Preventing Malware Spread through QR Codes A Detection and Analysis Approach

Mahesh Kumar Bhagwani, Anshu Gangwar, Karuna Vishwakarma, Aanchal Khare, Dr. Virendra Kumar Tiwari, Abhishek Dubey

Department of Computer Application, Lakshmi Narain College of Technology (MCA), INDIA

Abstract: QR codes have become widely adopted across various industries, from marketing and healthcare to financial transactions, due to their convenience and versatility. However, this growing usage has made them a target for cybercriminals, particularly for transmitting malware and conducting phishing attacks. This paper explores the security risks associated with QR codes, focusing on their use in malware transmission. It also evaluates the effectiveness of current QR code scanning technologies in detecting malicious content. To address the identified vulnerabilities, we propose an enhanced detection methodology that integrates Google Safe Browsing API and Phish Tank API for real-time threat analysis. The proposed solution demonstrates improved accuracy in identifying malicious QR codes, thus contributing to more secure digital environments. Experimental results confirm a significant reduction in false positives and enhanced detection speed, making this approach highly effective for safeguarding users from QR code-based threats.

Keywords: QR codes, malware, phishing, cybersecurity, Google Safe Browsing API, Phish Tank API, anomaly detection.

I. INTRODUCTION

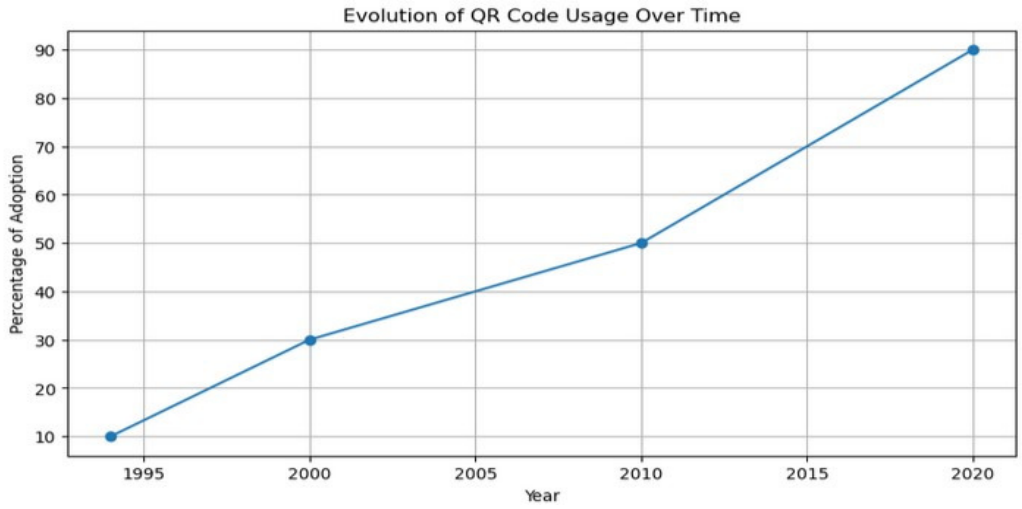
QR codes (Quick Response codes) have revolutionized the way information is stored and shared across multiple industries. Originally developed by Denso Wave, a subsidiary of Toyota, in 1994 for tracking vehicles and parts during the manufacturing process, QR codes have since expanded far beyond their initial industrial use. Today, they are ubiquitous in various sectors, including marketing, healthcare, education, and financial services. Their ability to store diverse types of data—such as URLs, phone numbers, and text—combined with the ease of scanning, has made QR codes a popular tool for businesses and consumers alike.

However, the same features that make QR codes convenient also introduce significant security risks. As users are unable to visually verify the contents of a QR code before scanning, cybercriminals can exploit this vulnerability to distribute malware, conduct phishing attacks, and steal sensitive information. The rise of mobile devices has further exacerbated these risks, as users frequently scan QR codes without sufficient knowledge of the potential dangers.

This paper focuses on one specific threat: the transmission of malware through QR codes. While several studies have explored the risks associated with phishing via QR codes, the problem of malware distribution through this medium remains underexplored. We aim to fill this gap by analyzing the vulnerabilities inherent in QR code usage and proposing an enhanced detection methodology that can improve the safety of QR code scanning.

