CamTalk: A Bidirectional Light Communications Framework for Secure Communications on Smartphones

Mengjun Xie[1], Liang Hao[1], Kenji Yoshigoe[1], and Jiang Bian[2]

[1] University of Arkansas at Little Rock, Little Rock, AR 72204, USA
{mxxie,lxhao,kxyoshigoe}@ualr.edu
[2] University of Arkansas for Medical Sciences, Little Rock, AR 72205, USA
jbian@uams.edu

Abstract. In this paper we present CamTalk, a novel bidirectional communications framework using front-facing cameras and displays of smartphones. In the CamTalk framework, two smartphones exchange information via barcodes: information is encoded into barcodes that are displayed on the screen of the origin device, and those barcodes are captured by the front-facing camera of the destination device and decoded; Both devices can send and receive barcodes at the same time. The general design of data transmission enables CamTalk to support a wide range of applications. More importantly, CamTalk's communications channels are short-range, highly directional, fully observational, and immune to electromagnetic interference, which makes CamTalk very appealing for secure communications and bootstrapping security applications. We have implemented CamTalk on the Android platform and conducted extensive experiments to evaluate its performance on both Android smartphones and tablets. Our experimental results demonstrate the efficacy of CamTalk in short-range wireless communications.

Keywords: bidirectional light communications, mobile application, secure communication, key exchange.

1 Introduction

With the popularity of camera equipped smartphones, a nonconventional communications channel through display and camera becomes more accessible to smartphone users. It is already common for a smartphone user to obtain information through the phone's camera. For example, we can use a smartphone to easily scan a barcode (e.g., a UPC code [8] or QR code [7]) printed on an item sold in a grocery store or on an ad wallpaper and then read the information encoded by the barcode. The visual data channel for this type of uses has been leveraged to create new schemes of data streaming from a screen (e.g., LCD) to a smartphone [10,9,5] and to design new mechanisms of authentication [11,18]. However, information flows through the visual channels in those schemes and mechanisms are all unidirectional, which seriously limits the functions of

T. Zia et al. (Eds.): SecureComm 2013, LNICST 127, pp. 35–52, 2013.

those applications. For example, the mutual authentication protocol proposed in [18] has to introduce a second channel due to the visual channel being one way.

In this paper, we present CamTalk, a novel bidirectional communications framework using front-facing cameras and displays of smartphones. Similar to previous camera phone based schemes, CamTalk also employs barcode for information transmission. Information is encoded into barcodes that are displayed on the screen of the sending smartphone, and those barcodes are captured by the front-facing camera of the receiving smartphone and then decoded. However, by using front-facing cameras and displays, in the CamTalk framework, the smartphone that is receiving information can simultaneously send its data by rendering the corresponding barcodes on its display, and those barcodes can also be captured by the other party that is doing sending at the same time. To our best knowledge, CamTalk is the first bidirectional camera-based communications scheme for smartphones (and tablets).

Thanks to the communications medium, i.e., visible light, communications through CamTalk are short-range, highly directional, fully observational, and immune to electromagnetic interference. These properties make CamTalk an appealing choice for secure communications and mutual authentication between two smartphones in close proximity. CamTalk can further bootstrap other security applications. Diffie-Hellman key exchange, a basic building block for secure communications, can be easily implemented based on CamTalk to securely share a secret between two smartphones without prior knowledge of each other. The shared secret can be used not only for the CamTalk communications, but also for other communications such as those through Bluetooth and Wi-Fi.

We have implemented a fully functional prototype of CamTalk on the Android platform. Our prototype adopts ZXing library [20], a popular open source barcode processing library, for handling low-level barcode encoding and decoding. We choose QR code as the underlying barcode technique as a recent study shows that QR code has the best decoding performance among the barcodes supported by ZXing library [19]. We have conducted extensive experiments to evaluate the impacts of different factors on communications of CamTalk and measure its performance on both Android smartphones and tablets. Our experimental results demonstrate the efficacy of CamTalk in short-range wireless communications.

The rest of this paper is organized as follows: Section 2 briefly describes the background and related work. Section 3 presents the structure of CamTalk framework and the design of communications mechanisms, especially the two transport modes. Sections 4 and 5 detail the implementation of the prototype and its evaluation using Android smartphones and tablets, respectively. Section 6 discusses the applications of CamTalk in security. Section 7 concludes this paper and discusses future work.

2 Background and Related Work

2.1 Visible Light Communication

Recently, visible light communication (VLC) has received strong interest as an alternative wireless communication channel. Generally speaking, VLC refers to wireless information transmission (usually in a short range) using visible light through free space. A number of studies on VLC have been conducted, e.g., high-speed (gigabit rate) VLC using light emitting diodes (LEDs) [16] and VLC based indoor positioning [17]. Compared to radio frequency (RF) based wireless communications technologies, VLC has the following advantages: using unlicensed spectrum, being immune to electromagnetic interference, and having no interference with RF systems. More importantly, the communications medium, visible light, can also be used for illumination, display, decoration, etc. Besides using LEDs and photodiodes, researchers have also leveraged liquid crystal displays (LCDs) and digital cameras for visible light communications [15,6]. Existing systems based on the LCD-camera pair require high-end digital cameras and large and high resolution LCD screens and involve high computational overhead, which is relatively difficult for smartphones.

2.2 Barcode Techniques

A barcode is an optical machine-readable representation of information. There are two types of barcodes: one-dimensional (1D) barcodes and two-dimensional (2D) barcodes. 1D barcodes are usually made up of parallel lines (bars) with various widths and spacings representing specific patterns. Universal Product Code (UPC) [8] is a very popular 1D barcode. 2D barcodes encode data in rectangles, dots, hexagons and other geometric patterns in two dimensions. Popular 2D barcodes include QR code [7], Data Matrix code, MaxiCode, etc. The main differences between 1D and 2D barcodes lie in the amount of encoded data and the error correction they provide. In the past, reading of barcodes required special optical scanners called barcode readers. Nowadays, more devices including camera equipped mobile phones support barcode scanning and information interpretation [13].

