QR Steganography
A Threat to New Generation Electronic Voting Systems

Jordi Cucurull[1], Sandra Guasch[1], Alex Escala[1], Guillermo Navarro-Arribas[2] and Víctor Acín[2]
[1]*Research and Security Department, Scytl Secure Electronic Voting, Barcelona, Spain*
[2] *Department of Information and Communications Engineering, Universitat Autònoma de Barcelona, Cerdanyola, Spain*

Keywords: QR, Security, Steganography, Electronic Voting.

Abstract: Quick Response (QR) codes, used to store machine readable information, have become very common nowadays and have found many applications in different scenarios. One of such applications is electronic voting systems. Indeed, some electronic voting systems are starting to take advantage of these codes, e.g. to hold the ballots used to vote, or even as a proof of the voting process. Nevertheless, QR codes are susceptible to steganographic techniques to hide information. This steganographic capability enables a covert channel that in electronic voting systems can suppose an important threat. A misbehaving equipment (e.g. infected with malware) can introduce hidden information in the QR code with the aim of breaking voters' privacy or enabling coercion and vote-selling. This paper shows a method for hiding data inside QR codes and an implementation of a QR writer/reader application with steganographic capabilities. The paper analyses different possible attacks to electronic voting systems that leverage the steganographic properties of the QR codes. Finally, it proposes some solutions to detect the mentioned attacks.

1 INTRODUCTION

Quick Response (QR) codes (ISO/IEC, 2006) have become very common nowadays. Its characteristics make them very suitable to store machine readable information which can be easily captured by any device equipped with a simple camera such as a smartphone. Their use has become quite pervasive and electronic voting systems are not an exception. Several recent electronic voting systems are using QR codes, e.g. to hold the encrypted vote so it can be used to vote, or even as a proof of the voting process. In any case we argue that the use of QR codes can expose a vulnerability in voting systems.

This vulnerability is related to the error correcting capacity of QR codes, which use an error correcting codification to be able to stand errors in their reading. Such errors might be produced when printing QR codes if a low quality printer is used, once the paper has already been printed due to, for example, crumpled paper, or when reading the QR code if a low quality camera is used or the environmental light is not sufficient. While this error correcting ability is desirable in most applications, in some scenarios it can be exploited as a covert channel to hide undetectable information.

QR steganography provides the ability to hide information in the QR code, in a way such that the code still provides the initial intended information. The legitimate reader reads the original information without noticing the hidden message, but a specially tuned reader can read this hidden message. The hidden message can be encoded as errors or concealed within the redundant data used to correct the errors.

This steganographic capability of the QR codes enables a covert channel in electronic voting systems that suppose an important threat. A misbehaving equipment (e.g. infected with malware) can introduce hidden information in the QR code with the aim of breaking voters' privacy or enabling coercion and vote-selling.

QR codes have been recently used in steganography. More precisely the main use is to encode the steganogram in the QR code, and then the QR code into an image (Chung et al., 2009; Dey et al., 2012; Huang et al., 2011; Chen and Wang, 2009). The QR code is used as the container of the steganogram, and its error correcting capabilities ensure that the errors introduced by encoding the QR into an image does not alter the message. That is, the hidden information is the information encoded in the QR. Contrary to these works, the hiding of the information in the QR

Cucurull J., Guasch S., Escala A., Navarro-Arribas G. and Acín V..
QR Steganography - A Threat to New Generation Electronic Voting Systems.
DOI: 10.5220/0005120404840491
In *Proceedings of the 11th International Conference on Security and Cryptography* (SECRYPT-2014), pages 484-491
ISBN: 978-989-758-045-1

has been scarcely explored in the literature (Lin et al., 2013). Furthermore we are not aware of any previous publication exposing the threat that QR codes can involve in electronic voting systems.

Although several types of two dimensional codes exist, QRs are the most popular and widely used. For example, all standard barcode reader applications for mobile phones can read QRs, while they may not be able to read other formats, such as PDF417 or Data Matrix. Thus, our analysis is centered on QR codes.

In this paper we argue that QR codes can introduce vulnerabilities in voting systems and can suppose an important threat if this implications are not carefully considered. The paper is organised in six sections: Section 2 gives an overview of the QR codes' design; Section 3 presents a steganography method and its implementation to include hidden data into the code; Section 4 analyses the use of QR codes in voting systems and associated attacks; Section 5 proposes solutions to detect and mitigate the presented attacks; and Section 6 presents the conclusions of the paper.

