Jive: Spatially-Constrained Encryption Key Sharing using Visible Light Communication

Jayanth Shenoy, Aditya Tyagi, Meha Halabe, Christine Julien
{jayanth.shenoy,adityatyagi6498,meha.halabe,c.julien}@utexas.edu
University of Texas at Austin
Department of Electrical and Computer Engineering

ABSTRACT

This paper investigates a novel encryption key sharing mechanism using the emerging wireless technology Visible Light Communication (VLC). Based on the idea of transmitting data by modulating light, we are able to (1) share a secret key within a constrained physical space and (2) communicate encrypted information among co-located mobile devices using the shared key. We present the demonstration Jive (Joint Integration of VLC and Encryption), a framework to support secret key sharing over Visible Light Communication. In defining Jive, we tackle challenges related to data encoding, message synchronization, and environmental noise to build a reliable, low complexity system using off-the shelf hardware. Our system is capable of sending encryption keys at speeds of more than 750bps using ultra short, high speed light pulses imperceptible to the human eye. Additionally, we have developed an application for Android that interfaces with the VLC device through serial communication so that applications running on mobile devices can subsequently use the keys to encrypt application data. Experimental results illustrate the high accuracy of our system across a variety of different variables. Finally, we position our system for use by a variety of applications that require a high-level of data security among physically co-located devices.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Embedded Software*; VLC; • **Networks** → Secure Communication.

KEYWORDS

embedded systems, VLC, communication, encryption, mobile applications

ACM Reference Format:
Jayanth Shenoy, Aditya Tyagi, Meha Halabe, Christine Julien. 2019. Jive: Spatially-Constrained Encryption Key Sharing using Visible Light Communication. In *16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous), November 12–14, 2019, Houston, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3360774.3360815

MobiQuitous, November 12–14, 2019, Houston, TX, USA

ACM ISBN 978-1-4503-7283-1/19/11…$15.00
https://doi.org/10.1145/3360774.3360815

1 INTRODUCTION

In many application scenarios, private or secure data needs to be exchanged among co-located devices in a way that is protected from eavesdroppers. In the current state of the practice, humans are heavily involved in securing the exchange of this data, for instance by exchanging a "key" using a whiteboard or by using human-controlled exchanges (e.g., through a USB key). Exchanging secure or private information in the clear over direct wireless links is potentially dangerous because eavesdroppers unseen to the communicating parties (e.g., nearby but outside of the room) may overhear the communication. As two concrete examples, consider a medical examination room in which several medical devices collect information about a patient's condition, and nurses and doctors insert notes in a medical record. This process could be completely automated and secured if all of the devices shared a private "session" key with which all of the information is encrypted. Similarly, in a company's board room, proprietary information may be shared among a set of meeting participants, though eavesdroppers outside of the conference room should not be able to extract the details.

In this work, we ask the seemingly simple question: "*What if the solution to establishing a novel secure session key is to simply turn on the light?*" Visible Light Communication (VLC) is a powerful way to transmit data that provides several benefits over traditional radio-based communication systems. Most pertinent to this work, transmitting data through visible light pulses results in both a high-speed and highly secure means of communicating information. In our context, "highly secure" refers to the ability to constrain the signal (i.e., light) using the physical space (i.e., walls). It is non-trivial for receivers to eavesdrop on the exchange without being noticed or detected. The low-cost of VLC and the possible integration of VLC devices into everyday lighting serve as further motivation for the developing technology. Moreover, in an increasingly technological world, the security of everyday information exchange activities has become a necessity. In our research, we provide an accessible means to communicate secure data between co-located mobile devices by introducing a VLC system that facilitates the communication of encrypted data from one device to another.

VLC is rapidly maturing as a research field; existing work has shown that smartphone cameras can be used as receivers [11], and VLC has also been extended to IoT applications and embedded systems including the ability to transmit data between vehicles in low ambient light environments [9]. Previous research efforts have used similar motivations to ours to explore supporting secure communication among co-located devices. For instance, prior work has introduced the idea of encoding data in images or bar codes to communicate secure information [5, 19]. Our work is premised on

the fact that the VLC domain is inherently more secure to eavesdropping simply because light signals, unlike radio signals, can be physically contained in a space.

In this paper, we seek to define a framework that relies on this inherent security of a VLC link to establish private communication on a secondary (e.g., radio) link. In particular, we propose Jive, the Joint Integration of VLC and Encryption. Jive enables co-located devices to transmit encrypted messages on a non-VLC medium using an encryption key supplied by VLC in the shared physical space. Jive supports a wide variety of applications that require establishing a secure communication channel among two or more co-located parties, making it ideal for transmitting data to groups of individuals isolated in (physically) secure environments.

This paper outlines the process of designing and evaluating a Jive prototype network that integrates a private key distributed via VLC with traditional communication among mobile applications. Implementing the VLC aspect of Jive required overcoming the challenges of encoding data in light, synchronizing the transmitter and receivers, and handling constantly varying environmental noise from ambient light. To make Jive's functionality accessible on mobile devices, we designed an application on the Android platform to interact with the Jive VLC system. Our system maintains a low implementation complexity in the sense that Jive's overall architecture is composed of readily available off-the-shelf components.

Finally, to evaluate the efficacy of our system and observe the design trade offs, we performed several experiments deriving our system's achievable data rates and communication distances. These experiments provide specific insights into our system to facilitate their development for proof-of-concept applications requiring high security VLC communication systems.

2 RELATED WORK

The world of VLC has rapidly developed ever since Harald Haas [4] coined the concept of LiFi and ignited a phenomenon that is being expanded by researchers across the globe. Prabu *et al.* proposed a VLC system that protects vehicles from theft using a VLC link among cars and tollbooths [14]. Karthik *et al* proposed a system in which drivers can communicate with other drivers at night time to provide safety information across vehicles using VLC [9]. Many systems use simple encoding schemes (e.g., light is on when the transmitted bit is 1 and off when it is 0). Our work differs in its use of Manchester encoding to reduce the visible flicker of the light during transmission; we evaluate the effectiveness of this scheme with real users in Section 4. Works such as Lartigues *et al.*'s on implementing a beacon-to-camera VLC system [11] and De Lausnay *et al.*'s evaluation of modulation techniques [3] also utilized Manchester encoding. Xu *et al.* constructed a passive VLC system that uses Miller encoding [18]. Although this approach does reduce the bandwidth, Miller encoding requires certain signal sequences to be avoided, and is thus not compatible with our design. Schmid *et al.* overviewed the basics of designing a visible light communication system [15]. Inspired by this work, we included a preamble that notifies the receiver of the start of a message. Our approach differs as we use this preamble to sense when enough data has been collected so that the key can be directly sent to the smartphone through serial connection.