Quick Response code (QR code) [7] is a popular 2D barcode. All major smartphone platforms including Android, iOS, Blackberry, and Windows Phone support QR code scanning either natively or through third-party applications. ZXing project [20] provides an open source cross-platform barcode scanning library, which fully supports QR code encoding and decoding. Compared to other 2D barcodes, QR code has more features including large capacity, small printout size, and high speed scan. Scheuermann *et al.* evaluated barcode decoding performance using ZXing library and reported that QR code delivers the best results [19]. The amount of data that can be stored in a QR code symbol depends on the data type (mode), version (indicating the overall dimensions of the symbol), and error correction level.

Traditionally, barcodes use only black and white for information encoding. With the popularity of the camera based barcode scanning techniques that are capable of detecting colors, more colors are used to develop new types of barcodes with higher information capacity. High Capacity Color Barcode (HCCB) [14] is such a colored 2D barcode that employs clusters of colored triangles for encoding data. Langlotz and Bimber proposed another type of colored 2D barcode called 4D barcodes, which essentially are time-multiplexing colored 2D barcodes. Liu *et al.* proposed a video barcode scheme called VCode in [10] and analyzed its data transmission capacity in [9]. Hao *et al.* presented another 2D barcode scheme for data streaming on smartphones, called COBRA, in [5]. Both VCode and COBRA rely on specially designed colored 2D barcodes and use those barcodes to achieve high-speed data streaming between a screen and a smartphone. CamTalk is distinct from VCode and COBRA schemes in that data communications in CamTalk are bidirectional while in those schemes are unidirectional. In addition, CamTalk can adopt VCode and COBRA barcode techniques as its communications building block.

2.3 Mobile Visual Channel

Mobile visual channel has been applied to security applications. McCure *et al.* proposed an authentication scheme called Seeing-is-Believing (SiB) [11], which leverages the visual channel between a 2D barcode and a camera phone for authentication and demonstrative identification of devices. The visual channel of SiB is unidirectional. Therefore, operations requiring bidirectional communications such as Diffie-Hellman key exchange have to be decomposed into multiple unidirectional operations and direction switches must be coordinated manually.

Sexena *et al.* proposed a secure device pairing protocol, VIC (Visual authentication based on Integrity Checking), based on a visual channel [18]. Similar to SiB, the visual channel in [18] is also unidirectional. To achieve mutual authentication in secure device pairing, another insecure channel, e.g., Bluetooth, is introduced in VIC.

SiB and VIC are two special authentication schemes built on top of a unidirectional mobile visual channel. Compared to them, the bidirectional communications capability makes CamTalk support Diffie-Hellman key exchange and mutual authentication in an easier and automatic manner. Moreover, CamTalk supports more general communications such as file transfer.

3 System Design

CamTalk is designed as a wireless communication framework for smart mobile devices (e.g., smartphones and smart tablets) that enables bidirectional communications between two devices solely through display-camera links. As an analogy of face-to-face talk between two persons, CamTalk aims to achieve a face-to-face "talk" between two mobile devices. Thus, CamTalk merely requires mobile devices with a front-facing camera and a display of reasonable resolution, which are ubiquitous for today's smartphones and smart tablets. CamTalk is designed for short-range communications. Depending upon the display size and camera

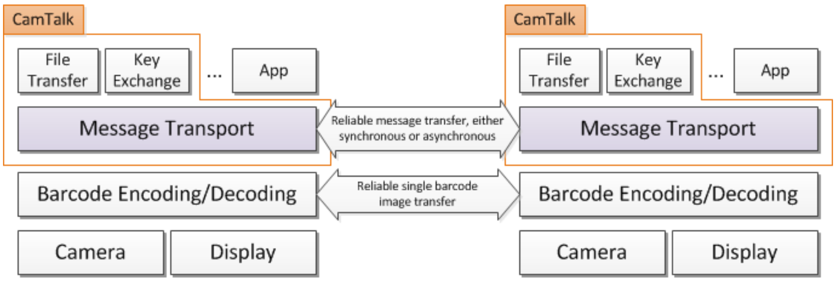

CamTalk
File Transfer
Key Exchange
...
App
Message Transport
Reliable message transfer, either synchronous or asynchronous
Barcode Encoding/Decoding
Reliable single barcode image transfer
Camera
Display
CamTalk
File Transfer
Key Exchange
...
App
Message Transport
Barcode Encoding/Decoding
Camera
Display

Fig. 1. The architecture of CamTalk framework

capability, the distance between two communicating devices can vary, e.g., from around ten centimeters to fifty centimeters in our experiments. Given the strong directional communication medium, fully observational communication process, and being entirely free from radio frequency interference, CamTalk provides a unique and advantageous channel for secure communications between two mobile devices.

Figure 1 depicts the architecture of CamTalk. The CamTalk framework relies on reliable single barcode image transfer provided by the underlying barcode encoding/decoding service, which is further supported by barcode scanning through front-facing camera and barcode rendering through device display. Making an analogy between CamTalk and a normal networking stack, CamTalk comprises the transport layer and part of the application layer. Based on the service of single barcode image transfer, the message transport layer of CamTalk realizes bidirectional reliable message transfer between two mobile devices, in either synchronous or asynchronous mode, and provides it as a service to the upper application layer. To demonstrate the efficacy and facilitate application development of CamTalk, two directly applicable applications, file transfer and Diffie-Hellman key exchange, are also incorporated into CamTalk. The design of CamTalk eases the development of other applications based on visible light communications, e.g., achieving secure file transfer through an encrypted channel by leveraging the existing file transfer and key exchange applications or building it directly on top of the message transport layer.

CamTalk framework is orthogonal to the underlying barcode technique, as illustrated in Figure 1. In other words, the design of CamTalk is generic, applicable to a variety of barcodes, and different implementations of CamTalk may use different barcodes that best suit the application requirements and given conditions (e.g., hardware capabilities, environment constraints, etc) for information transmission. In our prototype, we adopt QR code as the barcode encoding/decoding mechanism given its popularity and ubiquitous support on smart mobile devices.