2 QR CODES

A QR code (ISO/IEC, 2006) is a two dimensional code intended for storing information, which is composed of an array of squared modules arranged in a squared area. The code version determines the storage and error correction capacity. Maximum storage is 23.648 bits, while four error correction levels are available, up to 30% error correction capacity.

2.1 Overview

A QR code, also referred as symbol, is composed of several rows and columns, with sizes ranging from 21 x 21 modules (Version 1) to 177 x 177 modules (Version 40). A QR code is composed of different elements (see Figure 1):

- **Position Detection and Alignment Patterns.** Each QR code contains three position detection patterns on the upper left, upper right and lower left corners. A number of alignment patterns, dependant on the symbol version, are included.
- **Format Information.** Located around the position detection paterns, this includes information about the error correction codes and mask pattern used (see Section 2.3).
- **Version Information.** This determines the capacity of the QR code. It is also located around the position detection patterns.

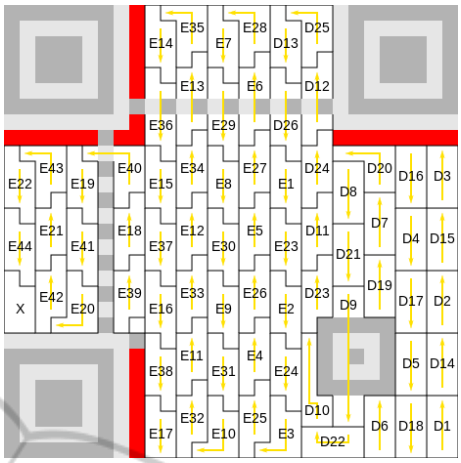

Figure 1: QR elements and data distribution. (src. Wikimedia Commons, CC0 1.0 Universal lic.)

- **Timing Patterns.** Used to determine the symbol density and version of the QR. They are marked in Figure 1 as the dotted lines crossing the QR both horizontally and vertically.
- **Data and Error Correction Codes.** Areas that contain the actual data content stored in the QR, as well as error correction codes of the data. These areas are organised in blocks, which are composed of a number of codewords, each of them of 8 bits. The number and size of blocks assigned to data and error correction codes is determined by the QR version. The data and error correction codes are placed in different areas of the QR (see blocks numbered as *Dx* for data, and *Ex* for errors, in Figure 1).

Depending on the type of data contained in the QR, different encodings can be used, e.g. numeric, alphanumeric, 8-bit Byte, etc. The encoding is selected with an optional header called Extended Channel Interpretation (ECI) included as part of the data blocks.

2.2 Blocks and Mask

Codewords from different blocks are interleaved in the QR. In Figure 1 there is an example of a QR code where there are two blocks of data, B1 and B2: the first block contains the data codewords D1-D13 and the second one the data codewords D14-D26. Nevertheless, the codewords are interleaved, i.e. they are written as D1, D14, D2, D15, and written from the right-bottom to the top-left parts of the code. Interleaving is used to increase the symbol robustness against error bursts.

At low level, a QR code is composed of a number of white and black squared modules. Depending

on the encoded data, there could be areas with many modules of the same colour, producing reading problems. In order to avoid these issues, a known mask pattern with a uniform distribution of white and black modules is XORed over the encoded data. This results in a well-balanced distribution of modules. Eight different mask patterns, identified with a three bit code called Mask Pattern Reference, are available.

2.3 Error Correction

QR codes provides error correction based on Reed-Solomon codes (Reed and Solomon, 1960). The QR code has a given error correction capacity, determined by the QR version, divided in four possible levels: L (7%), M (15%), Q (25%) and H (30%). The higher the correction capacity is, the higher is the number of codewords dedicated to error correction.

There are two types of erroneous codewords that can be corrected: *erasures* and *errors*. An erasure is produced due to a symbol character or module, the position of which is known, that cannot be scanned or decoded. An error is produced due to a misdecoded symbol character, the position of which is unknown.

The error correction capacity can be calculated using Equation 1, that shows the maximum number of erasures (e) and errors (t) that can be corrected given the number of error correction codewords (d) and the number of misdecode protection codewords (p)[1], both defined by the QR specification.

$$e + 2t \leq d - p \tag{1}$$

This capacity is calculated assuming an homogeneous distribution of the erasures and errors through the available data blocks. The reason is that each block of error correction codewords corrects the errors of a given block of data codewords.

