Towards Preventing QR Code Based Attacks on Android Phone using Security Warnings

Huiping Yao
Computer Science and Engineering
New Mexico Tech
Socorro, NM 87801, USA
hyao@nmt.edu

Dongwan Shin
Computer Science and Engineering
New Mexico Tech
Socorro, NM 87801, USA
doshin@nmt.edu

ABSTRACT
QR (Quick Response) code has become quite popular in recent years due to its large storage capacity, ease of generation and distribution, and fast readability. However, it is not likely that users will be able to find out easily the content encoded, typically URLs, until after they scan QR codes. This makes QR codes a perfect medium for attackers to conceal and launch their attacks based on malicious URLs. We believe that security hardening on QR code scanners is the most effective way to detect and prevent the potential attacks exploiting QR codes. However, little attention has been paid to the security features of QR code scanners so far in literature. In this paper, we investigated the current status of existing QR code scanners in terms of their detection of malicious URLs exploited for two well-known attacks: phishing and malware. Our study results show the existing scanners either cannot detect or can very poorly detect those two attacks. Hence, we propose a QR code solution called *SafeQR* that enhances the detection rate of malicious URLs by leveraging two existing security APIs to detect phishing and malware attacks: *Google Safe Browsing API* and *Phishtank API*. Additionally, a visual warning scheme was carefully designed and implemented to enable users to better heed warnings. A user study was designed and conducted to investigate the effectiveness of our scheme compared with the methods adopted by existing QR code scanners.

Categories and Subject Descriptors
K.6.5 [**Management of Computing and Information Systems**]: Security and Protection – *Invasive software, unauthorized access.*

Keywords
QR code security; phishing; malware; visual warning; user study

1. INTRODUCTION
As smartphones have become very popular recently, more and more services based on smartphones have been being developed and adopted. Quick response codes (QR codes), as one of such services, have been widely used due to their easy generation and distribution, large storage capacity, and fast readability [1]. They are often found on public signage, usually to direct users to a website with more information about a product, company, or location. According to a recent survey, 75% of retailers now offer customers this 2D technology to interact with and track potential buyers [2]. Customers can scan them with QR code scanners on their smart phone or tablet to receive coupons, discounts, or other information related to products or services.

However, due to the fact that their encoding scheme is more machine-readable than human-readable, the only way for users to figure out what information is encoded in a QR code is to scan it. This makes QR codes a perfect medium for attackers to conceal and launch their attacks based on malicious contents embedded such as phishing URLs [3]. Besides, people often cannot resist their own curiosity to scan the codes they have come across. This innate obfuscation of QR codes and the curiosity make QR codes one of the biggest hidden security threats [4]. So far, little attention has been paid to the security features of QR code scanners in literature. In this paper, we investigated the current status of existing QR code scanners in terms of their detection of malicious URLs exploited for two well-known attacks: phishing and malware. Our study results show that the existing scanners either cannot detect or can very poorly detect those two attacks. Hence, we propose a QR code solution called *SafeQR* that enhances the detection rate of malicious URLs by leveraging two existing security APIs for effectively dealing with phishing and malware attacks: *Google Safe Browsing API* and *Phishtank API*. Additionally, our solution offers an effective visual warning scheme to help users better informed of imminent threats. Finally, an experiment involving a user study was designed and conducted to investigate the effectiveness of our scheme compared with the methods adopted by existing QR code scanners.

The rest of the paper is organized as follows: Section 2 discusses the potential threats in QR codes, and reveals the problems of the existing QR code applications on Android phone. Section 3 presents our approach to tackle the existing problems. Section 4 discusses the methodology and design of a user study developed to test the effectiveness of our approach as well as some results from the user study. Section 5 concludes the paper with our future research work.

2. BACKGROUND
In this section, we first discuss the main potential threats brought about by QR codes, and then we present our study results which show that the vast majority of Android-based QR code scanners did not provide users any security information for the scanned URLs.

ASIACCS'13, May 8–10, 2013, Hangzhou, China.

2.1 Potential Threats of QR Codes

Usually, a QR code directs users to the website of their interest, which can provide further information or services. By scanning a carefully manipulated QR code, however, they may be taken to a malicious website. Attackers can use QR codes for various types of attacks. However, two critical attacks we are interested in for this study are phishing and malware attacks.

