB).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Agbo, C. C., Mahmoud, Q. H., and Eklund, J. M. (2019). Blockchain technology in healthcare: a systematic review. *Healthc. (MDPI)* 7, 56. doi:10.3390/healthcare7020056

Aleroud, A., and Zhou, L. (2017). Phishing environments, techniques, and countermeasures: a survey. *Comput. and Secur.* 68, 160–196. doi:10.1016/j.cose.2017.04.006

Al-Fayoumi, M., Alhijawi, B., Al-Haija, Q. A., and Armoush, R. (2024). Xai-phd: fortifying trust of phishing ural detection empowered by shapley additive explanations. *Int. J. Online and Biomed. Eng.* 20, 80–101. doi:10.3991/ijoe.v20i11.49533

Al-Haija, Q. A., and Al Badawi, A. (2021). "Url-based phishing websites detection via machine learning," in *2021 international conference on data analytics for business and industry (ICDABI)* (IEEE), 644–649.

Aljofey, A., Rasool, A., Jiang, Q., and Qu, Q. (2022). A feature-based robust method for abnormal contracts detection in Ethereum blockchain. *Electronics* 11, 2937. doi:10.3390/electronics11182937

Alzahrani, A. J., and Ghorbani, A. A. (2015). "Real-time signature-based detection approach for sms botnet," in *2015 13th annual conference on privacy, security and trust (PST)* (IEEE), 157–164.

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., et al. (2018). "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, 1–15.

Andryukhin, A. (2019). "Phishing attacks and preventions in blockchain based projects," in *2019 international conference on engineering technologies and computer science (EnT)* (IEEE), 15–19.

Apruzzese, G., Conti, M., and Yuan, Y. (2022). "Spaceish: the evasion-space of adversarial attacks against phishing website detectors using machine learning," in *Proceedings of the 38th annual computer security applications conference*, 171–185.

Aslam, S., Aslam, H., Manzoor, A., Chen, H., and Rasool, A. (2024). Antiphishstack: lstm-based stacked generalization model for optimized phishing ural detection. *Symmetry* 16, 248. doi:10.3390/sym16020248

Attaran, M. (2022). Blockchain technology in healthcare: challenges and opportunities. *Int. J. Healthc. Manag.* 15, 70–83. doi:10.1080/20479700.2020.1843887

Bell, S., and Komisarczuk, P. (2020). "An analysis of phishing blacklists: google safe browsing, openish, and phishtank," in *Proceedings of the australasian computer science week multiconference*, 1–11.

Buterin, V. (2014). A next-generation smart contract and decentralized application platform. *white Pap.* 3, 2–1.

Catal, C., Giray, G., Tekinerdogan, B., Kumar, S., and Shukla, S. (2022). Applications of deep learning for phishing detection: a systematic literature review. *Knowl. Inf. Syst.* 64, 1457–1500. doi:10.1007/s10115-022-01672-x

Chen, J., Xia, X., Lo, D., Grundy, J., Luo, X., and Chen, T. (2020a). Defining smart contract defects on Ethereum. *IEEE Trans. Softw. Eng.* 48, 327–345. doi:10.1109/tse.2020.2989002

Chen, W., Guo, X., Chen, Z., Zheng, Z., and Lu, Y. (2020b). Phishing scam detection on Ethereum: towards financial security for blockchain ecosystem. *IJCAI* 7, 4456–4462. doi:10.24963/ijcai.2020/621

Cole, R., Stevenson, M., and Aitken, J. (2019). Blockchain technology: implications for operations and supply chain management. *Supply chain Manag. An Int. J.* 24, 469–483. doi:10.1108/scm-09-2018-0309

Dasgupta, D., Akhtar, Z., and Sen, S. (2022). Machine learning in cybersecurity: a comprehensive survey. *J. Def. Model. Simul.* 19, 57–106. doi:10.1177/1548512920951275

Demirkan, S., Demirkan, I., and McKee, A. (2020). Blockchain technology in the future of business cyber security and accounting. *J. Manag. Anal.* 7, 189–208. doi:10.1080/23270012.2020.1731721