2.4 Code Generation

The QR code generation involves the following steps:

1. **Data Analysis.** Selects the most appropriate data encoding, error detection and correction level, and symbol version to use, given the information to store.
2. **Data Encoding.** Encodes the data into a data stream according to the rules selected in the previous step, splits the stream in codewords and adds the needed padding.
3. **Error Correction Coding.** Generates the error correction codes for the blocks of codewords generated in the previous step.
4. **Structure Final Message.** Interleaves the data blocks and the error correction blocks and adds filling bits if necessary.
5. **Module Placement in Matrix.** Places the codewords into a matrix together with the additional information required.
6. **Masking.** Applies different mask patterns to the matrix generated in the previous step and selects the one that provides a better result for further symbol processing.
7. **Format and Version Information.** Generates format and, if required, version information of the symbol.

[1] $p = 3$ in version 1-L symbols, $p = 2$ in version 1-M and 2-L symbols, $p = 1$ in version 1-Q, 1-H and 3-L symbols, $p = 0$ in all other cases.

3 QR STEGANOGRAPHY METHOD

This section explains the method designed to insert hidden information in the QR code.

3.1 Overview

The method proposed to insert hidden information in a QR code exploits the error correction capability so that it is not detected by a regular QR reader.

The method is simple, it consists on replacing a certain number of codewords of data in the QR code with the information to hide. When the QR is read, the replaced codewords are considered as errors and are corrected by the error correction mechanism when read. Therefore, the added information remains hidden to the regular QR reader. In order to take full advantage from the error correction mechanism and maximise the capacity for hidden information, the modification has to be performed without altering the error correction codewords.

3.2 Designed Method Implementation

The information to hide into the QR code overwrites a subset of its data codewords. In order for the stegranography method to work, it is important that both the modified QR generator and the modified QR reader agree on where to locate the hidden data, which means, where to overwrite the data codewords. In our implementation, the new data is sequentially placed, from the beginning of the QR matrix (bottom-right position), overwritting the interleaved data codewords

of the different data blocks. This guarantees the maximum error correction capacity usage, since the same number of data codewords ± 1 are overwritten for each block.

For simplicity, in this implementation both the hidden and the original data are encoded with the 8-bit Byte encoding.

As in case of the location of added data, both the modified QR generator and the modified QR reader have to agree on how to indicate the termination of the hidden data. In our implementation, the end of the added data is indicated with two contiguous semicolon symbols ("::").

Considering the steps to create a QR code (see Section 2.4), the new data is injected into the QR code just before the masking step. The reason is that at this step the error codes have already been created, hence the original data is recoverable, and it is still possible to add the masking to create a balanced distribution of white and black symbols.

In order to read a QR code generated according to this method, the opposite steps are followed. Just after applying the mask, the hidden data is read, and after that the overwritten original data is automatically recovered thanks to the error correction codes.

3.3 Capacity

The amount of information that can be hidden in the QR, while still being able to recover the original message, is limited by the error correction capacity of the QR (see Section 2.3) and the actual errors.

Since the introduced hidden text will overwrite the original data with legitime symbols, each codeword of hidden data will count as one error in the Equation 1. Therefore, the error correction capabilities of the QR are reduced proportionally to the amount of hidden data. In case the combination of hidden data and legitimate errors overcomes the error correction capacity, the hidden data may be partially exposed to the regular QR reader.

Finally, we want to remark that that the hidden data is not protected against errors by the error correcting codes, and therefore any error when reading this data would not be solved. The hidden data can be additionally protected by incorporing its own error correcting codes. However, because they introduce redundancy, this would dramatically reduce the space available in the QR to hide information.

3.4 Implementation of an Android App

In order to test and prove the method described, an Android application to generate and read codes with additional hidden information has been created. As it can be seen in Figure 2, the application allows the user to enter two texts. Once the QR code is generated, one of the texts is readable using any standard QR reader, while the other is hidden and only readable using a modified reader, e.g. the application implemented (see Figure 3).