- **Phishing**: phishing is an activity that tricks people to divulge their sensitive information by masquerading as a trustworthy entity [5, 6]. A QR code can redirect users to a fake bank that looks exactly like the real bank. A normal user cannot see the differences and type in his or her information and hand it to the attackers [7-9].
- **Malware**: A malicious QR code can be used to redirect users to a URL containing malware[3]. An early example of malware attacks through QR code was that people were fooled into scanning a QR code and downloading a malicious application, which sent off multiple text messages to a number that charged users $5 per SMS message [10].

There are other types of attacks such as social engineering [7, 11] and cross-site attacks [12]. Though interesting, a complete study to analyze these rather rare cases was out of purview of our study.

2.2 Defects of Existing QR Code Readers

QR code scanners are the only direct way to detect QR code based attacks. In this study, we first investigated the security related features of existing QR code scanners, and then focused on the detection rates of existing QR code scanners with security indicators.

2.2.1 Security Related Features of QR Code Apps

Google Play, the official Android Market, is used as a reliable source for users to download the apps. On Jun 6, 2012, we searched for QR code scanning apps from Google Play market using keywords "QR code", "QR code reader", and "QR code scanner," and we found 31 apps. We installed them in a HTC Nexus One with Android 2.3.6 and used them to scan QR codes with benign URLs randomly chosen from DMOZ [13], a manually edited directory, and QR codes with malicious URLs selected from PhishTank [14], a blacklist of phishing URLs, to find out the features of these scanners.

Table 1 shows the results of our study on the security related features of those scanners. Due to the space limitation, we only listed the software names. Though the names (Barcode Scanner) of No. 2 and No. 3 are the same, No. 2 scanner was developed by ZXing Team, while No. 3 was developed by George Android.

As we can see from Table 1, 74.19% (23/31) of the apps had a feature called user confirmation. The user confirmation feature provides users with a confirmation page that displays the URL encoded in the QR code, along with an option to continue/discontinue to visit the website. This allows users to find out where they will visit, and thus offers a way to involve users to potentially prevent attacks by inspecting the decoded URL. Interestingly, two readers, *QR Droid* and *QR Droid Private*, had an additional feature called preview. The preview feature provides the preview of the scanned website so that users will be able to figure out roughly if the site is actually the one that they are initially interested in (this is especially true in case that they actually visited the site previously.). On the other hand, an app called *QuickMark* had an interesting utility function based on the history of QR code usage. The function displays two usage data: *Scan* and *Click Rate*. The first is the total number of times that a QR code has been scanned using this app, and the second pertains to the number of times a QR code has been clicked divided by *Scan*.

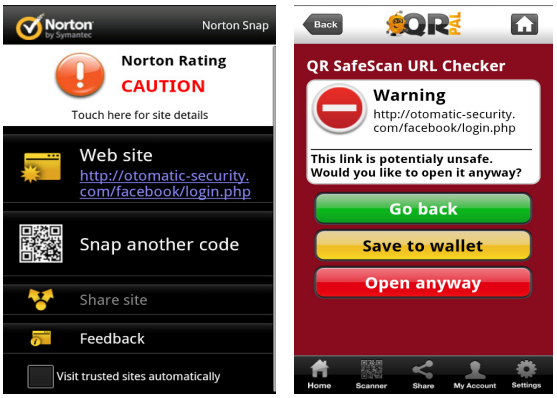
Table 1. Security related features of QR code scanners

No.	Application	User Confirmation	Preview	Security Warning
1	AT&T Code Scanner	✓		
2	Barcode Scanner	✓		
3	Barcode Scanner	✓		
4	BeeTagg QR Reader	✓		
5	RedLaser Barcode& QR Scanner	✓		
6	Codee QR Code Reader	✓		
7	DTEScanner			
8	Google Goggles	✓		
9	HandyShopping Barcode Scanner	✓		
10	HP CodeScan			
11	i-nigma Barcode Scanner	✓		
12	Mobiletag QR Code Scanner			
13	NeoReader	✓		
14	Norton Snap QR Code Reader	✓		✓
15	Scan			
16	ScanLife Barcode & QR Reader			
17	Scanner Pro	✓		
18	SHARP QR Code Reader	✓		
19	ShopSavvy Barcode Scanner			
20	QuickMark Barcode	✓		
21	QR Barcode Scanner	✓		
22	QR Barcode Scanner – Lite	✓		
23	QR barcode scanner			
24	QR Code Reader / Scanner			
25	QR Droid	✓	✓	
26	QR Droid Private	✓	✓	
27	QR Pal – QR & Barcode Scanner	✓		✓
28	QR Pro	✓		
29	QR Reader for Android	✓		
30	QR Rewords	✓		
31	UberScanner	✓		