Do, N. Q., Selamat, A., Krejcar, O., Herrera-Viedma, E., and Fujita, H. (2022). Deep learning for phishing detection: taxonomy, current challenges and future directions. *Ieee Access* 10, 36429–36463. doi:10.1109/access.2022.3151903

Fu, B., Yu, X., and Feng, T. (2022). Ct-gcn: a phishing identification model for blockchain cryptocurrency transactions. *Int. J. Inf. Secur.* 21, 1223–1232. doi:10.1007/s10207-022-00606-6

Hasanova, H., Baek, U.-j., Shin, M.-g., Cho, K., and Kim, M.-S. (2019). A survey on blockchain cybersecurity vulnerabilities and possible countermeasures. *Int. J. Netw. Manag.* 29, e2060. doi:10.1002/nem.2060

Hölbl, M., Kompara, M., Kamišalić, A., and Nemec Zlatolas, L. (2018). A systematic review of the use of blockchain in healthcare. *Symmetry* 10, 470. doi:10.3390/sym10100470

Jibat, D., Jamjoom, S., Al-Haija, Q. A., and Qusef, A. (2023). A systematic review: detecting phishing websites using data mining models. *Intelligent Converged Netw.* 4, 326–341. doi:10.23919/icn.2023.0027

Joshi, K., Bhatt, C., Shah, K., Parmar, D., Corchado, J. M., Bruno, A., et al. (2023). Machine-learning techniques for predicting phishing attacks in blockchain networks: a comparative study. *Algorithms* 16, 366. doi:10.3390/a16080366

Khan, A. G., Zahid, A. H., Hussain, M., and Riaz, U. (2019). "Security of cryptocurrency using hardware wallet and qr code," in *2019 international conference on innovative computing (ICIC)* (IEEE), 1–10.

Khonji, M., Iraqi, Y., and Jones, A. (2013). Phishing detection: a literature survey. *IEEE Commun. Surv. and Tutorials* 15, 2091–2121. doi:10.1109/surv.2013.032213.00009

Lee, W.-M., and Lee, W.-M. (2019). Using the metamask chrome extension. *Begin. Ethereum Smart Contracts Program. Examples Python, Solidity, JavaScript*, 93–126. doi:10.1007/978-1-4842-5086-0_5

Liu, D., Wang, W., Wang, Y., and Tan, Y. (2019). "Phishledger: a decentralized phishing data sharing mechanism," in *Proceedings of the 1st international electronics communication conference*, 84–89.

Maleh, Y., Shojafar, M., Alazab, M., and Romdhani, I. (2020). Blockchain for cybersecurity and privacy: architectures, challenges, and applications

Moosavi, J., Naeni, L. M., Fathollahi-Fard, A. M., and Fiore, U. (2021). Blockchain in supply chain management: a review, bibliometric, and network analysis. *Environ. Sci. Pollut. Res.*, 1–15. doi:10.1007/s11356-021-13094-3

Odeh, A., Al-Haija, Q. A., Aref, A., and Taleb, A. A. (2023). Comparative study of catboost, xgboost, and lightgbm for enhanced ural phishing detection: a performance assessment. *J. Internet Serv. Inf. Secur* 13, 1–11. doi:10.58346/jisis.2023.i4.001

Oest, A., Safaei, Y., Doupé, A., Ahn, G.-J., Wardman, B., and Tyers, K. (2019). "Phishfarm: a scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists," in *2019 IEEE symposium on security and privacy (SP)* (IEEE), 1344–1361.

Peng, P., Yang, L., Song, L., and Wang, G. (2019). "Opening the blackbox of virustotal: analyzing online phishing scan engines," in *Proceedings of the internet measurement conference*, 478–485.

PhishTank (2024). PhishTank. Available at: https://phishtank.org/ (Accessed on March 20, 2024).

