Ripple: Communicating through Physical Vibration

Nirupam Roy   Mahanth Gowda   Romit Roy Choudhury

University of Illinois at Urbana-Champaign

Abstract

This paper investigates the possibility of communicating through vibrations. By modulating the vibration motors available in all mobile phones, and decoding them through accelerometers, we aim to communicate small packets of information. Of course, this will not match the bit rates available through RF modalities, such as NFC or Bluetooth, which utilize a much larger bandwidth. However, where security is vital, vibratory communication may offer advantages. We develop *Ripple*, a system that achieves up to 200 bits/s of secure transmission using off-the-shelf vibration motor chips, and 80 bits/s on Android smartphones. This is an outcome of designing and integrating a range of techniques, including multi-carrier modulation, orthogonal vibration division, vibration braking, side-channel jamming, etc. Not all these techniques are novel; some are borrowed and suitably modified for our purposes, while others are unique to this relatively new platform of vibratory communication.

1 Introduction

Data communication has been studied over a wide range of modalities, including radio frequency (RF), acoustic, visible light, etc. This paper envisions vibration as a new mode of communication. We explore the possibility of using *vibration motors*, present in all cell phones today, as a transmitter, while *accelerometers*, also popular in mobile devices, as a receiver. By carefully regulating the vibrations at the transmitter, and sensing them through accelerometers, two mobile devices should be able to communicate via physical touch.

We are not the first to recognize this opportunity. Acoustic communication operates on the same fundamental principles and has been studied for decades (over air [24, 20] and under water [12]). In recent years, authors in [32] identified the possibility of using vibra-motors and accelerometers in mobile phones, as an opportunity to exchange information. The benefits were identified as security and zero-configuration, meaning that the two devices need not discover each other's addresses to communicate. The act of physical contact would serve as the implicit address. However, authors identified the drawbacks of such a system to be low bit rates ($\sim$ 5 bits/s), based on the "morse-code" style of ON/OFF communi-

cation with vibrations. Still, researchers conceived creative applications, including secure smartphone pairing and keyless access control [47].

This paper is aimed at improving the data rates of vibratory communication, as well as its security features. We design *Ripple*, a system that breaks away from the intuitive *morse-code* style ON/OFF pulses and engages techniques such as orthogonal multi-carrier modulation, gray coding, adaptive calibration, vibration braking, side-channel suppression, etc. While some techniques are borrowed from RF/acoustic communication, unique challenges (and opportunities) emerge from the vibra-motor/accelerometer platform, as well as from solid-materials on which they rest. For instance, the motor and the materials exhibit resonant frequencies that need to be adaptively suppressed; accelerometers sense vibration along 3 orthogonal axes, offering the opportunity to use them as parallel channels, with some degree of leakage. In addition to such techniques, we also design a receiver cradle – a wooden cantilever structure – that amplifies/dampens the vibrations in a desired way. A vibration based product in the future, say a point-of-sale equipment for credit card transactions, may potentially benefit from such a design.

From a security perspective, Ripple recognizes the threat of acoustic leakage due to vibration, i.e., an eavesdropper could listen to the sound of vibration and decode the transmitted bits. To thwart such side channel attacks, we design the transmitter to also listen to the sounds and adaptively play a synchronized acoustic signal (through its speaker) to cancel the sound. The transmitter also superimposes a jamming sequence, ultimately offering inherent protection from acoustic eavesdroppers. We observe that application layer securities may not apply in all such scenarios – public/symmetric key based encryption infrastructure may not scale to billions of phones and other use-cases such as internet of things (IoT). Blocking access to the signal, at the physical layer itself, is desirable in these spontaneous, peer-to-peer, and perhaps disconnected situations [41].

Its natural to wonder *what kind of applications will use vibratory communication, especially in light of NFC*. We do not have a killer app to propose, and even believe that most applications would prefer NFC, mainly due to its higher data rates (NFC uses 1.8MHz bandwidth achiev-

ing more than 100 Kbits/s, in contrast to 800Hz with today's vibra-motors). However, our hope is that bringing the vibratory bit rates to a respectable level – say credit card transactions in a second – may trigger new ideas and use-cases. In particular, strict security-sensitive applications may be the candidates. Despite the very short communication range in NFC, recent results [40, 28] confirm that security threats are real. Authors decode NFC transmissions from 1m away [14, 21, 22] and conjecture that high-gain beamforming antennas can further increase the separation. With the natural security benefits of touch-based communication (over RF), and supplemented with acoustic cancellation and jamming, we attempt to set a higher security bar for *Ripple*.

Moreover, the ubiquity of vibration motors in every cell phone, even in developing regions, presents an immediate market for vibratory communication. Peer to peer money exchange with recorded logs is a global problem, recently recognized by the Gates Foundation; hidden camera attacks on ATM kiosks have been rampant in many parts of India and south Asia [25]. Paying local cab drivers with phone-vibrations, or using phones as ATM cards can perhaps be of interest in developing countries. Clandestine operations may benefit where information need to be exchanged without leaving any trace in the wireless channel or in the Internet. Finally, if link capacity proves to be the only bottleneck, perhaps improved vibration motors can be included to mitigate it in the next phone models. While it's difficult to anticipate the needs of the future, we focus our attention on enabling and pushing forward this new modality of vibratory communication. To this end, our main contributions may be summarized as:

- Harnessing the vibration motor hardware and its functionalities, from a communication perspective.

- Developing an orthogonal multi-carrier communication stack using vibra-motor and accelerometer chips, and repeating the same for Samsung smartphones. Design decisions for the latter are different due to software/API limitations on smartphones, where vibra-motors were mainly integrated for simple alerts/notifications.

- Identifying acoustic side channel attacks and using signal cancellation and jamming to offer physical layer protection to eavesdropping.

2 Vibration Motors and Accelerometers

We begin with a high level overview of vibration motors and accelerometers (substantial details in [33, 34, 13]).

2.1 Vibration Motor

A vibration motor (also called "vibra-motor") is an electro-mechanical device that moves a metallic mass around a neutral position to generate vibrations. The motion is typically periodic and causes the center of mass (CoM) of the system to shift rhythmically. There are mainly two types of vibra-motors depending on their working principle:

(1) Eccentric Rotating Mass (ERM): This type of vibration generators uses a DC motor to rotate an eccentric mass around an axis as depicted in Figure 1(a). As the mass is not symmetric with respect to its axis of rotation, it causes the device to vibrate during the motion. Both the amplitude and frequency of vibration depend on the rotational speed of the motor, which can in turn be controlled through an input DC voltage. With increasing input voltages, both amplitude and frequency increase almost linearly and can be measured by an accelerometer.

(2) Linear Resonant Actuators (LRA) generate vibration by linear movement of a magnetic mass, as opposed to rotation in ERM (Figure 1)(b). With LRA, the mass is attached to a permanent magnet which is suspended near a coil, called "voice coil". Upon applying AC current to the motor, the coil also behaves like a magnet (due to the generated electromagnetic field) and causes the mass to be attracted or repelled, depending on the direction of the current. This generates vibration at the same frequency as the input AC signal, while the amplitude of vibration is determined by the signal's peak-to-peak voltage. Thus LRAs allow for regulating both the magnitude and frequency of vibration separately. Fortunately, most mobile phones today use LRA based vibra-motors.
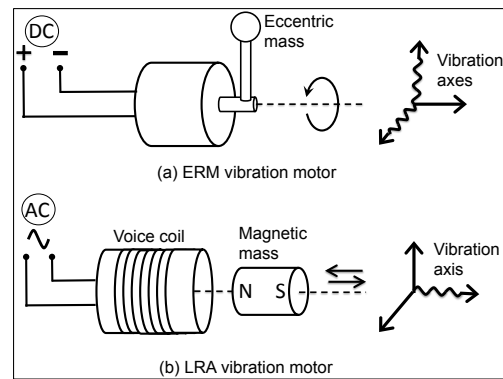

Figure 1: Basics of ERM and LRA vibra-motors.

