PEARL: **P**eer-to-peer **E**ffective **A**udio, **R**adio, and **L**ight Communications

Travis Peters, Ayan Azmat, Suleman Ashfaq, *Department of Computer Science at Dartmouth College*

***Abstract*—In recent years there has been an explosion of research in wireless communications, and our everyday lives and devices are moving more and more towards being untethered. This trend suggests that our devices will increasingly rely on wireless communication technologies to communicate; however, each technology offers its own unique set of trade-offs (e.g., low-power, short/long range, variable data rates) which make it an appealing method for communication for specific contexts, but not beyond those contexts. To further exacerbate the issue, some classes of devices may have certain constraints (e.g., limited power, security/privacy requirements, operating distance requirements) which limit the communicative capabilities of the device itself. What we believe is needed is a more capable communication framework and system that can (1) enable our devices to communicate with (nearby) devices by determining which communication channels are most effective given contextual information (e.g., about the environment, channel quality), and then (2) continuously adapt to use the communication channel(s) which yield the "best" and most reliable connection.**

In this paper, we propose a proof-of-concept implementation and evaluation of PEARL[1]: a first of its kind communication system that enables Peer-to-peer Effective Audio, Radio, and visible Light communication capabilities that all co-exist on a single platform. PEARL, at its core, is meant to present an abstraction of the communication technologies implemented in the underlying system, enabling application and system developers to focus more on *what* data is communicated, while delegating the responsibility of figuring out *how* to best communicate with other devices to the system itself.

Keywords*—*Audio, Amazing, Visible Light Communications, Awesome, Radio, Spectacular.

I. Introduction

Today, application developers are bound to rigid communication models such as Wi-Fi or Bluetooth. These communication technologies inherently have different trade-offs when using them and make more or less sense based on the context in which they are being used. For example, Bluetooth is good for personal area networks (PANs) and can actually work reasonably well with as much as 100 meters between the communicating devices [2]; Bluetooth is great for connecting smartphones and music players, or for connecting peripherals such as wireless mice/keyboards to your desktop or laptop computers, but the use of Bluetooth also has disadvantages. Namely, Bluetooth is not regarded as a power-friendly communication protocol for devices such as wearables and smartphones. In the case of Bluetooth applications, application developers for these platforms must decide between using Bluetooth LE, which is more conservative with its power usage but significantly reduces the overall data rate for communications, or using normal Bluetooth which offers higher data rates with greater power consumption, leading to a shorter battery life for devices and, overall, an annoying experience for users that must recharge their devices often. More concerning than issues with power are the security concerns that radio-based communications raise. Electromagnetic waves—used by radio communications—are known to pass through walls and travel significant distances before the signal loses enough power to no longer be discernable. This issue actually comes up frequently when considering security and privacy aspects of a system and is typically regarded as a significant vulnerability. Wi-Fi, in many regards, is worse than Bluetooth in situations like the one described above. Wi-Fi enables *much* greater data rates for communication, but is even more costly in terms of power and, to make matters worse, its signals actually carry further than Bluetooth signals, effectively working up to 300 meters.

On the other hand, there has been an explosion of research in recent years that demonstrate the viability of other, less conventional, communication technologies which offer their own set of advantages and disadvantages; namely ultrasonic and visible light communications. In light of this, we argue that it should be possible for devices to utilize the communication technology that makes the most sense in any given moment to effectively communicate with one another. In order to realize this vision, our devices must be able to (1) sense the *quality* of the various communication channels that are available (e.g., Wi-Fi, visible light, ultrasound), and (2) opportunistically switch between, or simultaneously utilize, the different communication technologies that are available in a way that is transparent to the user while being reliable and efficient in terms of power consumption and throughput.