For bidirectional communications, each CamTalk device is capable of both sending and receiving information, that is, encoding and rendering a barcode and scanning and decoding a barcode image. Figure 2 shows the data flow in CamTalk and those modules involved in data sending and receiving. The message to be

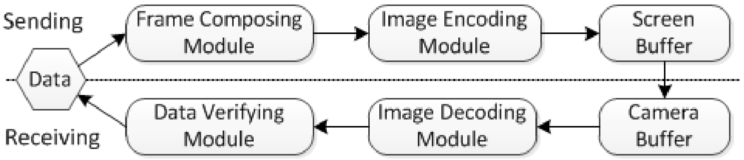


Fig. 2. Data flow in CamTalk

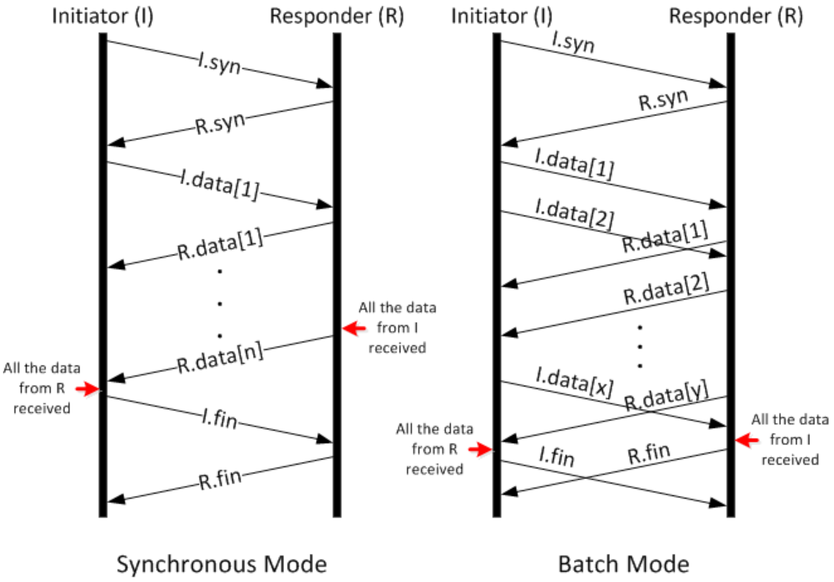


Fig. 3. Two message transport modes of CamTalk

transferred may be too large to be conveyed by a barcode image. Therefore, a large message will be split by the frame composing module into multiple smaller segments that can fit into a barcode image (also called frame). Then, each segment is encoded into a barcode by the image encoding module and copied into the screen buffer for rendering. When a picture is taken by the camera, the content in the camera buffer will be examined by the image decoding module. If the content contains a recognizable barcode and that barcode image can be successfully decoded, the decoded data will be validated and merged if necessary by the data verifying module. In practice, data sending and receiving can be carried out simultaneously by CamTalk.

A message can be transported in different fashions. To explore the display-camera channel capacity and provide flexibility, CamTalk incorporates two transport modes: synchronous mode and batch mode, which are illustrated in Figure 3. Alternation between sending and receiving is enforced for communications in the synchronous mode, in other words, frame $i + 1$ cannot be sent out before the frame i from the other party is successfully received. On the other hand, multiple frames can be sent out in a batch without waiting for the reception of the corresponding frames from the other party. For better presentation, we only show ideal scenarios of communications in the two modes in Figure 3. For example, duplicate or out-of-order frame transmissions that are possible in the Batch mode are not shown in the diagrams. In the diagrams, the party initiating the communication is called initiator and the other party is called responder. The initiator and responder are no difference in functionality and their roles are solely dependent on which initiates the communication.

There are two types of frames–control frame and data frame–in the communication. Control frames include (1) `syn` (for synchronization) and `fin` (for finish) frames sent at the beginning and ending of communications in either mode, respectively, and (2) status frames, which are used to notify the other party what frames are missing in the current batch, in the batch mode (not shown in the Batch Mode diagram in Figure 3). Each frame has a header and payload. To reduce overhead, a frame header has a sequence number and an acknowledgment number, each taking two bytes. The choice of small sequence number space is based on that CamTalk is intended for transferring a relatively small amount of data as an alternative to RF channels. The payload for `syn` frames contains the size of the whole message (in bytes), the capacity in a data frame (in bytes), and some other meta information. The payload for `fin` frames contains the SHA-256 hash value of the message for verifying data integrity. The exchange of `syn` frames before actual data transmission also has a practical consideration–to ensure the establishment of communication channels. If the display-camera links are not set up appropriately, `syn` frames will not be exchanged, i.e., the content in the screen will not change. A user usually needs to adjust the distance and positions of two CamTalk devices before communications and the visual changes of barcodes notify her of the establishment of the links.

4 Implementation

We have implemented a prototype system of CamTalk that runs on the Android platforms (API level ≥ 15) and works for both smartphones and tablets. Our implementation uses ZXing [20] (version 2.1) for underlying barcode encoding and decoding. The cross-platform compatibility of ZXing will ease the porting of our prototype to other mobile platforms such as iOS for iPhone. The prototype consists of around 2,300 lines of Java code (excluding ZXing library).

Our prototype is implemented as a standalone application (refer to as "CamTalk" in the remaining of the section for simplicity), supporting general information exchanges and also providing file transfer and Diffie-Hellmen key exchange functions. Thanks to the Android framework, CamTalk can also be invoked by other applications for information exchange through camera-display links, similar to how a barcode scanner application is invoked by another application such as Amazon mobile app on an Android smartphone.