Regulating Vibration

Ideally, a controller should be able to regulate the vibra-motor at fine granularities using any analog waveform. Unfortunately, micro-controllers produce digital voltage values limited to a few discrete levels. A popular technique to approximate analog signals with binary voltage

levels is called <mark>Pulse Width Modulation (PWM) [8]. This technique is useful to drive analog devices with digital data without a digital to analog converter (DAC).</mark>

PWM based Motor Control: The core idea in PWM is to approximate any given voltage V by rapidly generating square pulses and configuring the pulse's duty cycle appropriately. For example, to create a $1V$ signal with binary voltage levels of $5V$ and $0V$, the duty cycle needs to be 20%. Now, if the period of the square pulse is made very small (i.e., high frequency), the effective output voltage will appear as $1V$. <mark>Towards this goal, the PWM frequency is typically set much higher than the response time of the target device so that the device experiences a continuous average voltage.</mark> Importantly, it is also possible to generate varying voltages with PWM, say a sine wave, by gradually changing the duty cycles in a sinusoidal fashion.

2.2 Accelerometer

The accelerometer is a micro electro-mechanical (MEMS) device that measures acceleration caused by motion. While the inner workings of accelerometers can vary [7], the core working principle pertains to a movable seismic mass that responds to the vibration of the object it is attached to. Capacitive accelerometers, shown in Figure 2, are perhaps most popular in smartphones today. When vibrated, the seismic mass moves between fixed electrodes, causing differences in the capacitance c_1 and c_2, ultimately producing a voltage proportional to the experienced vibration.
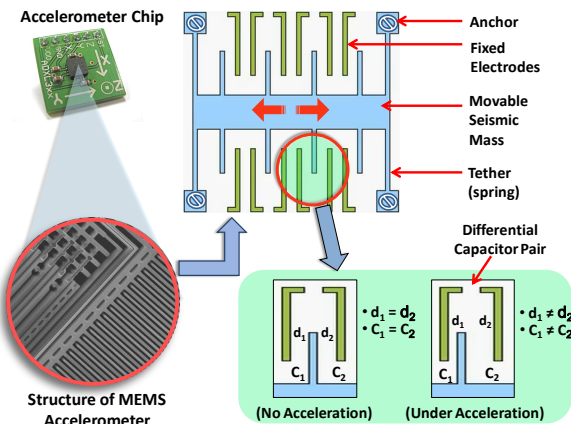

Figure 2: The internal architecture of MEMS accelerometer chip used in smartphones [19].

Sensing Acceleration

Modern accelerometers sense the movement of the seismic mass along 3 orthogonal axes, and report them as an $< X, Y, Z >$ tuple. The gravitational acceleration appears as a constant offset along the axis pointed to-wards the floor. The newest accelerometer chips support a wide range of adjustable sampling rates, typically from 100 mHz to $3.2KHz$. For this paper, we choose the ADXL345 [18] capacitive MEMS accelerometer, not only because it is used in most smartphones, but also because of programmability and frequency range.

3 Vibratory Transmission and Reception

Software/API limitations in smartphones prevent fully exploiting the vibra-motors and accelerometers. We design a custom hardware prototype using the same chips that smartphones use, and characterize/evaluate the system. We develop the constrained smartphone version in the next section.

3.1 Custom Hardware Setup

We control the vibra-motor and accelerometer through Arduino boards [1], an open source hardware development platform equipped with a ATmega328 8-bit RISC micro-controller [2]. Our first step is to precisely control the vibration frequency (and amplitude) through a time-varying sequence of voltage levels fed to the vibra-motor. Unfortunately, the micro-controller's output current fluctuates, leading to errors in the transmitted vibratory signals. Therefore, we power the vibra-motor with a stand-alone $6V$ DC power supply and use the Arduino micro-controller signal to operate a switch that regulates the voltage to the motor. We develop a simple circuit shown in Figure 3 – a NPN Darlington transistor (TIP122) serves as the switch and the controller signal goes to its base.
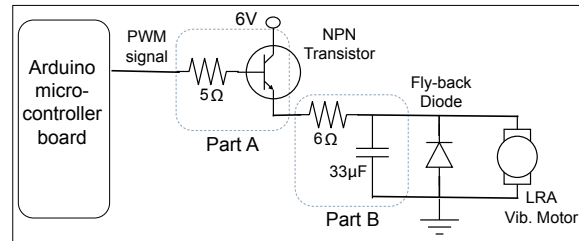

Figure 3: Transmitter hardware: the micro-controller controls a switch that regulates the $6V$ DC input.

Let's assume that we intend to regulate the vibra-motor in a sinusoidal fashion. We pre-load digital samples of the sine waveform into memory, and PWM uses them to determine the width of the square waves. When the sine wave frequency needs to be increased, the same digital samples need to be drawn at a faster rate and at precise timings. The switch uses the PWM output to regulate the $6V$ DC signal. We mitigate a number of engineering problems to run the set up correctly, including harmonic distortions due to the square pulses, spikes due to back EMF, etc. We move the PWM frequency to a

high $32KHz$ and use an RC filter (part B Figure 3) to remove the distortions; we use a 1N4001 fly-back diode to smooth out the spikes. We omit further details in the interest of space.

The accelerometer receiver is also controlled through Arduino via the I2C protocol [44] at 115200 baud rate. We set the accelerometer's sampling rate to 1600Hz and 10 bit output resolution. While higher sampling rates are possible, we refrain from doing so since the micro-controller records the accelerometer data at a slower rate. In particular, the chip produces a sample per 0.625ms, but the micro-controller takes around $8 - 12$ms to periodically read and write in memory. We handle this with a FIFO mode of the accelerometer, such that the queued-up data is read in a burst. We also mount an on-board SD card to store data via the SPI protocol.

Figure 4 shows the accelerometer output when the vibra-motor is driven by the sinusoid input and made to touch the accelerometer. The final system functions correctly, and the platform is now ready for design and experimentation.
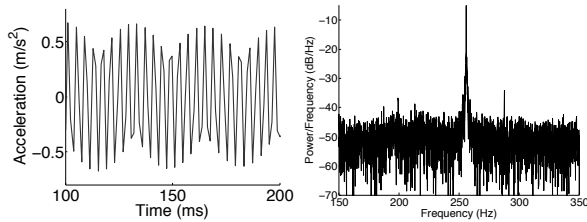

Figure 4: Accelerometer output (a) time and (b) frequency, when vibra-motor fed with a 250Hz sine wave.

3.2 Transmitter and Receiver Design

Ripple's design firmed up after multiple rounds of iterations. In the final version, the transmitter performs amplitude modulation on 10 different carrier signals uniformly spaced from 300 to $800Hz$ – each carrier is modulated with a bandwidth of $40Hz$. Further, the vibrations are also parallelized on orthogonal motion dimensions (X and Z) with appropriate signal cancellation. The design details are presented next.

Selecting the Carrier Signal
To reason about how data bits should be transmitted, we first carry out an analysis of the available spectrum. This available spectrum is actually bottlenecked by the maximum sampling rate of the accelerometer receiver – since this rate is 1600Hz, the highest frequency the transmitter can use is naturally 800Hz. Now, to test the system's frequency response in the $[0, 800]$ band, we perform a "sine sweep" test. The transmitter, with the help of a waveform generator, produces continuously increasing frequencies from 1Hz to 800Hz with constant amplitude (the frequency increments are at 1Hz). Figure 5

shows the corresponding vibration magnitudes recorded by the accelerometer. Evidently, the response is weak up to 60Hz (called the "inert band"), followed by improvements till around 200Hz, followed by a large spike at around 231Hz. This spike is near the resonant frequency of the vibra-motor (confirmed in the data sheet).
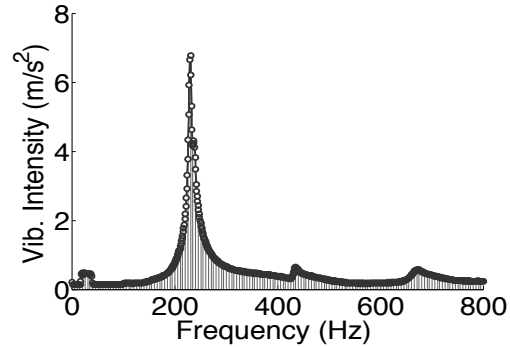

Figure 5: The vibra-motor's frequency response with the resonant frequency at around 231Hz.