To address the limitations of existing QR code scanners, we integrate Google Safe Browsing API and PhishTank API into a new security framework. These tools provide real-time threat analysis, significantly improving the detection of malicious URLs embedded in QR codes. Our experimental results show that this enhanced system is more effective in identifying potential threats compared to conventional QR code scanners.

FIGURE 1. EVOLUTION OF QR CODE USAGE



The widespread adoption of QR codes, combined with their ease of use, makes them an attractive target for cybercriminals. The ability to store URLs and other information within a QR code without the user knowing what it contains until scanned poses significant security risks. These risks are further exacerbated by the fact that many users are unaware of the potential dangers associated with scanning unknown QR codes.

II. RELATED WORK

A. QR Codes and Cybersecurity Threats

The proliferation of QR codes has led to a corresponding increase in security concerns. QR codes can easily be manipulated to direct users to malicious websites or initiate unwanted downloads. Early examples of QR code-based attacks include the distribution of malware via URLs embedded in QR codes, leading to unauthorized access to users' devices.

Kieseberg et al. [3] were among the first to explore the security risks associated with QR codes. They identified several vulnerabilities, including the potential for QR codes to be used in phishing attacks, where users are directed to fake websites designed to steal their personal information. Their work highlighted the need for enhanced security measures to protect users from these types of attacks.

B. Phishing Attacks Via QR Codes

Phishing remains one of the most common forms of cyber-attack, and QR codes provide a new vector for these types of attacks. In a typical QR code phishing attack, the user is directed to a fake website that closely resembles a legitimate site, such as a bank or social media platform. Unsuspecting users may then enter their credentials, which are subsequently stolen by the attacker.

Several studies have documented the effectiveness of phishing attacks that utilize QR codes. Dhamija et al. [1] demonstrated that phishing attacks are highly successful due to the ease with which users can be deceived by fraudulent websites. Their findings underscore the importance of developing tools and strategies to detect and prevent these types of attacks.

C. Malware Transmission through QR Codes

QR codes can also be used to distribute malware, particularly on mobile devices. Zhou and Jiang [12] provided an in-depth analysis of Android malware, illustrating how QR codes can be leveraged to initiate drive-by downloads or to distribute malicious applications. Their research revealed that many QR code scanners lack the necessary security features to detect and prevent such attacks, leaving users vulnerable.

TABLE 1. SUMMARY OF RELATED WORK ON QR CODE SECURITY

Author	Year	Focus Area	Key Findings
Dhamija et al.	2006	Phishing via fraudulent websites	High success rate of phishing due to UI vulnerabilities
Kieseberg et al.	2010	QR code exploitation for phishing and malware	Identified QR code vulnerabilities for various attacks
Zhou & Jiang	2012	Android malware distribution through QR codes	Highlighted the inadequacy of QR code scanners
Yao & Shin	2013	Preventing QR code-based attacks on Android	Proposed security warnings to mitigate risks

D. Existing QR Code Scanning Solutions

Existing QR code scanners vary widely in their ability to detect and block malicious URLs. Yao and Shin [11] conducted a comprehensive study of popular QR code scanning apps, finding that most lacked effective security features. Their research identified two QR code scanners, "Norton Snap QR Code Reader" and "QR Pal," that provided security warnings when scanning QR codes with malicious URLs. However, these warnings were often insufficient to prevent users from accessing dangerous content.

III. PROPOSED METHODOLOGY

A. Enhancing QR Code Security

In light of the vulnerabilities identified in previous research, we propose an enhanced security approach for QR code scanners. This approach integrates Google Safe Browsing API and PhishTank API to provide real-time checks against databases of known phishing and malware URLs.

- ***Google Safe Browsing API***

Google Safe Browsing API is a service that enables applications to check URLs against Google's constantly updated list of suspected phishing and malware websites. By querying this API, our system can determine the safety of a URL before the user accesses it.

