SBVLC: Secure Barcode-based Visible Light Communication for Smartphones

Bingsheng Zhang*†, Kui Ren*, Guoliang Xing‡, Xinwen Fu§, and Cong Wang¶
*Department of Computer Science and Engineering, State University of New York at Buffalo
†Department of Informatics and Telecommunications, National and Kapodistrian University of Athens
‡Department of Computer Science and Engineering, Michigan State University
§Department of Computer Science, University of Massachusetts Lowell
¶Department of Computer Science, City University of Hong Kong
Emails: *{bzhang26, kuiren}@buffalo.edu, ‡glxing@msu.edu, §xinwenfu@cs.uml.edu, ¶congwang@cityu.edu.hk

***Abstract*—As an alternative to NFC technology, 2D barcodes have been increasingly used for security-sensitive applications including payments and personal identification. However, the security of barcode-based communication in mobile applications has not been systematically studied. Due to the visual nature, 2D barcodes are subject to eavesdropping when they are displayed on the screen of a smartphone. On the other hand, the fundamental design principles of 2D barcodes make it difficult to add security features. In this paper, we propose SBVLC - a secure system for barcode-based visible light communication (VLC) between smartphones. We formally analyze the security of SBVLC based on geometric models and propose physical security enhancement mechanisms for barcode communication by manipulating screen view angles and leveraging user-induced motions. We then develop two secure data exchange schemes. These schemes are useful in many security-sensitive mobile applications including private information sharing, secure device pairing, and mobile payment. SBVLC is evaluated through extensive experiments on both Android and iOS smartphones.**

I. Introduction

Short-range communication technologies including *near field communication* (NFC) and 2D barcodes have enabled many popular smartphone applications such as contactless payments, mobile advertisements, and device pairing. Evolved from the RFID technology, NFC can enable reliable low-power communication between RF tags and readers. However, NFC requires additional hardware and has been supported by only a few smartphone platforms on the market. Recent studies have shown that NFC is subject to security vulnerabilities such as eavesdropping and jamming [7], [5]. Compared with NFC, 2D barcodes have enjoyed a significantly higher penetration rate in mobile applications. This is largely due to the extremely low barrier to adoption – almost every camera-enabled smartphone can read and process 2D barcodes. As an alternative to NFC, 2D barcodes have been increasingly used for security-sensitive applications including mobile payments and personal identification. For instance, PayPal recently rolled out a barcode-based payment service for retail customers [1]. As one of the most anticipated new features of iPhone 5, the Passbook App stores tickets, coupons, and gift/loyalty cards using barcodes.

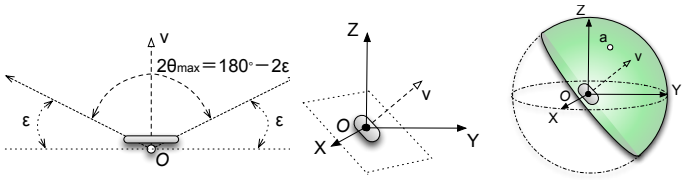
However, the security of barcode-based communication in mobile applications has not been systematically studied. Due to the visual nature, 2D barcodes are subject to eavesdropping when they are displayed on the screens. The proliferation of smartphones puts a portable camera in everyone's pocket, making eavesdropping much easier. This is exacerbated by widely spread use of surveillance cameras in public areas like shopping malls. On the other hand, the fundamental design principles of 2D barcodes make it difficult to add security features. First, a 2D barcode only contains a small amount of information and hence cannot simply adopt advanced encryption primitives. Moreover, most existing barcode applications are based on a single barcode exchange, which is insufficient for establishing a secure communication channel. Recently, several systems are designed to stream a series of barcodes between a LCD screen and smartphone camera [10][8]. These systems can enable high-throughput ad hoc communication between smartphones; however, they are designed based on highly customized barcodes which are not widely adopted in practice. In this paper, we investigate secure barcode-based communication for smartphones. Due to the inherent directionality, the *visible light communication* (VLC) channel of barcode exchanges yields interesting security properties. We formally analyze the security of VLC based on geometric models and propose security enhancement mechanisms such as manipulating view angles and leveraging user-induced motions. We then develop three secure data exchange protocols that encode information in barcode streams. We believe that the proposed protocols are useful in many mobile applications including private information sharing, device pairing, and mobile payment systems, etc.

Contributions. We propose SBVLC (Secure Barcode-based Visible Light Communication) – a novel secure wireless communication system for smartphones. Unlike NFC, SBVLC can be widely adopted by most off-the-shelf smartphones. It works across various smartphone platforms equipped with a color screen and a front-facing camera. Our system can also be easily extended to support other mobile and portable devices such as laptops and tablets. We use 2D/3D geometric models to examine the security of proposed SBVLC. To the best of our knowledge, this work is the first that focuses on modelling and analyzing the security of VLC channel and barcode-based communication between smartphones. Specifically, we first design a real-time duplex screen-camera VLC channel based on 2D barcode streaming. By embedding extra information into the color of *quick response* (QR) codes, we developed a fast QR filtering technique to quickly remove the non-QR and

978-1-4799-3360-0/14/$31.00

(a) Data exchange between smartphones. (b) Mobile payment systems.

Fig. 1: SBVLC System Architecture.



(a) 2D screen model. (b) 3D screen model. (c) 3D visible zone.

Fig. 2: Screen Model and Visible Zone.

duplicate QR frame images. On top of the duplex VLC channel, we further propose two secure communication schemes. The first scheme, *two-phase message transfer scheme*, is an ultra-lightweight solution to secure smartphone data exchange and mobile payment. The second scheme, *all-or-nothing data streaming scheme*, is tailored for secure temporary data transfer without key exchange. It adopts the secret sharing scheme to enhance the channel security — it preserves the confidentiality of all the transmitted data, if the eavesdropper misses at least one barcode frame during the entire communication. Both proposed schemes are evaluated through extensive experiments on both Android and iOS smartphone platforms.

Road Map. The rest of this paper is organised as follows. Sec. II introduces our design goal and system the architecture. In Sec. III, we present 2D and 3D geometric security models. In Sec. IV, we enable a real-time one-way screen-camera VLC channel based on color QR codes. In Sec. V, we propose various physical protection approaches; we then develop and analyse two secure communication schemes: (a) two-phase message transfer scheme; (b) all-or-nothing data streaming scheme. In Sec. VI, we study the compatibility, usability and robustness of SBVLC. Finally, Sec. VII summarises related work, and a conclusion is given in Sec. VIII.

II. Design Goal and System Architecture

Our goal is to enable secure barcode-based communication between smartphones. The focus is to achieve data confidentiality against eavesdropping. Designed for off-the-shelf smartphone platforms, SBVLC should be lightweight. For example, it is implausible to establish a secure channel for a single-barcode communication with overhead of multiple-round barcode exchange. In the case of exchanging large files, we still want to avoid the use of computationally expensive public-key cryptographic primitives. We note that the security of NFC relies on Diffie-Hellman key exchange [2], [3].

The communication mode of SBVLC is ad-hoc in that the sender and the receiver are not expected to have a common shared secret knowledge such as secret key in priori to the communication. Similar to NFC setting, there is an air interface between the sender and the receiver, and the typical reception distance is also a few inches. As shown in Fig. 1 (a) and (b), SBVLC supports secure data exchange for both smartphone-smartphone and smartphone-terminal scenarios. SBVLC works on top of a fully duplex VLC channel, and thus the smartphones must be equipped with a color screen and a front-facing camera as the sender and the receiver are required to 'talk' to each other simultaneously. SBVLC works among various mobile platforms without specific requirement on the screen size and camera resolution, but a better specification usually leads to higher communication throughput.