Intuitively, frequencies near the resonant band can serve as good carriers for amplitude modulated data because of a larger vibration range. However, when we plot the frequency versus time spectrogram of the sine sweep test (Figure 6), we find that the vigorous vibration around the resonant frequency spills energy in almost the entire spectrum. Therefore, transmitting on the resonant band can be effective for a single carrier system, but the interference ruins the opportunity to transmit data in parallel carriers. In light of this, we define a "resonant band" of 100Hz around the peak, and move the carrier signals outside this band. We select 10 orthogonal carriers separated by 40Hz from the non-resonant frequencies between 300Hz and 800Hz. The 40Hz separation ensures the non-overlapping sidebands for the carriers, allowing reliable symbol recovery with software demodulation.
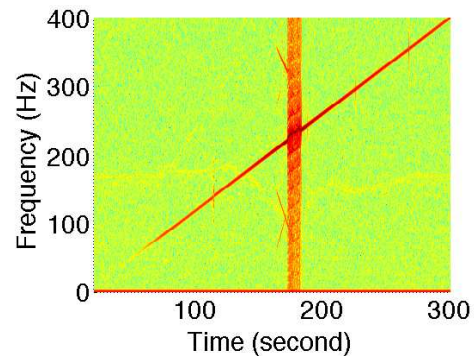

Figure 6: When excited with the resonant frequency, the vibra-motor spills energy across a wide frequency range.

Synchronization
Micro-controllers inject timing errors at various stages – variable delay in fetching digital samples from memory,

during time-stamping the received samples, and due to oscillator/crystal frequency shifts with temperature. The timing errors manifest as fluctuations in vibration frequency, causing error in demodulation. To synchronize time between the transmitter and receiver, we introduce a pilot frequency at 70Hz and transmit it in parallel to data bits. We choose 70Hz to be above the inert band and lower than the resonant band. During reception, the receiver detects the pilot frequency, measures the offset in sampling rate, and interpolates the received signal by adjusting for this offset. Of course, this operation also corrects all other frequencies in the spectrum needed for demodulation.

(De)Modulating the Carrier Signal

The carrier frequencies are modulated with *Amplitude Shift Keying* (ASK) in light of its bandwidth efficiency and simplicity over Frequency Shift Keying (FSK). We modulate each of the 10 carriers with binary data at a symbol rate of 20Hz. To prevent inter-carrier interference, we shape the pulses with a raised cosine filter for each carrier individually; the modulated carriers are then combined and fed to the vibration motor transmitter. The receiver senses the energy in the pilot carrier, calibrates and synchronizes appropriately to identify the beginning of transmission. We again filter the received spectrum with (the same) raised cosine filter to isolate each carrier, and proceed to demodulate individual carriers separately. Figures 7(a) and (b) show a part of the spectrum before and after filtering, for an example carrier frequency at 405Hz. The demodulation is performed with envelope detection and precise sampling at bit intervals. We will evaluate this custom-designed system in Section 6 and show ∼200 bits/second data rates through vibration.
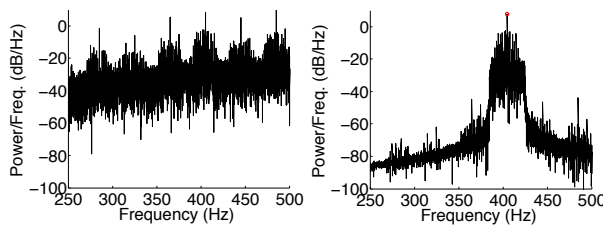

Figure 7: The spectrum (a) before and (b) after filtering for a single carrier frequency at 405Hz.

3.3 Orthogonal Vibration Dimensions

The above schemes, although adapted for vibra-motors, are grounded in the fundamentals of radio design. In an attempt to augment the bit rate, we observed that a unique property of accelerometers is its ability to detect vibration on 3 orthogonal dimensions (X, Y, and Z). Although vibra-motors only produce signals on a single dimension, perhaps multiple vibra-motors could be used in parallel. Unfortunately, due to some rigidity in our custom set up, accelerometer's motion along the X axis is minimal, precluding it for communication. Therefore, we orient two vibra-motors in the Y and Z dimensions and execute the exact multi-carrier amplitude modulated transmissions discussed above.

Measurements show that vibration from one dimension spills into the other. However, rather interestingly, this spilled interference exhibits a 180° phase lag with respect to the original signal, as well as an attenuation in the amplitude. Figure 8 shows an example in which the Z axis signal (solid black) has a spill on the Y axis, with a reversed phase and halved amplitude. The vice versa also occurs. Now, to remove Z's spilled interference and decode the Y signal, we scale the Y signal so that the interference matches Z's actual amplitude, and then add it to the Z signal. The Z signal is removed quite precisely, leaving an amplified version of Y, which is then decoded through the envelope detector. The reverse is performed with Z's signal, resulting in a 2*x* improvement in data rate, evaluated later.
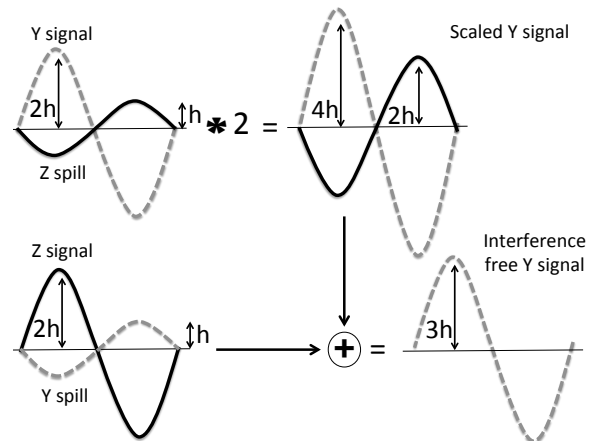

Figure 8: Orthogonal vibrations in X and Z axes.

4 Smartphone Prototype

This section shifts focus to vibratory communication on Android smartphones. Android is of interest since it offers APIs to a kernel level PWM driver for controlling the ON/OFF timings. We develop a user space module that leverages third-party kernel space APIs [5] to control the vibration amplitudes as well. However, this still does not match the custom set-up in the previous section. The PWM driver in Samsung smartphones is set to operate on the resonant band of the LRA vibra-motor, and the vibration frequency cannot be changed. This is understandable from the manufacturer's viewpoint, since vibra-motors are embedded to serve as a 1 bit alert to the user. However, for data communication, the non-linear response at the resonant frequencies presents difficulties. Nonetheless, Ripple has to operate under these constraints and hence is limited to a single carrier frequency, modulated via amplitude modulation.

4.1 Smartphone Tx and Custom Rx

One advantage of the resonant frequency is that it offers a larger amplitude range, permitting *n*-ary symbols as opposed to binary (i.e., the amplitude range divided into *n* levels). To further amplify this range, we also design a custom smartphone cradle – a cantilever based wooden bridge-like framework – that in contact with the phone amplifies specific vibration frequencies. While we will evaluate performance without this cradle, we were curious if (deliberately designed) auxiliary objects bring benefits to vibratory communication. Figure 9 shows the design – when the transmitter phone is placed on a specific location on this bridge, and the accelerometer connected to the other end, we indeed observe improved SNR. The key idea here is to make the "channel" resonate along with the smartphone to improve transmission capacity. We elaborate on the cantilever based design next, followed by the communication techniques.

Cantilever based Receiver Setup

Observe that every object has a *natural frequency* [46] in which it vibrates. If an object is struck by a rod, say, it will vibrate at its natural frequency no matter how hard it is struck. The magnitude of the strike will increase the amplitude of vibration, but not its frequency. However, if a periodic force is applied at the same natural frequency of the object, the object exhibits amplified vibration – resonance. In our set-up, we use a 1 foot long wooden beam supported at one end, called a cantilever (Figure 9). The smartphone transmitter placed near the supported end, impinges a periodic force on the beam, calculated precisely based on the beam's resonant frequency (inversely proportional to $\sqrt{(weight)}$). We adjust the weight of the structure so that its natural frequency matches that of the phone's vibra-motor (which lies between 190Hz to 250Hz). This creates the desired resonance.
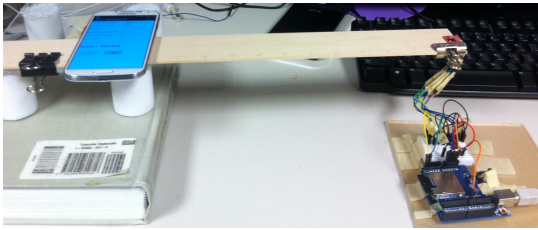

Figure 9: Cantilever based receiver platform for vibration amplification.