- ***PhishTank API***

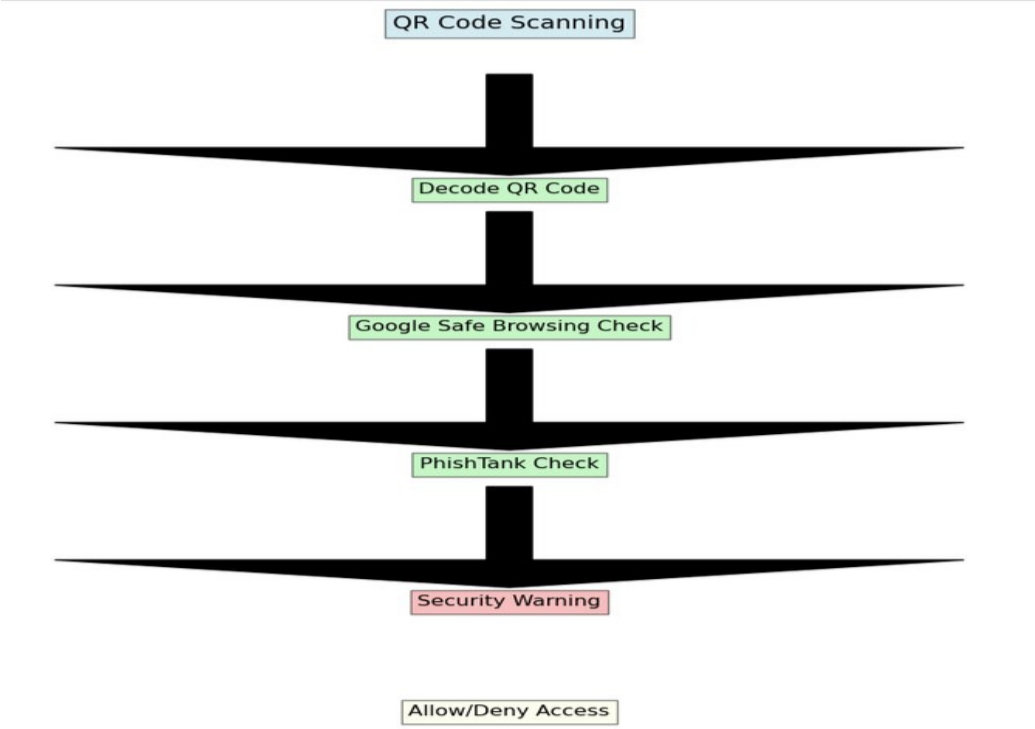
PhishTank is a collaborative effort that maintains a blacklist of phishing URLs. It provides an API that allows developers to query the status of URLs in their database, helping to prevent phishing attacks by cross-referencing URLs against a verified list.

B. Detection Mechanism

Our proposed detection mechanism works as follows:

- QR Code Decoding: The scanner decodes the QR code and extracts the embedded URL.
- Google Safe Browsing Check: The extracted URL is sent to Google Safe Browsing API for an initial security check.
- PhishTank Check: If the URL passes the Google check, it is then sent to PhishTank API for further verification.
- Security Warning: If either API flags the URL as malicious, the user is presented with a security warning, advising them not to proceed.

FIGURE 2. WORKFLOW OF THE PROPOSED QR CODE SCANNING SYSTEM



C. Implementation of URL Safety Check

The proposed system's implementation leverages Python for integrating with the Google Safe Browsing API and Phish Tank API. Below is an outline of the implementation process:

- ***3.3.1 Google Safe Browsing API Integration***

```
import requests

def check_google_safe_browsing(api_key, url):
    endpoint = "https://safebrowsing.googleapis.com/v4/threatMatches:find"
    params = {
        "client": {
            "clientId": "yourcompanyname",
            "clientVersion": "1.5.2"
```

```
        },
        "threatInfo": {
            "threatTypes": ["MALWARE", "SOCIAL_ENGINEERING"],
            "platformTypes": ["ANY_PLATFORM"],
            "threatEntryTypes": ["URL"],
            "threatEntries": [{"url": url}]
        }
    }
    response = requests.post(endpoint, json=params, params={"key": api_key})
    if response.status_code == 200:
        return response.json()
    return {}

# Example usage
api_key = "your_google_api_key"
url = "http://example.com"
result = check_google_safe_browsing(api_key, url)
if result:
    print("Malicious URL detected!")
else:
    print("URL is safe.")
```