III. Security Model

Successful defense against eavesdropping vastly depends on careful analysis of the attack scenarios and adopting suitable protection mechanisms based on the analysis. We first build formal 2D/3D geometric security models in this section. The 3D model reflects the situation in reality, but the 2D model is also useful and intuitive, because we can always take a projection map $P : \mathbb{R}^3 \to \mathbb{R}^2$ and project all the objects onto a plane, e.g. we can map any point (x, y, z) in the 3D space $\mathbb{R}^3$ to a point $(x, y, 0)$ as its projection on the x-y plane, which is the horizontal plane.

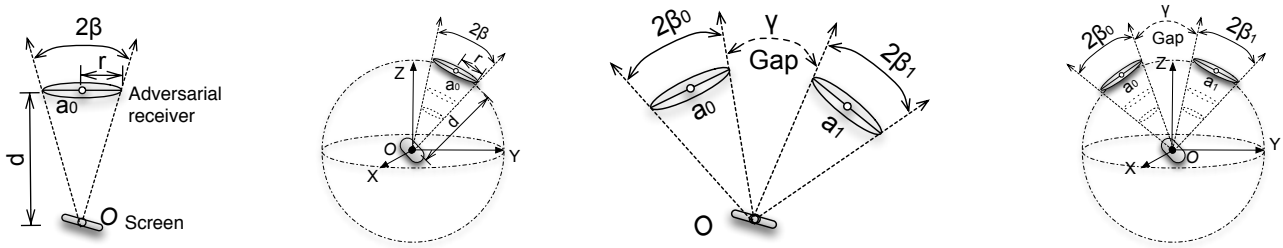
A. 2D/3D screen geometric model

The typical screen size of a mainstream smartphone platform is 3-5.5 inches. One important feature of a smartphone screen addressed in our model is its visible angle. A 2D screen model with visible angle $2\theta_{\mathsf{max}}$ is depicted in Fig. 2 (a), where the screen is represented as an interval, and the vertex of the screen visible angle is located at the origin $\mathcal{O}$. The visible angle is denoted as $2\theta_{\mathsf{max}} = 180° - 2\varepsilon$. Since ε is usually small, given a typical smartphone screen size, the distance between $\mathcal{O}$ and the screen center is less than 0.1 inch, which is negligible to an adversary who is far away; hence, we ignore the tiny offset between $\mathcal{O}$ and the screen center. Similarly, the screen can be modelled as a plane that passes through the origin in the 3D model. We describe the screen orientation by quantifying its normal vector $\boldsymbol{v} \in \mathbb{R}^3$. As shown in Fig. 2 (b), such plane is uniquely determined by its normal vector $\boldsymbol{v}$, so we denote the screen plane as $\mathsf{pl}(\boldsymbol{v})$. In order to address the notion of visibility, we define the *visible zone* in the 2D/3D model as follows.

Definition Let $t \in \{2, 3\}$. Let $\boldsymbol{v} \in \mathbb{R}^t$ be a normal vector and $\varepsilon \in [0°, 90°]$ be an angle. The visible zone of the screen plane $\mathsf{pl}(\boldsymbol{v})$ is denoted as:

$$\mathsf{Vis}_t(\boldsymbol{v}, \varepsilon) = \left\{ \boldsymbol{u} \in \mathbb{R}^t \,\middle|\, \frac{\boldsymbol{v} \cdot \boldsymbol{u}}{||\boldsymbol{v}||_2 \cdot ||\boldsymbol{u}||_2} \geq \sin(\varepsilon) \right\} \quad .$$

According to this definition, if a receiver is at location $\boldsymbol{a} \in \mathsf{Vis}_t(\boldsymbol{v}, \varepsilon)$, then the receiver is able to capture information emitted by the screen. (c.f. Fig. 2 (c).) Hence, the distance factor is not taken into account in our notion of visibility. The transmission rate decreases along with the increase of the distance between the transmitter and the receiver for a typical VLC channel. However, it only offers a fuzzy security guarantee, because it is hard to make assumptions on the attackers' devices. For the sake of uniformity, we don't differentiate the visibility in terms of distance.



(a) 2D single-rcvr adv. (b) 3D single-rcvr adv. (c) 2D multi-rcvr adv. (d) 3D multi-rcvr adv.

Fig. 3: Single-receiver and Multi-receiver Adversarial Models.

B. 2D/3D Single-receiver adversarial model

In the *single-receiver adversarial model*, the eavesdropper uses only one optical receiver during an attack. This is the most common attack scenario in practice: a curious eavesdropper first occasionally discovers a barcode based communication, and he/she then tries to eavesdrop the communication with his/her carried optical receiver, e.g. a camera or a smartphone. Without loss of generality, the optical sensors of those receivers can be in arbitrarily sharp; in the t-D model, $t \in \{2, 3\}$, for a given optical sensor $\mathcal{D} \subseteq \mathbb{R}^t$, there exists a point $\boldsymbol{a_0} \in \mathbb{R}^t$ such that $\mathcal{D} \subseteq B(\boldsymbol{a_0}, r)$ with a minimum radius $r \in \mathbb{R}$, where $B(\cdot, \cdot)$ denotes a ball. The adversarial receiver is represented by the ball $B(\boldsymbol{a_0}, r)$ in our security analysis, and we note that the adversarial capability is (presumably) increased by this approximation. We assume that the shooting angle of the adversarial receiver can be optimized instantly. Whereas, we don't consider the case that an adversary can physically move his/her receiver a long distance away from its initial position during a very short period. Hence, position of the adversarial receiver is supposed to be fixed during eavesdropping.

As shown in Fig. 3 (a), the adversary's receiver can be represented as an interval with length $2r$ in the 2D model. Let the phone screen be at the origin $\mathcal{O}$, and the distance between the screen and the adversary's receiver is $d = ||\boldsymbol{a_0}||_2$. One can easily deduce the adversary's capture cone aperture as $2\beta = 2 * \arctan\left(\frac{r}{d}\right)$. Recall that the distance d does not affect the eavesdropping successful rate in our security model. Therefore, in rest of this paper, we only quantify the adversary by the angle β and the position $\boldsymbol{a_0}$ when r and d parameters are not important in the context. Denote the single-receiver adversary as $\mathsf{Adv}_s(\boldsymbol{a_0}, \beta)$. We define the *adversarial capture cone* of $\mathsf{Adv}_s(\boldsymbol{a_0}, \beta)$ as follows.

Definition Let $t \in \{2, 3\}$. The adversarial capture cone of a single-receiver adversary $\mathsf{Adv}_s(\boldsymbol{a_0}, \beta)$ is

$$\mathsf{c}_t(\boldsymbol{a_0}, \beta) = \left\{ \boldsymbol{u} \in \mathbb{R}^t \,\middle|\, \frac{\boldsymbol{u} \cdot \boldsymbol{a_0}}{||\boldsymbol{u}||_2 \cdot ||\boldsymbol{a_0}||_2} \geq \cos(\beta) \right\} \quad .$$

Clearly, all the source beam emitted from the origin $\mathcal{O}$ that lies inside the adversarial capture cone $\mathsf{c}_t(\boldsymbol{a_0}, \beta)$ can be captured by the single-receiver adversary $\mathsf{Adv}_s(\boldsymbol{a_0}, \beta)$. Therefore, we can define 'visibility' as follows.