The accelerometer is attached at the unsupported end of the beam. Figure 10 plots the measured amplitude variation (over 3 axes of the accelerometer) as the smartphone is placed on different positions on the beam. We choose the position located 6 inches from the supported end, as it induces maximal amplification on all 3 axes of the accelerometer.
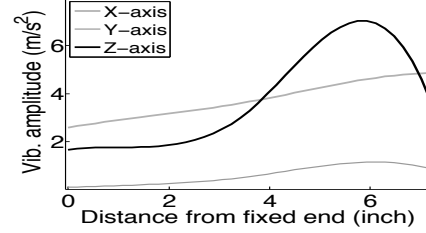

Figure 10: Vibration highest at a specific phone location.

Symbol Duration and the Ringing Effect

Ripple communicates through amplitude modulation – pulses of *n*-ary amplitudes (symbols) are modulated on the carrier frequency for a symbol duration. Ideally, the effect of a vibration should be completely limited within this symbol duration to avoid interference with the subsequent symbol (called inter-symbol interference). In practice, however, the vibration remains in the medium even after the driver stops the vibrator, known as the *ringing effect*. This is an outcome of inertia – the vibra-motor mass continues oscillating or rotating for some period after the driving voltage is turned off. Until this extended vibration dampens down substantially, the next symbol may get incorrectly demodulated (due to this heightened noise floor). Moreover, the free oscillation of the medium also contributes to ringing. Figure 11(a) shows a vibratory pulse of the smartphone, where the vibra-motor is activated from 20 to 50 ms. Importantly, the motor consumes 30 ms to overcome static inertia of the movable mass and reach its maximum vibration level. Once the voltage is turned off (at 50ms) the vibration dampens slowly and consumes another 70 ms to become negligible. This dictates the symbol duration to be around $30 + 70 = 100$ ms to avoid inter-symbol interference.
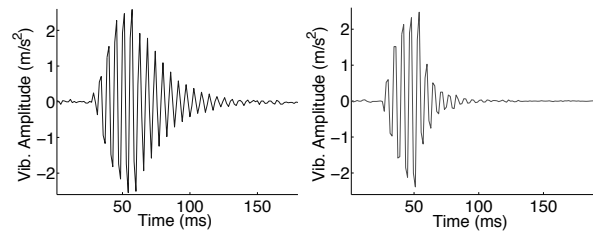

Figure 11: (a) Ringing effect in the channel. (b) Reduced ringing using a braking voltage.

Vibration Dampening

To push for greater capacity, we attempt to reduce the symbol duration by dampening the ringing vibration. The core observation is that the ringing duration is a function of the amplitude of the signal – a higher amplitude signal rings for a longer duration. If, however, the amplitude can be deliberately curbed, ringing will still occur but will decay faster. Based on this intuition, we apply a small *braking-voltage* to the vibra-motor right after the signal has been sampled by the demodulator

(30ms). This voltage is deliberately small so that it does not manifest into large vibrations, and is applied for 10ms. Once braking is turned off, we allow another 10ms for the tail of the ringing to die down, and then transmit the next symbol. Thus the symbol duration is 50ms now (half of the original) and there is still some vibration when we trigger the next symbol. While this adds slightly to the noise floor of the system, the benefits of a shorter symbol duration out-weighs the losses. Moreover, an advantage arises in energy consumption – triggering the vibra-motor from a cold start requires higher power. As we see later, activating it during the vibration tail saves energy.

(De)Modulation

The (de)modulation technique is mostly similar to a single carrier of the custom hardware prototype. The only difference is that it uses multiple levels of vibration amplitudes (up to 16), unlike the binary levels earlier. Figure 12 shows how we can vary the voltage levels (as a percentage of maximum input voltage) to achieve different vibration amplitudes. If adequately stable, the amplitude at each voltage level can serve as separate symbols. Given the linear amplitude slope from voltage levels 15 to 90%, we divide this range into n-ary equi-spaced amplitude levels, each corresponding to a symbol. However, due to various placements and/or orientations of the phone, this slope can vary to some degree. While this does not affect up to 8-ary communication, 16 symbols are susceptible to this because of inadequate gaps between adjacent amplitude levels. To cope, we use a preamble of two symbols. At the beginning of each packet the transmitter sends two symbols with the highest and lowest amplitudes (15 and 90). The receiver computes the slope from these two symbols, and calibrates all the other intermediate amplitude levels from them. The receiver then decodes the bits with a maximum likelihood based symbol detector.
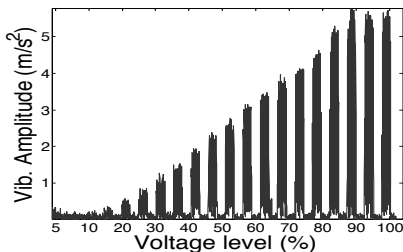

Figure 12: The change of vibration amplitude with the percentage of maximum input voltage.

5 Security

Vibrations produce sound and can leak information about the transmitted bits to an acoustic eavesdropper [29, 3, 9]. This section is aimed at designing techniques that thwart such side channel attacks. We design this as a real-time operation on the smartphone.

5.1 Acoustic Side Channel

The source of noise that actually leaks information is the rattling of the loosely-attached parts of the motor – the unbalanced mass and metals supporting it. Our experiments show that this *sound of vibration* (SoV) exhibits correlations of ∼0.7 with the modulated frequency of the data transmission. Although SoV decays quickly with distance, microphone arrays and other techniques can be employed to still extract information. Ripple attempts to prevent such attacks.

5.2 Canceling Sounds of Vibration (SoV)

One way to defend against eavesdropping is to jam the acoustic channel with a *pseudorandom noise* sequence, thus decreasing the SNR of the SoV. Since this jamming signal will not interfere with physical vibrations, it does not affect throughput. Upon implementation, we realized that the jamming signal was audible, and annoying to the ears. The more effective approach is perhaps to cancel/suppress the SoV from the source, and then jam faintly, to camouflage the residue.

Ideally, Ripple should produce an "anti-noise" signal that cancels out the SoV to ultimately create silence. The transmitter (and not the receiver) should generate this anti-noise since it knows the exact bit sequence that is the source of the SoV. Of course, acoustic noise cancellation is a well studied area – several headphones today use a microphone to capture ambient sounds and blends a negative version of it through the headphone speakers. The challenge of course is in detecting the ambient sound in real time and producing the precise negative (phase shifted) signals. However, unlike Ripple, headphones need to cancel the ambient noise only at the human ear, and not at all other locations around the human.

With Ripple, the problem is easier in the sense that the transmitter exactly knows the bit sequence that is causing the SoV. This can help in modeling the sound waveform ahead in time, and can potentially be synchronized. The issue, however, is that the SoV varies based on the material medium on which the phone is placed; also the SoV needs to be cancelled at all locations in the surrounding area. Further, the phase of the SoV remains unpredictable as it depends on the starting position of the mass in the vibra-motor and the delay to attain the full swing. Finally, Android offers little support for real-time audio processing [10], posing a challenge to develop SoV cancellation on off-the-shelf phones.

5.3 Ripple Cancel and Jam

The overall technique is composed of 3 sub-tasks: anti-noise modeling, phase alignment, and jamming.

(1) Anti-noise modeling
The core challenge is to model the analog SoV waveform corresponding to the data bits that will be transmitted through vibration. Since the motor's vibration amplitude and frequency are known (i.e., the carrier frequency), the first approximation of this model is simple to create. However, as mentioned earlier, the difficulty arises in not knowing how the unknown material (on which the phone is placed) will impact the SoV. Apart from the fundamental vibration frequency, the precise SoV signal depends also on the strength and count of the *overtones* produced by the material. To estimate this, the Ripple transmitter first transmits a short "preamble", listens to its FFT, and picks the *top-K* strongest overtones. These overtones are combined in the revised signal model. Finally, the actual data bits are modeled in the time domain, reversed in sign, and added to create the final "anti-noise" signal. This is ready to be played on the speaker, except that the phase of anti-noise needs to precisely match the SoV.