However, none of the features seems to provide users with information that will help them make a better security decision. As to the user confirmation feature, users see only a URL and they often click to continue to visit the site without much thinking about the security consequences. Unsophisticated users may even not be able to recognize malicious URLs. The preview feature can be easily abused, incapable of providing users with needed security information either; a well designed fraudulent website can only give a false sense of security to users.

Among the 31 scanners, only two (0.06%), "*Norton Snap QR Code Reader*" and "*QR Pal – QR & Barcode Scanner*", had a feature called security warning. Upon scanning a QR code containing a malicious URL, *Norton Snap* and *QR Pal* equipped

with the feature display the URL along with a warning message before loading the website of the URL, as shown in Figure 1.



(a) (b)

Figure 1. Warning messages displayed in two existing QR code scanners: (a) Norton Snap; (b) QR Pal

2.2.2 *Evaluation of Two Scanners*

Since *Norton Snap* and *QR Pal* provide various levels of security warnings for malicious URLs, it would be interesting to study the accuracy of those warnings. During the testing for measuring their detection rates, we applied the default settings of those two QR code scanners. We investigated two types of errors, false positive and false negative errors. Since we believe false negative errors are more serious that false positive ones, we conducted a more extensive testing on false negative errors.

2.2.2.1 *Benign URLs*

To study whether the two QR scanners had false positive errors or not, we used DMOZ Open Directory Project as data set, of which all entities are vetted by editors who also go through a vetting process [15], and randomly chose one URL from each of the 10 random subcategories under 15 categories as follows: Arts, Business, Computers, Games, Health, Home, Kids and Teens, News, Recreation, Reference, Regional, Science, Shopping, Society, and Sports. Out of 150 URLs, *Norton Snap* took 16 URLs as untested rating, 2 as malicious, and the remaining 132 as trusted ones. So the false positive rate of *Norton Snap* was 1.33% (2/150). *QR Pal* took all the 150 URLs as benign, thus we concluded that *QR Pal* did not have false positive errors.

2.2.2.2 *Phishing URLs*

Testing phishing URLs, drawn from PhishTank, was conducted based on the following plan: first, we conducted a preliminary testing using a small sample size to see if the two scanners had a desirable detection rate for phising URLs, by setting 90% detection rate as a baseline that is usually achieved by email-based phishing detection tools [16] (1st phase). Since their performance was way below than the baseline, we decided to conduct a more comprehensive testing with a larger sample size (2nd phase).

The First Phase

When analyzing a site, *Norton Snap* goes through the Norton Safe Web [17] to identify trusted and malicious sites, and gives the corresponding ratings. Specifically, Norton Safe Web will evaluate a site by computing the three factors, *computer threats*, *identity threats*, and *annoyance factors*. A site is identified malicious if Norton detects one or more threats from any of the above three factors. If none of the three threats was found, then the site is identified as a trusted one. *Norton Snap* has five security ratings as follows: WARNING_MALICIOUS, CAUTION, UNTESTED, SAFE, and SECURED. WARNING_MALICIOUS or CAUTION rating indicates that the URL is malicious. UNTESTED means the site has not been tested yet. SAFE or SECURED implies that the URL is trusted. The difference between WARNING_MALICIOUS and CAUTION is that: if a site is identified to have one or more *computer threats* regardless of the other two factors, it is rated as WARNING_MALICIOUS, otherwise CAUTION. If a site passes all the above three factors, Norton Safe Web then checks if it satisfies at least one item of Ecommerce Safety [18], if yes, the site is rated as SECURED, otherwise, rated as SAFE.