In this first work, we initially focus on building a system which integrates a few diverse communication technologies (e.g., radio-based communications via Bluetooth, visible light communications via OpenVLC, and acoustic communications using NearBytes) and effectively demonstrates their co-existance on a single platform which can be easily leveraged by application developers to write applications that can use the communication channel (or combination of channels) that makes the most sense for their needs. It should be noted that this work addresses principles that can be utilized in order to enable more adaptive and capable communications in general, i.e., these ideas are not constrained to personal area networks or peer-to-peer communications. In fact, it should

[1] It should be noted that while PEARL targets three specific communication "channels" (i.e., audio, radio, and visible light), this is simply to demonstrate the effectiveness of our design and offer a proof-of-concept implementation—in the future, we will integrate other novel communication technologies into the PEARL architecture.

be possible to utilize other communication technologies such as NFC (very short range), RFID (medium range), Wi-Fi (medium/long range), cellular networks (long range), infrared light (IR), vibration-based communications, etc., that are not explicitly addressed in this work, in order to leverage the best channels for communication based on what makes the most sense at any given moment in time.

Contributions. In this paper, we propose the following contributions:

- A prototype of PEARL built atop a Nexus 5 phone running the Android operating system (v5: Lollipop); the *audio* and *radio* communication are implemented in software running directly on the phone while the *visible light communication* runs on a separate ”Beagle Bone” computer and is connected to the phone over USB.
- Two sample applications that utilize PEARL for (1) a local chat-like application, and (2) network throughput testing application.
- An evaluation of PEARL which provides insight into the advantages and limitations of our system.

II. Background: Acoustic, Radio, and Visible Light Communication

These days, communication between devices can be done in many different ways. Devices that communicate wirelessly most commonly do so via radios that use electromagnetic waves to carry the signal, as mentioned in section I. Radio-based communication is not, however, the only way devices can communicate. In this paper, we implement PEARL, a system that can effectively use all devices to communicate over audio, visible light, *and/or* radio. In the remainder of this section we review some of the fundamental concepts behind PEARL's current communication channels.

A. Visible Light Communication (VLC)

Visible Light Communication (VLC) is being researched today as a complementary technology to traditional Radio Frequency (RF) technologies. VLC is believed to be useful in diverse range of application domains such as indoor localization, the Internet of Things (IoT), and more [18] — there have even been recent efforts to make low-cost, open-source platforms for rapid prototyping of VLC systems such as the OpenVLC platform [6].

VLC utilizes light emitting sources such as LEDs to transmit data, and photodiodes—or similar hardware components—to convert light into electrical current, which can be decoded into data. A range of LEDs can be used for communication, though each offers their own tradeoffs and the LED that best meets the needs of an application should be used. For example, high-powered LEDs can be used to communicate with multiple nearby nodes and works at greater distances, but is less secure (since it easier to observe) and requires more power to drive the high-powered LED. Low-power LEDs, on the otherhand, can be used simply for visual feedback or for very close-range communication which is desirable in some security applications; low-power LEDs also require less power.

B. High-frequency Audio Communication

High frequency audio communication involves encoding data over sound waves using a modulation and error correction scheme. The prevalence of microphones and speaker in our society today makes it a practical and interesting medium for short range communications between devices. Researchers have deployed audio communication systems on off the shelf smart phones as shown in [19]. The computational power in current smart phones is enough to support this audio communication using a software acoustic modem.

C. Radio Frequency Communication

To allow Radio Frequency Communication in PEARL, we opted for Bluetooth. It is relatively more energy efficient and can easily facilitate communication in a $10\ meter$ radius. For these reasons and many more, Bluetooth Network Stack is being supported by most wireless devices hence making it highly integrable. Since our initial prototype of PEARL is implemented on Android, Bluetooth communication is implemented with the Android Bluetooth API. Our implementation of the API makes use of Bluetooth pairing and helps send data over the RFCOMM channel once the devices are connected through the Bluetooth socket.

III. PEARL Design

PEARL can be thought of as many things and indeed, the name is used to speak to various aspects of our work. In this section we will discuss PEARL as a software architecture. As a software architecture, PEARL aims to serve as an abstract communication layer, primarily between application developers and the underlying communication technologies available on the system. Figure 1 shows an overview of the PEARL software architecture.