(2) Phase Alignment with Frequency Switch
Unfortunately, Android introduces a variable latency of up to 10ms to dispatch the audio data to the hardware. This is excessive since a 2.5ms lag can cause constructive interference between the anti-noise and the SoV. Fortunately, two observations help in this setting: (1) the audio continues playing at the specified sample rate without any significant fluctuation, and (2) the sample rate of the active audio stream can be changed in real-time. Thus, we can now control the frequency of the online audio by changing the playback sample rate.

We leverage this frequency control to match the phase of anti-noise with the SoV. The key idea is to start the anti-noise as close as possible to the SoV, but increase the sampling frequency such that the fundamental frequency of the anti-noise increase by δf. When this anti-noise combines in the air with the SoV, it creates the amplitude of the sound to vary because of the small difference in the fundamental frequencies. Obviously, the maximum suppression of the SoV occurs when the amplitude of this combined signal is at its minimum. The phase difference between the SoV and anti-noise is almost matched at this point. At exactly this "phase-lock" time, Ripple switches the fundamental frequency of the anti-noise to its original value (i.e., lower by δf). It recognizes this time instant by tracking the envelope of the combined signal and switching frequencies at the minimum point on the envelope. Figure 13 illustrates the various steps leading up to the frequency switch, and the sharp drop in signal amplitude. The suppressed signal remains at that level thereafter.

(3) Jamming
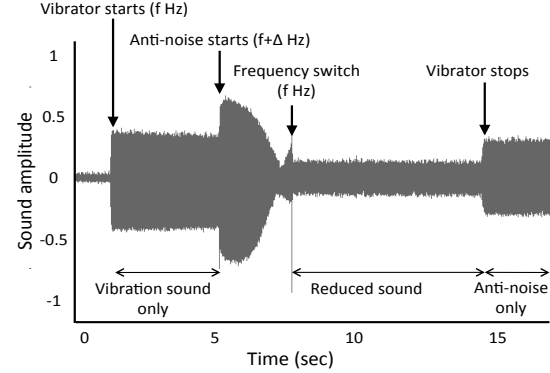The cancellation is not perfect because the timing of the operations are not instantaneous; microphone and

Figure 13: The anti-noise partially cancels the SoV, however, some mismatches result in some residual signal.

speaker noise also pollute the anti-noise waveforms, leaving a small residue. To prevent attacks on this residue, Ripple superimposes a jamming signal – the goal is to camouflage the sound residue. Conceptually it is simple, since a *pseudorandom noise* sequence can be added to the anti-noise waveform once it has phase-locked with the vibration sound. Unfortunately, Android does not allow loading a second signal on top of a signal that is already playing. Note that if we load the jamming signal upfront (along with the modeled anti-noise signal), the precise phase estimation will fail. We develop an engineering work-around. When modeling the anti-noise waveform, we also add the jamming noise sequence, but pre-pad the latter with a few *zeros*. Thus, when the SoV and anti-noise combine, the zeros still offer opportunities for detecting the time when the signals precisely cancel. We phase-lock at these times and the outcome is the residual signal from imperfect cancellation, plus the jamming sequence. We will show in the evaluation how the SoV's SNR degrades due to such cancellation and jamming, offering good protection to eavesdropping. Of course, the tradeoff is that we need a longer preamble now for this phase alignment process. However, this is only an issue arising from current Android APIs.

6 System Evaluation

We evaluate Ripple in three phases – the custom hardware, the smartphone prototype, and security.

6.1 Custom Hardware

Bit Error Rate (BER)
Recall that the custom hardware is composed of vibra-motors and accelerometer chips controlled by Arduino boards. We bring the two devices in contact and initiate packet transmission of various lengths (consuming between 1 to 10 seconds). Each packet contains pseudo-random binary bits at 20Hz symbol rate on 10 parallel carriers. The bits are demodulated at the receiver and compared against the ground truth. We repeat the exper-
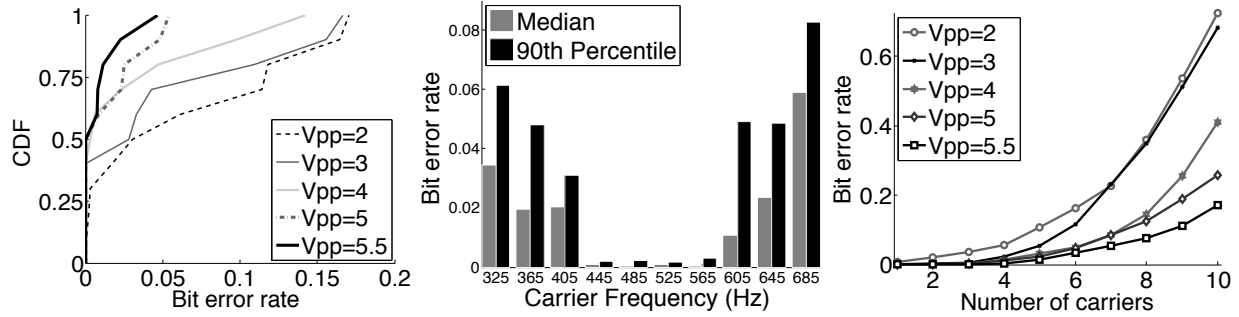
Figure 14: (a) BER as a function of the input signal peak-to-peak voltage (Vpp). Overall data rate ∼200 b/s. (b) Per-carrier BER across 10 frequencies. (c) BER as a function of no. of carriers used (each carrier bit rate = 20 bits/s).

iment for increasing signal energy (i.e., by varying the peak to peak signal voltage, Vpp, from 1V to 5V). Figure 14(a) plots the BER as a function of peak-to-peak input voltage (Vpp) to the vibra-motor and demonstrates how it diminishes with higher SNR. At the highest SNR, and aggregated over all carrier frequencies, Ripple achieves the 80th percentile BER of 0.017 translating to an average bit rate of 196.6 bits/s.

Behavior of Carriers

In evaluating BERs across different carrier frequencies, we observe that not all carriers behave similarly. Figure 14(b) shows that carrier frequencies near the center of the spectrum perform consistently better than those near the edges. One of the reasons is *aliasing noise*. Ideally, the accelerometer should low-pass-filter the signal before sampling, to remove signal components higher than the Nyquist frequency. However, inexpensive accelerometers do not employ anti-aliasing filters, causing such undesirable effects. Carriers near the resonant band also experience higher noise due to the spilled-over energy.

Increasing the number of carriers will enable greater parallelism (bit rate), at the expense of higher BER per carrier. To characterize this tradeoff, we transmit data on increasing number of carriers, starting from the middle of our spectrum and activating carriers on both sides, one at a time. Figure 14(c) shows BER variations with increasing number of carriers, for varying signal energy (peak-to-peak voltage, Vpp). As each carrier operates at fixed 20Hz symbol rate, this also shows the bit rate vs BER characteristics of our system. Figure 15 zooms on the best four carriers.

Temporal Stability

Given that vibra-motors and accelerometers are essentially mechanical systems, we intend to evaluate their properties when they are made to operate continuously for long durations. Given the low bit rates, this might be the case when relatively longer packets need to be transmitted. Towards this, we continuously transmit data for 50 sessions of 300 seconds each. Figure 16 plots the
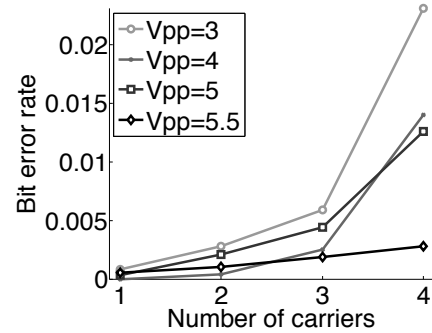

Figure 15: BER vs. number of carriers (4 carriers shown) per-carrier BER (computed in the granularity of 10 second periods) of a randomly selected session – the Y axis shows each of the carriers and the X axis is time. The BERs vary between 0.02 near the center to 0.2 near the edge. Overall results, omitted for the interest of space, show no visible degradation in BER even after running for 300 seconds.
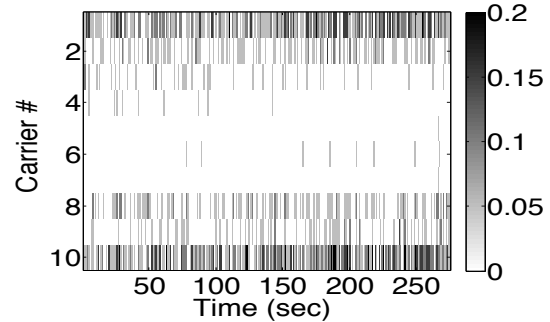

Figure 16: The BER per-carrier does not degrade after the motor is run for long durations.