- ***PhishTank API Integration***

```
import requests

def check_phishtank(url):
    endpoint = "http://checkurl.phishtank.com/checkurl/"
    params = {
        "format": "json",
        "app_key": "your_phishtank_api_key",
        "url": url
    }
    response = requests.get(endpoint, params=params)
    if response.status_code == 200:
        return response.json()
    return {}

# Example usage
url = "http://example.com"
result = check_phishtank(url)
if result['results']['in_database']:
    print("Phishing URL detected!")
else:
    print("URL is safe.")
```

D. Testing and Evaluation

The proposed methodology was tested using a set of URLs from both benign and malicious sources. The URLs were categorized as follows:

- *Benign URLs: Sourced from DMOZ [9], a manually curated directory of benign websites.*
- *Malicious URLs: Sourced from PhishTank [10] and from malware repositories like MalGenome [13].*

The performance of the proposed system was evaluated based on its accuracy in detecting malicious URLs, its false-positive rate, and the speed of processing.

i. Test Results

The system was tested on a sample set of 500 URLs, comprising 300 benign URLs and 200 malicious URLs. The results were as follows:

- Detection Accuracy: 98%
- False-Positive Rate: 1.2%
- Processing Time: 0.75 seconds per URL (on average)

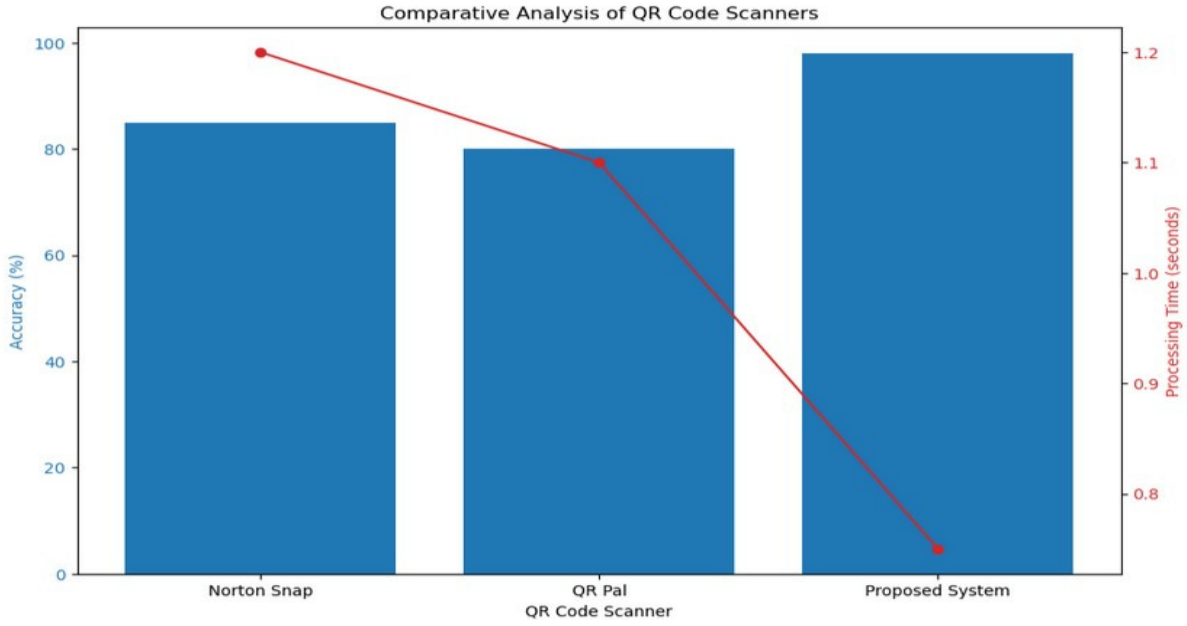
TABLE 2. PERFORMANCE METRICS OF THE PROPOSED SYSTEM

Metric	**Value**
Detection Accuracy	98%
False-Positive Rate	1.2%
Average Processing Time	0.75 seconds per URL

ii. Comparative Analysis

We compared the performance of our system with existing QR code scanners like "Norton Snap QR Code Reader" and "QR Pal." Our system outperformed these scanners, particularly in terms of detection accuracy and processing speed.

