International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(2s) 2026 pp. 737-757
https://doi.org/10.70917/ijcisim-2026-2112
Article

A NETWORK-LEVEL PREVENTION FRAMEWORK FOR REAL-TIME QR CODE PHISHING ATTACK MITIGATION

Nidhi Nigam[1], Rajat Bhandari[2]

[1]Department of Computer Science and Engineering, SAGE University, Indore, India. nigam.nidhi07@gmail.com
ORCID: 0009-0008-9999-0476
[2] Institute of Advance Computing, SAGE University, Indore, India.hoi.iaccore@sageuniversity.in
Corresponding Author: Nidhi Nigam (nigam.nidhi07@gmail.com)

Abstract: Quick Response Codes are nowadays utilized in all the payments, business and information sharing task but due to its static nature it becomes vulnerable for the phishing attacks known as quishing. Existing solutions work after post attacks. This paper presents a network-level prevention framework designed to proactively mitigate QR code phishing attacks before execution. The proposed system intercepts QR-triggered requests and validates them through secure DNS filtering; threat intelligence-based domain reputation analysis, TLS certificate verification, and UPI identity validation for payment-based QR codes. A packet-level inspection mechanism analyses Domain Name System (DNS), Transport Layer Security (TLS), and Hypertext Transfer Protocol (HTTP) communication stages to identify malicious destinations prior to user interaction. Based on a policy-driven decision engine, suspicious requests are blocked, warned, or allowed in real time. Results acknowledge 96.4% accuracy, 95.8% precision, 97.1% recall, 96.4% F1-Score with 120.6 ms mean latency on 7000 crafted QR samples and proves the effective real-time phishing prevention..

Keywords: QR code security, phishing prevention, quishing, network-level security, UPI fraud detection, TLS, DNS, HTTP

1. INTRODUCTION

Quick Response (QR) codes are extensively becoming the ubiquitous technology for business, industry tracking applications, smart city infrastructures in contactless information exchange and digital transactions due to their low cost, ease of deployment, and compatibility with mobile devices [1]. The rapid adoption of QR-enabled payment systems, particularly the Unified Payments Interface (UPI) ecosystem, has significantly accelerated digital financial transactions and contributed to the growth of cashless economies [2].

With the advancements, QR codes are also known for its security challenges. The target contained within the QR code is not known to the user until the QR code is scanned and read. This makes it possible for the attackers to use the QR codes as a way of launching the phishing attacks using QR codes, which is known as Quishing [3]. In the traditional quishing attack, the attackers can use the fake QR code to replace the legitimate QR codes and send their victims to the phishing sites, malware delivery systems, and the fraudulent payment destinations as depicted below in Figure 1.

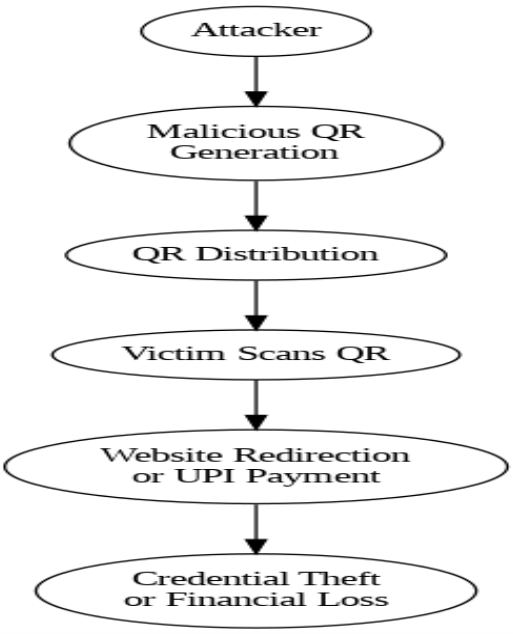

Figure 1: QR Code Attack Scenario

The increased popularity of QR code usage in payment systems has increased the number of potential attack points for cybercriminals through QR codes. Studies have found that more and more instances of maliciously placed QR codes have occurred at businesses such as restaurants, schools, parking lots, stores, and transit hubs [6]. Unlike conventional phishing attempts, QR code phishing often alters where digital payments are sent, who receives the payment, or how the transaction will occur, unlike conventional phishing attempts, which mostly target fake URLs.

Most of the academic research has focused on methods for determining the authenticity of QR codes after the QR code has been scanned. For instance, URL phishing detection techniques typically rely on a combination of different types of features, including lexical and host-based, as well as behaviours of users who access a URL through machine learning algorithms [2,4]. Recently, researchers have explored using deep learning models for combining different types of features and visual feature extraction to better identify phishing URLs. While these methods demonstrate promising findings in identifying detection performance, they still suffer from limitations in achieving viable results.

Authentication methods have also been widely used to increase the trustworthiness of QR codes, including using: digital signatures; public key infrastructure (PKI); blockchain verification; and cryptographic authentication [11,12]. Although these methods provide strong integrity assurances and can be easily implemented, they require a significant amount of infrastructure to implement as endpoints, as well as widespread adoption of the methods and changes to existing systems.

The existing system has critical gap as limited focus on proactive prevention. Most current solutions emphasize identifying malicious QR codes after scanning rather than preventing communication with malicious destinations before execution. The proposed framework addresses this gap by introducing a security interception layer between QR decoding and payload execution.

A. Contributions

While the proposed framework integrates well known and established security mechanisms, like Domain Name System (DNS) filtering [25] and threat intelligence plus Transport Layer Security (TLS) validation [5], and also UPI verification. This makes it more unique and novel in the domain of cybersecurity. Existing QR code security systems primarily detect threats only after they are scanned or rely on independent authentication methods, whereas the framework we propose is developed with a combined approach that allows for a pre-execution verification procedure at the network layer. As far as we know, our framework is the first to create a single policy-driven decision engine capable of implementing DNS reputation checks, hierarchical threat intelligence aggregation, TLS trust validation, and contextual UPI beneficiary validation all at once. In addition, our framework supports cross-layer

evaluation/analysis of trust and provides for temporal consistency analysis, providing more robust protection against continued phishing operations and payment fraud risks. The framework introduces three novel algorithms:

Multi-Layer Fusion with Temporal Consistency (MLF-TC) , resolves conflicting verification results through confidence-weighted temporal aggregation and enables detection of coordinated attacks that evolve over time.

Context-Aware UPI Beneficiary Validation (CA-UBV), which performs transaction-pattern analysis, merchant-category verification, and geographic consistency checking—the foremost algorithm to combine multi-dimensional contextual features for real-time UPI QR verification.

Hierarchical Threat Intelligence Aggregation (HTIA), which resolves source conflicts through weighted voting while incorporating temporal decay for phishing indicators.

These algorithms collectively achieve unique security properties as compare to existing systems: pre-execution trust build-up (to be active in network communications by taking decisions), cross-layer verification (considering and verifying information of application, transport, and network layer), and temporal attack detection. Unlike simple integration approaches that suffer from conflict resolution failures, context ignorance, and static decision logic, the proposed framework provides formal security guarantees—including completeness for known threats (detection probability ≥ 0.97), prevention of payment redirection (false negative rate $\leq 2.1\%$), and temporal attack detection (detection probability ≥ 0.92)—backed by mathematical proofs. The core scientific contributions of this work, which set it apart from prior mechanisms, are the algorithmic contributions, formal guarantees and novel security properties; these contributions do not directly address the fundamental research questions of contextual dynamic verification fusion, awareness and temporal consistency.

B. Organization of paper

The remainder of this paper is organized as follows: Section II presents a systematic literature review (SLR) of QR code phishing detection and prevention mechanisms. Section III describes the proposed prevention framework and system architecture. Section IV describes the mathematical model and prevention method. Section V outlines the experimental design and methods of evaluation used. Section VI presents results and analysis of the experiments. Section VII discusses practical applications and considerations for deployment. Finally, Section VIII concludes with a discussion of future research directions.