Many past works have described useful and novel communication technologies [4], [11], [13], [16], [17], [18]; to the best of our knowledge, however, no other work has sought to unite all of the diverse communication methods and make them readily available to application developers on a single platform, through an intuitive interface. The role of the system, then, is to intelligently utilize communication channels that make the most sense for any given type of communication (e.g., based on contextual information). Alternatively, such an interface can allow application developers to explicitly express the type of communication channel they seek to use, and the system can then ensure that channel is used for specific communications. Our goal is to design a system the presents both of these features, however the system which we have built to-date primarily achieves the later objective.

A. PEARL Abstraction Layer

The PEARL Abstraction Layer essentially boils down to an interface which application developers can utilize to communicate with other devices in a consistent manner. Most programmers are familiar with *socket programming* for communication between devices, so we designed an interface which tries to allow applications to communicate with other devices via a

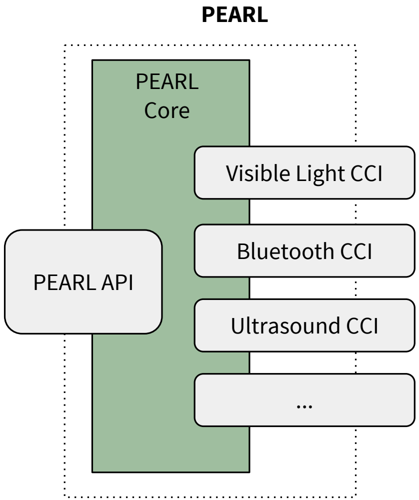


Fig. 1: Proposed software architecture of PEARL as a simple interface to the underlying communication interfaces. Notice that new interfaces can be added by implementing a new communication component interface (CCI).

similar API—basic socket routines are essentially all the make up the "PEARL API."

The more interesting interface is the Communication Component Inteface (CCI). In order for the PEARL system to support various communication channels, a system developer must implement a CCI which details how to manage the underlying hardware which controls the communication technology in accordance with the interface sketched in the code sample shown below.

```
1  // Methods for managing the CCI from the core.
2  setup()
3  enable()
4  getState()
5  setState()
6  disable()
7  teardown()
8  isEnabled()
9
10 // CCI connection-oriented methods.
11 connectRemoteDevice()
12 isRemoteDeviceConnected()
13 disconnectRemoteDevice()
14
15 // Methods for communicating over an enabled CCI.
16 open()
17 close()
18 send()
19 recv()
```

We've tried to design a very simple yet expressive interface for designing communication components that may be made available on a given system. Specifically, we require that each new communication component implement a set of methods for managing the the component (*lines 2-8*), methods for managing a connection with a remote device (*lines 11-13*), and methods for accessing the actual interface where data is written and received (*lines 16-19*). The PEARL CCI is described more in section III-B.

Our abstract communication interface is still being refined. One of the challenges in defining a truly consistent abstraction is that some communications are inherently meant to be connection-oriented, while others are connectionless.

Currently, some of our communication channels (e.g., visible light and Bluetooth) work in a connection-oriented fashion, whereby devices rely on "connecting" to a specific MAC or IP address before communicating; this is always the case for Bluetooth, but it is our understanding that VLC can easily be modified to run in a more broadcast-oriented manner. Our remaining communication channel (e.g., audio) does not try to connect to a particular device—any device that is close enough and is listening on its microphone is capable of decoding acoustic communications generated by our system.

B. PEARL Communication Component Interfaces (CCI)

PEARL Communication Component Interfaces (CCIs) implement the minimum interface required to describe a communication device (e.g., open/close connections, read/write to an interface that handles receiving/sending data). CCIs are essentially middleware plugins that implement the details for interacting with lower-level drivers or wrap system APIs (if any exist) in order to provide a consistent view of all of the available communication interfaces present on the system. The implementation of a CCI can also extend the minimum interface to provide more fine-grained control over the underlying implementation of the communication interface. This poses a challenge for the PEARL core which must be updated in order to utilize any enhanced functionality that goes beyond the minimum interface described above.