Exploiting Vibration Dimensions

Recall that Ripple used 2 vibra-motors in parallel to exploit the orthogonality of vibrations along the Y and Z axes of the accelerometer. Figure 17(a) and (b) show the distribution of BER achieved across carrier frequencies on the Y and Z axes, respectively. We also attempt to push the limits by modulating greater than 20 bits/s, however, the BER begins to degrade. In light of this, Ripple achieves median capacity of around 400 bits/s (i.e., 20 bits/s per carrier x 10 carriers x 2 dimensions). While the

tail of the BER distribution still needs improvement, we believe coding can be employed to mitigate some of it.
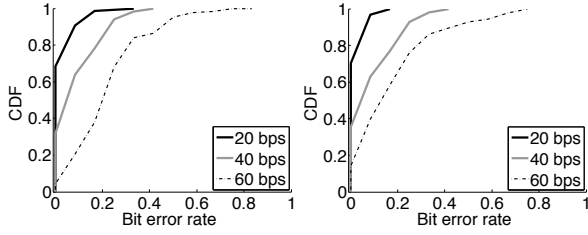

Figure 17: BER per carrier for parallel transmissions on orthogonal dimensions: (a) Y axis and (b) Z axis.

6.2 Smartphone Prototype

Calibration

Vibrations will vary across transactions due to phone orientation, humans holding it, different vibration medium, etc. As discussed earlier, the demodulator calibrates for these factors, but pays a penalty whenever the calibration is imperfect. We evaluate accuracy of calibration using the error between the estimated amplitude for a symbol, and the mean amplitude computed across all received symbols. Figure 18 plots the normalized error for various n-ary modulations – the normalization denominator is used as the difference between adjacent amplitudes.
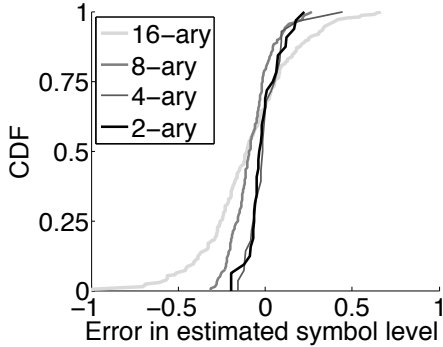

Figure 18: CDF of estimated symbol level error as a fraction of the mean inter-symbol difference.

BER with Smartphones

Figure 19(a) plots the confusion matrix of transmitted and received (or demodulated) symbols, for 16-ary modulation. While some errors occur, we observe that they are often the symbol adjacent to the one transmitted. In light of this, Ripple uses Gray codes to minimize such well-behaved errors. With these codes and calibration, Figure 19(b) shows the estimated BER for different bit rates, for each of the 4 modulation schemes. As comparison points, the "Basic" symbol detector uses predefined thresholds for each symbol and maps the received sample to the nearest amplitude. The "Ideal" scheme identifies the bits using the knowledge of all received symbols. Ripple's performs well even at higher bit rates, which is not the case with Basic.

Figure 19(c) shows the BER per symbol for 16-ary modulation, showing that symbols corresponding to the high vibration amplitudes experience higher errors. The reason is that the consistency of the vibration motor degrades at high amplitudes – we have verified this carefully by observing the distribution of received vibration amplitudes for large data traces.

Impact of Phone Orientation

The LRA vibra-motor inside Galaxy S4 generates linear vibration along one dimension – the teardown of the phone [11] shows the motor's axis aligned with the Z axis of the phone. Thus, an accelerometer should mostly witness vibration along the Z axis. The other two axes do not exhibit sufficient vibration at higher bit rates. This is verified in Table 1 where the first 4 data points are from when the phone is laid flat on top of the cantilever. However, once the phones are made to stand vertically or on the sides, its X and Y axes align with the accelerometers Z axis, causing an increase in errors. This suggests that the best contact points for the phones are their XY planes, mainly due to the orientation the vibration motor.

Table 1: BER with 16-ary for various orientations.

Orientation	Hor. A	Hor. B	Hor. C	Hor. D	Ver. A	Ver. B
Mean BER	0.025	0.029	0.002	0.029	0.197	0.178

Phone Held in Hand (No Cantilever)

We experiment a scenario in which the accelerometer based receiver is on the table, and the hand-held phone is made to touch the top of the receiver. The alignment is crudely along the Z axis. This setup adversely affects the system by (1) eliminating the amplitude gain due to the cantilever, and (2) the dampens vibration due to the hand's absorption. Figure 20 shows the results – unsurprisingly, the total vibration range is now smaller, pushing adjacent symbol levels to be closer to each other, resulting in higher BER.
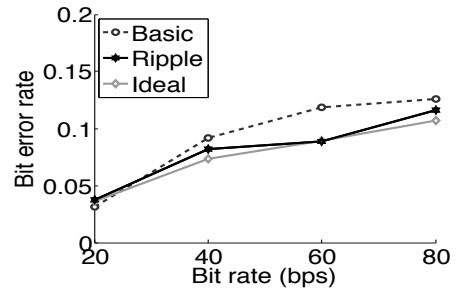

Figure 20: The BER with a hand-held phone.

6.3 Security
Acoustic Signal Leakage

To characterize the maximum acoustic leakage from vibrations, we run the vibra-motor at its highest intensity and record the SoV at various distances, using smartphone microphones sampled at 16KHz. This leakage is
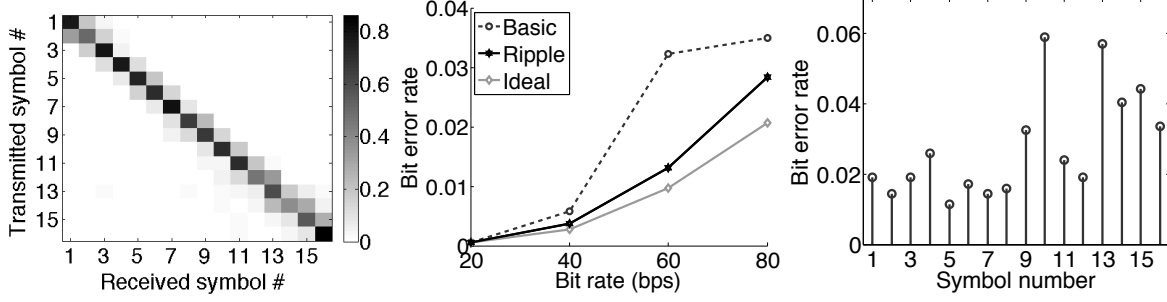
Figure 19: (a) This heat-map shows the confusion matrix of the transmitted and received symbols. (b) Ripple's BER compared to the Basic, Ideal. (c) Per symbol BER with 16-ary communication.

naturally far higher than a typical vibratory transmission (composed of various intensity levels), so mitigating the most severe leakage is stronger security. We also realize that the material on which the smartphone is placed matters, therefore, repeated the same experiment by placing the phone on (a) glass plate, (b) metal plate (aluminum), (c) on the top of another smartphone, and (d) our custom wooden cantilever setup. Figure 21 shows the contour plots for each scenario. Evidently, glass causes the strongest side channel leak, and wood is minimum. Following experiments are hence performed on glass.
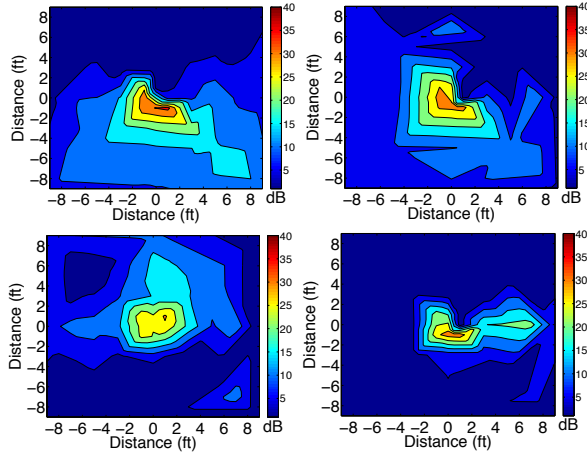

Figure 21: Acoustic side channel leakage on: (a) glass, (b) metal, (c) on another phone, and (d) wood.