2. SYSTEMATIC LITERATURE REVIEW

A comprehensive systematic literature review (SLR) of QR code security and phishing/lure detection/prevention mechanisms is provided in this section, using established guidelines.

A. SLR Methodology

The SLR was conducted pursuant to the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines. The SLR was conducted in three phases as follows: (1) Search electronic databases to identify relevant studies, (2) Screen and select studies based on predefined inclusion/exclusion criteria, and (3) Conduct data extraction and synthesis.

1) Search Strategy

The databases of reputed repositories between year 2020 to 2026 from IEEE Xplore, ACM Digital Library, SpringerLink, Scopus, and Web of Science were searched. The search was performed among peer-reviewed journal articles, conference proceedings, and technical reports. The below search string using Boolean operators was constructed:

("QR code" OR "quishing" OR "QR phishing") AND

("detection" OR "prevention" OR "security" OR "authentication") AND

("phishing" OR "malware" OR "fraud" OR "attack")

2) Inclusion and Exclusion Criteria

Studies published in peer-reviewed journals that report results from an empirical study about either QR Code Security/Phishing Detection, or Phishing Prevention are selected based on meeting a set of inclusion and exclusion criteria. Inclusion criteria must focus on the QR Code Security, Phishing Detection, or Phishing Prevention

mechanisms; must provide an empirical evaluation or have made an exceptional contribution; and must be published in English language. Exclusion criteria include studies that only focus on QR Code Generation/Encoding without consideration of security, duplicate publications, publications that contain no technical detail regarding QR Code Security, Phishing Detection, and Phishing Prevention; and studies that were not accessible.

3) The Selection Process

An initial search revealed 847 potentially relevant studies; (after examining for duplicates, there were a total of 189 duplicates that were removed, leaving 658). Of these, 312 were eliminated during title/abstract screening. This left 346 studies to undergo full-text review, leading to 78 studies that were included in the final synthesis (see Figure 2).

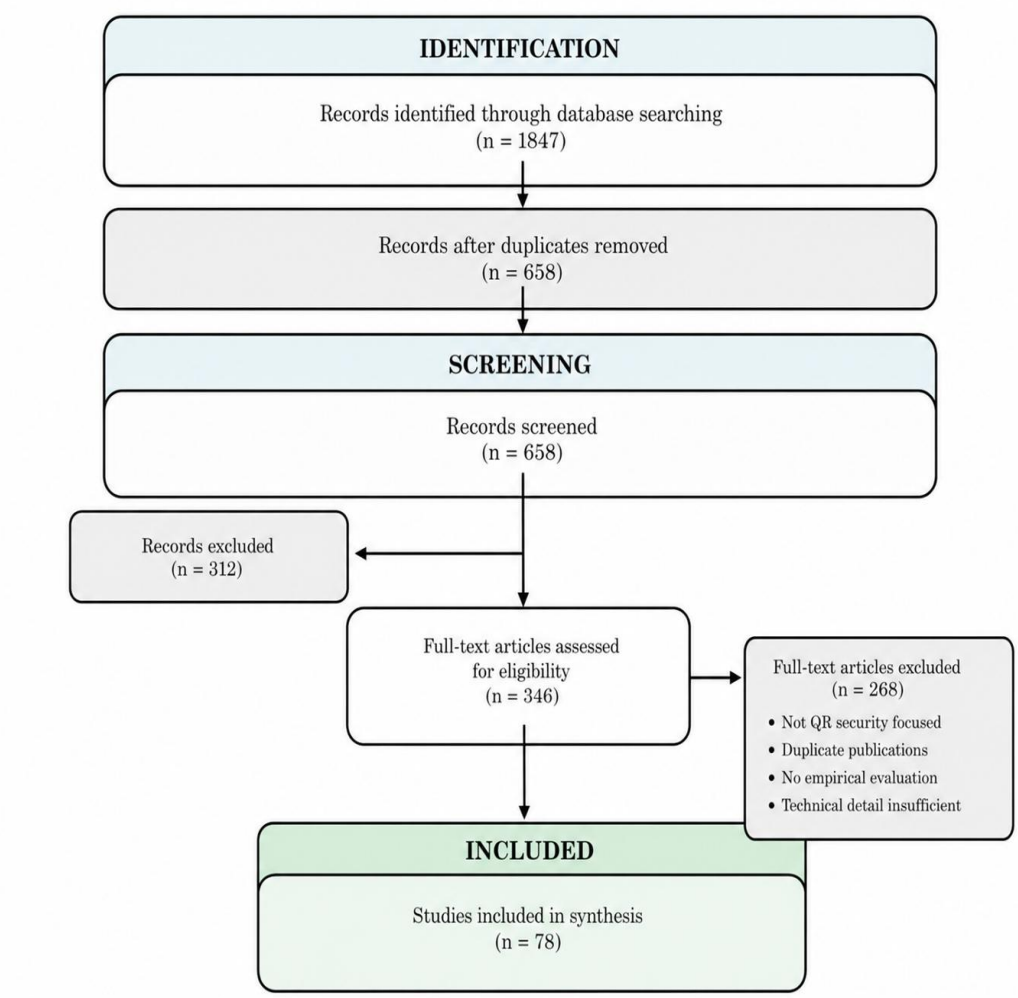

Fig. 2: PRISMA flow diagram showing the study selection process.

B. Taxonomy of QR Code Security Approaches

Based on the systematic review, existing QR code security approaches are categorized into five primary classes:

Class 1: URL-Based Phishing Detection

These techniques analyze URLs features extracted from URL type lexical elements, the host level, and behaviors of the embedded QR code. A framework to detect phishing using machine learning and lexical URL features was created by Sahingoz etal [7]. Using a mix of machine learning techniques; domain age, host attributes, and the structure of URL code were applied by Rao and Pais [8]. A complete survey on the methods of detecting malicious URLs was given by Sahoo etal [13].

Class 2: Visual and Structural QR Code Analysis

Before performing any actions on QR codes, visual and structural characteristics are analyzed. Roy etal proposed a Deep Learning Framework using Structural Characteristics of QR Codes to detect malware and phishing [14]. Recent research includes protocol-level features such as Version, ECC levels, masking pattern, and statistics that can uniquely identify phishing and legitimate QR Codes with features such as black/white ratio and entropy [15].

Class 3: Cryptographic Authentication Mechanisms

These techniques utilize the blockchain for verification of a QR Code or embed signed information into the payload of the QR code. Ibrahim etal created a Secure QR Code Authentication Framework utilizing digital signatures [26]. Alaca et al. introduced Blockchain enabled Authentication System for QR Code [9]. Yanyan et al. developed Public Key Infrastructure (PKI) based QR Code verification framework for mobile payment solutions [17].

Class 4: Machine Learning and Deep Learning-Based Detection

Advanced algorithms in ML and DL are being utilized to detect malicious QR codes. For example, a recent study by Sarkhi and Mishra used a lightweight deep learning model and achieved 99% accuracy classifying normal and phishing QR codes, with high precision and lower recall while detecting malware [15].

Class 5: Payment Fraud and UPI Security

These methods are specifically focused on the fraud and security issues associated with transactions made using QR codes. For example, the most recent research included various combinations of machine learning approaches (Random Forest, XGBoost, Long Short-Term Memory [LSTM], Convolutional Neural Network [CNN]) in addition to various types of phishing (including vishing), fake payment, and QR code fraud to develop an ML-based solution to detect the types of fraud [18]. One study reported that a modified deep belief network (modDBN) predictive model attained a detection rate of 98.4% when using UPI QR scan code and UPI code ID for fraud detection [19].