In our current work, and under the design described above, we have implemented three PEARL CCIs: AudioCCI, BluetoothCCI, and VLCCCI.

C. PEARL Core

The PEARL core is a part of our system that is left largely unimplemented currently. The core itself is responsible for processing contextual information and automatically switching between—or deciding to simultaneously utilize—communication interfaces present on the system to communicate with other devices.

At a minimum, PEARL's core must implement at least basic logic for obtaining access to channel contention and carrier sensing information at the application layer—though these are typically the responsibility of the MAC and PHY layers, the core needs this information to make informed decisions about channel selection. A proper implementation of the PEARL core is left to future work.

IV. Implementation

PEARL is currently a research-grade platform that is implemented primarily on two platforms: a Nexus 5 [5] phone and a BeagleBone Black (BBB) [1] board which is a development

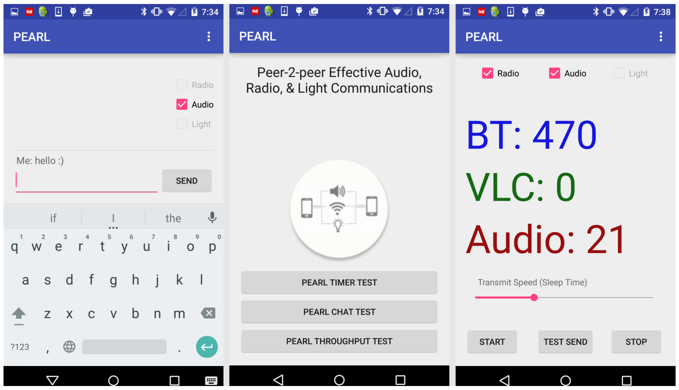


Fig. 2: PEARL Application. We implemented an Android library which provides access to Bluetooth, Audio, and VLC communication channels. Atop our library we built an Android application (middle) that demonstrates a user-friendly local chat application (left) and throughput testing application (right) which we used to produce a demo video of our system in action.

platform for quick prototyping that runs Linux OS. In order for VLC to work, we use the OpenVLC1.0 cape, a front-end transceiver that can be attached directly to the BBB, and run their driver that implements the underlying software for VLC network communications. Communication over the radio (i.e., Bluetooth) and acoustic channels is implemented directly on the Nexus 5 phone using built-in hardware components and pre-installed drivers for the Bluetooth radio, microphone, and speakers.

One of our goals with PEARL is to provide a single interface to various communication channels. We found that with reasonable effort we could implement interfaces to Bluetooth and high-frequency audio communications on the Nexus 5 running Android—we were not, however, able to implement VLC on the smartphone. Above we mentioned the use of the OpenVLC system which runs on an independent platform. In order for PEARL to provide an abstracted view of the underlying communication channels, we connected the beaglebone black board to our Nexus 5 via USB (using PyUSB [7]) and ran a custom lightweight server on the BBB that pipes data received over VLC up to the smartphone and pipes data generated on the smartphone down to the BBB which transmits the data over VLC.

To demo our proof-of-concept implementation of PEARL, we created a multi-functional application that helps us illustrate the fact that our system is working effectively. Figure 2 illustrates a few screenshots taken of the application that we've developed.

We chose to implement one example, "PEARL Chat" (left), that only utilizes communication technologies as they are presented by the PEARL interface described in section III. This application allows users to send text messages over a desired interface (or interfaces). Another application that we implemented by using the underlying PEARL APIs is our "Throughput Testing" application (right) which automatically constructs random packet payloads and sends them on the selected interface(s). This application was used in a demo video we produced and shows effective communication over each communication channel in isolation, as well as working together simultaneously. A delay in sending the packets was introduced so that we could show how the transmit rates could vary; the display increments a counter corresponding to the respective communication channel to indicate that another packet was received. Sliding the "seek bar" in our application allows the user to increase/decrease the delay.