Queiroz, M. M., Telles, R., and Bonilla, S. H. (2020). Blockchain and supply chain management integration: a systematic review of the literature. *Supply chain Manag. An Int. J.* 25, 241–254. doi:10.1108/scm-03-2018-0143

Rao, R. S., Vaishnavi, T., and Pais, A. R. (2020). Catchphish: detection of phishing websites by inspecting urls. *J. Ambient Intell. Humaniz. Comput.* 11, 813–825. doi:10.1007/s12652-019-01311-4

Salem, A., Banescu, S., and Pretschner, A. (2021). Maat: automatically analyzing virustotal for accurate labeling and effective malware detection. *ACM Trans. Priv. Secur. (TOPS)* 24, 1–35. doi:10.1145/3465361

Shahrivari, V., Darabi, M. M., and Izadi, M. (2020). Phishing detection using machine learning techniques. *arXiv preprint arXiv:2009.11116*

Sheng, S., Wardman, B., Warner, G., Cranor, L., Hong, J., and Zhang, C. (2009). An empirical analysis of phishing blacklists

Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., et al. (2021). Towards crisp-ml (q): a machine learning process model with quality assurance methodology. *Mach. Learn. Knowl. Extr.* 3, 392–413. doi:10.3390/make3020020

Sun, Y., Chong, N., and Ochiai, H. (2022). "Federated phish bowl: lstm-based decentralized phishing email detection," in *2022 IEEE international conference on systems, man, and cybernetics (SMC)* (IEEE), 20–25.

Thota, A. R., Upadhyay, P., Kulkarni, S., Selvam, P., and Viswanathan, B. (2020). "Software wallet based secure participation in hyperledger fabric networks," in *2020 international conference on COMmunication systems and NETworkS (COMSNETS)* (IEEE), 1–6.

Tikhomirov, S. (2018). "Ethereum: state of knowledge and research perspectives," in *Foundations and practice of security: 10th international symposium, FPS 2017, nancy, France, october 23-25, 2017, revised selected papers 10* (Springer), 206–221.

Trad, F., and Chehab, A. (2024a). Large multimodal agents for accurate phishing detection with enhanced token optimization and cost reduction [online]. *arXiv.org*. Available at: https://arxiv.org/abs/2412.02301

Trad, F., and Chehab, A. (2024b). Prompt engineering or fine-tuning? a case study on phishing detection with large language models. *Mach. Learn. Knowl. Extr.* 6, 367–384. doi:10.3390/make6010018

Trad, F., and Chehab, A. (2024c). To ensemble or not: assessing majority voting strategies for phishing detection with large language models [online]. *arXiv.org*. Available at: https://arxiv.org/abs/2412.00166.

Varma, J. R. (2019). Blockchain in finance. *Vikalpa* 44, 1–11. doi:10.1177/0256090919839897

Varshney, G., Misra, M., and Atrey, P. K. (2016). A survey and classification of web phishing detection schemes. *Secur. Commun. Netw.* 9, 6266–6284. doi:10.1002/sec.1674

Vidyakeerthi, S., Nabeel, M., Elvitigala, C., and Keppitiyagama, C. (2022). "Phishchain: a decentralized and transparent system to blacklist phishing urls," in *Companion proceedings of the web conference 2022*, 286–289.

Vijayalakshmi, M., Mercy Shalinie, S., Yang, M. H., and U, R. M. (2020). Web phishing detection techniques: a survey on the state-of-the-art, taxonomy and future directions. *Iet Netw.* 9, 235–246. doi:10.1049/iet-net.2020.0078

Vranken, H. (2017). Sustainability of bitcoin and blockchains. *Curr. Opin. Environ. Sustain.* 28, 1–9. doi:10.1016/j.cosust.2017.04.011

Yang, P., Zhao, G., and Zeng, P. (2019). Phishing website detection based on multidimensional features driven by deep learning. *IEEE access* 7, 15196–15209. doi:10.1109/access.2019.2892066

Zhang, D., Chen, J., and Lu, X. (2021). "Blockchain phishing scam detection via multi-channel graph classification," in *Blockchain and trustworthy systems: third international conference, BlockSys 2021, guangzhou, China, august 5–6, 2021, revised selected papers 3* (Springer), 241–256.

Zhang, L., Xie, Y., Zheng, Y., Xue, W., Zheng, X., and Xu, X. (2020). The challenges and countermeasures of blockchain in finance and economics. *Syst. Res. Behav. Sci.* 37, 691–698. doi:10.1002/sres.2710