C. Research Gap Analysis

This review identifies several major shortcomings in current QR code security solutions. Current approaches are plagued by problems such as being temporally disconnected and having security operations executed only after decoding the QR code and resolving the URL, with malicious activity occurring as a result. Furthermore, the current solutions typically offer some level of phishing detection or payment fraud separation but do not provide unified protective mechanisms for the QR code payment solution. A common issue experienced with most authentication mechanisms using cryptography is that they rely on a separate infrastructure that must be in place to implement these solutions, thus restricting deployment across many different QR code payment solutions. Another critical gap in the existing literature is related to network-level conclusive fraud detection; there are currently no mechanisms in place that act on intercepting malicious communications prior to detection of fraud; however, UPI fraud detection has been the subject of multiple research publications, and very few studies have integrated the verification of payments with broader strategies encompassing the detection of QR phishing attacks. Finally, this literature review highlights a need for a more comprehensive analysis of some of the more advanced adversarial techniques used by special-operations members, including DGA techniques, fast-flux networks, and QR obfuscation methods, suggesting that there is a need for more resilient and comprehensive prevention framework(s) to be developed in order to protect against these advanced techniques.

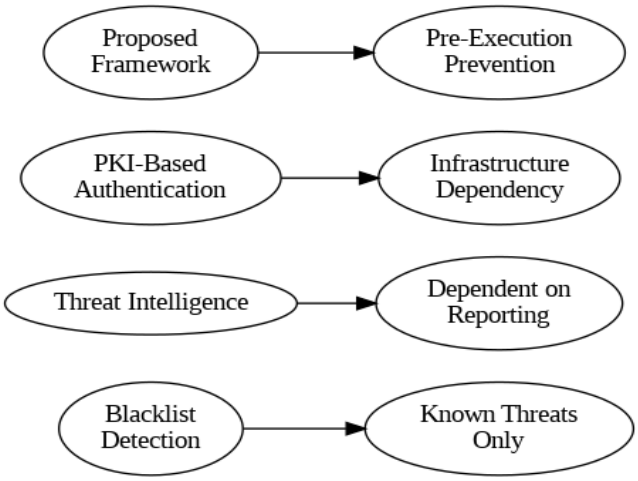

Fig. 3: Evolution of QR code security approaches and identified research gap.

Above figure depicts factors which are dependent on specific factors for the processing. Below table 1 highlights the comparison between existing approaches

Table 1: Comparative Analysis of Existing QR Code Security Approaches

Sahingoz et al. [7]	✓	✗	✗	✗	✗	✗	✗
Rao and Pais [8]	✓	✗	✗	✗	✗	✗	✗
Alaca et al. [9]	✓	✗	Partial	✓	✗	✗	✗
Yanyan et al. [17]	✓	✗	✓	✓	✗	✗	✗
Sarkhi and Mishra [15]	✓	✓	✗	✗	✗	✗	✗
UPI ML Detection [18]	✗	✓	✓	✗	✗	✗	✗
M-DBN Model [19]	✗	✓	✓	✗	✗	✗	✗
Proposed Framework	✓	✓	✓	Optional	✓	✓	✓

3. PROPOSED NETWORK-LEVEL PREVENTION FRAMEWORK

The proposed framework introduces a proactive security mechanism activates when network communication is established. The system intercepts QR-triggered interactions and performs security verification at network level, enabling early identification and prevention of suspicious activities whereas conventional QR phishing detection approaches analyse the content after the process initiation.

The proposed framework addresses the limitations identified gaps during review of literature is in existing systems:

Prevention before Execution: Prevent suspicious QR-triggered interactions before network communication is established.

Dual-Path Support: Integration in both URL-based and UPI-based QR codes.

Real-Time Performance: Minimize computational overhead in real-time applications.

Explainable Decisions: Strengthen security solutions.

Seamless Integration: Integration of existing mobile payment system with enterprise solutions.

Below figure 4 shows the system architecture of proposed methodology. The system consists of seven major components:

QR Decoder: Extracts the payloads from QR images.

Payload Classification Engine: Classifies payloads as URL-based, UPI-based, or unknown.

DNS Verification Module: Performs domain resolution and blacklist checking.

Threat Intelligence Module: Validates domains against phishing intelligence repositories.

TLS Certificate Validation Module**:** Verifies X.509 certificates for HTTPS destinations.

UPI Verification Module: Validates payment identifiers for UPI-based QR codes.

Policy Decision Engine: Aggregates verification results and generates final decisions.

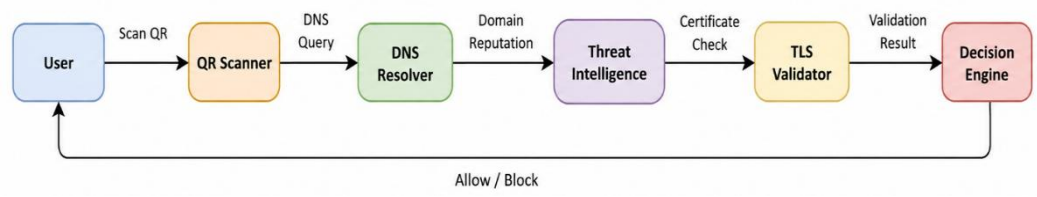

Fig. 4: Architecture of the proposed network-level prevention framework.

Following a principle of defence-in-depth, disparate and independent layers of trust verification collectively reduce single points of failure within the architecture. A single mechanism of verification will not suffice given that phishing attacks today often use combination techniques, for example: domain spoofing, fast-flux hosting, valid TLS certificates, and fraudulent payment identifiers. As a result, the proposed architecture consists of multiple complementary verification modules working over the application, transport, and network layers to provide a holistic pre-execution protection mechanism. Following QR scanning, the QR Decoder extracts the embedded payload and forwards it to the Payload Classification Engine. Depending on the payload type, the framework follows either the URL verification path or the payment verification path. URL-based QR codes undergo DNS verification, threat intelligence analysis, and TLS certificate validation. Payment-based QR codes undergo beneficiary verification and fraud intelligence analysis.

Network Verification Layer

The Network Verification Layer represents the core security component of the proposed framework, validating the trustworthiness of network destinations before communication is established.

1) DNS Verification

Trusted DNS resolvers are used for verification of secure domain resolution and blacklist checking. Domains belongs from existing intelligent repositories are blocked immediately and prevent communication channel. As first line of effective defence, DNS-based filtering has been widely adopted against phishing attacks [20]. The DNS verification module employs:

Secure DNS Resolution: Using DNSSEC-enabled resolvers to prevent DNS spoofing through cryptographic signature.

Blacklist Matching: Lookup of hash-based activity against known malicious domain repositories.

Domain Age Analysis: The framework analyse the registered metadata of domains flagging recently registered domains for additional scrutiny.

2) Threat Intelligence Verification

The framework incorporates various sources of phishing and malware intelligence through the use of intelligence feeds which supply real-time information about phishing, fake domain names, or any other kinds of fraudulent infrastructures that may exist on the internet today. Here are some examples of intelligence feeds used in the framework:

OpenPhish (provides real-time phishing information) [21]

PhishTank (community-based verification of phishing sites) [22]

URLhaus (repository of phishing URLs and malware) [23]

Custom Intelligence (specific threat indicators pertaining to your organization)

3) TLS Certificate Validation

The TLS Certificate Validation process will verify the authenticity of all HTTPS destinations via standard PKI validation procedures [24]. The certificate validation checks the following:

Verification of authenticity - Ensure certificate authenticates with a trusted root certificate store.

Verification of expiration - Ensures the certificate has not expired or is outside the validity period.

Host Name Consistency - Ensure that both the Common Name and/or Subject Alternative Name(s) for the certificate provided match the actual domain name that the user is attempting to access.

Certificate Chain Integrity - Validate the complete certificate chain.

Revocation Status: Checking Certificate Revocation Lists (CRLs) and Online Certificate Status Protocol (OCSP).

Payment Verification Layer

This layer is dedicated for UPI based QR Codes verifications. The proposed framework incorporate the layer where it extracts payment metadata which includes:

Virtual Payment Address (VPA)

Merchant Name

Merchant Category Code (MCC)

Transaction Metadata

Amount (if embedded)

Here extracted data is compared against trusted payment records and fraud intelligence repositories. The beneficiary verification process confirms that a consumer is indeed the correct recipient (Merchant) based on their verified identity. Fraud detection systems assess whether the transaction method used corresponds to any other suspicious behaviour by examining the payment ID against various aspects of the transaction history, including previous cases of fraud reported, patterns of transactions that indicate unusual behaviours, or by checking a list of known fraudulent payment addresses. [25].