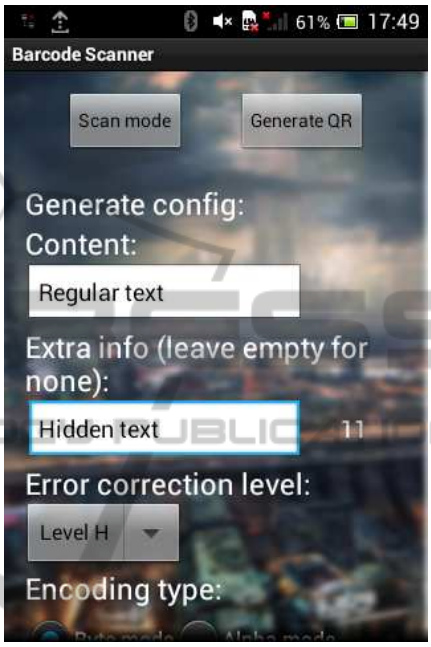

Figure 2: Application: generating code.

Figure 3: Application: reading code.

The application is based on the ZXing library (Zebra Crossing Barcode Scanner Library) (ZXing, 2014), which was modified to write and read QR codes with hidden information according to the method described using the 8 bit encoding. The application has been tested with the QR Versions 1-20 and the four levels of error correction. The rest of the QR Versions should also work.

4 QR CODES IN ELECTRONIC VOTING SYSTEMS

In the previous section we have seen a very simple method to apply steganography to the QR codes. This provides opportunities for attackers to exploit systems that make use of these codes, by implementing a hidden channel on them.

An interesting case of use is electronic voting. Some electronic voting systems integrate QR codes for representing some pieces of data to be used by the system and/or by the voters. In these systems a QR code can be used as a:

- Voter Credential: it is an authorization that is issued to the voter after being identified by the poll workers as an eligible voter, usually in poll-site electronic voting systems. The voter is required to present a valid voter credential to the voting device, in order to be able to cast a vote.
- Candidate: it is a representation of the candidate, information related to him, or the data required to cast the vote for him by scanning the code.
- Filled Ballot: it contains the voting options selected by the voter. The voting options can be in plaintext or encrypted, depending on the particular voting system.
- Proof: some electronic voting systems provide mechanisms for the voter to verify some parts of the voting process. Proofs for the voter are used to prove that the content of the vote to be cast matches the voter selections (*cast-as-intended* verification) or that the vote has been correctly stored in the ballot box (*recorded-as-cast* verification).

The following sections describe potential attacks to electronic voting systems using QR steganography. These attacks may affect security properties of the schemes, such as voter privacy or verifiability, depending on the use of the QR code.

4.1 QR as a Voter Credential

Some electronic voting systems, and in general those which are poll-site based, separate the identification of the voter from the vote casting process, in what is known as the two agencies model (Chaum, 1981; Fujioka et al., 1993), in order to provide voter privacy.

In such schemes, a poll-worker identifies the voter and checks her eligibility. After that, a temporal credential is issued to the voter. This credential will later be used by the voter to cast a vote in the voting booth in an anonymous way. Therefore the credential must not contain any personal voter's information, or any data which allows to trace a cast vote to a specific voter. QR codes can be used to represent such voter credential.

The proposed method for using steganography in QR codes can be used by a malicious voter credential issuer colluding with a malicious voting booth, in order to perform an attack against the privacy of the voter: the malicious voter credential issuer adds voter's personal information, or data which can be related to her (such as a timestamp or a sequential voter number), as hidden information in the QR containing the voter credential. The audit of the QR content using a regular QR reader does not reveal the hidden voter data. However, when the QR code is read by the malicious voting booth, it reads the hidden data and relates it to the preferences selected by the voter. Therefore, votes can be traced back to specific voters, breaking their privacy.

4.2 QR for Candidate Selection

A QR code printed in an electoral advertisement could be used to select the candidate or set of candidates in the voting booth (by scanning it). Despite QR codes have been used in political electoral publicity, those only included links to the candidates' websites. Currently, no relevant poll-site or remote electronic voting systems allow the usage of QR codes for casting a vote for a candidate. Only proof of concepts limited to academic environments (von Bergen, 2012) exist.

When using QR codes to encode the candidates to be selected, two cases have to be distinguished: generic codes and personalised codes. If the codes provided are generic, i.e. single copy of the code massively printed or present in printed public advertisements, no attacks against the election consisting of exploiting a hidden channel in the QR codes seems to be feasible. The reason is that in this case the QR codes cannot be personalised to include personal voter information with the vote cast. Nevertheless, if these QR codes are electronically distributed, e.g. using an smartphone application or via e-mail, there is a chance to personalise the code to include information about the voter to whom the code is delivered for breaking her privacy, e.g. the name or e-mail of the voter.