In the first phase, we randomly picked and scanned 15 phishing URLs on Jun 28, 2012 using the two apps. Our analysis result showed that, for *Norton Snap*, only 26.67% (4/15) URLs were predicted correct, 60% (9/15) scans did not provide any security information by displaying "UNTESTED", which actually does not provide helpful warnings for users to decide to open a site, 13.33% (2/15) cases gave the wrong security information. *QR Pal* either provides warning information for malicious URLs or opens the URL for the benign URLs. Our analysis result on *QR Pal* showed that 66.67% (10/15) were predicted correct, and it opened the remaining 5 malicious URLs directly without any user confirmation.

Through the preliminary testing, we discovered that neither *Norton Snap* nor *QR Pal* had a desirable detection rate for phishing URLs. Their performance was way below the threshold of 90%.

The Second Phase

In the second phase, we randomly chose 400 out of 6131 entries (6.52%) from the PhishTank dataset downloaded on Jul 7, 2012.

Table 2. Detection of phishing attacks

	Correct		Neutral		Incorrect	
	#	%	#	%	#	%
Norton Snap	112	28.00	150	37.50	137	34.25
QR Pal	111	27.75	0	0.00	289	72.25

The results in Table 2 show that both *Norton Snap* and *QR Pal* were quite disappointing in terms of their detection of phishing attacks. *Norton Snap* only detected 28% (112/400, 95% Confidence, CI-4.25%) of the URLs correctly. For 37.5% of (150/400) the scans, *Norton Snap* did not provide any security warning for users. What is worse, it provided wrong security information for the remaining 34.25% (137/400) cases. For *QR Pal*, it did not contain the neutral case. The detection rate was only 27.75% (111/400, 95% Confidence, CI-4.24%).

2.2.2.3 *Malware URLs*

Malware attack is another popular attack on Android. We tested the two QR code scanners against the attacks using malware obtained from http://malgenomeproject.org [3]. Specifically, we used malware under the drive-by download categories (*GGTracker*, *Jifake*, *Spitmo*, and *Zitmo*) and repackaging malware categories (*AnserverBot* and *DroidKungFu*). Each of the four

drive-by downloaded categories contains only one malware. We chose all these four malware for testing. For *AnserverBot* and *DroidKungFu*, we randomly picked one malware from each category. We uploaded these six malware to a personal homepage http://infohost.nmt.edu/~hyao, and tested the corresponding URLs, thus simulating a zero-day attack.

We found that both *Norton Snap* and *QR Pal* detected very poorly for all the six situations. *Norton Snap* only checked if the domain of the URL was secure or not. Therefore, since our test domain was not classified as malicious, it gave the security rating "Safe", so that users could easily download the malware. On the other hand, *QR Pal* did not download any of these malware with its default setting. However, it could download all these six malware if its default *Built-in browser* setting in the general settings was turned off, which could be easily done by users who just followed the instructions displayed by the app, upon receiving the download request from the users.

3. OUR SOLUTION

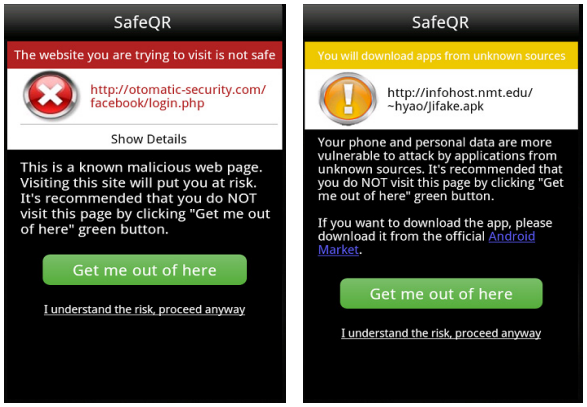
Our approach called *SafeQR* aimed at two goals in order to address the poor performance problem of the existing QR code scanners. The first was to enhance the effectiveness of detecting malicious URLs used for phishing and malware attacks. The second was to improve user perception of security when a QR code is scanned and used so that users can make a better security decision. For this purpose, we focused on how to improve the effectiveness of the security warnings by providing a better mobile UI design.

3.1 Malicious URLs Detection

Two well-known security APIs, *Google Safe Browsing API* [19] and *PhishTank API* [20], were adopted for our solution to improve the effectiveness of detecting malicious URLs.