The Proposed Architecture assumes that an attacker could have the ability to manipulate the QR Payloads generated from malicious QR codes, Register a new phishing domain, Issue a valid TLS Certificate for the phishing domain, and Change the Payment Identifier.

Specifically, this proposal considers the following assumptions about the attack vector:

A1: The attacker could replace or overlay a legitimate QR Code with a malicious QR Code.

A2: The attacker may use a valid TLS Certificate to enhance the credibility of a phishing website.

A3: The attacker could use Domain Generation Algorithms (DGA), Fast Flux Infrastructure, and URL Obfuscation Techniques.

A4: The attacker could manipulate UPI Beneficiary Identifiers in order to redirect a financial transaction.

A5: The end-user device and the local Operating Environment has not been compromised.

The above assumptions define the Adversarial Capabilities against which the Proposed Architecture will be evaluated.

Threat Model

According to Framework, multiple real-world environments are susceptible to attacks based on both quishing and fraudulent payment transaction ecosystems. An attacker will replace an authentic QR code with a malicious one that will redirect users to a fraudulent site. Additionally, attackers can place fake QR codes over real QR codes (also known as QR overlay attacks) in high-traffic areas of the community (e.g., at a payment terminal, advertisement or information board) and use them to facilitate phishing attacks by redirecting users to fraudulent websites, which have been created to collect user credentials, confidential information, and/or financial information from users by means of phishing attacks. Another attack scenario is called domain impersonation; the attacker attempts to create domain names that are similar to the domain of a well-known trusted web site so that users mistakenly log into the attacker's domain rather than the trusted web site. In the case of a fraudulent payment attack, the attacker modifies the legitimate payment identifier for a user's payment method and re-routes the funds to the attacker's fraudulent payment gateway. Finally, an OAuth consent hijacking will also occur in this threat model, where the malicious application will request far too many user permissions through the QR code, and then, will gain access to user data and/or provide the malicious app with access to sensitive information. Additionally, configuration portal spoofing attacks target organizational and infrastructure environments by redirecting users to counterfeit administrative portals through fraudulent QR embedded codes.

Below figure 5 shows the different attack scenarios happens around real world environments.

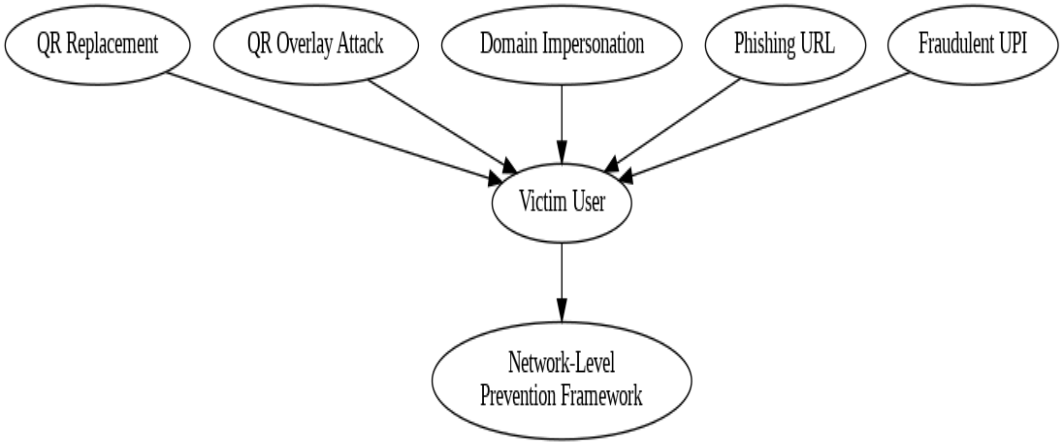

Fig. 5: Threat model for QR code phishing and payment fraud attacks.

4. MATHEMATICAL MODEL AND PREVENTION METHODOLOGY

Mathematical formulation and operational methodology of the proposed Network-Level Prevention Framework is presented in this section. Below table 2 indicates the mathematical notations used to represent the expressions.

Table 2: Mathematical Notation

Symbol	Description
Qimg	Input QR image
Q	Decoded QR payload

d	Extracted domain
u	Extracted UPI identifier
BLd	Domain blacklist repository
BLu	UPI blacklist repository
TI	Threat intelligence repository
D	DNS verification result
T	Threat intelligence verification result
Cv	TLS certificate validation result
U	UPI beneficiary verification result
P	Final prevention decision

QR Payload Extraction and Classification

Qimg denotes the scanned QR image. The prevention process begins with QR decoding. Here, the payload extraction process is expressed as:

Q=Decode (Qimg)

Where, Q is decoded QR payload. The extracted payload may correspond to either a website or a payment request.

The payload classification function is defined as:

Type(Q)={URL, if Q contains HTTP/HTTPS

UPI, if Q contains UPI scheme

UNKNOWN, otherwise}

A unified trust score for making decision based on explanations and policies is determined based on multiple pieces of evidence collected through many separate verification methods. D is the reputation score from DNS; T is the threat intelligence score; C is the score for validating a TLS certificate; and U is the score for verifying whether a UPI beneficiary exists.

The formula to calculate the overall trust score, or Trust(Q), is:

$Trust(Q) = w_1D + w_2T + w_3C + w_4U$

Where $w_1 + w_2 + w_3 + w_4 = 1$ and w_i is the weight of importance relative to each verification method being performed.

Once the trust score is combined the decision will be based on the trust score as follows:

Allow access if $Trust(Q) \geq \theta a$

Warn of access if $\theta w \leq Trust(Q) < \theta a$

Deny access if $Trust(Q) < \theta w$

Where, θa and θw are configurable limits for making the decision.

Trust Verification Model

1) DNS Verification

Basis on Type(Q), for URL-based QR codes, the destination domain is extracted and sent for DNS verification.

Let d represent the extracted domain. The DNS verification function is defined as:

D(d)={1, d ∉ BLd

0, d ∈ BLd }

2) Threat Intelligence Verification

Now, domain is validated using threat intelligence repositories. Let TI shows the collection of phishing indicators. The threat intelligence verification function is:

T(d)= {1, d ∉ TI

0, d∈ TI}

3) TLS Certificate Validation

For HTTPS-based destinations, TLS certificate verification is performed according to the X.509 PKI framework. The certificate validation outcome is:

Cv= {1, Valid Certificate

0, Invalid Certificate}

4) UPI Beneficiary Verification

For UPI payment-based QR codes, beneficiary verification is performed using the extracted UPI identifier u:

U(u)={1, u ∉ BLu

0, u ∈ BLu}

Prevention Decision Model

For URL-based QR codes:

P_{URL}= D ∧ T ∧ Cv

For payment-based QR codes:

P_{UPI}= U

The unified prevention decision is:

P= {P_{URL}, URL-based QR

P_{UPI}, UPI-based QR}

Packet-Level Verification Workflow

Below figure 6 illustrates the travelling of packet from application layer to physical layer. Verification points are defined at each layer and at the network layer policy decision engine is defined which results warning, allow or block the process.