V. Evaluation

Our current prototype is in it's very early stages and does not support smart switching between the different communication technologies that PEARL employs. To get a better understanding of the current technologies we are leveraging, and in order to movite future schemes, we carried out independent throughput testing on each of the mediums. Figure 3 shows how radio (bluetooth), audio and visible light perform in terms of throughput as the payload increases. All measurements were taken at an application level. Figure 4shows how radio (bluetooth), audio and visible light perform in terms of throughput as the payload increases. All measurements were taken at an application level.

A. Experimental Setup

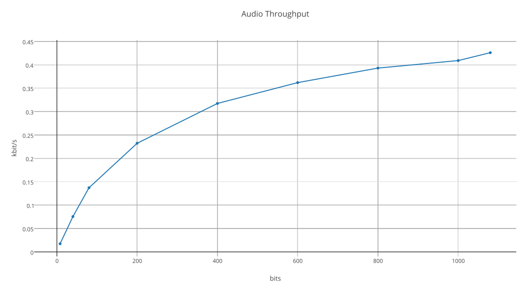
In our experimental set up we configured the devices to send an arbitrary payload over any of the interfaces we identified. The application at the receiving end would then echo back the transmission over the same medium, and the round trip time was recorded and used to calculate the throughput for a corresponding payload.

B. Independent Operation Modes

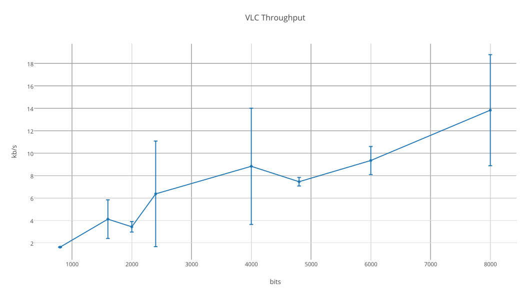
For each of the experiments, we placed the communication technologies in ideal conditions and measured the throughput vs the payload.

Bluetooth is the most reliable of all the channels as the orientation of the device has no impact on the overall performance of bluetooth. We have observed a maximum throughput of 1.08 Mbps for bluetooth. In the case of VLC and audio communication the successful transmission of data is highly sensitive to the device orientations and distance. They also have significantly lower overall throughput. In our use of VLC we have observed a maximum throughput of 13.82 Kbps and for audio transmissions throughput can go up to 0.43 Kbps.

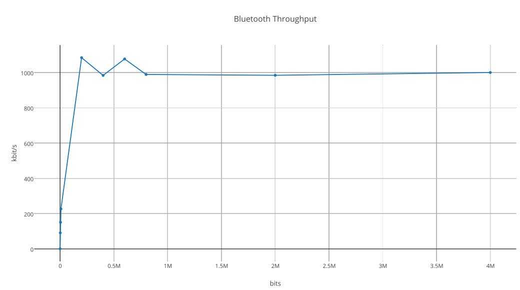
For VLC we also observed that repeating transmissions of the same payload had a large standard deviation and often varied significantly in throughput. We've identified the cause for this to be the Android to BeagleBone USB communication model that we have used. Since we poll in both the Android application and the BeagleBone to receive data this introduces a significant overhead. The transmissions take less time when they both poll in a closer time frame, and are longer when the polling does not coincide. We plan to address this by moving to an asynchronous event based model for communication over USB.



(a) Audio Throughput. We evaluate the data rate of Nearbytes High Frequency audio communication API after configuring it with PEARL and running it through our test cycle.



(b) VLC Throughput. Repeated transmissions of the same payload had significant affect on the data rate as observed in the graph.



(c) Bluetooth Throughput. The data rates for Bluetooth were fairly consistent as per our expectation. The data rate was roughly consistent around 1Mbps.