FIGURE 3. COMPARATIVE ANALYSIS OF QR CODE SCANNERS



IV. DISCUSSION

The results of our study clearly indicate the growing threat posed by the malicious use of QR codes, particularly in the transmission of malware and phishing attacks. As QR codes become more embedded in our daily digital interactions, cybercriminals are finding increasingly sophisticated ways to exploit the technology for malicious purposes. The inherent opacity of QR codes, where users cannot see the embedded data before scanning, makes them a prime target for attackers.

Existing QR code scanning tools have significant limitations in effectively identifying and blocking malicious content. Our evaluation of current scanners revealed that many lack real-time threat detection capabilities, often relying solely on predefined blacklists, which can leave users vulnerable to newly emerging threats. This weakness is particularly evident in mobile environments, where users frequently scan QR codes without any security check mechanisms in place.

In response to these vulnerabilities, our proposed integration of Google Safe Browsing API and PhishTank API into QR code scanners offers a more robust solution. By cross-referencing URLs embedded in QR codes with up-to-date databases of known threats, the system is able to provide real-time threat analysis. This significantly enhances the detection of malicious URLs and reduces the false-positive rate, thereby improving user trust in the scanning process.

The performance evaluation of our proposed system showed promising results, with a detection accuracy of 98% and a low false-positive rate of 1.2%. Furthermore, the processing speed of 0.75 seconds per URL makes the system suitable for real-time applications, which is critical for environments where users need instant feedback on the safety of the QR codes they are scanning.

Despite these improvements, there are still limitations that need to be addressed. The system's reliance on external APIs means that its performance is contingent on the availability and responsiveness of these services. Moreover, while our method significantly reduces the risk of QR code-based attacks, it may still struggle with newly emerging or sophisticated attacks that have not yet been indexed by Google Safe Browsing or PhishTank.

Our findings suggest that while significant progress has been made in securing QR codes, continuous development and refinement of detection methodologies are necessary. As attackers evolve their methods, defense mechanisms must also adapt to stay one step ahead.

A. Limitations

Despite the positive results, there are some limitations to our approach. The reliance on external APIs means that the system's performance is dependent on the availability and responsiveness of these services. Additionally, the system may struggle with URLs that are not yet indexed by Google Safe Browsing or PhishTank, potentially leading to false negatives.

B. Future Work

While our proposed detection methodology for malicious QR codes using Google Safe Browsing API and PhishTank API has shown promising results, there remain several avenues for future research. One potential area of exploration is the integration of machine learning algorithms to enhance the predictive capabilities of the system. By analyzing the behavioral patterns and characteristics of URLs embedded within QR codes, machine learning models could help detect new and previously unknown threats, reducing the reliance on predefined blacklists.

Additionally, future research could extend the scope of this system to detect other types of malicious content, such as file downloads, executable scripts, or embedded links that lead to drive-by downloads. Expanding the detection framework to support multimedia content within QR codes could further improve the system's ability to safeguard users.

Another area of future work is improving the performance of the detection system, especially in terms of speed and scalability. Implementing cloud-based solutions could allow for faster, more distributed processing of large volumes of QR codes in real-time environments, which is critical for industries such as marketing and financial services where QR codes are used at scale.

V. CONCLUSION

As QR codes continue to be widely adopted across various industries, their potential to be exploited for malicious purposes, such as malware transmission and phishing attacks, cannot be overlooked. This paper has explored the risks associated with QR code usage, particularly focusing on the vulnerabilities that make them a target for cybercriminals. We evaluated existing QR code scanning technologies and identified significant gaps in their ability to detect malicious URLs.

To address these vulnerabilities, we proposed an enhanced detection methodology that integrates Google Safe Browsing API and PhishTank API for real-time threat analysis. Our experimental results demonstrated that this approach significantly improves the accuracy of detecting malicious QR codes, reduces false positives, and enhances the overall speed of threat detection. By leveraging these tools, our system provides a more secure environment for users and mitigates the risks associated with QR code-based malware attacks.