Definition Let $t \in \{2, 3\}$. We say that the screen is visible to a single-receiver adversary $\mathsf{Adv}_s(\boldsymbol{a_0}, \beta)$, if and only if $\mathsf{Vis}_t(\boldsymbol{v}, \varepsilon) \cap \mathsf{c}_t(\boldsymbol{a_0}, \beta) \neq \emptyset$.

The radius r is usually very small in practice. Whenever $r << d$, we have $\beta \approx 0$. We refer this special type of single-receiver adversary $\mathsf{Adv}_s(\boldsymbol{a_0}, 0)$ as *single-point adversary*.

C. 2D/3D Multi-receiver adversarial model

We now model a more powerful type of adversaries, who are able to control multiple optical receivers during an attack. We begin with *two-receiver adversary*, and Fig. 3 (c) and (d) illustrates the situation in the 2D/3D model. There are a gap with angle γ (on the $\boldsymbol{a_0}$-$\boldsymbol{a_1}$ plane) between two adversarial capture cones $\mathsf{c}_t(\boldsymbol{a_0}, \beta_0)$ and $\mathsf{c}_t(\boldsymbol{a_1}, \beta_1)$, where $t \in \{2, 3\}$ and

$$\gamma = \arccos\left(\frac{\boldsymbol{a_0} \cdot \boldsymbol{a_1}}{||\boldsymbol{a_0}||_2 \cdot ||\boldsymbol{a_1}||_2}\right) - \beta_0 - \beta_1 \quad .$$

Denote $\mathsf{Adv}_m(\boldsymbol{a_0}, \beta_0, \boldsymbol{a_1}, \beta_1, \gamma)$ as the two-receiver adversary.

We now reduce a two-receiver (or multi-receiver) adversary to a single-receiver adversary by the following theorem.

Theorem 3.1: In the 2D model, if $\gamma < 2\theta_{\mathsf{max}}$, for a screen with visible angle $2\theta_{\mathsf{max}}$, there exists $\boldsymbol{a^*}$ such that

$$\mathsf{Adv}_s(\boldsymbol{a^*}, \beta_0 + \beta_1 + \gamma) \equiv \mathsf{Adv}_m(\boldsymbol{a_0}, \beta_0, \boldsymbol{a_1}, \beta_1, \gamma) \quad .$$

Proof: We want to show that a two-receiver adversary $\mathsf{Adv}_m(\boldsymbol{a_0}, \beta_0, \boldsymbol{a_1}, \beta_1, \gamma)$ is equivalent to a single-receiver adversary $\mathsf{Adv}_s(\boldsymbol{a^*}, \beta_0 + \beta_1 + \gamma)$ for some $\boldsymbol{a^*}$. Consider an adversary who uses additional devices to fill the blind spot between those two adversarial capture cone, so that he/her can also capture the source beam from the screen that falls into the gap. This modified adversary has a continuous capture aperture $\beta_0 + \beta_1 + \gamma$, so he/she can be considered as a single-receiver adversary $\mathsf{Adv}_s(\boldsymbol{a^*}, \beta_0 + \beta_1 + \gamma)$, where $\boldsymbol{a^*}$ lies on the angle bisector. We need to show that this modified adversary has the same capture capability as the original two-receiver adversary. Indeed, they are different if and only if there exists $\boldsymbol{v}$ such that the visible zone $\mathsf{Vis}_2(\boldsymbol{v}, \varepsilon)$ has intersection with the gap but has no intersection with either capture cones $\mathsf{c}_2(\boldsymbol{a_0}, \beta_0)$ or $\mathsf{c}_2(\boldsymbol{a_1}, \beta_1)$. Since $\gamma < 2\theta_{\mathsf{max}}$, such $\boldsymbol{v}$ does not exist. Hence, $\mathsf{Adv}_s(\boldsymbol{a^*}, \beta_0 + \beta_1 + \gamma) \equiv \mathsf{Adv}_m(\boldsymbol{a_0}, \beta_0, \boldsymbol{a_1}, \beta_1, \gamma)$. ■

IV. SBVLC CHANNEL CODING SCHEME DESIGN

First of all, we need to enable a one-way real-time VLC channel between smartphones. Note that all kinds of 1D and 2D barcodes can be the channel coding candidates. Our prototype adopts QR code due to its advantages over other conventional barcodes, including high information density per code and low sensitivity to varying lighting conditions and

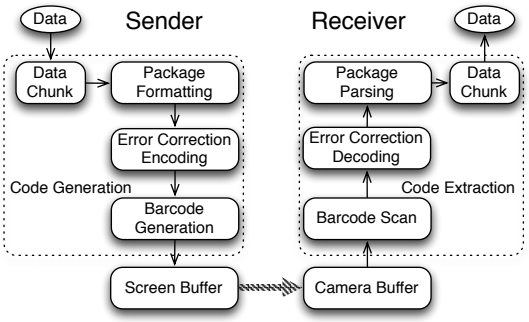


Fig. 4: 2D Barcode Streaming.

angles. As depicted in Fig. 4, the barcode streaming system runs between a sender and a receiver. At the beginning of a data transmission, the sender divides the data string into several data chunks. The size of each data chunk depends on the system parameters such as the maximum storage capacity of a single barcode and the rate of the employed error correcting codes (ECC). Each data chunk is formatted to a package by adding a 32-bit sequence number in the header. Let ℓ_{max} be the maximum package size, which is the storage capacity of a single barcode. The data chunk size is the payload size $\ell_p = \ell_{\mathsf{max}} - 32$ bits. The package is encoded by ECC to a frame block, which is then processed to generate a barcode. The prepared barcodes are sequentially displayed on the sender's screen at a certain frame refresh rate. The receiver starts the decoding process as soon as the first barcode frame is captured by its front-facing camera. The successful barcode decoding process outputs a frame string, which is then decoded by ECC. Finally, the data string is assembled from those data chunks.

A. Determining optimal system parameters

SBVLC uses the 8-bit binary mode (mode indicator '0100') for QR code generation. The main system parameters that need to be decided includes the QR version, error correction level and frame refresh rate. In order to determine the proper ECC level, we did statistical test from QR version 1 to 20 on iPhone 4S, Google Nexus S and Samsung Galaxy S3. The result shows that low ('L') ECC level is sufficient in our usage scenario, and there is no correlation between the barcode decoding success rate and the error correction level even for high QR versions. Hence, we pick low ('L') ECC level for better throughput.