In spaces directly related to our approach, Fu *et al.* proposed a system in which a 2D color barcode is used to communicate data across smartphones [19]. Similarly, Hao *et al.* presented COBRA [5], a VLC system that provides a secure means to communicate data from a smartphone to another smartphone by designing an app that scans and blurs a 2D barcode image containing data. These efforts rely on directionality and short range of distance to preserve the security and data that is communicated in the system. Li *et al.* created a two-way VLC channel in which two LED devices communicate with each other by synchronizing their frame rate [12]. Furthermore, Hewage *et al.* designed an open-sourced VLC system available for use for numerous diverse applications [7]. This body of work together shows that VLC is a maturing field, with components increasingly available to be used in research deployments. In contrast, we build Jive using common off-the-shelf components at low cost, in an attempt to motivate the inclusion of Jive in readily available light-bulbs.

There also exist previous efforts that attempt to determine the physical security of the VLC link. For instance, Classen *et al.* benchmark the ability for VLC signals to escape rooms through cracks under doors or through keyholes [1], positing that a significant vulnerability in VLC designs is in that designers attribute stronger security guarantees to the VLC link than truly exist [2]. We recognize this potential pitfall in that Jive assumes the VLC link is secure. In particular, we assume that Jive will operate in physically secure, enclosed rooms. Future work will investigate how to further secure Jive's key exchange, even in the light of these potential light "leakages". In this sense, we demonstrate Jive using a "box" whose material and design do not allow light to escape.

Instead, we are motivated by prior work that details the importance of encrypted communication in in today's technological world. Kounavis *et al.* motivate improving and expanding the use of encrypted communication in networked applications, in general [10]. This is a widely accepted view; however resource limitations of mobile and IoT devices make some modern encryption techniques based on asymetric cryptography intractable [8]. For this reason, we are motivated to find a way to use the resource-intensive symmetric encryption.

As discussed above, prior research endeavors regarding VLC have often involved systems with high cost and implementation complexity to distribute highly sophisticated data. Our VLC link design differs in that it strives to develop a simple, scalable, and highly reliable system to communicate basic yet secure information across devices in a constrained space. Our framework's novel encryption key sharing mechanism via VLC has been studied minimally in previous mechanisms that either focus on the physical layer aspects of VLC or highly specific situations. Jive is more broadly applicable in daily secure communication transactions as it does not require the complex overhead installations and ad hoc use cases of existing mechanisms.

3 JIVE SYSTEM

In this section, we describe the entire Jive system, which we designed with the intent to create a system with a simple implementation and accuracy. Figure 1 depicts an overview of the entire

system. The system makes two assumptions: (1) for the time required to exchange a single secret key, the ambient light level does not change and (2) the exchanged keys will be carried in messages with a constant, known length.

From a high level perspective, the operation of Jive proceeds from the top of Figure 1 to the bottom. When initially powered on, the VLC transmitter generates a randomized bit string (the secret key) and communicates the data one-way to the receivers. This transmission is simply continuously repeated until the transmitter is powered off. Each receiver communicates the received data from the VLC transmitter with an Android application using a direct connection between the receiving microcontroller and the Android device. In the future, we envision the receiver itself to be incorporated into the device hosting the application. As a result of this approach, only device receivers that are exposed to the light will have knowledge of the shared secret key. The last step of the Jive system is shown at the bottom of Figure 1, in which the receiver devices use the secret key to exchange encrypted application-level information. Devices can encrypt and decrypt messages in some other communication medium (e.g., WiFi) using this shared symmetric key. Whenever the transmitter is powered off and back on, it generates a new random key, which it commences sharing. In this way, to share a secret "session key", two Jive receivers must be co-located in both space and time. The remainder of this section describes the Jive system design and methodology in further detail.

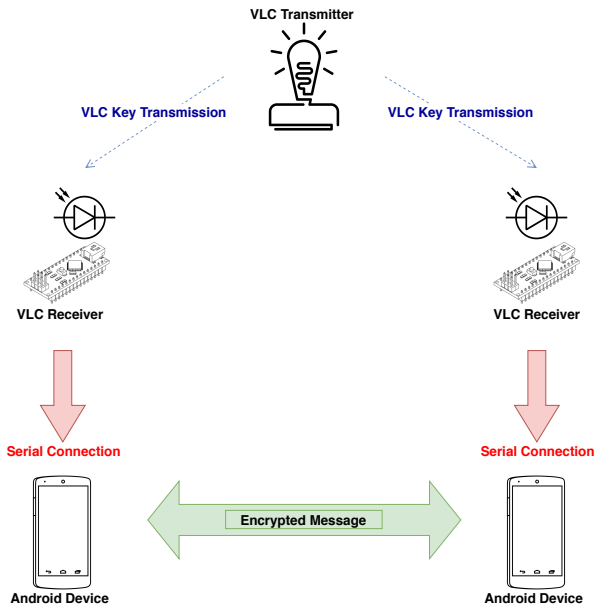

Figure 1: High-level Jive System Architecture. The VLC transmitter (at the top) generates and transmits a random sequence of bits that define the space's secret session key. Any VLC receivers that capture the transmission share it with their connected Android device, which can subsequently use it for encrypted communication over another communication medium. When the transmitter is powered off and back on, it generates a new key and restarts the process.