Fig. 3: Pictures of animals

C. Simultaneous Operation Modes

Our current prototype is able to simultaneously transmit and receive on radio, audio, and light. The communication is bi-directional for each channel, however none of the channles are full-duplex. Fig 4 shows the two smart phones in the set up communicating on radio, audio, and light. The count for each interface identifies the total number of messages that have been received on that particular interface.

Since the communication technologies we have selected are completely orthogonal and do not interfere with each other, the bottleneck in simultaneous operation comes down to the design of our controller. For simplicity of implementation, our current prototype uses a single thread to manage all the communication interfaces. This however is not ideal and we plan to have each communication technology run in its own thread in the future. The overhead involved in switching transmission from one technology to another comes down to the architecture of our implementation.

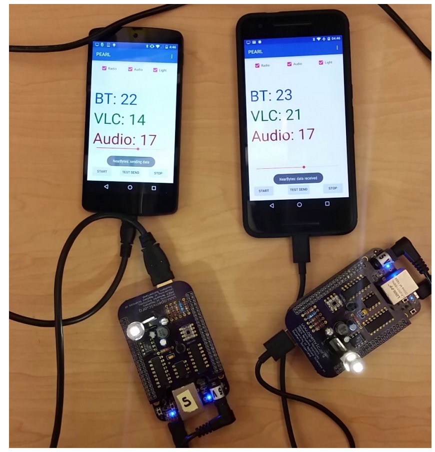


Fig. 4: A picture of the experimental setup demonstrating simultaneous communication over radio, audio, and visible light. Each smartphone is connected via USB to a BeagleBone for VLC. The counters on each device show the number for messages received on each interface.

VI. Discussion & Future Work

At this point we are very pleased with the progress of our work. Throughout our work this term it was apparent that a significant effort was needed to learn and integrate all of our proposed communication technologies under the control of a single system and design an intuitive interface for application developers to utilize the diverse communication channels that we currently provide (and plan to provide in the future). At this stage we believe we are now positioned well to continue to improve and optimize our implementation of PEARL, as well as demonstrate the benefits of a system such as PEARL in real-world application domains. The remainder of this section discusses some of the future work we envision.

Additional Communication Interfaces. In future work we want to explore integrating other novel and interesting methods for communication, including: the use of accelerometers/motors to communicate over vibrations, NFC (inductive and/or capacitive), Infrared Light, WiFi, cellular networks, and so forth. This work could be done in future version of our system which we are referring to as PEARL+(Vibrations/NFC/IR/WiFi/Cellular). This work supplements our vision for enabling devices to use the "best" communication method for a given context by enabling more diversity in the communication capabilities of the systems themselves. Ultimately, we believe that system designers should strive to make diverse communication resources more easily available to application developers in order to suit the needs of our increasingly connected and smart world.

Bootstrap Secure Connections. With access to multiple ”out of band” communication channels there are possibilities for interesting security applications. Consider a scenario where two devices want to establish secure communication on a channel by exchanging cryptographic keys. Using multiple out of band channels to exchange these keys raises the bar for an attacker to eaves drop on this exchange. The attacker would have to be sophisticated enough to eaves drop on all mediums simultaneously, and even if the attacked has that capability, communication technologies like light and audio require much closer proximity to be able to eaves drop, unlike radio which has far greater signal propagation.

Internet of Things (IoT) Communications. Another relevant domain that PEARL impacts is the *Internet of Things (IoT)*. In an IoT context we generally consider various objects in our everyday lives having computational capabilities. For example, consider the light bulbs in your house providing high-throughput communications for devices that don’t have radio-based communications, but do have cheap light sensors or photo diodes for decoding information that is encoded in visible light. In this example, PEARL could actually be implemented on a router-type device which can be connected to other communication-capable devices in the environment (e.g., light bulbs, sound systems) and can act as a “controller” for all communications whereby it is used to (1) communicate with devices that have limited communication capabilities (e.g., perhaps IoT devices in the environment aren’t connected to power and, for power-saving reasons, only communicate over lower-power mechanisms such as visible light communications or acoustic communications), and (2) determine the best way to communicate with nearby devices based on the types of devices that are present, the type of data they are requesting/sending, and so forth. The point being that we should enable our devices to make intelligent decisions for how best to communicate based on information they can obtain and context they can utilize in order to improve the overall quality, efficiency, and reliability of how our devices communicate.