In order to achieve a real-time system, we must ensure that each barcode can be encoded and decoded on time. The charts in Fig. 5 show the performance evaluation of single-thread encoding and decoding running time tested on both Nexus S and Galaxy S3. To determine the proper frame refresh rate, we first tested the screen refresh rate and camera capture rate. Our experiment shows that the average time taken to refresh a QR frame screen is roughly the same among different platforms, ranging from 20 to 22 ms. Hence, displaying QR codes is not the system bottleneck unless the frame refresh rate is above 40 frames per second (FPS). On the country, the major challenges are brought by the low camera capture rate. Our system prototype fetches camera image preview using standard callback API on Android systems and avcapturesession API on iOS systems. The corresponding image capture rates of the front-facing cameras with image size 640×480 on Nexus S, Galaxy S3 and iPhone 4S are 8.3, 25.4 and 30.3 FPS, respectively. Since SBVLC requires a fully duplex two-way VLC communication between smartphones, the front-facing camera capture rate is crucial. We did channel robustness test to determine the frame refresh rate cap, and the result confirms our conjecture that the frame refresh rate cap τ_{max} should be roughly half of the camera capture rate. Denote $t_{\mathsf{enc}}(i)$ and $t_{\mathsf{dec}}(i)$ be the average encoding and decoding running time (in seconds) of a version-i QR code. We can estimate the ideal frame refresh rate as

$$\tau_f(i) = \max\left(\tau_{\mathsf{max}}, \frac{1}{\max(t_{\mathsf{enc}}(i), t_{\mathsf{dec}}(i))}\right) .$$

B. Constructing fast QR filtering

Since the frame refresh rate cap is about half of the camera capture rate, it is expected to have multiple camera frame images for the same QR code. So we have to construct an efficient filter to remove duplicated QR frame images. In addition, the filter should also be able to remove those images that does not contain a QR code before submitting them for decoding. We now propose a novel fast QR filtering technique to remove those non-QR and duplicated QR frame images with only a few image pixel sample lookups.

We utilize the color screen of a smartphone, and let the sender display the QR codes in blue and red alternating order such that any two consecutive QR codes are in different colors. Therefore, we can embed extra information into the colors of the QR codes while maintaining the traditional QR code functionality. Once the receiver captures a frame image, it randomly picks N pixel samples in the central area of the image. According to the RGB value of each pixel, the receiver then classifies the pixels into three bins: 'blue', 'red' and 'none'. The receiver will then make decisions based on the weight of those bins. Let $\boldsymbol{p_i} = [R_i, G_i, B_i]^T$ be the RGB vector of the i-th sampled pixel. Define the RGB vectors of red and blue as $\boldsymbol{p_r} = [255, 0, 0]^T$ and $\boldsymbol{p_b} = [0, 0, 255]^T$ respectively. Denote σ as a threshold value, the image is classified as 'red' if $||\boldsymbol{p_i} - \boldsymbol{p_r}||_1 < \sigma$ and 'blue' if $||\boldsymbol{p_i} - \boldsymbol{p_b}||_1 < \sigma$; otherwise, it is classified as 'none'. According to the largest weight of these three bins, the classifier will return 'Red', 'Blue' or 'None', indicating that the image contains a red QR code, a blue QR code or no QR code, respectively. In the context of our system, no QR code means there is no red or blue QR code.

We set the parameter $N = 80$ and run experiments to determine the proper threshold σ. During our experiment, we found that both 'red' and 'blue' bins are constantly empty when the test images contain no QR code, even with threshold $\sigma = 150$. The classifier fails to correctly detect the 'red' color when $\sigma \leq 70$, but the weight of 'red' bin catches up quickly along with the increase of threshold. Aftermath, we select $\sigma = 110$ to tolerate the chromatic aberration caused by different smartphones' display screens and cameras. Our empirical result shows that our classifier can distinguish a image that contains no QR, a red QR or a blue QR with 100% accuracy. Its JAVA implementation on Android systems runs in < 0.1 ms on all tested smartphone platforms. Equipped with this classifier, the receiver is able to quickly filter the duplicated QR images with nearly no computational overhead by removing the duplicated QR images in the same color.

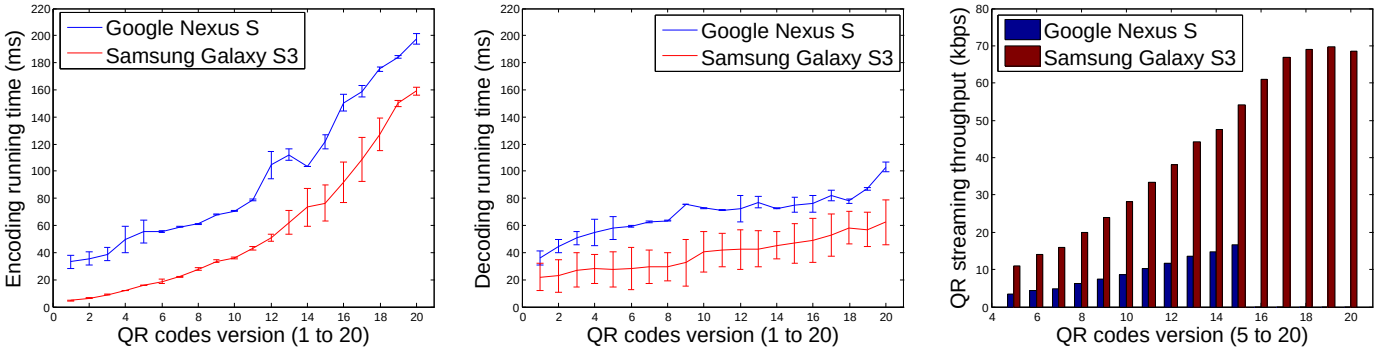


Fig. 5: QR barcode streaming performance.

C. Channel realization

We implemented the system on both Android and iOS, borrowing some parts of the open source QR library [4]. We set the frame refresh cap $\tau_{\max}$ as 5 FPS and 13 FPS for Google Nexus S and Samsung Galaxy S3, respectively. For single-thread encoding/decoding version, we found that the throughput bottleneck becomes the encoding time at the sender-end for higher QR versions in the Samsung Galaxy S3 case. Fortunately, most latest mainstream smartphones are equipped with multi-core CPUs, for instance, iPhone 4S is armed with a dual-core CPU and Galaxy S3 is armed with a quad-core CPU. To explore the benefit of multi-core CPUs, we deploy multiple encoding/decoding threads. On Galaxy S3, with 3 encoding threads, the amortized encoding time for QR version 20 is reduced under 90 ms, which is sufficient to send 10 QR codes per second. At the receiver end, once a frame image is captured by the camera, the receiver first uses our fast QR filter to remove the duplicated QR frames and non-QR frames. The filtered image will be pushed into the decoding queue to be decoded by multiple decoding threads.

Because small camera preview image size leads to higher camera capture rate and lower CPU usage. Our system uses adaptive camera preview image resolutions ranging from 192×144 to 800×600 for different QR versions. We tested the QR streaming throughput on both Google Nexus S and Samsung Galaxy S3 from QR version 5 to 20. As illustrated in the right bar chart of Fig. 5, the channel throughput for Samsung Galaxy S3 reaches its peak at 70 kbps with QR version 19. The throughput bottleneck switches from the frame refresh cap to the limited computation resource after QR version 18, and that's why the throughput starts to drop after version 20. On Nexus S, it can only decode the QR codes up to version 15 due to its poor front-facing camera resolution; thus its maximum throughput is below 20 kbps.

V. The Proposed SBVLC Schemes

A. Two-phase message transfer scheme

After building a high-throughput real-time VLC channel, we are ready to focus on the security aspects. In particular, we are going to show that the communication system can achieve much higher security level once it has a duplex VLC channel. Consider the following scenario: Bob wants to share dozens of his contacts with his friend Alice. VLC seems to be an adequate tool to accomplish this task, because it is extremely simple to setup. However, an eavesdropper can shoulder sniff all the information if he/she can 'see' Bob's smartphone screen. To overcome this security issue, we propose the first scheme of SBVLC – two-phase message transfer scheme.