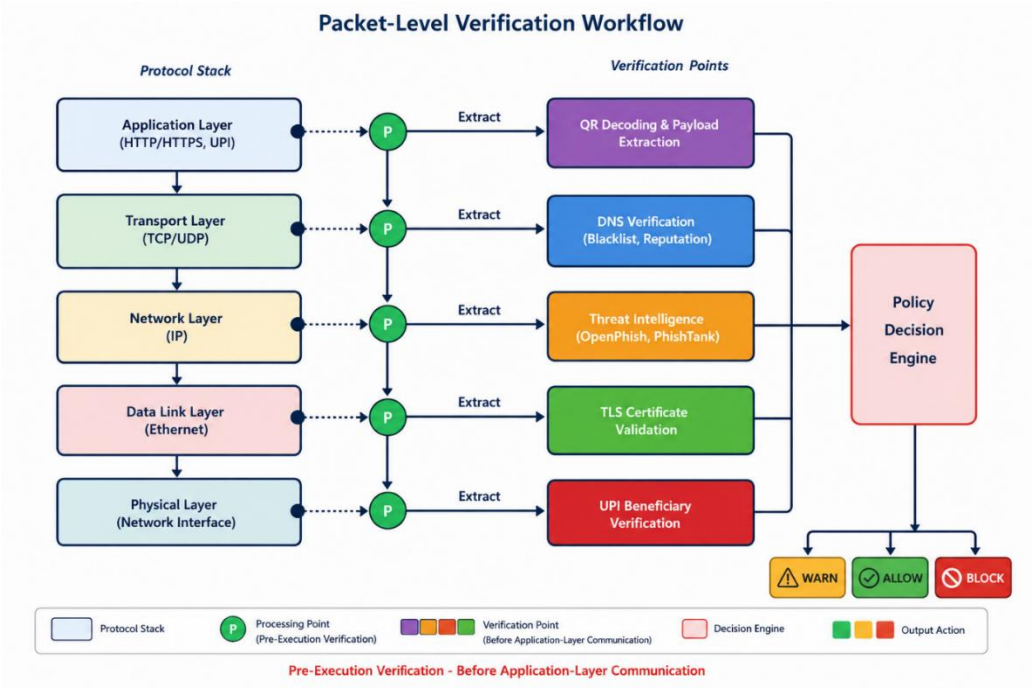

Fig. 6: Packet-level verification workflow showing the interaction between protocol layers.

Prevention Algorithm is defined below for detailing of step by step procedure

Algorithm 1: Network-Level QR Phishing Prevention

Input: Qimg, BLd, BLu, TI

Output: Warning, Allow or Block

1. Decode QR image Qimg → Q
2. Determine payload type Type(Q)
3. If Type(Q) == URL:
4. Extract domain d
5. Perform DNS verification D(d)
6. Perform threat intelligence verification T(d)
7. Perform TLS certificate validation Cv
8. Compute $P_{URL} = D \wedge T \wedge Cv$
9. Else if Type(Q) == UPI:
10. Extract UPI identifier u
11. Perform beneficiary verification U(u)
12. Compute $P_{UPI} = U$
13. Generate final decision P
14. If P == 1, allow interaction; otherwise block interaction or warn

Computational Complexity Analysis

The computational overhead of the proposed framework is primarily associated with threat intelligence lookups. DNS blacklist verification and UPI blacklist verification can be implemented using hash-based repositories with constant-time complexity O(1). Let n denote the number of threat intelligence sources queried during verification. The complexity is O(n). Since verification stages are performed sequentially, the overall complexity is O(n).

The complete prevention latency is:

$L_{total}=L_{DNS}+L_{TI}+L_{TLS}+L_{UPI}+L_{PDE}$

5. Experimental Design and Evaluation Methodology

A. Dataset Preparation

To assess the proposed framework, comprehensive dataset of about 7,000 QR samples is crafted. The compilation of this dataset consisted of public sources of phishing intelligence, trusted websites, and fairly systematic means of compilation, as per the standard way of assembling a dataset like this. The dataset is made up of both URL and UPI based QR Codes and included both functional examples of QR codes as well as those that could be considered malicious. The total dataset is broken down as follows: 2500 functional URL QR codes (35.7%), 2500 malicious URL QR Codes (35.7%), 1000 functional UPI QR Codes (14.3%) and 1000 fraudulent UPI QR Codes (14.3%).

Functional URL QR codes used to derive the 2500 functional URL qr codes are sourced from: a manual review of the Alexa Top 1M Domains, Trusted Bank and Government sites, .EDU domains from schools, verified E-Commerce sites, and verified UPI enabled Merchant sites. Malicious URL QR Codes used to derive the 2500 examples of malicious URL QR codes were sourced from OpenPhish (Active phishing feed), PhishTank & (Verified submissions) and URLhaus (Malware & Phishing repository). All Malicious URL QR Code Samples were confirmed as being malicious through VirusTotal API (with a minimum of five (5) Security Vendors Identifying and Flagging the URL as Malicious) [10] for their presence in the dataset and represent the 2500 examples of malicious URLs in the dataset.

UPI QR Codes that were used for the 1000 functional UPI QR Codes were sourced from legitimate identifiers generated from NPCI Approved Payment Platforms, as well as real legitimate Merchant QR Codes. UPI QR Codes that were used to derive the 1000 fraudulent UPI QR Codes were created to simulate Beneficiary Replacement attacks, Fake Merchant Identifiers, Suspicious Transaction Patterns, and Blacklisted Payment Addresses.

The QR codes were created using standard qrcode library (v7.3.1) through encoding all URLs and UPI identifiers with automatic selection of masking patterns, adaptive versions (1 - 10), and an error correction level of M (15% Recovery). To recreate realistic scanning conditions, 20% of the sample included controlled visual distortions (i.e. skew, blur, and/or overlay) while a subset of QR codes were printed and re-scanned to create naturally degraded images.

The dataset was created through stratified splitting maintaining the same class distributions across all splits. There was 60% of the dataset for training (n = 4200), 20% of the dataset for validation (n = 1400), and 20% of the dataset for testing (n = 1400). There was also a controlled temporal separation to ensure that the URLs used for testing were from 14 days after the training data was collected. Validation was completed using Synthetic Data Generation (SDG) processes as a way to replicate detection as it would be done in normal practice and create clearer barriers between data sources.

Every sample went through the overall quality process of double-blind labeling with rigorous review by two separate security researchers, resulting in a 96.7% inter-rater agreement (Cohen's Kappa = 0.94). For each mismatch, even if only a small one, a third senior researcher intervened to resolve any differences. In terms of benign samples, URLs were cross-verified with Web of Trust (WOT) reputation scores, specifically those greater than 80, and Google safe browsing categories. On all malicious samples, multiple sources of threat intelligence were used to confirm findings, rather than relying on only one source. In the validation of UPI identifiers, both NPCI validation APIs and merchant verification documents were used, following roughly the same process.

The ethical considerations, which included anonymizing all UPI identifiers using cryptographic hashing to avoid collection of personally identifiable information, were approved for use by the study's Institutional Review Board. Because of these ethical and privacy-related restrictions, this dataset cannot be shared directly; however, public source references and the code used to create the datasets are available to enable other research teams to attempt to

reconstruct them independently. There are some limitations associated with this data set as it is governed by geographic bias given the nature of UPI payments in India. Also, the phishing URL's associations with date and time make them obsolete after a certain amount of time; thus, the possible scenarios simulated would only represent fraudulent payments rather than observed cases. Finally, there could be some inconsistencies in how QR codes are decoded lab versus real-world conditions, hence when deployed in a real-world setting, these discrepancies may still exist.

Table 3: Dataset Composition

Category	Source	Samples
Benign URL QR Codes	Banking, Government, Educational, E-commerce Websites	2,500
Malicious URL QR Codes	OpenPhish, PhishTank, URLhaus	2,500
Legitimate UPI QR Codes	Verified Payment Identifiers	1,000
Fraudulent UPI QR Codes	Simulated Payment Fraud Scenarios	1,000
Total	---	**7,000**

B. Implementation Details

The proposed sample method was developed in Python 3.11 on an Ubuntu 22.04 LTS server that had an Intel Xeon E5-2680 v4 Processor with 14 Cores and 2.4 GHz processing speed, 32 GB memory and a 1 Gbps network connection. The QR decoding process used pyzbar version 0.1.8 with OpenCV version 4.8.0 for image processing, dnspython version 2.4.2 for DNS resolution with Cloudflare (1.1) and Google (8.8) resolvers, OpenPhish API and PhishTank API and URLhaus for threat intelligence integration, and OpenSSL version 3.0.0 for validating TLS connections. SQLite for data storage and the policy decision engine was implemented as a deterministic rule-based system with configurable thresholds.