Figure 4 (in Section 5.1) shows the communication interface of CamTalk on a Motorola Atrix 4G smartphone. The phone screen is split into two parts: The top half (in portrait/vertical orientation) is for communication by displaying the barcode in an ImageView instance. The bottom half shows the view captured by the front-facing camera using a SurfaceView instance. The rationale for displaying a barcode at the top half (or the half closer to the front-facing camera) is that most of smartphones and smart tables place their front-facing cameras at the top. Thus, it is relatively easier to capture a barcode at the top than at the bottom of the display when two phones are placed face to face for communication. The bottom camera view is primarily to help a user adjust the distance between and/or positions of the two devices for better communication quality.

The prototype of CamTalk follows modular and layered implementation practice. The display and capture of single barcode forms the basic building block of communication, similar to the transfer of an Ethernet frame between two LAN nodes. We implement the single barcade transfer function based on ZXing. Two transport modes–synchronous mode and batch transfer mode–are implemented as two modules in parallel on top of single barcode transfer service. The upper level application can decide which transport mode to use for a specific information transfer task.

The CamTalk prototype is a multi-threaded application, which employs multiple threads to speed up data processing and offload computation from the main thread (also called UI thread), following Android programming guide. The communications between threads are through messages (and handlers) and each module (encoding, decoding, transport) is developed in an event (or message) driven manner. The main thread is responsible for UI rendering and message dispatch. Two work threads are used by ZXing for encoding and decoding respectively. Transport module itself is implemented as a separate thread, handling message segmentation/assembly.

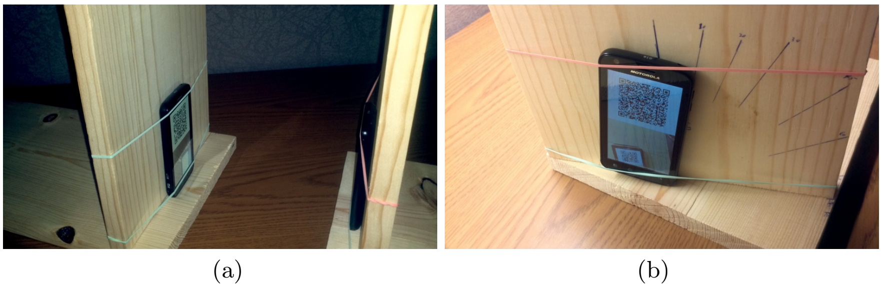
5 Evaluation

In this section, we describe how we evaluate the CamTalk prototype. We use three types of mobile devices all with both front-facing camera and display, and their hardware parameters related to our evaluation are listed in Table 1. From the table, we can see that the front-facing cameras have rather low resolutions and their positions vary. The Android platform versions on Atrix 4G, Nexus S, and Nexus 7 are 4.1.2, 4.1.1, and 4.2.2, respectively. As Android 2.3.6 is the latest version officially supported for Atrix 4G, we install a customized cyanogenmod version on our Atrix 4G phones for meeting the API level requirement of our prototype. Most of our experiments are carried out between two devices of the same type, that is, Atrix phones or Nexus 7 tablets. We also test the CamTalk between two different phones and between one phone and one tablet, and those devices are held in hand. Our experiments confirm that CamTalk can be used between two different devices in practice.

Since bidirectional communications through visible light between smart mobile devices have not been reported yet, our evaluation mainly focuses on the communication performance of CamTalk. We categorize the factors that can affect the communication performance into two categories: external and internal.

Table 1. The smart mobile devices used in the experiments

Device (type)	Motorola Atrix 4G (phone)	Google Nexus 7 (tablet)	Google Nexus S (phone)
Release Time (in the U.S.)	Feb. 2011	July 2012	Dec. 2010
CPU	1 GHz ARM A9 (dual-core)	1.2 GHz ARM A9 (quad-core)	1 GHz ARM A8 (single-core)
Front-facing Camera	0.3 Megapixels (top left)	1.2 Megapixels (top middle)	0.3 Megapixels (top right)
Display Size	4.0 inch	7.0 inch	4.0 inch
Display Resolution	540 × 960	1280 × 800	480 × 800


(a)

(b)

Fig. 4. The experiment testbed. These two pictures are for demonstration purpose, therefore not taken in a real experiment. The light was turned off for better display of phone screen in Figure (a). The phone in Figure (b) was rotated around both Y-axis and Z-axis.

The external category consists of those factors that are not CamTalk specific, including distance, rotations, lighting, etc. The factors in the internal category are related to the CamTalk design, including barcode image size, barcode capacity (i.e., number of bytes encoded in a barcode), barcode error correction level, etc. Note that we only study those factors that are not implementation specific. In the following, we first describe how CamTalk is affected by the external factors and then detail the impact made by the internal factors. After that, we present the experiment results of CamTalk's throughput with the two transport modes described in Section 3.

To conduct the experiments in CamTalk evaluation, we make a simple testbed as shown in Figure 4. Figure 4 (a) shows the scenario of CamTalk communication between two Atrix phones when they are aligned face to face, and Figure 4 (b) illustrates the rotation of device. Without further notice, all the experiments in the following are conducted on the testbed.

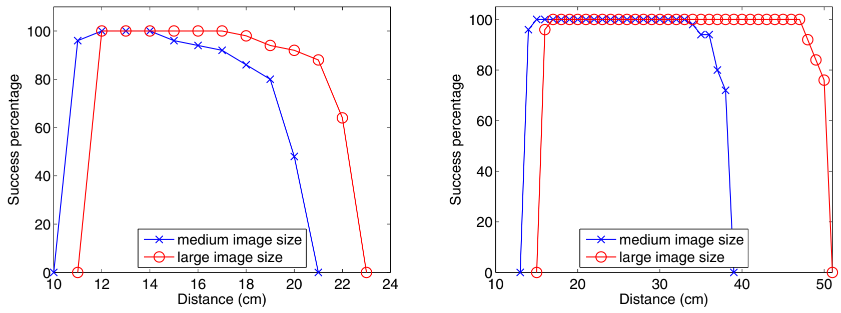
5.1 Impact of External Factors

As CamTalk employs visible light as its communication medium, many external factors can affect CamTalk, including ambient light, screen brightness, screen reflection, distance, rotations, etc. Among them, we have quantitatively studied the impact of distance and rotations on the communication, as those two factors can be easily controlled by a user and their effects can be instantly observed.