1) Protocol design: By combining two opposite-directional one-way screen-camera VLC channels, we can enable a fully duplex two-way VLC channel such that both smartphones are able to 'talk' to each other simultaneously. We utilize this feature to construct a more secure message transfer protocol as follows. Let ℓ_p be the payload capacity of a single barcode. The sender first divides the data into n chunks with size ℓ_p. Suppose that the sender wants to send the receiver one data chunk $M_i \in \{0,1\}^{\ell_p}$. They do the following steps: (1) The receiver first randomly picks $R_i \leftarrow \{0,1\}^{\ell_p}$ and sends R_i to the sender through the receiver-sender VLC channel; (2) The sender fetches R_i and sends $C_i := M_i \oplus R_i$ to the receiver through the sender-receiver VLC channel; (3) The receiver fetches C_i and returns $M_i := C_i \oplus R_i$.

Naively, both smartphones can repeat the above procedure n times to send n data chunks. For $i \in [n]$, both the sender and receiver set a counter $\mathsf{ctr} = i$ and put the counter in the frame header while transferring the i-th data chunk. The receiver first encodes the i-th random frame to a barcode and displays it on its screen. Meanwhile, the receiver keeps checking each frame image captured by the camera, trying to decode a new incoming barcode. Once C_i is received, the receiver extracts $M_i = C_i \oplus R_i$ and repeats the same procedure for data chunk M_{i+1}. Similarly, the sender tries to decode a incoming barcode for R_i. Upon success, the sender encodes $C_i = M_i \oplus R_i$ to a barcode and displays it on its screen. After that, the sender is waiting for the next incoming barcode. In such way, for QR version j, transferring each data chunk M_i takes

$$t(j) = 2 * (t_{\mathsf{enc}}(j) + t_{\mathsf{dec}}(j)) + t_{\mathsf{delay}} \ ,$$

where $t_{\mathsf{enc}}(j)$ and $t_{\mathsf{dec}}(j)$ are the running time of encoding and decoding for QR version j and t_{delay} is the system delay.

To improve the performance, we propose the *lazy decoding technique* as follows. First of all, since the random frames are independent to the messages, the receiver can prepare the QR codes for random frames during any spare time or even offline. Secondly, we notice that the QR decoding success rate is very high, and thus the image can usually be decoded correctly once it passes our fast QR filter. Therefore, upon receiving a NewBarcode, the receiver can first display the prepared the

QR code for the next random frame and then try to decode the NewBarcode. If decoding fails, the receiver can simply set the counter ctr of the next random frame to be the missing sequence number, and the sender will try to send the indicated data chunk again. After decoding, the receiver first recovers the message and then prepare the random QR for the next round. By applying our lazy decoding technique, the system time to transfer each data chunk is reduced to $t(j) = t_{\mathsf{enc}}(j) + t_{\mathsf{dec}}(j) + t_{\mathsf{delay}}$.

2) User interface design: We put a small camera preview window at the top of the screen to help the user to quickly align two smartphones such that both QR frame areas are captured by each others' front-facing cameras. Once the alignment is done, the preview window is shadowed at the beginning of the data transmission due to security concerns. Alternatively, we can also blur the preview images on the fly such that the preview images can still assistant users for alignment but the blurred QR codes in the preview window can't be decoded. Hence, we can keep the blurred preview all the time during the whole process. However, we found that the real-time blurring process leads significant computational overhead, and it effects the performance in current smartphone environment. Therefore, we prefer the shadowing based solution. The two smartphones are expected to be in opposite direction during a communication. Our experiment shows this alignment can minimise the image distortion caused by the viewing angle, and thus the system is more robust.

3) Security analysis: It is easy to see that C_i itself does not reveal any information about M_i. Therefore, the eavesdropper has to 'see' both screens in order to recover the message M_i; however, the time interval between sending R_i and C_i is only a few milliseconds. We can consider both smartphones are sending the corresponding QR codes at roughly the same time. We now show that our two-phase message transfer scheme preserves confidentiality of the transmitted data string against single-point adversaries in distance. Recall that 'visibility' is defined as the intersection between the adversarial capture cone and the visible zone(s). As depicted in the left of Fig. 6, the distance between two smartphones is around 10 cm, and we define the middle of two phones as the origin. An eavesdropper must be in the shadowed area to simultaneously 'see' both phone screens. We bound this shadowed area as a minimum ball $B(\mathcal{O}, d_{\mathsf{save}})$, where $d_{\mathsf{save}} = \frac{5}{\tan(\varepsilon)}$ cm. Hence, $\forall a_0 \notin B(\mathcal{O}, d_{\mathsf{save}})$, a_0 cannot be in both visual zones simultaneously, In other word, if the single-point adversary is more than d_{save}-distance away, then the data confidentiality is preserved. Plugging in the widest smartphone screen visible angle, $\varepsilon = 2°$, we have $d_{\mathsf{save}} \approx 143$ cm. It means that all the single-point adversaries who are more than 1.4 m away cannot eavesdrop the message regardless the quality of their optical devices. On the other hand, any adversary within the range can be easily detected by the user in most circumstances.

If the smartphones are equipped the privacy screen projectors, (which is widely available in current market e.g. [6]), the system achieves much stronger security guarantees. According to [6], the contrast ratio drops to nearly 0 when the viewing angle is more than 60°. It means that the maximum visible angle is $2\theta = 2 * 60°$ for a screen equipped with a privacy screen projector. Hence, we have $d_{\mathsf{save}} \approx 8.66$ cm. It is almost impossible for an adversary to be in this range without being noticed in practice. During our experimental validation of the security guarantees, we found that there exists no angle such that the camera can 'see' both screen simultaneously in about 1 foot distance. In general, SBVLC is secure against an adversary with receiver radius r in distance d such that $\arctan\left(\frac{r}{d-d_{\mathsf{save}}}\right) < \varepsilon \approx 30°$. When $d_{\mathsf{save}} << d$, we can approximate $d \approx d - d_{\mathsf{save}}$; thus the system can tolerate any single-receiver adversary with $\beta < 30°$. We can also generalize the result to the 3D case by the following theorem.

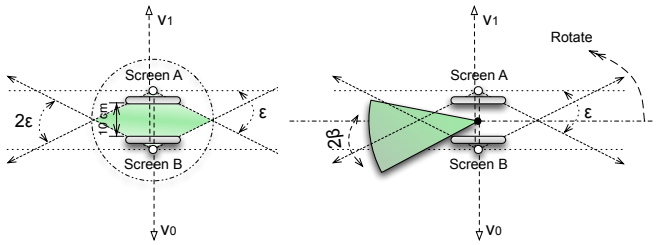


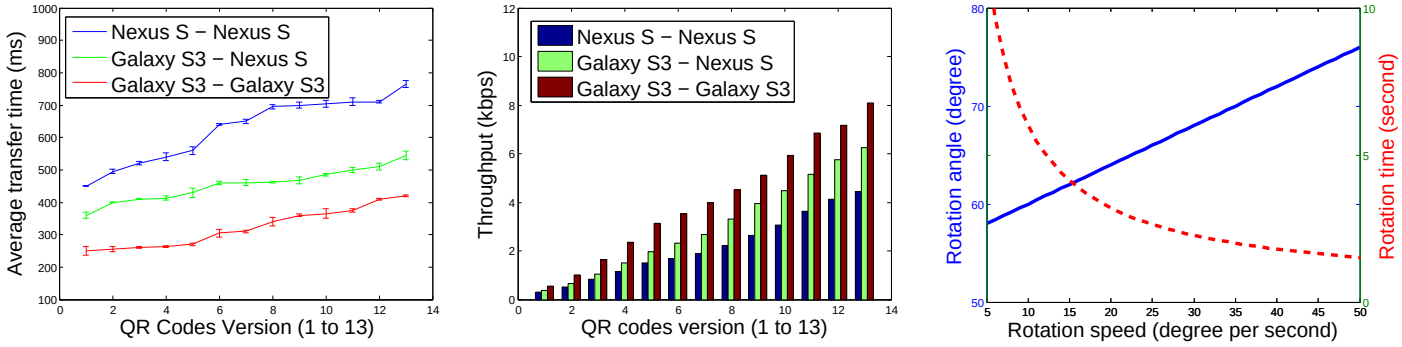
Fig. 6: Security of Two-phase Message Transfer.