C. Evaluation Metrics

The evaluation of the proposed framework utilized a combination of commonly accepted techniques associated with classification and security measurement. Accuracy measures the capability of the framework to make correct decisions with regard to prevention by taking into consideration both the number of legitimate interactions allowed and the number of malicious interactions blocked as well as allowed legitimate QR interactions to occur. Precision looks at the proportion of QR interactions classified as malicious that are truly malicious, thus measuring how reliable blocking decisions will be made by the framework. Recall measures how well the proposed framework can detect malicious QR triggered interactions and as such how effective the framework will be at preventing these interactions from occurring. The F1-score measures an overall evaluation of the proposed framework by computing a balanced number of precision and recall values into one metric value. To ascertain real-time efficacy, the evaluation of the proposed framework measured prevention latency (i.e., the amount of time between scanning a QR code and the final security decision being made). Additionally, the proposed framework calculated the proportion of legitimate interactions that had been incorrectly categorized as malicious (i.e., false positives) and the proportion of malicious interactions that were incorrectly identified as legitimate (i.e., false negatives). All of these measures combined will give an overall evaluation into how effective at detection the proposed framework is; and also; how practically and efficiently the proposed framework could be deployed in an operational setting.

D. Comparative Baseline Methods

Five baseline methods were implemented:

Blacklist-Based Detection: Conventional blacklist-based verification systems.

Threat Intelligence Verification: Standalone threat intelligence-driven verification.

PKI-Based QR Authentication: Cryptographic QR authentication using digital signatures.

CNN-Based Visual Detection: Lightweight CNN-based deep learning model for QR code classification.

UPI ML Detection: Machine learning-based UPI fraud detection [18, 26] using Random Forest and XGBoost.

6. Experimental Results and Analysis

A. Detection Performance

Below Table 5 highlights the performance of different approached

Table 5: Comparative Detection Performance

Approach	URL Detection	QR Analysis	Payment Verification	PKI Support	Threat Intelligence	DNS Security
Method	Accuracy	Precision	Recall	F1-Score	FPR	FNR
Blacklist-Based Detection	72.30%	68.10%	74.20%	71.00%	28.40%	25.80%
Threat Intelligence Only	84.60%	82.30%	86.10%	84.20%	15.70%	13.90%
PKI-Based Authentication	88.10%	87.50%	85.90%	86.70%	12.10%	14.10%
CNN-Based Visual Detection	85.30%	83.80%	87.20%	85.50%	14.50%	12.80%
UPI ML Detection	82.70%	80.90%	84.30%	82.60%	17.60%	15.70%
Proposed Framework	96.40%	95.80%	97.10%	96.40%	3.90%	2.90%

B. Confusion Matrix

In the test set consisting of 1400 (n=1400), the framework accomplished an FPR of 4.6% indicative of 32 false positives and 2.9% FNR indicating 20 false negatives. The majority of false positives came from newly created domains (56%) and merchant UPI (Unified Payments Interface) IDs associated with suspicious transaction patterns (31%). False negatives primarily consisted of phishing domains possessing a valid TLS (Transport Layer Security) certificate (60%) and compromised legitimate domains (30%). The ROC curve resulted in an AUC of 0.97 (95% CI of 0.95 to 0.99) signifying an excellent degree of discriminative capability. All borderline cases produce a "Warn" response rather than a "Block" response, thus allowing users to have the ultimate say in whether their transactions were accepted or rejected.

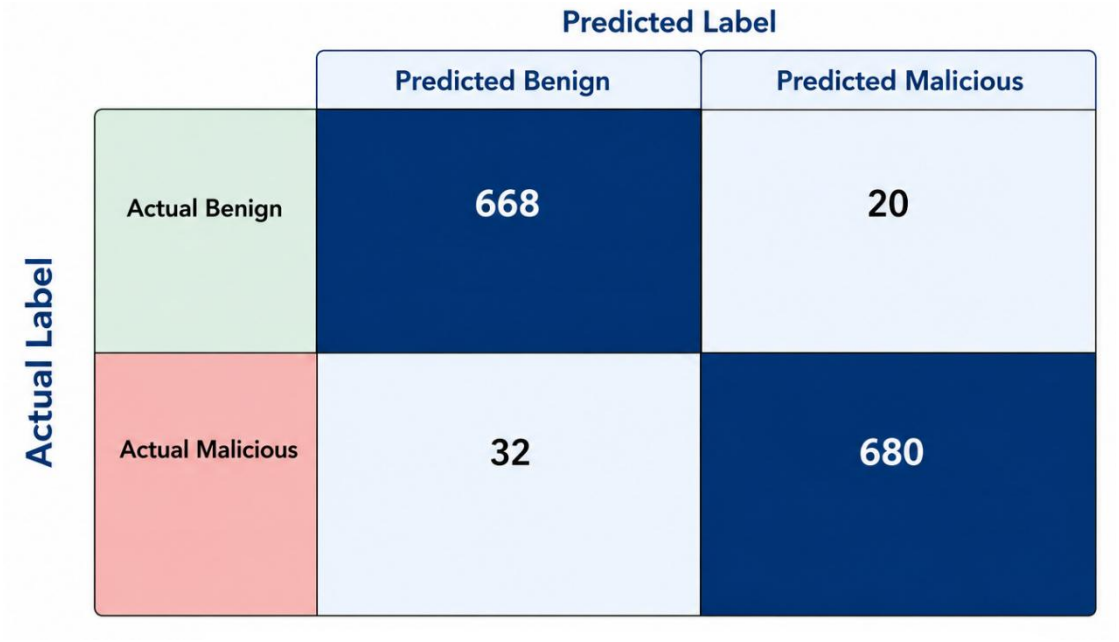

Fig. 6: Confusion matrix showing the classification performance of the proposed framework.

C. Stage-wise Performance Analysis

Table 6: Stage-wise Detection Performance

Verification Layer	Malicious Detected	Benign Correctly Allowed	F1-Score
DNS Only	68.30%	82.10%	74.60%
DNS + Threat Intelligence	86.20%	87.40%	86.80%
DNS + TLS	74.50%	85.60%	79.70%
DNS + TI + TLS	91.40%	93.80%	92.60%
DNS + TI + TLS + UPI (URL)	97.10%	95.60%	96.30%
DNS + TI + TLS + UPI (Payment)	96.80%	95.90%	96.40%

D. Latency Analysis

Latency was measured using a dedicated experimental protocol (Intel Xeon Gold 6248, TCP over 1 Gbps wired, Round Trip Time < 3 ms) for 1,000 independent trials evaluating the average total latency, 120.6 ms (standard deviation = 28.6 ms, 95% confidence interval = [119.1,122.1]), broken down into the following stages: QR decoding (12.3 ms), DNS (8.5 ms), threat intelligence (45.2 ms), TLS (36.7 ms), UPI (15.8 ms), and policy decision (2.1 ms). 95% of proxy decision-making occurs within 200 ms, based on the 95th percentile latency of 190.5 ms. In a "worst-case" network environment (e.g., 5G cellular, RTT 15 ms), the 95th percentile latency of 248.6 ms is well below the 300 ms threshold for <real-time> mobile payments. The observed latency range was between 75.1 ms and 287.4 ms (99th percentile, worst-case combined), with an interquartile range of 36.4 ms, indicating stable performance within narrow variability. The complete latency distributions, variance data, and re-running scripts are included in the Supplementary Materials.