4.3 QR Containing a Filled Ballot

Several electronic voting systems use ballots that include QR codes to represent the voting preferences. Depending on the system, voting preferences are represented (encrypted or unencrypted) in the QR code

and/or in human readable form outside the QR code. Different attacks can be performed by taking advantage of the steganography technique described in Section 3.

4.3.1 Ballots with Plaintext Voter Selections

Some poll-site voting systems use paper ballots with QR codes encoding the voter preferences, in order to be able to speed up the counting phase: the QR code in the paper ballot may be scanned while the ballot is put in the ballot box, so that votes are electronically accumulated during the voting phase (Vegas, 2012). Another alternative is to scan the QR codes in the paper ballots stored in the ballot box after the voting phase has ended, in order to accumulate the votes electronically and obtain the results faster than by manual count (Volkamer et al., 2011).

Some voting systems use QR codes to encode the voter preferences in plaintext or, if the preferences are encrypted, they also include them printed as human readable text. These kind of ballots cannot include any information about the identity of the voter, since they would violate her privacy. An example of a voting system that includes a QR and the preferences printed in cleartext is the Belgian e-voting system (Vegas, 2012). In this case, to ensure the privacy of the voter, the voter authenticates and generates the ballot in a separate system than the one that reads and processes the ballot to store it in the ballot box.

In these type of systems, QR steganography provides a covert channel which may include the identity of the voter or other information that can be related to her (such as the time of ballot generation). Thus, this information can be used to trace voter preferences back to a voter.

This hidden information would be included by a malicious component in charge of generating the ballot. Then, a malicious electronic ballot box scanning the QR codes in the ballots would be able to generate a relation of ballots and voters, or a sorted list of ballots. In case the QR codes are not scanned by the ballot box, but manually, a malicious poll worker could obtain this same information. In both cases, the privacy of the voters would be seriously compromised.

4.3.2 Ballots with Encrypted Voter Selections

Other electronic voting systems use QR codes in the ballot to represent the encrypted vote. An example of this type of system is Wombat Voting (Farhi, 2013; wom, 2014). In Wombat, the voter uses a voting booth to select the vote preferences and then the vote is encrypted and printed into a QR code together with the vote preferences in plaintext (outside the QR code). The voter folds the ballot (to cover the plaintext printed preferences) and hands the ballot to the voting committee. The voting committee scans the QR code and, after it, the voter tears the ballot in half and inserts the printed preferences into a physical ballot box. The printed preferences stored in the physical ballot box are used for further auditing the result of the count of the votes scanned from the QR codes. The voter keeps the QR code as a vote receipt.

In these type of systems, the voter identity or any information that can be related to her can be included in the QR code or in the ballot, since the voter preferences are encrypted, as far as this information is removed before the decryption (for example by means of a mixnet (Chaum, 1981; Sako and Kilian, 1995)). However, QR steganography may give the chance to include the vote preferences in plaintext as hidden information in the QR code. In this case, it would still be possible to connect a voter with her vote in clear, so her privacy would be broken. Also, it can be used to obtain partial election results before it is allowed (when the votes are still supposed to be encrypted).

These attacks require the manipulation of both the voting booth and the electronic ballot box. In detail, when the ballot is generated by the voting booth the vote preferences in cleartext are hidden in the QR code. When the QR code is read and stored in the electronic ballot box, the machine can obtain partial counts that are not supposed to be performed and it can also link the vote preferences to the voter if the voter identification is also included among the legitime information.

4.4 QR Containing a Proof

Some electronic voting systems provide mechanisms for the voter to verify some parts of the voting process. Proofs for the voter are used to prove that the content of the vote to be cast matches the voter selections (*cast-as-intended* verification) or that the vote has been correctly stored in the ballot box (*recorded-as-cast* verification) (Gharadaghy and Volkamer, 2010).