Theorem 5.1: $\forall \boldsymbol{a} \in \mathbb{R}^3$, the $(\boldsymbol{a}, \beta)$-single-receiver adversary with $\beta < \varepsilon$ is not capable of eavesdropping any information about the data transmitted by the two-phase message transfer scheme.

Proof: Since the visibility is defined as the intersection between screen visual zones and the adversarial capture cone. It is easy to see that when $\beta < \varepsilon$ the adversarial capture cone $\mathsf{c}_3(\boldsymbol{a}, \beta)$ cannot simultaneously intersect with both screen visible zones $\mathsf{Vis}_3(\boldsymbol{v_0}, \varepsilon)$ and $\mathsf{Vis}_3(\boldsymbol{v_1}, \varepsilon)$, where $\boldsymbol{v_0} = -\boldsymbol{v_1}$. Therefore, at least one of the two phone screens is invisible to the adversary at any given time, so the claim holds for all $\boldsymbol{a} \in \mathbb{R}^3$. ■

4) Implementation and performance: Using fully duplex VLC channel, our two-phase message transfer scheme naturally confirms message delivery, so that we don't need a frame refresh cap to avoid missing QR frames. The scheme requires that both the sender-receiver and receiver-sender VLC channels, so its computational requirement is nearly twice higher than the conventional one-way message transfer. The left chart in Fig. 7 (a) shows the average time taken for one data chunk transfer on Galaxy S3 and Nexus S. In the Galaxy S3 case, the average time is between 200 and 300 ms for low QR versions, and it grows gradually as long with the increase of QR versions. The communication throughputs for QR version 13 is above 8 kbps in the Galaxy S3 case.

The system throughput depends on 3 important factors: the smartphone front-facing camera capture rate, the encoding/decoding time and the storage capacity per single barcode. Hence a smartphone with high camera capture rate may lead to high throughput, for example it was said that iPhone 4S running on iOS 5 can capture up to 60 FPS; however, iOS 6 limits the maximum camera capture rate to be 30 FPS. In theory, we expect better throughput on iOS devices if we can remove this limitation. In terms of barcode scheme, since QR code is not specifically designed for smartphone environment, its encoding/decoding running time is relatively high for legacy devices. Considering that the frame refresh rate is limited by the camera capture rate, we have to increase the storage capacity per single barcode in order to improve the system



(a) Two-phase Transfer Performance.

(b) Rotation angle and time trade-off.

Fig. 7: Two-phase transfer performance and rotation angle and time trade-off.

throughput. As future work, we would like to replace QR codes with color barcodes [8] for shorter encoding/decoding time and higher storage capacity. It uses multiple colors for each information block, so it can encode more information. For instance, the storage capacity of a single color barcode for the 1280×720 resolution screen of Galaxy S3 with 7×7-pixel block size is about 34K bit. According to [8] the encoding/decoding time is less 20 ms, which is significantly faster than QR codes. We estimate that our system is able to reach above 200 kbps throughput if it adopts color barcode as its coding scheme.

Remark on mobile payment systems. This scheme can be also used for mobile payment systems, where one party is a smartphone and another is a terminal equipped with a screen and camera, e.g. the users can securely 'show' their movie tickets/coupons to the terminal. In those usage scenarios, only a little bandwidth is required. Hence, the current system throughput is sufficient in practice. Assume the user's smartphones are not equipped with privacy screen projector but the terminal screen does, we can estimate the safe distance as $d_{\mathsf{save}} \approx 17$ cm, which is promising to prevent shoulder sniffing.

B. All-or-nothing data streaming scheme

In this section, we are going to deal with those adversaries whose $\beta \geq \varepsilon$. In presence of such strong adversaries, can we still protect the data confidentiality? We propose the all-or-nothing data streaming scheme, which is specifically tailored for secure data transfer without key exchange.

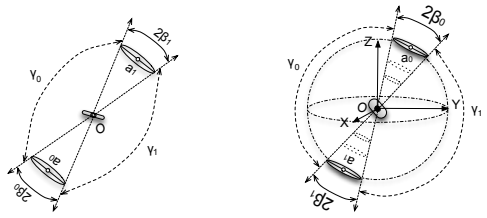
1) Protocol design: The aim of this scheme is to amplify the security such that the confidentiality of the entire transmitted data is guaranteed if the eavesdropper fails to capture at least one data frame. To achieve this goal, the sender first picks a random key and encrypts its data. Then the sender splits the key into many key shares and gradually sends those key shares together with the encrypted data chunks frame by frame. If the adversary miss one frame, then he/she cannot recover the key; subsequently, he/she cannot decrypt the captured data. We would like to employ a secret sharing scheme to split the key. In a (t, n)-secret sharing scheme, the data dealer runs $\{S_i\}_{i=1}^n \leftarrow \mathsf{Deal}(D)$ to split his/her data D into n shares S_i. The original data D can be recovered from any collection of more than t shares by $D \leftarrow \mathsf{Rec}(\{S_j\}_{j\in\mathcal{J}})$, where $\mathcal{J} \subseteq [n]$ and $|\mathcal{J}| \geq t$. In terms of security, the distribution of any less than t shares $\{S_i\}_{i\in\mathcal{I},|\mathcal{I}|<t}$ must be independent to the data D, i.e., any collection of $t-1$ shares reveals no information about D. Although Shamir's scheme [16] is a widely used secret sharing scheme in the literature, we adopt another xor-based scheme proposed by Kurihara *et.al.* [9], whose performance is 900 folders faster than the Shamir's. More specifically, we modify the package format by adding one extra ℓ_k-bit slot for key shares. As shown in Fig. 8, the sender first picks a random key $sk \xleftarrow{\mathsf{R}} \{0,1\}^{\ell_k}$. Let $\mathsf{PRG} : \{0,1\}^{\ell_k} \to \{0,1\}^{n\ell}$ be a pseudo-random number generator. The data is then encrypted by xoring it with generated random string. The sender splits the key sk into n key shares, using secret sharing algorithm $\mathsf{Deal}(sk)$. Next, the key shares and encrypted data chunks are sent to the receiver, utilising our two-phase message transfer protocol. After the transmission, the receiver recovers the secret key by $sk \leftarrow \mathsf{Rec}(k_1, \ldots, k_n)$ and decrypts the received data string.