Safe Browsing, developed by Google, is a service that enables applications to check URLs against Google's constantly updated lists of suspected phishing and malware websites. For simplicity, we chose *Safe Browsing Lookup API* [21], queried the URLs through HTTP GET request, and got the state of the URL(s) directly. PhishTank contains a blacklist of phishing URLs consisting of manually verified websites. PhishTank provides API for developers to lookup a URL's status in their database. We used the API to query a URL's status, thus further enhancing the capability for detecting phishing scams.

In addition, if the URL string ends with ***.apk***, this means that a non-official Android market application will be downloaded to a user's mobile phone. Specifically, if "Unknown Sources" setting is checked in Android, the app will be automatically downloaded to the user's Android device. If the setting is not checked, a dialog will pop up to ask the user if she wants to check the option to download the app. Users can easily tick "Unknown Sources" option just by following the instructions in the dialog, and then download and install the application. Our solution checked if the URL ended with "***.apk***", and if yes, then we provided potential warnings to users, as shown in Figure 2.



(a) (b)

Figure 2. Our UI designs: (a) UI for phishing or malware attacks; (b) UI for the URL ending with ".apk"

The algorithm for our solution is outlined in the following pseudo code.

```
Algorithm detection(url)
1   status = 0;
2   if google-safebrowsing-code(url)==200
3      if contains-phishing && contains-malware
4         status = 3;
5      else if contains-malware
6         status = 2;
7      else if contains-phishing
8         status =1;
9   if in-phishtank-list(url)
10     if status == 0 || status==2
11        status++;
12  if status == 0 && end-with-apk(url)
13     status = 4;
14  return status;
```

In the code, the *status* variable representing the security status of a URL has 5 different values. The *status* variable is initialized to 0, implying a safe URL. Its value becomes 1 when a URL is detected as a phishing one, 2 when a URL is detected as a malware website, 3 when a URL is a both phishing and malware site, 4 when a URL is checked safe through the Safe Browsing and PhishTank but ends with "***.apk***".

3.2 Security Warning Design

Studies have shown that warning designs affect user's decision to obey or ignore the warnings [22, 23], thus designing an effective security warning by providing better risk perception is a significant part of our solution. We applied existing warning design recommendations [22, 24] and Microsoft's NEAT (Neat, Explained, Actionable, Tested) & SPRUCE (Source, Process, Risk, Unique, Choices and Evidence) [25] into our warning message design as follows: we first set the warnings mainly with black and red colors, since this color combination is quite effective to prevent users from attacks [22]; we made the safest button "Get me out of here", as a recommended action, most visible by setting its background green, which also helps users

think of safe actions; since users are likely to ignore lengthy text, we only include necessary words in the warning, giving users a chance to click "Show Details" button for viewing details. In addition, only simple words are used, for users will not understand or will misinterpret technical jargons.

Two screenshots of our solutions are captured in Figure 2 (a) (b) respectively. Figure 2 (a) shows the display of our solution when a user scans a QR code with a phishing or malware URL. Figure 2 (b) shows the screenshot when a user scans a QR code whose encoded URL ends with "***.apk***".

If the website addressed by the URL is a phishing website or contains malware, we immediately return a negative evaluation, which is shown in Figure 2(a). If the URL ends with "***.apk***", we return an uncertain evaluation, shown in Figure 2(b), for ***.apk*** file may be not a malware. Compared to *Norton Snap* and *QR Pal*, our security warning design is better understood by providing the sufficient risk details and the recommended action without using technical jargons, and this is confirmed by the results of our user study in the next section.

4. USER STUDY

To investigate the effectiveness of our solution, we designed a user study to explore the effectiveness of the security warning in our solution. Our user study also was to compare our solution against the existing two solutions we discussed in the previous section. We also wanted to study how all of these solutions compare against the absence of any visual security warning.

4.1 Design and Recruitment

We defined four separate user groups, each of which was exposed to a different warning provided by a different QR code scanner.