We use the decoding rate, the percentage of successfully decoded barcodes, as the metric to measure the impact of distance and rotations. As the barcode image size can affect decoding, we measure the decoding rate with two different sizes of barcode image: medium size and large size. The medium and large sizes are relative to the device display dimension; therefore, the dimension of a medium-sized or large barcode may vary on different devices. The length of a medium-sized QR code is 65% of the half of the longer side of the display and that of a large QR code is 90%. Both medium-sized and large barcodes contain the same amount of payload (32 bytes) in the experiments.

Ambient Light and Screen Brightness We observe that dark or very bright ambient lighting conditions can significantly degrade and even disable the CamTalk communications. When the ambient light is too bright, e.g., the phone screen under direct light of a fluorescent tube, the screen reflection will become very strong and therefore sharply worsen the quality of images taken by a front-facing camera. We conduct all the experiments in an indoor environment with illuminance of 400 to 500 *lux*, measured by Mastech Light Meter LX1010B.

With this ambient lighting, we find that it is easier to successfully decode a barcode image when the screen brightness is relatively low (e.g., half full brightness or less). The better decoding rate with lower brightness is attributed to the reduced screen reflection. When two phones or tablets are placed face to face in close proximity (tens of centimeters), high screen brightness will cause strong screen reflection. The impact of screen brightness on the communication performance is negligible when the brightness is relatively low (between 30 *lux* and 160 *lux*). The device screen brightness in the following experiments is between 40 *lux* and 80 *lux*, measured by the same light meter.



(a) Atrix 4G phone

(b) Nexus 7 tablet

Fig. 5. The impact of distance

Distance. We measure the impact of distance on both smartphones and tablets. The two devices, one as sender and the other as receiver, are aligned face to face without shifting or rotation in the experiments, as illustrated in Figure 4 (a). We measure the decoding rate with different distances and show them in Figure 5. Apparently, the larger the barcode image, the longer the distance allowed for communication. The practical communication distance for Atrix smartphone ranges approximately from 11cm to 22cm, while that for Nexus 7 tablet is from 15cm up to 50cm. The ideal distances for Atrix and Nexus 7 are 12-14cm and 17cm-33cm respectively, where a barcode image (either medium-sized or large) can always be decoded successfully.

The effective communication distance of CamTalk can vary in practical use but should not significantly deviate from the measured range under similar conditions. The short-range and observational communication properties of CamTalk offer high security assurance.

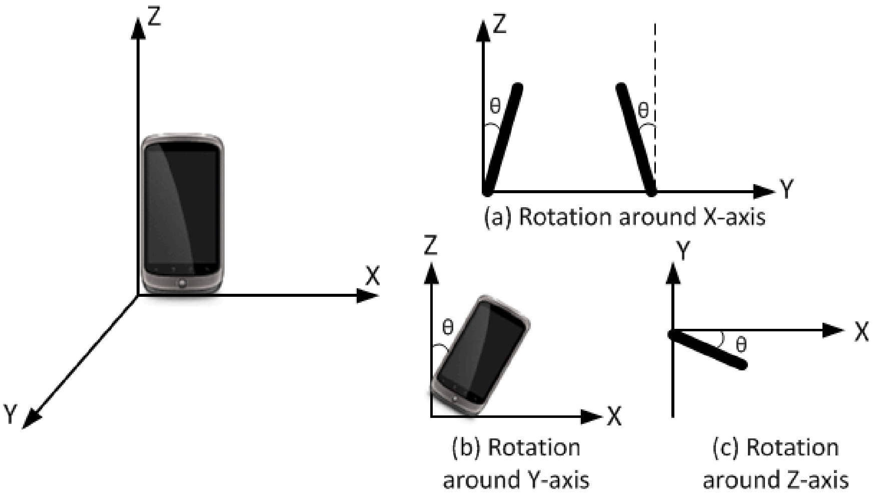
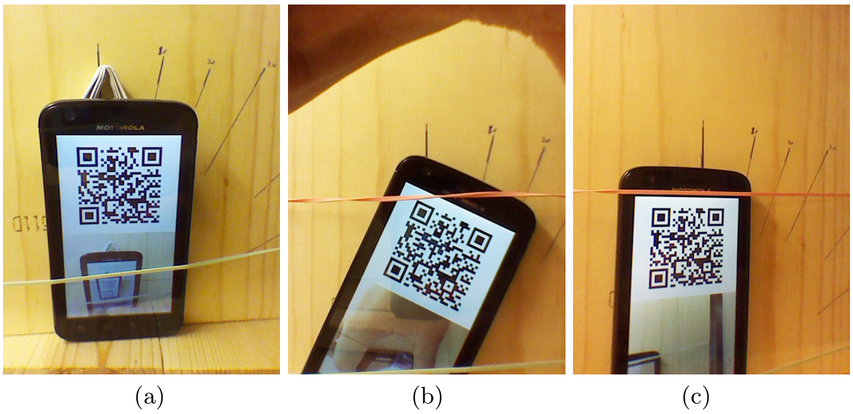

Z
X
Y
Z
θ
θ
Y
(a) Rotation around X-axis
Z
θ
X
(b) Rotation around Y-axis
Y
X
θ
(c) Rotation around Z-axis

Fig. 6. Illustration of rotations around X-, Y-, and Z-axis. θ represents the rotation degree.