Results indicate that the SoV is well below the socially acceptable noise level. At a distance of $2ft$, SoV is less than $25dB$, comparable to a soft whisper as per human perception of loudness [6]. We further quantify this by comparing SoV against the ambient noises recorded in 5 common locations – departmental store, inside a moving car, coffee shop, class room, and computer laboratory. Table 2 shows that the ratio remains close to 2.

Table 2: Ratio of power of SoV signals to ambient noise at public places.

Location	Dep. Store	Car	Coffee Shop	Class	Lab
Power ratio	1.57	1.81	2.01	2.10	2.31

Acoustic Leakage Cancellation

Recall that the Ripple receiver records the sound and produces a synchronized phase-shifted signal to cancel the sound, and superimposes a jamming sequence to further camouflage the leakage. Figure 22 shows the impact of cancellation using a ratio of the power of the residual signal to the original signal, measured at different distances. Evidently, the cancellation is better with increasing distance. This is because the generated "anti-noise" approximates the first few strong harmonics of the sound. However, the SoV also contains some other low-energy components that fade with distance making the anti-noise signal more similar to the vibration's sounds. Hence the cancellation is better at a distance, until around 4ft, after which residual signal drops below the noise floor and our calculated power becomes constant. The original signal also decreases but is still above the noise floor past 4ft, hence, the ratio increases.
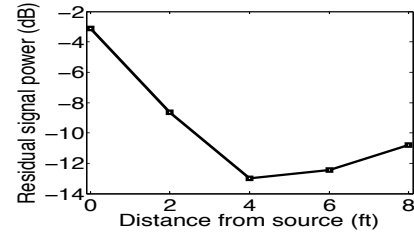

Figure 22: Ratio of residual to original signal power (in dB) at increasing distances from the source.

Acoustic Jamming

Ripple applies jamming to further camouflage any acoustic residue after the cancellation. To evaluate the lower bound of jamming efficiency, we make the experiment more favorable to the attacker. We transmit only two amplitude levels (binary data bits) at 10 bits per second. We place the phone on glass, the scenario that creates loudest sound. The eavesdropper microphone is placed as close as possible to the transmitter, without touching it. To quantify the efficacy of the jamming, we correlate the actual transmitted signal with the received jammed signal and plot the correlation coefficient in the Table 3. A high correlation coefficient indicates high probability

of correctly decoding the message by the adversary, and the vice versa. The table shows the correlation values for various ratios of the jamming to signal power. Evident from the table, the correlation coefficient sharply decreases when Ripple increases the jamming power.

Table 3: The mean and std. dev. of the correlation coefficient for increasing jamming to signal power ratio.

Power ratio	0	0.4	0.8	1.2	1.6	2
Corr. mean	0.68	0.55	0.35	0.19	0.18	0.09
Corr. std. dev.	0.027	0.015	0.017	0.008	0.003	0.003

7   Limitations and Future Work

Needless to say, this paper is an early step – some aspects need deeper treatment, as discussed below.

Bounds and Optimality. We have not derived an upper bound on the capacity of vibratory communication, nor do we believe that our design decisions are optimal. We have taken an engineering approach and developed an end-to-end solution using techniques borrowed from RF/acoustic communication. Further work is needed to "tighten" the design towards optimality, including gains from coding and cancellation (on X, Y, Z dimensions).

Energy. Given that vibra-motors can be energy consuming, its important to characterize the energy versus throughput tradeoffs. For smartphone applications, vibrations are likely to be used occasionally for short exchanges, so perhaps energy is not a major hurdle. Nonetheless, when the phone battery is low, the ability to adapt can be a valuable feature.

Other Side-channels. An attacker could exploit the visual channel with a high-speed camera [17] to decode the vibratory bits. Even physical eavesdropping may be a threat, where the attacker sneakily attaches an accelerometer to the surface on which the Ripple devices are located. A probable solution to such attacks can be "vibratory jamming". Essentially, the receiver's vibra-motor could generate a pseudo-random jamming vibration while receiving the data from the transmitter. Of course, the transmitter is unaware of this and performs normal transmission. The net vibration video-recorded by the attacker's camera is actually the sum of two vibrations, hiding the actual transmitted bits. However, since the receiver knows the pseudo-random jamming sequence it has deliberately injected, it can cancel it out. Of course, this pseudo-random vibration should have enough power to create desirable entropy at the transmitter, else the eavesdropper can focus only on the transmitter's vibration. We leave the viability of these attacks and mitigations to future work.

8   Related Work

Vibration generation and sensing: Applications in haptic HCI for assisted learning, touch-augmented environments, and haptic learning have used vibrations for communication to humans [39, 23, 31, 42, 16]. However, the push for high communication data rates between vibrators and accelerometers is relatively unexplored. Off late, personal/environment sensing on mobile devices has gained research attention. Applications like (sp)iPhone [36] and TapPrints[38] demonstrate the ability to infer keystrokes through background motion sensing. While many more efforts are around activity recognition from vibration signatures, this paper aims to modulate vibration for communication.

Vibratory communication: The papers [45] and [32] are probably closest to Ripple. They both encode vibrations through ON-OFF keying, with ON/OFF durations in the range of a second (i.e., around 1 bits/s). This is adequate for applications like secure pairing between two smart phones, or sending a tiny URL over tens of seconds. However, unlike Ripple, they do not focus on the wide range of PHY and cross-layer radio design issues and possible security leaks. Dhwani [41] is an elegant work on acoustic NFC and addresses conceptually similar problems, however, their acoustic platform are appreciably different from Ripple.

Technologies like Bump [4, 37, 43, 15, 30, 27, 35] use accelerometer/vibrator-motor response to facilitate secure pairing between devices. However, these techniques are primarily designed to exchange small signatures, as opposed to the arbitrary data transmission in Ripple. As indicated by researchers [45, 26], the lack of the dynamic secret message in Bump-like techniques makes them less secure in the wild. These modes also require Internet connectivity and trusted third party servers to function, none of which is needed in Ripple.

9   Conclusion

This paper is an attempt to explore a new modality of communication – vibration. Through multi-carrier modulation, orthogonal vibration division, and leakage cancellation, our system, Ripple, is able to achieve 200 bits/s alongside a strong level of security against side channel attacks. While there is room for improvement, we believe this paper could serve as a stepping stone for exciting vibration-based technologies and applications.

Acknowledgement

References

[1] Arduino. `http://arduino.cc`.

[2] Atmel microcontroller. `http://www.atmel.com/Images/doc8161.pdf`.

[3] Audible noise reduction. `http://www.danfoss.com/NR/rdonlyres/87FE3A20-0598-48C1-8A4B-75CFA8B25FC2/0/AudibleNoiseReduction.pdf`.

[4] Bump technologies. `http://bu.mp`.

[5] Cyanogenmod. http://www.cyanogenmod.org.

[6] Loudness comparison. `http://www.gcaudio.com/resources/howtos/loudness.html`.

[7] Practical guide to accelerometers. `http://www.sensr.com/pdf/practical-guide-to-accelerometers.pdf`.

[8] Pulse width modulation. `http://homepage.cem.itesm.mx/carbajal/Microcontrollers/ASSIGNMENTS/readings/ARTICLES/barr01_pwm.pdf`.

[9] Reducing audible noise in vibration motor. http://www.precisionmicrodrives.com/tech-blog/2013/08/15/reducing-audible-noise-in-vibration-motors.