- Group 1: Exposed to the attack with no warning, using *QR-code Scanner*
- Group 2: Exposed to the attack with *Norton Snap*'s warning
- Group 3: Exposed to the attack with *QR Pal*'s warning
- Group 4: Exposed to the attack with our designed warning

In the design of our experiment, we performed a power analysis to determine the minimum sample size that we would require to test our hypotheses. We chose an error of 0.05 and a power of 0.8, common among such experiments, and determined a minimum sample size of 19 subjects across the four user groups. Based on this analysis, we chose to recruit 80 participants with which 20 subjects in each of the four groups. It was necessary for participants to have a Facebook account, so that we can test their reactions after scanning a QR code containing the URL of the Facebook authentication page.

Our participants, 20 females and 60 males, have a high education rate, with all having completed at least high school degree, and 83.75% (67/80) having or currently pursuing undergraduate college degrees. Age groups of our participants include 38 from 10-20 years old, 32 from 21-30 years old, 5 from 31-40 years old, 2 from 41-50 years old, 2 from 51-60 years old, 1 from 61-70 years old. Relative to security knowledge, the participants in our sample are very sophisticated, with only 7 claiming they have poor security knowledge. 54% (43/80) of the participants replied that they had good security knowledge or above.

4.2 Experiment

During the experiment, we randomly assign 20 participants to each of four test groups. Each group was exposed to the phishing attack; three groups were given a specific warning respectively by *Norton Snap*, *QR Pal*, and *SafeQR*, and the fourth group was not warned at all by using *QR-code Scanner*. To avoid the framing effect, we did not want the users to be aware that we were testing their reaction to a security warning. In addition, we wanted the users to be exposed to the warnings as an abnormality or exceptional condition. To achieve both goals, we told the participants that we were investigating whether they made full use of smart phone apps, and evaluating the usability of using QR code scanners to access websites.

The hypotheses we wanted to test were on (1) the user unawareness of malicious QR codes, (2) the effectiveness of QR code based phishing attacks, and (3) the helpfulness of security warnings. Finally, we wanted to verify that our proposed warning design would be more effective than the existing solutions.

The participants were given an Android smart phone where the QR code apps were installed and were told that we would be interested in improving the usability of the apps, so they were encouraged to use the app with their real credentials to help us achieve the goal. Besides, users were asked to act as if they were using their own phone, in that all decisions they made should be the same as if they were being made on their own private phone. Security was never explicitly mentioned. Although we initially thought the "make all the decisions as if this was your phone" statement could bring focus on security issues, the results showed us that this was not the case.

4.3 User Study Results

Each user was asked to take an exit survey, which was also used to test our hypotheses. Our first hypothesis was tested by asking the subjects whether they had ever thought of any security problems caused by QR codes. Out of 80 subjects, 67.5% (54/80) were not aware that malicious QR codes existed. Hence, we concluded that most of the users were unaware of malicious QR codes. From the results of our second hypothesis testing, we learned that without any added security mechanisms, the phishing attack is highly effective, for 100% (20/20) of the participants opened the link and 75% (15/20) of them submitted their Facebook username and password. We also found that the security warnings provided by the QR code scanners, i.e. *Norton Snap*, *QR Pal*, and *SafeQR,* helped users perceive potential dangers and avoid phishing attacks, as shown in Table 3. Hence, our third hypothesis was confirmed. The results obtained for our proposed solution *SafeQR* proved more efficient than the existing solution. In terms of opening the link, *Norton Snap* only led 5 out of 20 users to not open the link, *QR Pal* led 11 out of 20 users to not open the link, while our solution *SafeQR* led 17 out of 20 users to not open the link. In terms of submission, both *Norton Snap* and *QR Pal* led 13 out of 20 users to not submit their credentials. Our solution *SafeQR* led 18 out of 20 users to not submit their credentials.

Table 3. For different study groups, # of open and submit

	Open	Submit
QR-code Scanner	20	15
Norton Snap	15	7
QR Pal	9	7
SafeQR	3	2

5. CONCLUSION AND FUTURE WORK

In this paper, we presented an approach to preventing QR code-based phishing and malware attacks. Specifically, we first studied the current status of existing QR code scanners in terms of their detection rate for malicious URLs. Then we proposed our solution to detect malicious URLs more effectively by using two well-know security APIs along with visual security warning design. Lastly we discussed our user study design to evaluate the effectiveness of the proposed solution.