3.1 Hardware

As a foundation for the VLC system, many previous efforts for implementing VLC have chosen to develop on top of FPGAs [7, 17]. In contrast, we were highly motivated to construct Jive using off-the-shelf components for ease of programming, use, and replication. Other efforts have also relied on off-the-shelf microcontrollers but have typically relied on the Arduino Uno as a microcontroller of choice [15]. However, we found that the Arduino uses an oscillator rather than a crystal clock. In a VLC application, this creates issues relating to interrupt timing and jitter, causing calibration drift and disrupting system synchronization, which in turn results in a dramatic decrease in the overall system accuracy. To avoid these issues, we implemented Jive using the Teensy 3.5 microcontroller, which in turn uses an ARM Cortex M4 processor. Although the software implementations on the Teensy and the Arduino remain mostly the same, a key notable difference is the Teensy's Interval Timer library, which allows for highly accurate and fast periodic interrupt triggers based on the Teensy's 120Mhz clock (in contrast to the Arduino's 16Mhz clock).

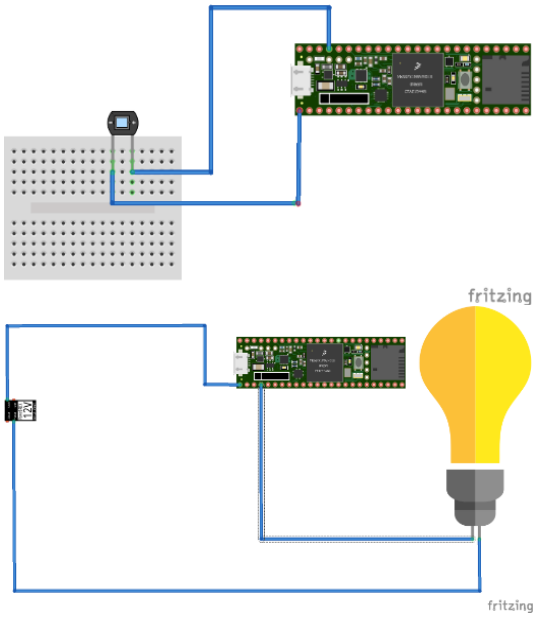

Figure 2: Hardware schematics of VLC receiver (top) and transmitter (bottom). Both involve a Teensy microcontroller connected to a very simple circuit using inexpensive off-the-shelf components.

The Teensy is connected to a simple VLC transmitter or receiver circuit. In both cases, the simple circuit design includes a minimal number of off-the-shelf components. The transmitter (see the bottom of Figure 2) includes a Teensy microcontroller connected to a small DC LED panel through a GPIO output for rapid toggle ability. The receiver (see the top of Figure 2) consists of a Teensy microcontroller connected to a PIN photodiode through a GPIO input. Both the LED and photodiode should include fast response time. Further, the proper focus of light on the photodiode is critical

to the success of the VLC system. Table 1 depicts the total cost of building both complete circuits. The affordability of these items gives way for an accessible means of designing a convenient, yet accurate VLC system.

Table 1: Bill of Materials. Both transmitter and receiver use inexpensive and widely-available off-the-shelf components.

Item	Part Number	Cost in USD
Teensy Chip	MK64FX512VMD12	$15.13
Teensy 3.5 MCU	Teensy 3.5	$25
PIN Photodiode	PD438C	$2
Transmitting LED	Linrose LED Module	$6.69
12-volt Battery	A23	$4.80

3.2 Encoding Algorithm

When choosing an encoding algorithm for Jive, we aimed to maximize the accuracy of data transmission while minimizing the ability for a user to perceive the light pulses. Although previous VLC systems attempt to use On-Off keying (OOK) [7, 16] as a simple encoding process with high-bandwidth, a general problem with OOK occurs when the system attempts to transmit long, consecutive sequences of zeroes. OOK will cause a VLC system to output low until a long, consecutive sequence of zeroes has completed. These extended periods of time when the light is off can cause flicker issues and a possible strobing effect. As a result, the VLC light pulses would be perceptible to the human eye, preventing the integration of the VLC system into the general lighting environment. Consequently, we chose to use Manchester encoding, which mitigates the flicker effect by switching between high and low output states at every bit in the message. Figure 3 depicts the Manchester encoding scheme and how bits are ciphered into rising and falling edges rather than high and low states.

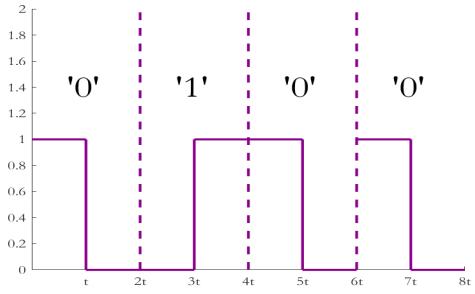

Figure 3: Figure of Manchester encoded signal. Effectively, every bit window is divided into two segments; the bit encoded is determined by whether the signal falls (0) or rises (1) in the middle of a given window. The signal can use the edges of the window to adjust in preparation to rise or fall within the next window.

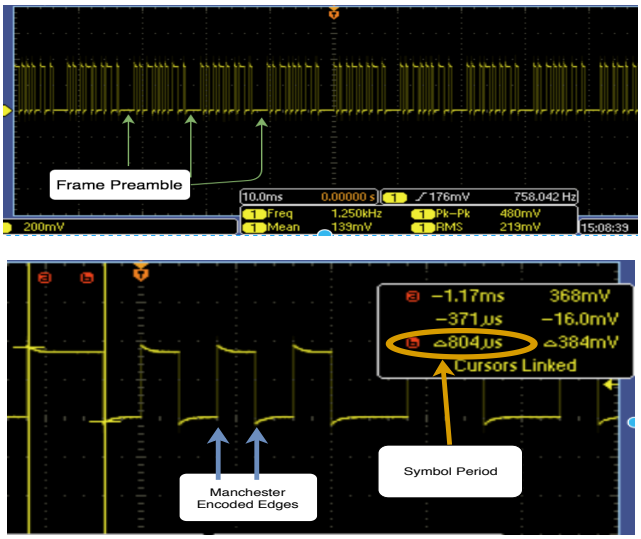

Figure 4: Waveforms of Manchester Encoded Signals with Inverted Logic. The figures show captures from the system at an 800 microsecond symbol period. The top figure clearly depicts the six high half-bit preamble for every packet. The bottom figure shows a zoomed in view of a sequence of Manchester encoded symbols generated in Jive.