[10] Reducing audible noise in vibration motor. http://createdigitalmusic.com/2012/07/android-high-performance-audio-in-4-1-and-what-it-means-plus-libpd-goodness-today/.

[11] Samsung galaxy s4 teardown. `https://www.ifixit.com/Teardown/Samsung+Galaxy+S4+Teardown/13947`.

[12] BRADY, D., AND PREISIG, J. C. Underwater acoustic communications. *Wireless Communications: Signal Processing Perspectives 8* (1998), 330–379.

[13] BRAUER, J. R. *Magnetic actuators and sensors*. John Wiley & Sons, 2006.

[14] BROWN, T. W., DIAKOS, T., AND BRIFFA, J. A. Evaluating the eavesdropping range of varying magnetic field strengths in nfc standards. In *Antennas and Propagation (EuCAP), 2013 7th European Conference on* (2013), IEEE, pp. 3525–3528.

[15] CASTELLUCCIA, C., AND MUTAF, P. Shake them up!: a movement-based pairing protocol for cpu-constrained devices. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services* (2005), ACM, pp. 51–64.

[16] CHO, Y.-J., YAND, T., AND KWON, D.-S. A new miniature smart actuator based on piezoelectric material and solenoid for mobile devices. In *The 5th International Conference on the Advanced Mechatronics, ICAM* (2010), pp. 615–620.

[17] DAVIS, A., RUBINSTEIN, M., WADHWA, N., MYSORE, G., DURAND, F., AND FREEMAN, W. T. The visual microphone: Passive recovery of sound from video. *ACM Transactions on Graphics (Proc. SIGGRAPH) 33*, 4 (2014), 79:1–79:10.

[18] DEVICES, A. Adxl345 datasheet. *USA: Analog Devices* (2010).

[19] DEY, S., ROY, N., XU, W., CHOUDHURY, R. R., AND NELAKUDITI, S. Accelprint: Imperfections of accelerometers make smartphones trackable. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)* (2014).

[20] DHANANJAY, A., SHARMA, A., PAIK, M., CHEN, J., KUPPUSAMY, T. K., LI, J., AND SUBRAMANIAN, L. Hermes: data transmission over unknown voice channels. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking* (2010), ACM, pp. 113–124.

[21] DIAKOS, T. P., BRIFFA, J. A., BROWN, T. W., AND WESEMEYER, S. Eavesdropping near-field contactless payments: a quantitative analysis. *The Journal of Engineering 1*, 1 (2013).

[22] EUN, H., LEE, H., AND OH, H. Conditional privacy preserving security protocol for nfc applications. *Consumer Electronics, IEEE Transactions on 59*, 1 (2013), 153–160.

[23] FEYGIN, D., KEEHNER, M., AND TENDICK, F. Haptic guidance: Experimental evaluation of a haptic training method for a perceptual motor skill. In *Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2002. HAPTICS 2002. Proceedings. 10th Symposium on* (2002), IEEE, pp. 40–47.

[24] FUNKE, H. D. Acoustic body bus medical device communication system, May 19 1992. US Patent 5,113,859.

[25] GIRI, M., AND SINGH, D. Theoretical analysis of user authentication systems. *International Journal of Innovative Research and Development 2*, 12 (2013).

[26] HALEVI, T., AND SAXENA, N. On pairing constrained wireless devices based on secrecy of auxiliary channels: The case of acoustic eavesdropping. In *Proceedings of the 17th ACM conference on Computer and communications security* (2010), ACM, pp. 97–108.

[27] HALPERIN, D., HEYDT-BENJAMIN, T. S., RANSFORD, B., CLARK, S. S., DEFEND, B., MORGAN, W., FU, K., KOHNO, T., AND MAISEL, W. H. Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on* (2008), IEEE, pp. 129–142.

[28] HASELSTEINER, E., AND BREITFUSS, K. Security in near field communication (nfc). In *Workshop on RFID security* (2006), pp. 12–14.

[29] HENDERSHOT, J. Causes and sources of audible noise in electrical motors. In *Proc. 22nd Incremental Motion Control Systems and Devices Symposium* (1993), pp. 259–270.

[30] HOLMQUIST, L. E., MATTERN, F., SCHIELE, B., ALAHUHTA, P., BEIGL, M., AND GELLERSEN, H.-W. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *Ubicomp 2001: Ubiquitous Computing* (2001), Springer, pp. 116–122.

[31] HUANG, K., DO, E.-L., AND STARNER, T. Pianotouch: A wearable haptic piano instruction system for passive learning of piano skills. In *Wearable Computers, 2008. ISWC 2008. 12th IEEE International Symposium on* (2008), IEEE, pp. 41–44.

[32] HWANG, I., CHO, J., AND OH, S. Privacy-aware communication for smartphones using vibration. In *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2012 IEEE 18th International Conference on* (2012), IEEE, pp. 447–452.

[33] KAAJAKARI, V. Practical mems: Design of microsystems, accelerometers, gyroscopes, rf mems, optical mems, and microfluidic systems. *Las Vegas, NV: Small Gear Publishing* (2009).

[34] KRISHNAN, R. *Electric motor drives: modeling, analysis, and control*. Prentice Hall, 2001.

[35] LESTER, J., HANNAFORD, B., AND BORRIELLO, G. are you with me?–using accelerometers to determine if two devices are carried by the same person. In *Pervasive computing*. Springer, 2004, pp. 33–50.

[36] MARQUARDT, P., VERMA, A., CARTER, H., AND TRAYNOR, P. (sp) iphone: decoding vibrations from nearby keyboards using mobile phone accelerometers. In *Proceedings of the 18th ACM conference on Computer and communications security* (2011), ACM, pp. 551–562.

[37] MAYRHOFER, R., AND GELLERSEN, H. Shake well before use: Intuitive and secure pairing of mobile devices. *Mobile Computing, IEEE Transactions on 8*, 6 (2009), 792–806.

[38] MILUZZO, E., VARSHAVSKY, A., BALAKRISHNAN, S., AND CHOUDHURY, R. R. Tapprints: your finger taps have fingerprints. In *Proceedings of the 10th international conference on Mobile systems, applications, and services* (2012), ACM, pp. 323–336.

[39] MORRIS, D., TAN, H. Z., BARBAGLI, F., CHANG, T., AND SALISBURY, K. Haptic feedback enhances force skill learning. In *WHC* (2007), vol. 7, pp. 21–26.

[40] MULLINER, C. Vulnerability analysis and attacks on nfc-enabled mobile phones. In *Availability, Reliability and Security, 2009. ARES'09. International Conference on* (2009), IEEE, pp. 695–700.

[41] NANDAKUMAR, R., CHINTALAPUDI, K. K., PADMANABHAN, V., AND VENKATESAN, R. Dhwani: secure peer-to-peer acoustic nfc. In *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM* (2013), ACM, pp. 63–74.

[42] NIWA, M., YANAGIDA, Y., NOMA, H., HOSAKA, K., AND KUME, Y. Vibrotactile apparent movement by dc motors and voice-coil tactors. In *Proceedings of the 14th International Conference on Artificial Reality and Telexistence (ICAT)* (2004), pp. 126–131.

[43] SAXENA, N., AND WATT, J. H. Authentication technologies for the blind or visually impaired. In *Proceedings of the USENIX Workshop on Hot Topics in Security (HotSec)* (2009), vol. 9, p. 130.

[44] SEMICONDUCTORS, P. The i2c-bus specification. *Philips Semiconductors 9397*, 750 (2000), 00954.

[45] STUDER, A., PASSARO, T., AND BAUER, L. Don't bump, shake on it: The exploitation of a popular accelerometer-based smart phone exchange and its secure replacement. In *Proceedings of the 27th Annual Computer Security Applications Conference* (2011), ACM, pp. 333–342.

[46] WIDNALL, S. Lecture l19-vibration, normal modes, natural frequencies, instability. *Dynamics* (2009).

[47] YONEZAWA, T., NAKAHARA, H., AND TOKUDA, H. Vibconnect: A device collaboration interface using vibration. In *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2011 IEEE 17th International Conference on* (2011), vol. 1, IEEE, pp. 121–125.