This research is by no means complete. Firstly, our immediate future work is to analyze the data from user study to provide some insightful guidelines about designing effective security warnings on mobile phones. Secondly, in this paper, we mainly focused on security hardening on QR code scanners. Our future research direction will include an extensive study of whether it is possible to enhance the security of QR code itself. This could be achieved by considering some cryptographic methods applied to QR codes for the purpose of certification and identification. Lastly, the phishing URLs that we used for this study came from PhishTank. In our immediate future work, we are going to use sample URLs from other sources such as Spamscatter to evaluate our proposed solution.

6. ACKNOWLEDGMENTS

This work was partially supported at the Secure Computing Laboratory at New Mexico Tech by the grant from the National Science Foundation (NSF-IIS-0916875). The authors would like to thank Chen Sun from NM Tech and three anonymous reviewers for their valuable comments and suggestions.

7. REFERENCES

[1] C. Woo Bong, H. Keon il, L. Won Gyu, P. Won Hyung, and C. Tai Myoung, "The New Vulnerability of Service Set Identifier (SSID) Using QR Code in Android Phone," in International Conference on Information Science and Applications (ICISA), Washington, DC, USA, 2011.

[2] CNET. (2012). *The Dark Side of QR Codes*. Available: http://news.cnet.com/8301-1009_3-57464276-83/the-dark-side-of-qr-codes/.

[3] Z. Yajin and J. Xuxian, "Dissecting Android Malware: Characterization and Evolution," Security and Privacy (SP), 2012 IEEE Symposium on, 2012.

[4] D. Winder. (2012). *Five Hidden Security Threats*. Available: http://www.pcpro.co.uk/features/374896/five-hidden-security-threats.

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

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

[7] P. Kieseberg, M. Leithner, M. Mulazzani, L. Munroe, S. Schrittwieser, M. Sinha, and E. Weippl, "QR code security," Proceedings of the 8th International Conference on Advances in Mobile Computing, Paris, France, 2010.

[8] V. Sharma, "A Study of Malicious QR Codes," International Journal of Computational Intelligence and Information Security (IJCIIS), vol. 3, May 2012.

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

[10] *Monthly Malware Statistics: September 2011*. Available: www.securelist.com/en/analysis/204792195/Monthly_Malware_Statistics_September_2011

[11] L. Borrett. (2011). *Beware of Malicious QR Codes*. Available: http://www.abc.net.au/technology/articles/2011/06/08/3238443.htm

[12] A. Kieyzun, P. J. Guo, K. Jayaraman, and M. D. Ernst, "Automatic creation of SQL Injection and cross-site scripting attacks," Proceedings of the 31st International Conference on Software Engineering, 2009.

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

[14] *PhishTank*. Available: http://www.phishtank.com

[15] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: learning to detect malicious web sites from suspicious URLs," Proceedings of the 15th ACM SIGKDD, Paris, France, 2009.

[16] S. Abu-Nimeh, D. Nappa, W. Xinlei, and S. Nair, "Distributed Phishing Detection by Applying Variable Selection Using Bayesian Additive Regression Trees," in Communications, 2009. ICC '09.

[17] *Norton Safe Web*. Available: http://safeweb.norton.com/

[18] Norton. *Ecommerce Safety*. Available: https://safeweb.norton.com/help/ecommerce_safety

[19] *Safe Browsing API*. Available: https://developers.google.com/safe-browsing

[20] OpenDNS. *PhishTank API Information*. Available: http://www.phishtank.com/api_info.php

[21] *Safe Browsing Lookup API Developer's Guide*. Available: https://developers.google.com/safe-browsing/lookup_guide

[22] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor, "Crying wolf: an empirical study of SSL warning effectiveness," Proceedings of the 18th conference on USENIX security symposium, Montreal, Canada, 2009.

[23] S. Egelman, L. F. Cranor, and J. Hong, "You've been warned: an empirical study of the effectiveness of web browser phishing warnings," Proceedings of Conference on Human Factors in Computing Systems, Italy, 2008.

[24] L. Zeltser. (2011). *How to Design Security Warnings to Protect Users*. Available: http://blog.zeltser.com/post/3638747689/designing-security-warnings

[25] R. Reeder, E. C. Kowalczyk, and A. Shostack, "Helping Engineers Design NEAT Security Warnings," presented at the Symposium On Usable Privacy and Security (SOUPS) Pittsburgh, PA, USA, 2011.