3.3 Synchronization

As described in more detail below and shown in Figure 5, for each "session" (i.e., when the transmitter is powered on), a Jive transmitter generates a *message* that contains the entire key to be transmitted for that session. These messages are then fragmented into *frames*, or smaller segments that are actually transmitted. Jive requires synchronization at both the frame level and the message level.

Frame Level Synchronization. Jive segments the generated key into 8 bit frames for transmission. Before sending a frame of data, the transmitter sends a sequence of six high "half-bits" (i.e., three full bit windows that do not have any rise or fall in the middle). Additionally, each message is prefaced with a one bit (i.e., a rising bit) at the beginning and a zero bit (i.e., a falling bit) at the end to ensure that the message begins on low and ends on low. This step is necessary for the preamble algorithm to function properly. In Manchester encoding, this sequence of six high half-bits will never be construed to be any part of a message since bits are encoded through rising and falling edges. As a result, this preamble sequence is efficient for accurately transmitting and synchronizing data at the frame level. The image at the top of Figure 4 displays the transmitter signal with this frame preamble labeled.

Message Level Synchronization. For the receiver to be able to rebuild the full message from the received 8-bit segments, a Jive transmitter prepends a message level preamble. This preamble allows a newly entering receiver to detect the start of a message and assists transmitters and receivers to maintain synchrony over time. In Jive, we use the 8-bit sequence 0xAA for this preamble. After receiving the message preamble, the receiver assembles the next set of

consecutive packets based on the known Jive message length. This has the added restriction that the sequence 0xAA cannot appear in any of the randomly generated keys; after randomly generating a session key, the transmitter ensures this is the case before sharing it. If the randomly generated key contains 0xAA, the Jive transmitter simply creates a new key. In conjunction, these two synchronization mechanisms boost system accuracy and dramatically reduce packet drops.

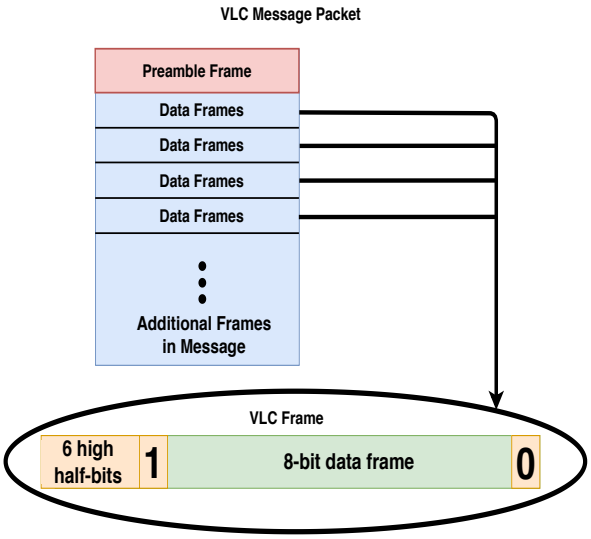

Figure 5: Jive Message and Frame. Jive generates a random sequence of bits to use as a shared session key, which it puts in a message, prefaced by a preamble (top of figure). Jive then fragments this message into smaller frames (8 bits), which are transmitted via VLC (bottom).

3.4 VLC Software

The software implementation of the VLC link is fairly straightforward. The software for both the transmitter and receiver rely on carefully timed periodic interrupts.

The transmitter software generates a random new encryption key upon each microcontroller reset and continuously transmits this same key until externally reset (e.g., through a power cycle). This password is represented as a sequence of bits and stored in a message buffer (Figure 5). The size of the buffer and message are parameters that can be specified; the length determines the length of the shared key and thus the strength of the security between the connected mobile applications. After the transmitter codes fills the buffer in the setup phase, its main loop enters a low power mode as it waits for the periodic interrupt to trigger each half symbol period. When this interrupt fires, the transmitter reads bit data from the buffer, encodes each bit with Manchester encoding, and outputs the appropriate logic level signal.

The receiver software likewise runs in low power mode until its periodic interrupt is triggered at every half symbol period. The half symbol period and frame parameters on both the transmitter and receiver must be identical. During each interrupt, the receiver's photodiode samples light data; each sample is then processed by the Teensy's 10-bit ADC. The receiver determines whether the received ADC value should be high or low based on a pre-specified analog threshold. We find in the evaluation section that this threshold can be modified to improve the accuracy of our link in different environments with respect to both ambient light and distance; future work will explore an auto-calibration phase in the receiver to allow it to adapt to changing ambient lighting conditions. Once sufficient bits are successfully received, the interrupt decodes and rebuilds each frame, finally assembling the received frames into the larger message

Given Jive's high accuracy and low numbers of dropped frames, we implement only a basic error correction algorithm. In particular, a Jive receiver decodes the repeated message multiple times and compares the received messages. If the same message has been consistently received for pre-specified count, the VLC receiver reports it as the correct message to the Android device. This implementation aids us in filtering messages that have only rare bit errors associated with them.

3.5 Android Application

To demonstrate Jive in a complete encryption system, we developed an Android application that takes a shared secret key received over VLC and uses it to encrypt application-level data sent between multiple Android devices.

We incorporated symmetric encryption algorithms in our application using the BouncyCastle API [13]. Via a serial connection to the Teensy, the Android app receives the bits from the VLC receiver and uses SecretKeySpec to initialize a new secret key with the received sequence of bits. The app allows the user to input a string to be encrypted; it then uses the secret key to encrypt the user-provided string using the Advanced Encryption Standard (AES) algorithm (a symmetric key encryption algorithm). This encrypted text can then be sent to a neighboring Android device via the WiFi interface. Upon being received on the other side, the Android application employs AES decryption to recover the originally entered text. Decryption uses the same key to recover the message; because both receiver devices can "see" the key transmitted via VLC, they can successfully and securely communicate.