2) Security analysis: The scheme should be combined with proactive rotation mechanism to enhance its security. Namely, the users can rotate their smartphones during the transmission period to prevent the adversary from 'seeing' all the barcode frames. We first show some impossibility results in case that an eavesdropper with two of more optical receivers can setup his/her receivers in optimal positions before-hand. It is easy to see that the optimal adversarial strategy against proactive rotation should always be distributing his/her receivers uniformly over the 360° cycle. We now show that if $\beta_0 + \beta_1 > 2\varepsilon$, there exists optimal receiver positions such that the screen is always visible to the adversary regardless the screen orientation.

Theorem 5.2: In 2D model, if $\beta_0 + \beta_1 > 2\varepsilon$, for all $\boldsymbol{v}$, the

Receiver		**Sender**
		$sk \xleftarrow{\mathsf{R}} \{0,1\}^{\ell_k}$; $\{k_i\}_{i=1}^n \leftarrow \mathsf{Deal}(sk)$; $U_1, \ldots, U_n \leftarrow \mathsf{PRG}(sk)$;
For $i = 1, \ldots, n$ **do:**		
$R_i \xleftarrow{\mathsf{R}} \{0,1\}^{\ell+\ell_k}$;	$\xrightarrow{R_i}$	$C_i = M_i \oplus U_i$;
	$\xleftarrow{F_i}$	$F_i = R_i \oplus (k_i \| C_i)$;
$k_i \| C_i = R_i \oplus F_i$;		
$sk \leftarrow \mathsf{Rec}(k_1, \ldots, k_n)$; $U_1, \ldots, U_n \leftarrow \mathsf{PRG}(sk)$; For $i \in [n]$, **return** $M_i = C_i \oplus U_i$;		

Fig. 8: All-or-nothing data streaming scheme.



(a) 2D optimal positions. (b) 3D optimal positions.

Fig. 9: Optimal positions for two-receiver adversary.

screen $\mathsf{pl}(\boldsymbol{v})$ with visible angle $2\theta_{\mathsf{max}} = 180^\circ - 2\varepsilon$ is visible by the two-receiver adversary $\mathsf{Adv}_m(\boldsymbol{a_0}, \beta_0, \boldsymbol{a_1}, \beta_1, \gamma)$, where the line a_0-a_1 passes through the origin $\mathcal{O}$.

Proof: As shown in Fig. 9 (a), when the line a_0-a_1 passes through the origin $\mathcal{O}$, we have $\gamma_0 = \gamma_1$ due to its symmetry. Given $\beta_0 + \beta_1 > 2\varepsilon$, we can deduce that

$$\gamma_0 = \gamma_1 = \frac{360^\circ - 2(\beta_0 + \beta_1)}{2} < 180^\circ - 2\varepsilon = 2\theta_{\mathsf{max}} \ .$$

According to Thm. 3.1, we can reduce both cases $\mathsf{Adv}_m(\boldsymbol{a_0}, \beta_0, \boldsymbol{a_1}, \beta_1, \gamma_0)$ and $\mathsf{Adv}_m(\boldsymbol{a_0}, \beta_0, \boldsymbol{a_1}, \beta_1, \gamma_1)$ to the single-receiver adversaries. Subsequently, the adversary can cover the entire 360° cycle, so the screen $\mathsf{pl}(\boldsymbol{v})$ is always visible to the adversary for all $\boldsymbol{v}$. ■

We now show that similar result holds in the 3D model as well. (cf. Fig. 9 (b).) Recall that the visibility is defined as the intersection between the screen visible zone $\mathsf{Vis}_3(\boldsymbol{v})$ and the adversarial capture cones. It is easy to see that

$$\forall \boldsymbol{v} \in \mathbb{R}^3 : \ \mathsf{Vis}_3(\boldsymbol{v}, \varepsilon) \cap (\mathsf{c}_3(\boldsymbol{a_0}, \beta_0) \cup \mathsf{c}_3(\boldsymbol{a_1}, \beta_1)) \neq \emptyset \ .$$

Hence, the screen is always visible by the adversary.

However, we claim that it is difficult for a the adversary to setup his/her devices at optimal positions before-hand due to the mobility of smartphones. We reduce any non-optimal multi-receiver adversary to a single-receiver adversary, whose device is modelled as the minimum ball that contains those receivers. Hence, we will only analyse security in the single-receiver adversarial model.

As illustrated in right of Fig. 6, since the adversarial capture cone must have intersection with both visible zones, when the rotation angle $\omega' > 2(\beta - \varepsilon) + \mu$ the adversary must lose vision of one of the screens at some moment, where μ is the additional rotation angle to guarantees that there is at least one barcode frame refreshed while rotating μ angle. Therefore, the rotation time for angle μ should be at least $\frac{2}{\tau_f}$, where τ_f is the system frame refresh rate. We can calculate the total rotation time for a given rotation speed ρ as $t = \frac{2(\beta-\varepsilon)+\mu}{\rho} = \frac{2(\beta-\varepsilon)}{\rho} + \frac{2}{\tau_f}$. [1]

By the security of (n, n)-secret sharing scheme, the distribution of any collection of less than n shares $\{k_j\}_{j \in \mathcal{J}, |\mathcal{J}| \leq n-1}$ is independent to the shared secret key sk. Hence, the eavesdropper learns no information about sk; so the confidentiality of the transmitted data string is preserved under the secure PRG assumption.

[1] Although the rotation speed does not effect the total rotation time, the higher rotation speed leads to the larger rotation angle μ. In practice, there is a trade-off between the rotation time and rotation angle.

3) Implementation and performance: The performance of our all-or-nothing data streaming scheme is very similar to the two-phase message transfer scheme. For (n, n)-secret sharing scheme, we use a more efficient $\mathsf{Deal}(sk)$ algorithm, i.e. picking n random key shares k_i such that $\oplus_{i=1}^n k_i = sk$. The corresponding recover algorithm $\mathsf{Rec}(k_1, \ldots, k_n)$ is defined as $\oplus_{i=1}^n k_i$. Clearly, any collection of $n-1$ key shares reveals no information about sk. One may also use the all-or-nothing data streaming scheme on top of the standard one-way VLC channel. In that case, we have to adopt a (t, n)-secret sharing scheme to tolerate the potential frame loss. In the general (t, n)-secret sharing case, we have to ensure that the adversary misses at least $n - t + 1$ frames during the transmission to preserve data confidentiality.

In terms of user-induced rotation motion, if the rotation speed is ρ, then in order to guarantee security, the users have to rotate $\omega > 2(\beta - \varepsilon) + \frac{2\rho}{\tau_f}$. For instance, let $\tau_f = 5$ FPS, $\varepsilon = 2^\circ$ and $\beta = 30^\circ$, we can obtain the angle and time trade-off chart as shown in Fig. 7 (b). For example, if a user rotates at speed 40 degrees per second, he/she has to rotate 72°, which takes 1.8 seconds to finish; whereas, if a user rotates at speed 10 degrees per second, he/she only needs to rotate 60° but it takes 6 seconds. On the other hand, a user may always rotate a certain angle ω^* at certain speed ρ^* in practice, and we can deduce the system security level from ω^* and ρ^*.