QR codes can also be used for providing verifiability by holding such proof. An example of an electronic voting system that uses these codes for providing verifiability is the Estonian i-Voting system (Maaten and Hall, 2008). In this case the QR codes provide cast-as-intended verifiability (Elektroonilise hletamine komisjon, 2014): When a voter casts a vote, the vote preferences are encrypted with the Election Key using some randomness, and sent to a remote voting server. Then, the server returns a short receipt code that can be used to laterly retrieve

the encrypted vote. The randomness used for the encryption and the receipt code are encoded within a QR code that is provided to the voter, to be able to verify that her vote has been cast as intended. In order to perform the verification, the voter captures the QR code with a smartphone. The smartphone retrieves the encrypted vote from the server using the receipt present in the QR code as a reference. Later, it generates one encryption for each possible combination of voter preferences, using the Election Key and the randomness also included in the QR code. The encryptions generated are compared to the one received from the server, and the one that matches will determine the content of the vote cast by the voter. Then, the voter preferences are shown to the voter on the mobile screen. The voter makes sure the preferences shown match the intended ones (those she selected at the vote casting time).

In this case, QR steganography provides the chance to invalidate the verification mechanism. A compromised voting client and mobile application for verification can collude to pretend a vote is cast as intended when it is not, in the following way: the voting client stores the voting preferences in clear as hidden data in the QR code that will be read by the verification application. This application is able to present to the voter the selected voter preferences, even if the ones encrypted, received from the server, are different. Since this data is hidden in the QR, this cannot be detected by a regular QR reader.

Another possible attack is not based on deceiving the verification mechanism, but in taking advantage of the proofs given to the voter, in order to include some extra information which can allow her to sell her vote. As we explained before, in some voting systems such as in Wombat Voting (Farhi, 2013; wom, 2014), voters get a QR code as a vote receipt, which they can take back home after voting. This receipt can be later used by the voter to check that her vote has been correctly stored in the ballot box, by checking it against the ballot box contents which are published in a public bulletin board (Gharadaghy and Volkamer, 2010). A malicious voting booth can use QR steganography to hide the voter preferences of the vote cast in clear in the QR code the voter gets, so that a vote buyer with a modified QR reader can check them. Some proof of authenticity may be added in order to ensure that the QR code has been generated from a voting booth, such as a digital signature or a MAC. In this case, the hidden information could also be included in the signature or MAC, in order to ensure its authenticity in front of the vote buyer.

5 ATTACK DETECTION AND MITIGATION

In Section 4, we have shown some attacks that can be performed in electronic voting schemes, by using QR steganography. These attacks can be detected and, up to certain extent, mitigated. Despite the attacks cannot be totally avoided, some additional security measures can be applied.

A method to detect hidden information (or errors) in the QR is the following: the QR code is scanned by a regular QR reader, which returns the information without errors or hidden data since, if present, they are removed by the error correction mechanism. After that, a new QR code is generated using the information read from the first one. The two QR codes are compared, and any differences between them can be attributed to the presence of errors or hidden data. Thus auditors with trusted devices can analyze a sample of the QR codes generated, and detect if an unusual number of errors are found. This may alert that some kind of attack is happening, and allow triggering the activation of security measures accordingly.

Identification of hidden data is not trivial. In the design presented here, the hidden data follows a regular distribution and the hidden message ending is marked with a specific character. However, if other methods are used to distribute it, e.g. by pseudo-randomly locating it, the hidden data may be indistinguishable from errors.

The attacks can be mitigated by breaking the secret channel created between the malicious QR code writer and the reader. This can be performed by putting a trusted device in the middle of these two elements that removes from the issued QR codes any discrepancy with the original message, with the method described above.

The attacks described cannot be totally avoided, but some general security measures can be applied:

1. Trusted Build: The different software components used in the servers are reviewed by auditors and compiled and signed under their supervision. The application surface corresponding to QR code generation and reading is limited, so that it can be easily audited. Then, any manipulation of the software components would be detected.
2. Logical sealing of the systems: The different software components used in the servers can be protected with logical sealing, i.e. measures to detect manipulation of the software. This is only valid for the applications running in controlled environments, without including software running in the voters' computers.

Finally, another option is to modify the design of electronic voting systems using approaches not vulnerable to the secret channel attacks. In this cases it is worth to change the design for better security.

6 CONCLUSION

The use of QR Codes has increased during the last years, specially thanks to the possibility to scan them using smartphones. In addition, the usage of these codes has been integrated in applications, such as ticketing systems, tracking systems, and so on. As explained in this paper the field of electronic voting systems has not been an exception. QR codes have been integrated in different electronic voting solutions.