In more detail, the function of the application proceeds as follows. When the user opens the app and begins to enter text to be encrypted, the app opens the USB serial (with the help libraries) [6] connection to the Teensy to receive the key shared via VLC. Once the key is received, the user is prompted to confirm that he or she is ready to encrypt the message with the received key. With the plain text in the box and the key received, the user can tap the button to encrypt, and the button triggers a call to start the AES encryption. Figure 6 depicts what the user sees the moment they start the app. The app begins with the screen shown and awaits for the teensy to send a random key for encryption. Once the "Encrypt Message" button is pressed, it triggers the algorithm and prints out the encrypted message (from the plain text user enters) in the text view below the button. In practice, displaying the encrypted message to the user is not useful; this is implemented in the app just for demonstration purposes. We also transmit this encrypted

data to another Android device with a similar app meant to decrypt the encrypted data.

This second device is also connected to a Jive receiver and receiving the same VLC light, ergo the same key. Upon receiving the ciphered text (e.g., via WiFi or some other communication medium outside of VLC), this app displays the cipher text in the textbox. The user can tap the decrypt button, and the AES decryption algorithm can proceed to use the key received over VLC to revert the cipher text back into plain text for the user to view at the end of the app.

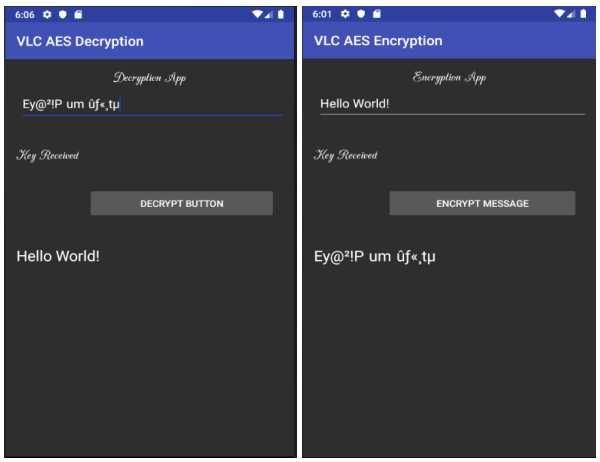

Figure 6: User Interface for Android Encryption application. The left image shows the decryption application, outputting a deciphered message after receiving the key and encrypted message inputs. The right image shows the encryption application outputting an encrypted message after receiving the key and message inputs.

4 EVALUATION

In this section, we evaluate the results produced from our prototype of the Jive system. We examine system performance across several parameters that effect our system including the half-bit symbol period[1], the link distance, and the capture threshold. We attempt to determine the consistency, reliability, and user perception under these changing conditions.

4.1 Experimental Setup

We implemented Jive as described in the previous section. All of the code used to implement Jive is publicly available[2]. We conducted our experiments in a laboratory setting where the ambient light can be easily controlled. Figure 7 shows the setup, with the VLC transmitter on the left hand side and the receiver to the right.

In every scenario described below, we tested the system for accuracy by sending 180 identical keys to the receiver. The number of keys received correctly divided by 180 (the total number of messages sent) is considered to be the accuracy of Jive at those

[1]In the remainder of this section, when we refer to "symbol period", we always intend the period for a half-bit.
[2]https://github.com/UT-MPC/JIVE_VLC

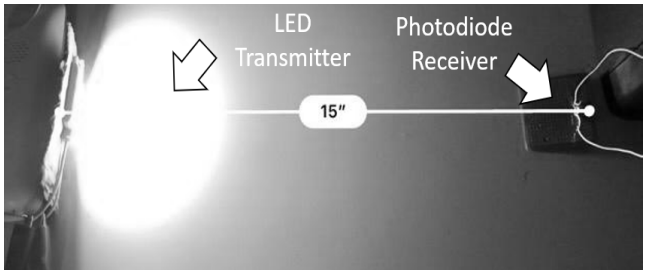

Figure 7: Experimental Setup. The figure shows an experiment under low ambient light; the transmitter is on the left, and the receiver is on the right.

settings. We use this accuracy metric to evaluate our system given that reliable transmission is critical to our target application.

To isolate the quality of our prototype and algorithms in our evaluation, we turned off our error correction mechanism for these experiments. In all experiments the receiver was placed in direct line of sight to the light, and we did not vary the incidence angle. Our experimental results record distance between VLC transmitter and receiver in inches (1 inch = 2.54 centimeters).

A primary goal of our evaluation was to determine the reliability of our VLC link when used to share symmetric encryption keys. Thus, we chose to focus on analyzing distinct factors that potentially play a role in our system's accuracy.

4.2 User Perception

To systematically determine the minimum half-bit symbol period that maximizes the accuracy but minimizes perception of flicker by users, we surveyed 7 individuals between the ages of 20-22, asking them to view the transmitter under different settings. Our approach was similar to that taken in the DarkLight case study [17]. In our case, we asked the participants to observe the transmitter while it was transmitting 3 messages containing 3 different randomly generated 128 bit keys. We attached the transmitter's LED to a constant 12 volt source to ensure consistent measurements and asked the participants to stand 5 feet back from the transmitter's LED. We then varied the symbol period and the environment's ambient light and asked the participants to state whether they were able to discern a flicker in the light while the data was being transmitted.

Each set of experiments started with a high ambient light level and the symbol period set to 5000μs. We repeated the experiment with the same symbol period and low ambient light. Then we decreased the symbol period and repeated the experiments with high ambient light and low ambient light, in turn. Table 2 shows the average detection rates across all participants for the different conditions we tested. The results at a 1000μs symbol period are interesting; although nearly half of the participants were able to see the flicker when the ambient lighting was low, almost no one was able to detect the flicker in high ambient light. We believe that this is because the environment distracts the participant from noticing the subtle flicker.

Table 2: Results of User Study. This table shows the percentage of participants who detected a flicker in the VLC transmission across all tested settings.

Symbol Period (μs)	Ambient Light Level	% Detection
5000	High	100.00
5000	Low	100.00
1000	High	14.20
1000	Low	42.86
800	High	0.00
800	Low	0.00
400	High	0.00
400	Low	0.00

It is also interesting to note that all the people who were able to see the light flicker in low ambient light in symbol period 1000μs were wearing glasses. With high ambient light, many of these same participants were not able to distinguish the flashing. At symbol periods of both 800μs and 400μs, none of the participants could discern the flicker.