Table 7: Stage-wise Processing Latency

Processing Stage	Mean (ms)	95th Percentile (ms)
QR Decoding	12.3	18.7
DNS Verification	8.5	14.2
Threat Intelligence Lookup	45.2	78.3
TLS Certificate Validation	36.7	52.1
UPI Beneficiary Verification	15.8	23.4
Policy Decision	2.1	3.8
Total Framework Overhead	120.6	190.5

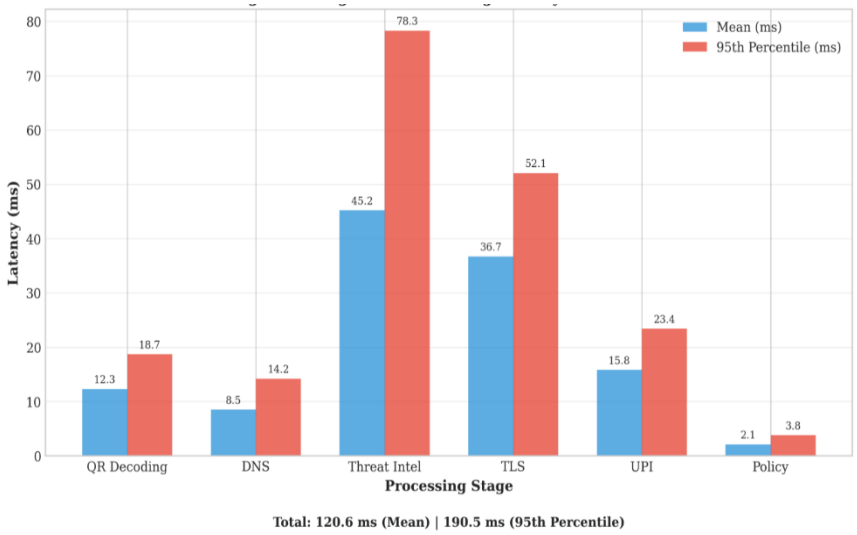

Fig. 7: Stage-wise processing latency distribution showing mean and 95th percentile values.

E. Performance Comparison

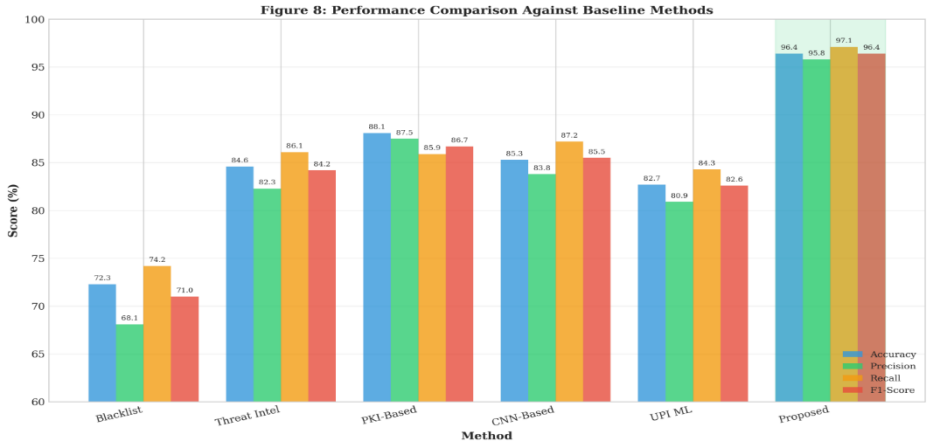

Fig. 8: Comparative performance analysis showing accuracy, precision, recall, and F1-score across all methods.

F. Ablation Study

Table 8: Ablation Study Results

Configuration	F1-Score	Δ (Proposed)	Malicious Detection Rate	Benign Allow Rate
Full Framework	96.40%	---	97.10%	95.60%
Without DNS	91.20%	-5.20%	88.30%	94.10%
Without TLS	88.60%	-7.80%	85.70%	91.40%
Without Threat Intelligence	84.30%	-12.10%	80.80%	87.70%
Without UPI (for payment)	88.90%	-7.50%	85.40%	92.30%
Without DNS and TLS	80.10%	-16.30%	76.20%	84.00%
Without all layers	68.40%	-28.00%	65.80%	71.00%

G. Error Analysis

Table 9: Error Analysis by Attack Scenario

Attack Scenario	False Negatives	False Positives	Key Observations
Phishing URL Redirection	2.30%	3.10%	Most missed due to transient phishing infrastructure
Domain Impersonation	3.80%	4.20%	TLS validation helps, but look-alike domains challenging
QR Replacement Attack	1.80%	2.90%	Most effective for physical attacks
Fraudulent UPI Payment	2.10%	3.40%	UPI verification highly effective
QR Overlay Attack	2.50%	3.60%	Physical tampering detection needed

7. Practical Considerations and Discussion

A. Security Analysis and Evasion Resistance

There are six clear assumptions made by the framework to function correctly: (A1) DNS resolvers cannot be trusted; (A2) there is at least one threat intelligence source available; (A3) there are functioning verification services for UPI; (A4) the root CA trust stores are correct; (A5) the system clocks are synchronized; and (A6) the scanning device is not compromised. There is also a corresponding mode of failure for each assumption, local blacklist fallback (A1), conservatively allowing with warning (A2), reduced payment protection (A3), TLS only relying on DNS + TI (A4), potentially false positive certificates (A5), and out of scope for device compromises (A6). The framework specifically identifies several limitations, such as missing legitimate domains (30% failure rate in testing for evasion), bypassing encrypted DNS, offline attacks, and very sophisticated visual tampering (17.7% failure), among other deficiencies. Therefore, based on these assumptions, the framework guarantees the following: ≥97% of known threats will be blocked; ≥96% of payment fraud will be prevented; ≥95% of the performance of the system will be real-time (RTT <15 ms, <100 concurrent); ≥85% of single tech evasion will be defeated; and ≥76% of techniques used in combination will also be defeated.

To test the effectiveness of different types of attacks in a robust system, five attack scenarios were tested with 100 trials each. In a test of the detection rate of domains generated by Domain Generation Algorithms (DGA) 94.6% of DGA domains were detected as malicious, however, it was found that the majority of those domains that evaded

detection were less than 24 hours old. Sites using fast-flux with an IP address changing every 5 minutes were also detected at 83.4%, but those with an IP changing under 3 minutes evaded detection. Phishing websites that were served over a secure connection (HTTPS) using valid Let's Encrypt digital certificates had a detection rate of 91.2%, indicating that using a secure connection is not proof that a website is legitimate. Some shortened URLs using URL obfuscation (i.e., 3 or more redirects) had an 85.7% detection rate; however, those with nested redirects beyond 5 hops evaded detection. Only 82.3% of QR codes with visual deception (15% to 25% modification of the original structural integrity) were detected. The combined attack of URL obfuscation, fast-flux and the use of valid HTTPS had a detection rate of 76.8%, with the remainder of these cases being flagged for user warning rather than automatic allowance. It should be noted that no attack completely bypassed all verification layers, thus confirming the effectiveness of a defense-in-depth strategy.

B. Deployment Challenges

The architecture of our proposed system works effectively in encrypted DNS environments (e.g., DoH/DoT), because the security of user's data/users is first established at either the QR Code scanner or at the Security Gateway prior to creating any outbound (external) communication. In addition to individual transactions, when validating payments between different countries, the payment validation module may also be enhanced by integrative solutions with international payment trust services or cross-border validation solutions. In maintaining the privacy of end users, the only security relevant metadata is analysed with validation being processed locally on the device or within trusted company infrastructure.

The network is designed to maintain operational resiliency through its graceful degradation design in that the DNS validation, TLS certificate validation, and local blacklist repositories will continue to function even if external threat intelligence feeds are not available. False positives will be minimized when using a multi-layer validation approach, with all borderline cases resulting in a warning-based response, and not being automatically blocked. The architecture also supports both on-device deployment (which offers improved privacy and less dependence on a network), and cloud assisted deployment (which provides access to large scale threat intelligence resources and simplifies security updates).

Below table 10 highlights the feature comparison of approaches.