QR codes can be easily manipulated to include hidden information. In this paper we have demonstrated this by modifying a QR reader/writer application for smartphones that inserts and reads hidden messages to the QR codes. Thus, as we explained the usage of QR codes can be exploited to attack the privacy of voters and verifiability properties of electronic voting systems.

The threat and explotation technique of the electronic voting system using QR codes depends on the place and purpose of the code usage. Most of the attacks can be detected, and in some cases prevented, at QR level. Some additional security controls can be applied to increase the security of the system. As a conclusion we claim that QR codes can be used in electronic voting systems, but always considering the possibility these codes have to include hidden information. So the systems have to be designed accordingly and the corresponding additional measures put in place.

REFERENCES

(2014). Wombat Voting System. http://www.wombat-voting.com.

Chaum, D. L. (1981). Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90.

Chen, W.-Y. and Wang, J.-W. (2009). Nested image steganography scheme using QR-barcode technique. *Optical Engineering*, 48(5).

Chung, C.-H., Chen, W.-Y., and Tu, C.-M. (2009). Image hidden technique using QR-Barcode. In *Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2009. IIH-MSP '09*, pages 522–525.

Dey, S., Mondal, K., Nath, J., and Nath, A. (2012). Advanced steganography algorithm using randomized intermediate QR host embedded with any encrypted secret message: ASA_QR algorithm. *International Journal of Modern Education and Computer Science(IJMECS)*, 4(6).

Elektroonilise hletamine komisjon (2014). Verification of Internet votes. https://www.valimised.ee/eng/nutitelefon.

Farhi, N. (2013). An implementation of dual (paper and cryptograhic) voting system. Master thesis, Tel Aviv University.

Fujioka, A., Okamoto, T., and Ohta, K. (1993). A practical secret voting scheme for large scale elections. In Seberry, J. and Zheng, Y., editors, *Advances in Cryptology AUSCRYPT '92*, volume 718 of *Lecture Notes in Computer Science*, pages 244–251. Springer Berlin Heidelberg.

Gharadaghy, R. and Volkamer, M. (2010). Verifiability in electronic voting - explanations for non security experts. In Krimmer, R. and Grimm, R., editors, *Electronic Voting*, volume 167 of *LNI*, pages 151–162. GI.

Huang, H.-C., Chang, F.-C., and Fang, W.-C. (2011). Reversible data hiding with histogram-based difference expansion for QR code applications. *IEEE Transactions on Consumer Electronics*, 57(2):779–787.

ISO/IEC (2006). ISO/IEC 18004:2006. information technology – automatic identification and data capture techniques – qr code 2005 bar code symbology specification.

Lin, P.-Y., Chen, Y.-H., Lu, E., and Chen, P.-J. (2013). Secret hiding mechanism using QR barcode. In *2013 International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, pages 22–25.

Maaten, E. and Hall, T. (2008). Improving the transparency of remote e-voting: The estonian experience. In Krimmer, R. and Grimm, R., editors, *3rd international Conference on Electronic Voting 2008*, volume 131 of *LNI GI*, pages 31–43. Gesellschaft fr Informatik (GI).

Reed, I. S. and Solomon, G. (1960). Polynomial codes over certain finite fields. *Journal of the Society for Industrial & Applied Mathematics*, 8(2):300–304.

Sako, K. and Kilian, J. (1995). Receipt-free mix-type voting scheme. In Guillou, L. and Quisquater, J.-J., editors, *Advances in Cryptology EUROCRYPT 95*, volume 921 of *Lecture Notes in Computer Science*, pages 393–403. Springer Berlin Heidelberg.

Vegas, C. (2012). The new belgian e-voting system. In Kripp, M., Volkamer, M., and Grimm, R., editors, *5th International Conference on Electronic Voting 2012 (EVOTE2012)*, volume P-205 of *LNI GI*, pages 200–213. Gesellschaft fr Informatik (GI).

Volkamer, M., Budurushi, J., and Demirel, D. (2011). Vote casting device with VV-SV-PAT for elections with complicated ballot papers. In *Requirements Engineering for Electronic Voting Systems (REVOTE), 2011 International Workshop on*, pages 1–8.

von Bergen, P. (2012). Swissivi: Proof-of-concept for a novel e-voting platform.

ZXing (2014). Zxing (Zebra Crossing Barcode Scanner Library). https://github.com/zxing/zxing.