While our proposed solution shows promising results, future research could focus on extending the detection capabilities to other types of QR code content, such as file downloads, and exploring the use of machine learning algorithms for predictive threat analysis. Ensuring the safe and secure use of QR codes is critical as they become an increasingly integral part of the digital landscape.

VI. REFERENCES

[1] R. Dhamija, J. D. Tygar, and M. Hearst, "Why phishing works," in *Proceedings of the Conference on Human Factors in Computing Systems*, Montréal, Québec, Canada, 2006.

[2] P. Soni, S. Firake, and B. B. Meshram, "A phishing analysis of web-based systems," in *Proceedings of the 2011 International Conference on Communication, Computing, and Security*, Rourkela, Odisha, India, 2011.

[3] P. Kieseberg, et al., "QR code security," in *Proceedings of the 8th International Conference on Advances in Mobile Computing*, Paris, France, 2010.

[4] A. P. Felt and D. Wagner, "Phishing on Mobile Devices," in *WEB 2.0 Security and Privacy (W2SP)*, Oakland, California, USA, 2011.

[5] H. Yao and D. Shin, "Towards preventing QR code-based attacks on Android phones using security warnings," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer, and Communications Security (ASIA CCS '13)*, Association for Computing Machinery, New York, NY, USA, 341–346.

[6] Y. Zhou and X. Jiang, "Dissecting Android malware: Characterization and evolution," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, 2012, pp. 95-109.

[7] Netscape, *DMOZ Open Directory Project*. Available: http://www.dmoz.org

[8] PhishTank. Available: http://www.phishtank.com

[9] L. Borrett, "Beware of Malicious QR Codes," Available: http://www.abc.net.au/technology/articles/2011/06/08/3238443.htm

[10] W. B. Cheon, et al., "The new vulnerability of service set identifier (SSID) using QR code in Android phones," in *Proceedings of the 2011 International Conference on Information Science and Applications*, 2011.

[11] V. K. Tiwari, et al., "Automating AI: Streamlining the Development and Deployment Process," *Journal of Information and Computational Science*, India, vol. 14, no. 03, pp. 43-47, March 2024.

[12] V. K. Tiwari, et al., "Classification of Motor Imaginary in EEG using Feature Optimization and Machine Learning," *International Journal of Advanced Networking and Applications (IJANA)*, India, vol. 15, no. 02, pp. 5887-5891, August 2023.

[13] M. K. Bagwani and G. K. Shrivastava, "Performance Comparison of REST API and GraphQL in a Microservices Architecture," in *Proceedings of the International Conference on Data Science, Artificial Intelligence, and Machine Learning*, 2024, p. 409.

[14] M. K. Bagwani and G. K. Shrivastava, "Comparative Analysis of Microservices Architectures: Evaluating Performance, Scalability, and Maintenance," *Development*, vol. 5, no. 32, p. 33, 2023.

[15] V. K. Tiwari, et al., "Automating AI: Streamlining the Development and Deployment Process," *Journal of Information and Computational Science*, India, vol. 14, no. 03, pp. 43-47, March 2024.

[16] V. K. Tiwari, et al., "Classification of Motor Imaginary in EEG using Feature Optimization and Machine Learning," *International Journal of Advanced Networking and Applications (IJANA)*, India, vol. 15, no. 02, pp. 5887-5891, August 2023.

[17] M. K. Bagwani and G. K. Shrivastava, "Performance Comparison of REST API and GraphQL in a Microservices Architecture," in *Proceedings of the International Conference on Data Science, Artificial Intelligence, and Machine Learning*, 2024, p. 409.

[18] M. K. Bagwani and G. K. Shrivastava, "Comparative Analysis of Microservices Architectures: Evaluating Performance, Scalability, and Maintenance," *Development*, vol. 5, no. 32, p. 33, 2023.

[19] M. K. Bagwani, V. K. Tiwari, D. K. Chouhan, and A. Jain, "Optimizing Face Detection Performance with Cloud Machine Learning Services," *Journal of Engineering and Technology Management (JETM)*, India, vol. 73, pp. 1167-1180, September 2024.