As a result, we determined that a symbol period of 800μs is sufficiently fast to avoid human detection but slower rates are not. This informs the design of Jive since some design choices are incompatible with this symbol period. We now investigate the impact of this symbol period on Jive's data rate, and, in turn, the accuracy of the system.

4.3 Data Rate

Our first experiments attempted to determine the performance of Jive under different data rates for the VLC link. In Jive, the data rate is specified by modifying the symbol period of every half-bit of transmission. Although Jive does not require a link with extraordinarily high bandwidth, we did desire to design our system to communicate data fast enough such that the human eye could not detect the VLC transmitter pulses. We are therefore motivated to minimize the user perceptibility of the system.

Figure 8 shows Jive's accuracy when varying the symbol period (x-axis) from 400μs up to 1000μs. We use 400μs as the lower end because this was the fastest data rate that our transmitter and receiver could support at reasonable distances[3]. As the figure shows, the highest data rate (400μs) produced the most accurate and precise results. This is likely because Jive was able to transmit more messages in a shorter period of time, leaving less overall transmission time for noise to negatively impact the system.

The following equation relates Jive's data rate to the symbol period, where T is our pre-specified half-bit symbol period:

$$data\ rate = \frac{8}{(6 * T) + (2 * T) + (16 * T) + (2 * T)} \quad (1)$$

The numerator's 8 bits accounts for the useful information sent per frame. The four terms of the denominator account for the actual components of the frame: the preamble (6 half bits), the 1 symbol that precedes the data (2 half bits), the data itself (16 half bits), and the closing 0 symbol (2 half bits). This computes the data rate of the individual frames; we consider the single frame preamble at the start of a message to be negligible. Based on this equation, Jive's maximum data rate, given a symbol period of 400μs, is just over 750bps.

[3]Jive does support a 100μs symbol period (3 kbps data rate), but only with a lower capture threshold and much shorter distance; we opted to use the 400μs period because it allowed for more reasonable capture thresholds and link distances.

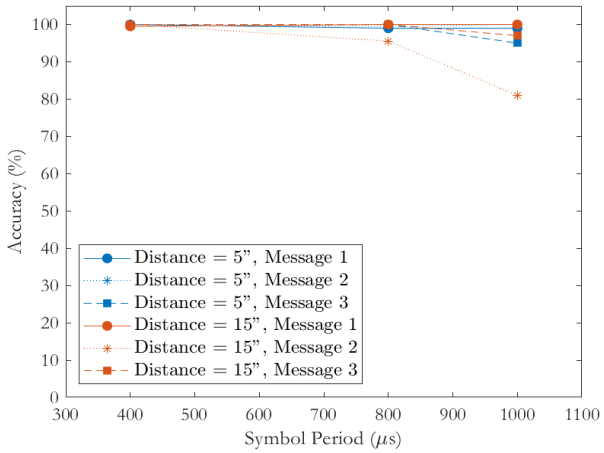

Figure 8: Jive's Performance Under Varying Symbol Periods and Link Distances. These experiments use a capture threshold of 75 and were conducted in our low ambient light environment. The distances used were 5 inches (12.7cm) and 15 inches (38.1cm).

Coupled with the above experiments on user perception, we opt to use a default symbol period of 400μs because it (i) provides the fastest data rate our transmitter can support; (ii) results in a flickering that is imperceptible to humans; and (iii) provides the highest accuracy in terms of data delivery

4.4 Distance

We also analyzed the impact of varying the distance between the transmitter and receivers. We chose our lengths to test the system in small, medium, and large distances. Figure 8 also shows these results. We make two observations, first, the system tends to perform better at shorter distances. Based on the inverse square law for light:

$$I \propto \frac{1}{d^2} \quad (2)$$

the trend makes sense as the system's receiver has an easier time distinguishing the high and low half bits when a higher intensity light signal is transmitted at a closer distance. A wider range of threshold values would result in a higher accuracy at low distances. Our second, less trivial observation is that the variance in accuracy across different messages increases as distance increases. All experiments in this section were conducted with a constant, calibrated threshold value of 75 ADC; this value represents the binary number generated by converting the voltage signal from the photodiode through the data acquisition system. It is directly proportional to the intensity of the transmitted light: a stronger light requires a

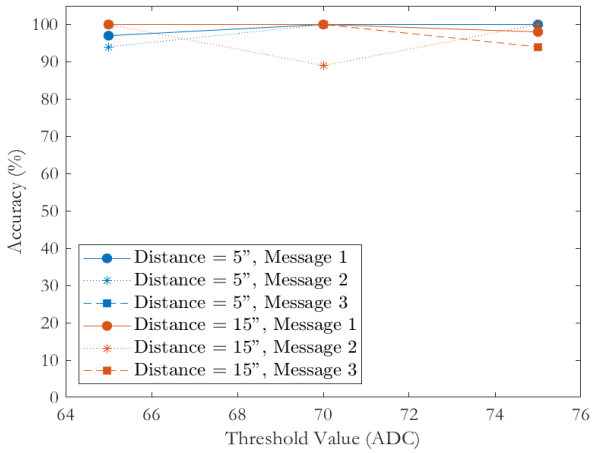

Figure 9: Jive's Performance Under Varying Capture Thresholds. The accuracy of the link decreases as distance and capture threshold increase. The distances used were 5 inches (12.7cm) and 15 inches (38.1cm).

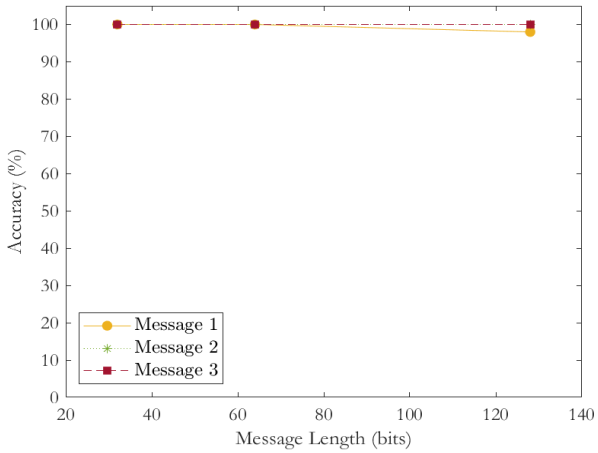

Figure 10: Jive's Performance for Varying Length Messages. The chart shows that Jive is virtually unaffected by increasing message length.

higher threshold, which was far from ideal at larger distances[4]. These results prompted us to better understand the relationship between the light capture threshold and the distance of the VLC link.