Rotations. Devices may be rotated in different manners and those rotations have different impacts on the decoding rate. We measure the impact of rotation around X-, Y-, and Z-axis respectively. Figure 6 illustrates how a device is rotated around X-, Y-, and Z-axis with a certain degree. Note that both the sending device and receiving device are rotated with the same degree in the experiments for rotations around X-axis, as depicted in Figure 6 (a), while in the experiments for rotations around Y- and Z-axis, only the sending device is rotated and the receiving device stays fixed. Without further notice, both devices in the following experiments are aligned initially. Figure 7 shows the pictures



(a) (b) (c)

Fig. 7. The snapshots of rotations around (a) X-axis, (b) Y-axis, and Z-axis

Table 2. The impact of rotations around X-axis

Device	Size & Distance	0°	5°	10°	15°
Atrix 4G phone	medium, 20cm	48%	94%	100%	100%
	large, 22cm	64%	86%	90%	100%
Nexus 7 tablet	medium, 37cm	80%	92%	94%	98%
	large, 50cm	76%	90%	92%	100%

Table 3. The impact of rotations around Y-axis

Device	Size & Distance	5°	10°	15°	20°	25°	30°	35°	40°	45°
Atrix 4G phone	medium, 13cm	100%	98%	94%	0%	0%	0%	0%	0%	0%
	large, 15cm	100%	95%	94%	80%	0%	0%	0%	0%	0%
Nexus 7 tablet	medium, 21cm	100%	100%	100%	94%	90%	0%	0%	0%	0%
	large, 30cm	100%	100%	100%	100%	100%	100%	90%	84%	0%

taken at the receiving phone when the sending phone is rotated 15°, 20°, and 20° around X-, Y-, and Z-axis, respectively.

Because the front-facing camera is at the top of the device in the portrait orientation, when Atrix phones or Nexus 7 tablets are tilted forward (device bottom fixed) towards each other in a small degree, the distance between the barcode and the camera is shortened. Thanks to the tolerance of QR code to the perspective distortion brought by the rotation, shortened distance increases the possibility of barcode image being successfully decoded. Table 2 shows the impact of X rotation on the decoding rate when two devices are placed in such a distance that only partial barcodes can be decoded without rotation. Clearly, small scale rotations around X-axis help decoding.

Tables 3 and 4 show the average decoding rates when the sending device is rotated a certain degree around Y- and Z-axis, respectively. We can see that those rotations affect the decoding rate negatively. Due to this reason, the devices are placed in an ideal distance for each experiment. When the sending device is

Table 4. The impact of rotations around Z-axis

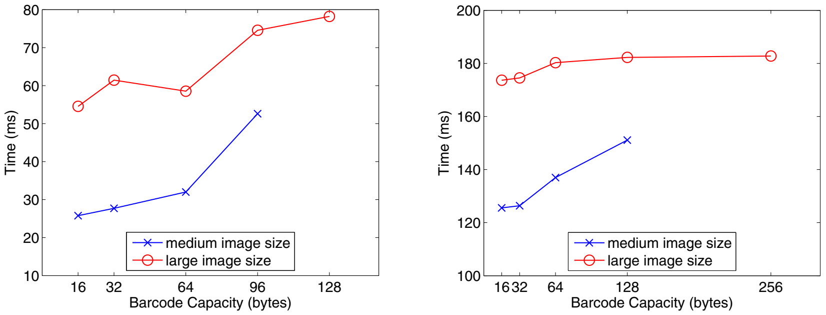
Device	Size & Distance	5°	10°	15°	20°	25°	30°
Atrix 4G phone	medium, 13cm	100%	100%	96%	90%	0%	0%
	large, 15cm	100%	98%	94%	94%	0%	0%
Nexus 7 tablet	medium, 21cm	100%	100%	96%	94%	82%	0%
	large, 30cm	100%	100%	98%	94%	74%	0%

rotated beyond a certain degree, the barcode image is either moved out of the camera view partially or entirely or distorted too much so that image decoding will fail.

5.2 Impact of Internal Factors

The internal factors we mainly consider include QR code error correction level, QR code data capacity, and QR code image size. We use encoding time and decoding time as the metrics for measuring the impact of each of those factors on CamTalk performance.

We first consider the QR code error correction level and its impact on encoding and decoding. QR code uses Reed-Solomon error correction algorithm with four levels: low (L), medium (M), quartile (Q), and high (Q). The higher the level, the more errors can be corrected. We measure the times of encoding and decoding at each of these four levels and find no significant difference among them. Therefore, we use error correction level M in the rest of experiments.

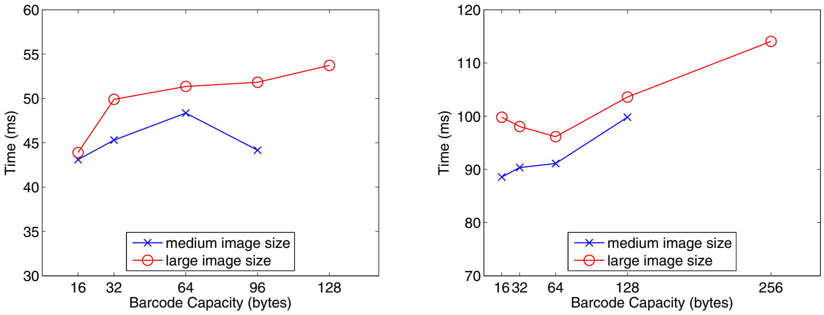


(a) Atrix 4G phone

(b) Nexus 7 tablet

Fig. 8. Average encoding time

Barcode data capacity and image size are two major internal factors that affect the encoding and decoding performance. Barcode capacity refers to the amount of data, including both header and payload, carried by each data frame (excluding the error correction bits). Figures 8 and 9 display how the average encoding time and decoding time vary with different capacity and image size,



(a) Atrix 4G phone (b) Nexus 7 tablet

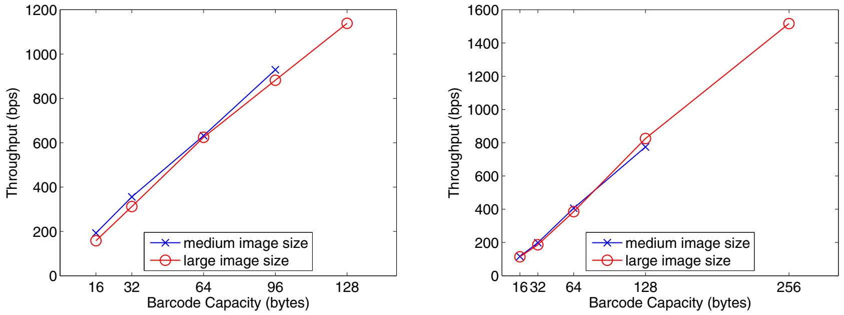
Fig. 9. Average decoding time

respectively. In those experiments, two devices are well aligned and in an appropriate distance. Intuitively, larger image size with more pixels will take more time for encoding and decoding, which is confirmed by our experiment results. The increase of encoding time is more evident than that of decoding time, which is true for both phone and tablet.