VI. Compatibility, Usability and Robustness

We tested the compatibility of SBVLC on iPhone 4/4S/5 and many Android smartphone platforms in various environments such as indoor, outdoor. The experiment shows that SBVLC works seamlessly across platforms under different lighting conditions. In terms of usability, we found that the rotation task is hard if two users hold their own phones and try to accomplish the rotation in a collaborative manor. The challenge is brought by maintaining the alignment of those two smartphones such that they are able to 'see' each other's barcode during the rotation. However, it is easy for a person to accomplish the rotation task if he/she holds these two smartphones in his/her both hands respectively. For instance, one can easily keep his/her upper body still and rotate his/her waist for a 90°-rotation task. We tested our system on 40 candidates randomly selected from the campus. The following table shows the average time taken by a user to align two smartphones such that both phones can 'see' the other's barcode at the first attempt and after 5-min training. Typically, it takes longer for a user to align two smartphones that are in different size and sharp such as the iPhone 4S-Galaxy S3 pair, but it becomes easier once the users get used to it. Given our single-person rotation instructions, 97.5% candidates can accomplish the 90°-rotation task within the first 2 attempts.

Study case	First attempt	After training
Galaxy S3 - Galaxy S3	$5''$	$2''$
iPhone 4S - iPhone 4S	$6''$	$2''$
iPhone 4S - Galaxy S3	$14''$	$3''$

In terms of system robustness, since our focus is data confidentiality against eavesdropping, the scenarios where a barcode itself contains malicious information, e.g. URL, are orthogonal to this work. Many other active attacks, e.g. data modification and injection can be easily detected if the attack

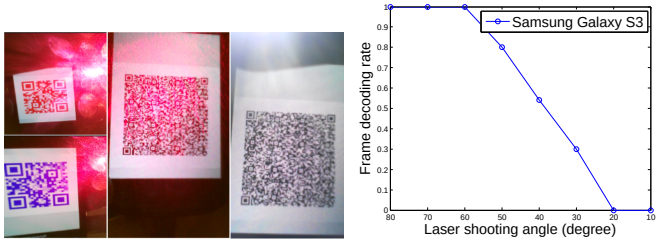


Fig. 10: Jamming experiment. (Tested on Galaxy S3.)

devices are near or in between the victims' smartphones; on the other hand, it is hard to implant a fake barcode from distance, for majority of the receiver's camera view is occupied/blocked by the sender's screen. We performed various jamming attacks to test the robustness of SBVLC. For instance, we use a laser pointer to shoot the receiver's camera at different angles. As shown in Fig. 10, the laser beam does not effect our system when the shooting angle is $\geq 60°$. On the flip side, the shooting angle can't be $\leq 30°$ in practice, because of the angle blocking by the other smartphone. In general, due to the usage of *visible light*, the jamming attacks can be easily detected and avoided, utilising the mobility of smartphones or physical blocking.

VII. Related work

Smartphones are widely used to scan 1D or 2D barcodes, such as UPC code, QR codes and Data Matrix. QR Droid [14] is a smartphone App related to this work. In QR Droid, the sender phone encodes a short message into a QR code and displays on its screen; the receiver uses its camera to capture the QR code and decodes it back to the message. The message can be encrypted with DES algorithm under a common secret key configured by both parties. However, that there is no automatic key exchange step in the implementation of QR Droid. By taking advantage of more colors, some new color barcodes are designed to increase the capacity, e.g., *high capacity color barcode* (HCCB) [12]. 4D barcode [10] is recently proposed for robust data transmission between smartphones. However, its throughput on smartphone platforms is as little as 100 bits/s due to the heavy computational overhead of operations like border detection and barcode rectification. PixNet [13] can build a wireless link using LCDs and cameras. The system can achieve high throughput over a long distance based on *orthogonal frequency division multiplexing* (OFDM) and complex computer vision algorithms. Unfortunately, PixNet is not suitable for smartphones dut to its high computation overhead. Meanwhile, COBRA system [8] can high speed barcode streaming between smartphones based on lightweight image processing techniques. But it improves system throughput by using highly customized barcodes, which are not widely adopted in practice. Moreover, the security of barcodes-based communication is not studied in [8]. Several previous studies have utilized VLC channel for security purposes. For example, McCune *et al.* proposed the Seeing-is-believing (SiB) system [11] for human authentication. It also can be used for secure device pairing [15]. However, these studies do not address the security of the barcode-based VLC channel for smartphone platforms.

VIII. Conclusion

We proposed SBVLC, utilizing a fully duplex smartphone VLC channel based on 2D barcode. On top of the duplex VLC channel, we further propose three secure communication schemes. All SBVLC schemes are evaluated through extensive experiments on Android smartphones, and the results show that our system achieves high level security and NFC-comparable throughput. The system can be used for private information sharing, secure device pairing and secure mobile payment, etc. To our best knowledge, this work is the first one that formally defines and studies the security of a smartphone VLC system. It serves as a milestone for further development in secure VLC systems for smartphones. In future work, we would like to increase the system throughput, using color barcode streaming [8] as discussed in Sec. V-A4. We will also extend our system to support other mobile and portable devices, e.g. laptops and tablets.

Acknowledgement

The first author was supported by University at Buffalo, SUNY, Project FINER, Greek Secretariat of Research and Technology, and ERC project CODAMODA. The second author was supported in part by NSF grants CNS-1262277 and CNS-1262275. The fourth author was supported by NSF grants CNS-0958477 and CNS-1116644. The last author was supported by City University of Hong Kong grant with project No. 7200320.

References

[1] Barcode payment service, http://gigaom.com/2012/05/30/paypal-rolls-out-barcode-payments-in-the-uk/.

[2] Norm ECMA-385. NFC-SEC: NFCIP-1 Security and Protocol, 2010.

[3] Norm ECMA-386. NFC-SEC-01: NFC-SEC Cryptography Standard using ECDH and AES Reference, 2010.

[4] Zxing (open source qr library), 2012. http://code.google.com/p/zxing.

[5] Mohamed Allah. Strengths and weaknesses of near field communication (nfc) technology. *GJCST*, 11(3), 2011.

[6] R. Russel Austin. Privacy filter for a display Device, June 1996. US Patent No. US5528319.

[7] Lishoy Francis, Gerhard Hancke, Keith Mayes, and Konstantinos Markantonakis. Practical relay attack on contactless transactions by using nfc mobile phones. ePrint Archive, Report 2011/618, 2011.

[8] Tian Hao, Ruogu Zhou, and Guoliang Xing. Cobra: color barcode streaming for smartphone systems. In *MobiSys*, 2012.

[9] Jun Kurihara, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka. A new (k,n)-threshold secret sharing scheme and its extension. In *ISC*, 2008.

[10] Tobias Langlotz and Oliver Bimber. Unsynchronized 4d barcodes: coding and decoding time-multiplexed 2d colorcodes. In *ISVC*, 2007.

[11] Jonathan McCune, Adrian Perrig, and Michael Reiter. Seeing-Is-Believing: using camera phones for human-verifiable authentication. *Int. J. Secur. Netw.*, 4(1/2):43–56, 2009.

[12] Devi Parikh and Gavin Jancke. Localization and segmentation of a 2d high capacity color barcode. In *WACV*, 2008.

[13] Samuel Perli, Nabeel Ahmed, and Dina Katabi. Pixnet: interference-free wireless links using lcd-camera pairs. In *MobiCom*, 2010.

[14] QRDroid, 2012. http://qrdroid.com/.

[15] Nitesh Saxena, Jan erik Ekberg, Kari Kostiainen, and N. Asokan. Secure device pairing based on a visual channel. In *S & P*, 2006.

[16] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11), 1979.