4.5 Capture Threshold

In Jive, the capture threshold determines whether a given sampled half bit should be considered as a high or low value. In general, the threshold value should correspond to the center point of the ADC samples received in the communication system. A well-calibrated threshold is critical to the accuracy and success of the VLC link. Since threshold is directly related to light intensity, we conducted an experiment relating distance to threshold in our system. In this experiment, we transmitted our three distinct messages with default parameters of: 128 bit message length, 8 bit frame size, and 400μs symbol period.

Figure 9 shows the results of the experiment. We observe that accuracy is typically worse on trials with higher distance and higher threshold. Equation 2 describes why the phenomenon occurs. A high threshold signifies that the incoming light intensity will increase, causing overall higher ADC values. As a result, we found that high thresholds at high distances not only reduced accuracy, but in many instances prevented transmission of messages all together.

4.6 Message Content

Because Jive must be expected to successfully transmit any randomly generated message, we desired to ensure that the settings are not over-tuned to a specific message content. To evaluate this, we computed Jive's accuracy in transmitting three distinct randomly generated messages. For the three messages we generated, across all combinations of parameters (i.e., distance, symbol period), we found the accuracy to be within ±2%, which is within reasonable error bounds.

4.7 Message Length

Our final setup involved an analysis of sending messages of varying length. Although we varied the total message length, we maintained our frame size of 8 bits because larger frame sizes did not synchronize as often as 8 bit frames, causing calibration drifts as the message was transmitted. Default parameters for the experiment include a 400μs symbol period and a link distance of 14 inches (35.56cm). Figure 10 shows the results; at least for messages up to 128 bits (the target size of our encryption key), the message length does not have a demonstrable impact. [5]

4.8 Discussion

While designing a VLC system, it can be easy to overlook extreme cases that can occur, especially when transmitting randomly generated data. To measure Jive's resilience to such edge cases, we tried messages consisting of entirely 0s or entirely 1s and found no differences from the results reported in the preceding section.

Although we collected data on several important aspects of the system, there are many additional factors we could still measure that have been captured in others' experiments [7, 18], including more widely varying ambient light conditions, the incidence angle between the transmitter and receiver, and the signal to noise ratio on the transmission lines. It is important to keep in mind that although many of these additional factors may affect our VLC link, they are not essential to measure for our vision of transmitting encryption keys through a secured space. The focus of our VLC

[4]With the photodiode we use for our experiments, we read and calibrated the ADC for a capture threshold of 75, which was ideal for our low ambient light environment. We then used this same value for consistent data collection.

[5]Upon testing with higher message lengths (256 bits), we were able to transmit successfully by changing the message length parameter, thereby indicating that Jive is scalable to longer length messages.

system is to reliably transmit data through low implementation complexity within a highly enclosed and controlled environment.

5 APPLICATIONS

With current communication mechanisms highly susceptible to eavesdropping, a significant amount of effort is exerted in protecting wireless communication. These efforts often require heavyweight and user-intensive mechanisms for exchanging encryption keys (in the case of symmetric encryption) or resource-powerful devices capable of the computation necessary for asymmetric key encryption. Our approach to symmetric key exchange over VLC helps to solve this issue in a variety of application domains. In particular, with Jive, only devices that can simultaneously "see" the same light can share encrypted data. The following list contains several examples in which this exchange has a strong potential.

- **Health Care**: Imagine a "smart exam room" of the future, in which a patient, a nurse, and a doctor all have mobile devices that can exchange medical information securely only when located together in the same exam room. Information about the patient cannot "leak" outside of the exam room because any overheard communication will be encrypted with a key that is only "visible" within the exam room. Further, IoT devices are also revolutionizing the healthcare space. Smart health devices (e.g., blood pressure monitors, lab machines, etc.) could similarly be equipped with the ability to receive VLC and insert measures into the patient's health record.
- **Trade Secrets**: The concept of "secure" rooms can also be applied in both business domains and in defense applications. Consider a meeting room equipped with VLC lighting executing the Jive system. Only meeting participants with access to the room *during the meeting* will have received the shared secret transmitted via VLC. All communications, documents, etc., generated in the meeting can therefore be restricted to only those devices that were also present during the meeting and able to "see" the secret key created for that meeting. When the meeting ends and the participants depart, they simply have to turn off the light to protect the secrets generated during the meeting.
- **Smart Home Authentication of IoT Devices**: As a final example, consider the challenge of setting up a smart home device. These efforts usually involve the homeowner installing the device (e.g., light), a bridge from the device manufacturer, and interacting with the bridge and device during setup by demonstrating that the user has physical access to both (e.g., by pressing a physical button and/or entering a long sequence of randomly generated characters). With Jive, this process can be reduced to both the user's device and the smart home device "seeing" the same VLC signal to finalize the authentication. With a simple VLC communication, the user is assured that they are the only ones who can have access to the device and can complete the set up worry-free.

6 CONCLUSION AND FUTURE WORKS

We successfully implemented and tested Jive, a system that bootstraps secure local communication by sending symmetric encryption keys using visible light communication. Jive is built entirely with easy-to-acquire, off-the-shelf components. We constrained Jive's design to a low-powered LED and further improved on our efficiency by swapping the oscillator clock of the commonly used Arduino with a crystal one of a Teensy microcontroller, thus eliminating calibration drifts and significantly increasing accuracy in transmitting over distances of around two feet. A Jive transmitter randomly generates an encryption key, which is then sent to any co-located receivers using VLC. Relying on an existing encryption library for Java [13], we created simple apps for Android that allow a received key to be used to encrypt information sent in a secondary communication medium. The apps allow users to view how Jive can be employed as part of a security process for information exchange among co-located mobile applications. In the paper, we also discussed a variety of benchmarks used to measure the accuracy of the system over varying data rates, distances, capture thresholds, and properties of the message itself. Jive is positioned to support a wide variety of applications, from protecting trade secrets or personal health records to making smart spaces more usable and easier to set up.