To study the impact of barcode capacity, we measure the average encoding and decoding times with different capacity but same image size. Constrained by the camera capability (e.g., resolution), the maximum capacity for different barcode image size varies. For example, a medium-sized QR code on the Atrix phone cannot contain 128-byte data, which simply cannot be decoded at the receiving phone. We note that 96 bytes and 128 bytes are not the maximum capacities on the phone for medium-sized and large barcodes respectively. The focus here is to show how encoding/decoding time varies with different barcode capacity instead of finding the maximum barcode capacity. In general, we can see that larger barcode capacity, i.e., denser barcode image, will render encoding and decoding to become longer. There exist a few points where processing time actually becomes slightly smaller in Figures 8 and 9. Those dips may be attributed to the dynamics of the running environment, e.g., OS scheduling and Java memory management.

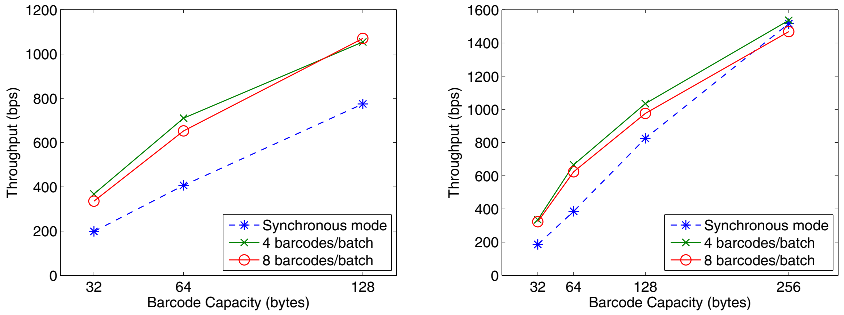
5.3 Throughput

We implement the file transfer function in the prototype and use it to measure the throughput in each of the two transport modes. The throughput is obtained by dividing the size of file over the duration from sending the first data frame to sending the `fin` frame (indicating that all data frames have been received). Note that we only measure unidirectional file transfers to simplify the experiments, while CamTalk supports bidirectional information transmission. Thus, the effective throughputs should be the double of those results.



(a) Atrix 4G phone (b) Nexus 7 tablet

Fig. 10. Average throughput in the synchronous mode



(a) Medium size barcode image (b) Large size barcode image

Fig. 11. Average throughput in the batch mode on Nexus 7 tablet

Figure 10 presents the relationship between throughput and barcode capacity (size) in the synchronous mode. Interestingly, a linear relation between throughput and barcode capacity exhibits, and appears insensitive to the size of barcode, which applies to both the phone and the tablet. As doubling the barcode capacity halves the number of barcodes to be sent, that is, reducing the overall transmission time approximately to the half of the original, throughput is mainly affected by barcode capacity in the synchronous mode.

As multiple barcodes are sent in a batch in the batch mode, intuitively, the throughput in the batch mode should be higher than that in the synchronous mode. We compare the throughputs of 4 barcodes/batch and 8 barcodes/batch in the batch mode and the corresponding throughput in the synchronous mode on the tablet and show them in Figure 11. We can see that sending multiple barcodes in a batch does improve the throughput. However, the improvement becomes very small when doubling the number of barcodes from 4 to 8. There is even a slight performance degradation when doubling the number for large barcode capacities. The limitation of improvement can be attributed to the hardware capacity. When too many barcodes are sent consecutively, the throughput is limited by the computing capacity, the majority of which is occupied by encoding, decoding, rendering, etc.

6 Discussions

Diffie-Hellman (D-H) key exchange [3] allows two parties without prior knowledge of each other to establish a shared secret key over an insecure communications channel. The importance of D-H key exchange to secure communications is beyond question. However, the D-H key exchange (in the original description) is vulnerable to the man-in-the-middle (MITM) attack even when the two communication parties are in proximity, e.g., two wireless devices communicating through Bluetooth. CamTalk, given its short-range, highly directional, and fully observational communications characteristics, can use D-H protocol for key exchange without worrying about MITM attacks. Once the shared key is securely exchanged through CamTalk, it can be used as the session key to encrypt the communications through either the display-camera links or RF wireless channels including Bluetooth and Wi-Fi. Therefore, CamTalk can not only provide a self-contained secure communications channel, but also service other communications channels. For example, we can use CamTalk to assist in pairing two smartphones with Bluetooth before using Bluetooth for normal communications.

We have implemented D-H key exchange on top of synchronous mode in the prototype. Since the Andorid SDK includes the popular Bouncy Castle Crypto suite, which provides easy-to-use and lightweight cryptography APIs, our D-H implementation and cryptographic operations such as encryption and decryption are based on Bouncy Castle. The generation of D-H parameters can be computationally expensive. To minimize resource consumption and reduce the time of key pair generation, we use the pre-generated safe 1024-bit prime modulus for D-H as suggested in [1] on the devices of CamTalk. Each device independently generates a large random number as its private key and then exchange the public key. Combining the other party's public key and its private key, a shared secret key can be derived. Thanks to the layered implementation of CamTalk, the public keys can be easily transferred as normal messages in both directions. The overhead of D-H key exchange mainly lies in computing the public keys and shared secret key. However, compared to the communication latency, computational overhead of D-H key exchange is minor. A D-H key exchange can be completed within 10 seconds using Atrix phones with barcode capacity being 64 bytes in the synchronous mode.