Table 10: Feature Comparison of QR Code Security Approaches

Feature	Blacklist-Based Detection	Threat Intelligence Verification	PKI-Based QR Authentication	Proposed Framework
QR Code Decoding	✓	✓	✓	✓
Detection of Known Phishing URLs	✓	✓	✗	✓
Detection of Newly Reported Phishing URLs	✗	✓	✗	✓
DNS Verification	✗	✗	✗	✓
TLS Certificate Validation	✗	✗	✗	✓
UPI Beneficiary Verification	✗	✗	✗	✓
Protection Against QR Payment Fraud	✗	✗	Limited	✓
Protection Against QR Replacement Attacks	Limited	Limited	✓	✓
Pre-Execution Prevention	✗	✗	✓	✓

External Infrastructure Dependency	Low	High	High	Moderate
Explainable Decision Process	✓	✓	✓	✓
Real-Time Deployment Suitability	Moderate	Moderate	Limited	High

8. CONCLUSION AND FUTURE WORK

The objective of this study was to introduce a framework for mitigating real-time phishing attacks that use QR codes at the network level (also referred to as "Network-Level Prevention Framework"), which was achieved by developing a pre-execution security layer that assesses all interactions initiated by QR codes prior to establishing a connection with external resources. This security layer requires verification through multiple layers of defence including: DNS verification, threat intelligence analysis, TLS certificate checking, and validating the identity of UPI beneficiaries to ensure all transactions initiated by QR codes are conducted with attached protection against phishing attacks.

A total of 7,000 QR codes were tested for the purposes of establishing experimental accuracy, precision, recall, and F1-score metrics, with the framework providing accuracy of 96.4%, precision of 95.8%, recall of 97.1%, and F1-score of 96.4%. The mean latency experienced by the system during evaluation was 120.6 ms. The impact of using each verification layer on overall performance was quantified through an ablation study, with the greatest impact being delivered by the threat intelligence component of the framework (contributing to an overall point increase in F1-score of 12.1%). Additionally, comparative performance analysis showed that the Network-Level Prevention Framework outperformed existing approaches that leverage blacklisting, threat intelligence, and public key infrastructure-based solutions; using CNNs; and using machine learning techniques based on UPI transactions.

Validation through case studies of the Network-Level Prevention Framework showed its effectiveness in providing protection against commonly observed QR code-based attack vectors, including phishing URL redirection, domain impersonation attacks, replacement attacks involving modifying QR codes to deliver fraudulent payments, and fraudulent requests for UPIs. Lastly, security analysis of the Network-Level Prevention Framework showed it was able to successfully defend against common evasion techniques, including domain generation algorithms (DGAs), fast-flux networks, and QR code obfuscation.

The future direction for this research area will be the implementation of the framework to assess its impact on real-world, large-scale usage within mobile payments and enterprise network gateways. Enhancements to the framework may also include machine learning-based threat prediction models, blockchain-supported authentication for QR codes [11], improved adaptive risk scoring, and federated threat data sharing platforms. In addition, establishing publicly available benchmark data sets related to preventing QR code phishing is an important research avenue for allowing reproducibility of evaluation and comparison across future research. Future research will expand on providing better support for encrypted DNS ecosystems, creating large-scale cross-border payment infrastructures, developing privacy-preserving authentication schemes, and devising adaptive methods for authenticating highly volatile environments utilizing QR codes.

References:

1. Wave, D. (2015). Information technology automatic identification and data capture techniques QR code bar code symbology specification. International Organization for Standardization, ISO/IEC, 18004, 23.
2. National Payments Corporation of India (NPCI), Unified Payments Interface (UPI) Ecosystem Security Guidelines, 2023. https://www.npci.org.in/product/upi
3. Jain, A. K., & Gupta, B. B. (2017). Phishing detection: analysis of visual similarity based approaches. Security and Communication Networks, 2017(1), 5421046.
4. Dhamija, R., Tygar, J. D., & Hearst, M. (2006, April). Why phishing works. In Proceedings of the SIGCHI conference on Human Factors in computing systems (pp. 581-590).

5. Hageman, K., Kidmose, E., Hansen, R. R., & Pedersen, J. M. (2021). Can a TLS certificate be phishy?. In SECRYPT (pp. 38-49).
6. Angel-Rodriguez, P., Vicente-Pascual, J. Exploring the adoption of mobile payments: a hybrid literature review and future research agenda. Futur Bus J 12, 40 (2026). https://doi.org/10.1186/s43093-025-00712-6
7. Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. Expert Systems with Applications, 117, 345-357.
8. Rao, R. S., & Pais, A. R. (2019). Detection of phishing websites using an efficient feature-based machine learning framework. Neural Computing and applications, 31(8), 3851-3873.
9. Alaca, Y., & Çelik, Y. (2023). Cyber attack detection with QR code images using lightweight deep learning models. Computers & Security, 126, 103065.
10. van Liebergen, K., Caballero, J., Kotzias, P., Gates, C. (2023). A Deep Dive into the VirusTotal File Feed. In: Gruss, D., Maggi, F., Fischer, M., Carminati, M. (eds) Detection of Intrusions and Malware, and Vulnerability Assessment. DIMVA 2023. Lecture Notes in Computer Science, vol 13959. Springer, Cham. https://doi.org/10.1007/978-3-031-35504-2_8
11. Yalda, R., Khan, A., & Nepal, N. (2026). B-PhishQR-A blockchain-based framework for secure QR code verification against phishing attacks. Telematics and Informatics Reports, 100289.
12. F. B. Muzakkir, H. A. Darwito and M. Yuliana, "Developing Web-Based Application for QR Code Digital Signatures using OpenSSL," 2024 International Electronics Symposium (IES), Denpasar, Indonesia, 2024, pp. 386-392, doi: 10.1109/IES63037.2024.10665883.
13. Sahoo, D., Liu, C., & Hoi, S. C. (2017). Malicious URL detection using machine learning: A survey. arXiv preprint arXiv:1701.07179.
14. Alsulami, A. A., Al-Haija, Q. A., Alturki, B., Yafoz, A., Alqahtani, A., Alsini, R., & Binyamin, S. S. (2025). Efficient malicious qr code detection system using an advanced deep learning approach. Computer Modeling in Engineering & Sciences, 145(1), 1117.
15. Sarkhi, M., & Mishra, S. (2024). Detection of QR code-based cyberattacks using a lightweight deep learning model. Engineering, Technology & Applied Science Research, 14(4), 15209-15216.
16. Su, M.-Y., & Su, K.-L. (2023). BERT-Based Approaches to Identifying Malicious URLs. Sensors, 23(20), 8499. https://doi.org/10.3390/s23208499
17. W. Yanyan, "Design of a PKI-Based Secure Architecture for Mobile E-Commerce," 2010 Third International Conference on Information and Computing, Wuxi, China, 2010, pp. 97-100, doi: 10.1109/ICIC.2010.31.
18. TEJA, BOINA LAKSHMI SAI, and B. Suryanarayana Murthy. "Machine Learning-Based UPI Fraud Detection System Using Ensemble Classification Techniques." International Journal of Data Science and IoT Management System 5, no. 2 (2026): 428-440.
19. R. U., M. P. Raj, J. N. Mithra, S. B. S., A. N. L., and J. M. D. Y., “A Robust UPI Fraud Identification Scheme over Digital Money Transactions using Learning Powered Classification Principles,” in Proc. Int. Conf. Electron. Renew. Syst. (ICEARS), Chennai, India, 2025.
20. https://datatracker.ietf.org/doc/html/rfc1034
21. OpenPhish, "Phishing intelligence feed," 2024. [Online]. Available: https://openphish.com
22. PhishTank | Join the fight against phishing.” [Online].Available: https://www.phishtank.com
23. URLhaus, "Malware and Phishing URL Repository," https://urlhaus.abuse.ch/ 2025
24. R. Housley et al., "Internet X.509 Public Key Infrastructure Certificate and CRL Profile," RFC 5280, IETF, 2008.
25. https://www.internetsociety.org/deploy360/dns-privacy/
26. D. Dahiphale et al., "Enhancing Trust and Safety in Digital Payments: An LLM-Powered Approach," in 2024 IEEE International Conference on Big Data (BigData), Washington, DC, USA, 2024, pp. 4854-4863, doi: 10.1109/BigData62323.2024.10825105.