It is, however, important to acknowledge areas for potential improvement. To make the system more robust, we could improve Jive in a variety of ways. Firstly, the LED that Jive's transmitter currently uses limits the distances at which the photodiode can reliably receive the message because the LED drives low current. A different LED could provide a means for a brighter VLC effect. To compensate for a stronger LED, we would conduct further research with a more sensitive photodiode to capture the pulses as accurately as possible, which would allow Jive to function more reliably in widely varying ambient lighting conditions. In the future, our goal would be to work with superior microcontrollers that can provide even faster clock speeds. Finally, there have been shown to be some caveats related to the inherent security of the VLC link [1]; future work should investigate the resilience of Jive to similar "leaks" of the VLC transmission. Overall, however, Jive demonstrates a very practical, simple, and realizable application of VLC across several mobile computing domains of significant interest to a wide variety of users.

REFERENCES

[1] J. Classen, J. Chen, D. Steinmetzer, M. Hollick, and E. Knightly. 2015. The Spy Next Door: Eavesdropping on High Throughput Visible Light Communications. In *Proceedings of the 2Nd International Workshop on Visible Light Communications Systems*. 9–14. https://doi.org/10.1145/2801073.2801075

[2] J. Classen, D. Steinmetzer, and M. Hollick. 2016. Opportunities and Pitfalls in Securing Visible Light Communication on the Physical Layer. In *Proceedings of the 3rd Workshop on Visible Light Communication Systems*. 19–24. https://doi.org/10.1145/2981548.2981551

[3] S. De Lausnay, L. De Strycker, J. Goemaere, N. Stevens, and B. Nauwelaers. 2014. Matlab based platform for the evaluation of modulation techniques used in VLC. In *2014 International Conference on Development and Application Systems (DAS)*. 57–61. https://doi.org/10.1109/DAAS.2014.6842427

[4] H. Haas. 2015. The Future of Wireless Light Communication. In *Proceedings of the 2nd International Workshop on Visible Light Communications Systems*. https://doi.org/10.1145/2801073.2801692

[5] T. Hao, R. Zhou, and G. Xing. 2012. COBRA: Color Barcode Streaming for Smartphone Systems. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*. ACM, 85–98. https://doi.org/10.1145/2307636.2307645

[6] Felipe Hernandez. 2014. USBSerial. https://github.com/felHR85/UsbSerial.

[7] K. Hewage, A. Varshney, A. Hilmia, and R. Voigt. 2016. modBulb: A Modular Light Bulb for Visible Light Communication. In *Proceedings of the 3rd Workshop on Visible Light Communication Systems*. 13–18. https://doi.org/10.1145/2981548.2981559

[8] Q. Jing, A. Vasilakos, J. Wan, J. Lu, and D. Qiu. 2014. Security of the Internet of Things: perspectives and challenges. *Wireless Networks* 20, 8 (Nov 2014), 2481–2501. https://doi.org/10.1007/s11276-014-0761-7

[9] P. Karthik, B. M. Kumar, B. A. Ravikiran, K. Suresh, and G. Toney. 2016. Implementation of visible light communication (VLC) for vehicles. (May 2016), 673–675. https://doi.org/10.1109/ICACCCT.2016.7831724

[10] M.E. Kounavis, X. Kang, K. Grewal, M. Eszenyi, S. Gueron, and D. Durham. 2010. Encrypting the Internet. In *Proceedings of the ACM SIGCOMM 2010 Conference*. 135–146. https://doi.org/10.1145/1851182.1851200

[11] C. Lartigue, J. Green, H. Perez-Olivas, J. Garcia-Marquez, and S. Topsu. 2018. High-efficient manchester coding for beacon-to-CMOS camera in visible light communications. In *2018 Global LIFI Congress (GLC)*. 1–4. https://doi.org/10.23919/GLC.2018.8319121

[12] Shuai Li, Ashish Pandharipande, and Frans M. J. Willems. 2016. Two-Way Visible Light Communication and Illumination With LEDs. *IEEE Transactions on Communications* 65, 2 (nov 2016), 740 – 750. https://doi.org/10.1109/TCOMM.2016.2626362

[13] Tau Ceti Co operative Ltd. 2013. Legion of the Bouncy Castle, Inc. Retrieved May 23rd, 2019 from https://www.bouncycastle.org/

[14] P. Prabu, N. G. Bhuvaneswari, and G. Annapoorani. 2015. Interfacing Visible Light Communication with GSM Networks to Prevent the Theft of the Vehicle. *ICTACT Journal on Communication Technology* 06 (09 2015), 1136–1140. https://doi.org/10.21917/ijct.2015.0165

[15] S. Schmid, G. Corbellini, S. Mangold, and T.R. Gross. 2013. LED-to-LED Visible Light Communication Networks. In *Proceedings of the Fourteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 1–10. https://doi.org/10.1145/2491288.2491293

[16] S. Schmid, B. von Deschwanden, S. Mangold, and T.R. Gross. 2017. Adaptive Software-Defined Visible Light Communication Networks. In *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation*. 109–120. https://doi.org/10.1145/3054977.3054983

[17] Z. Tian, K. Wright, and X. Zhou. 2016. The Darklight Rises: Visible Light Communication in the Dark: Demo. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. 495–496. https://doi.org/10.1145/2973750.2987384

[18] X. Xu, Y. Shen, J. Yang, C. Xu, G. Shen, G. Chen, and Y. Ni. 2017. PassiveVLC: Enabling Practical Visible Light Backscatter Communication for Battery-free IoT Applications. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*. 180–192. https://doi.org/10.1145/3117811.3117843

[19] B. Zhang, K. Ren, G. Xing, X. Fu, and C. Wang. 2016. SBVLC: Secure Barcode-Based Visible Light Communication for Smartphones. *IEEE Transactions on Mobile Computing* 15, 2 (Feb 2016), 432–446. https://doi.org/10.1109/TMC.2015.2413791