Offering a unique bidirectional communications channel, CamTalk can be extended beyond communications between mobile devices. For example, we envision that CamTalk may be applied to the communications between a smartphone and a PC or laptop equipped with a camera. Such extensions can catalyze new applications that leverage CamTalk for secure communications and authentication. Today, smartphones have become a popular choice of the second factor in a two-factor authentication (TFA) system. In conventional phone-based TFA systems, e.g., Mobile-OTP [12] and Google 2-step verification [4], the authentication process requires a user to manually enter a one-time password (OTP) received by the smartphone on a PC to authenticate her identity. However, this manual input process may create usability issues and the OTP is usually short for easy type. Using CamTalk, we can replace manual typing by putting the smartphone in front of the PC camera and complete the authentication in an automatic manner. Without typing, longer OTPs can be employed to enhance the security. Bluetooth is a popular choice for the communications between a PC and a smartphone in many phone-based TFA systems, e.g., PhoneAuth proposed by Czeskis *et al.* [2]. However, Bluetooth is vulnerable to the RF interference and subjected to MITM and jamming attacks. CamTalk is free of those concerns and can replace Bluetooth as the communications channel. We note that this replacement cannot be achieved by previous visual channel based approaches (e.g., [11,18]) as those channels are unidirectional.

7 Conclusion

We have presented CamTalk, a novel bidirectional communications framework leveraging display-camera links on smart mobile devices, and discussed its application to secure communications in this paper. We have described the design and implementation of CamTalk in detail and conducted extensive experiments to evaluate its performance and understand the factors affecting its performance. Our experiments show the throughput of CamTalk can reach 3Kbps using a Nexus 7 tablet, which provides a reasonable user experience for transferring a small amount of sensitive data in a fairly secure manner.

The relatively low throughput is mainly attributed to the low capacity of front-facing cameras (less than 2 megapixels and no auto focus) and the underlying barcode technique, i.e., QR code, which is not designed for high-throughput data transfer. We plan to replace the QR code with other barcodes designed for data streaming such as COBRA [5] and assess the performance change of CamTalk.

Our future work also includes a usability study of CamTalk. We have tested practical uses such as file transfer using CamTalk on both smartphones and tablets by holding those devices in close proximity, in which placing two devices in an appropriate position takes several seconds. We want to know the experience of an average user in using CamTalk and improve CamTalk based on the feedback in the future.

References

1. Aziz, A., Markson, T., Prafullchandra, H.: IETF internet-draft: Simple key-management for internet protocols, skip (1995), `http://tools.ietf.org/html/draft-ietf-ipsec-skip-06`
2. Czeskis, A., Dietz, M., Kohno, T., Wallach, D., Balfanz, D.: Strengthening user authentication through opportunistic cryptographic identity assertions. In: Proceedings of the 2012 ACM CCS, pp. 404–414 (2012)
3. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theor. 22(6), 644–654 (2006)

4. Google. Google 2-step verification, http://www.google.com/landing/2step/
5. Hao, T., Zhou, R., Xing, G.: COBRA: color barcode streaming for smartphone systems. In: Proceedings of MobiSys 2012, pp. 85–98 (2012)
6. Hranilovic, S., Kschischang, F.: A pixelated mimo wireless optical communication system. IEEE Journal of Selected Topics in Quantum Electronics 12(4), 859–874 (2006)
7. International Organization for Standardization. ISO/IEC 18004:2006: Information technology – automatic identification and data capture techniques – QR code 2005 bar code symbology specification (2006)
8. International Organization for Standardization. ISO/IEC 15420:2009: Information technology – automatic identification and data capture techniques – EAN/UPC bar code symbology specification (2009)
9. Liu, X., Doermann, D., Li, H.: A camera-based mobile data channel: capacity and analysis. In: Proceedings of the 16th ACM International Conference on Multimedia, pp. 359–368 (2008)
10. Liu, X., Doermann, D., Li, H.: Vcode - pervasive data transfer using video barcode. IEEE Trans. Multi. 10(3), 361–371 (2008)
11. McCune, J.M., Perrig, A., Reiter, M.K.: Seeing-is-believing: Using camera phones for human-verifiable authentication. In: Proceedings of the 2005 IEEE Symposium on Security and Privacy, pp. 110–124 (2005)
12. Mobile-OTP Project. Mobile one time passwords, http://motp.sourceforge.net/
13. Ohbuchi, E., Hanaizumi, H., Hock, L.A.: Barcode readers using the camera device in mobile phones. In: Proceedings of the 2004 International Conference on Cyberworlds, pp. 260–265 (2004)
14. Parikh, D., Jancke, G.: Localization and segmentation of a 2d high capacity color barcode. In: Proceedings of the 2008 IEEE Workshop on Applications of Computer Vision, WACV 2008, pp. 1–6 (2008)
15. Perli, S.D., Ahmed, N., Katabi, D.: Pixnet: Interference-free wireless links using lcd-camera pairs. In: Proceedings of MOBICOM, pp. 137–148 (2010)
16. Pisek, E., Rajagopal, S., Abu-Surra, S.: Gigabit rate mobile connectivity through visible light communication. In: Proceedings of IEEE International Conference on Communications, pp. 3122–3127 (2012)
17. Rahman, M.S., Kim, K.-D.: Indoor location estimation using visible light communication and image sensors. International Journal of Smart Home 7(1), 99–114 (2013)
18. Saxena, N., Ekberg, J.E., Kostiainen, K., Asokan, N.: Secure device pairing based on a visual channel: Design and usability study. IEEE Trans. Info. For. Sec. 6(1), 28–38 (2011)
19. Scheuermann, C., Werner, M., Kessel, M., Linnhoff-Popien, C., Verclas, S.A.W.: Evaluation of barcode decoding performance using zxing library. In: Proceedings of the Second Workshop on Smart Mobile Applications, SmartApps 2012 (2012)
20. ZXing Project. ZXing–Multi-format 1D/2D barcode image processing library with clients for Android, Java (2013), https://code.google.com/p/zxing/