Simultaneous Throughput & Dedicated Control/Data Channels. One advantage of PEARL is the fact that it enables simultaneous communication across multiple channels (e.g., radio, acoustic, and visible light). As a result, systems that utilize a system such as PEARL can achieve increased data throughput by leveraging multiple, non-interfering communication channels simultaneously. In essence, individual channels have data rate limitations, but when effectively combined, diverse communication channels can be used to increase the overall amount of data that can be communicated between two or more devices.

One utilization of this idea that we envision is to use higher-throughput channels as data channels so that we can offload control-related communications to lower-throughput channels[2]. It is well understood that today, devices that communicate generally communicate two types of data: control information and actual data. *Control channels* are setup for communicating information about setting up, maintaining, and tearing down a connection, and *data channels* are setup between devices for communicating the actual data. These communications are generally considered to be separate channels that exist within the communication bandwidth between devices. The fact that both of these types of communications must happen in the same bandwidth suggests that the overall availability of that bandwidth for transmitting and receiving the actual data is actually smaller than the reported data rates for technologies like Bluetooth and Wi-Fi, for example.

We propose to split those logical channels (e.g., control and data) into separate physical channels. By leveraging the fact that control information is generally exchanged less frequently and doesn’t require large packets, we can deduce that a lower-bandwidth channel would be sufficient for exchanging this kind of information. Generally speaking, radio-based communications such as Bluetooth and Wi-Fi can achieve higher data rates than other less-conventional communication methods such as ultrasound and visible light. With PEARL, lower data rate channels, such as acoustic and visible light, could be used to transmit and receive control information, while the higher data rate channels, such as radio (e.g., Bluetooth, Wi-Fi), could be used to communicate the data itself. This approach also has the added benefit that it provides a nice security property in that the channels that are usually considered to be more secure, such as ultrasound and visible light, are being used to communicate the channel control information.

VII. Related Work

Novel communication technologies have been heavily researched in recent years, but wireless communication technologies are far from new. To the best of our knowledge, however, there has been no other work that has explored the co-existance of various communication interfaces for local networking; specifically, no previous work has implemented or evaluated a system that successfully integrates multiple communication channels—such as visible light, acoustic, and radio-based communications—under a single controller in order to present an abstract communication layer that application and system developers can use to effectively leverage diverse networking technologies for local communications. To this end, we feel our work with PEARL is opening a new research direction that combines ideas from a wide range of topics—here we discuss the most salient prior work in each area.

Sensing & Opportunistic Channel Switching. Though we haven’t yet been able to implement sophisticated sensing capabilities for our current prototype, this is part of the ultimate

[2]Splitting data at a more fine-grained level and sending it over multiple interfaces would be possible by using MultiPath TCP [3]. This was an appealing idea in our early work, but research that we found (and discuss in the Related Works section) may suggest that MultiPath TCP is not ideal for local communications since the added overhead yields marginal throughput gains compared to the increased CPU workload and battery consumption that MultiPath TCP communications incur.

vision for PEARL. The general idea is to use feedback on channel quality as context which informs our system on how to adaptively select the best method for communication.

One of the most applicable and closely-related topics is research on methods for opportunistically switching between Wi-Fi and LTE networks to achieve the best throughput possible [10], [12], [15]. Deng et al. actually found that MultiPath TCP [3] yields worse throughput for "short flows" as compared to normal TCP; this is certainly relevant since PEARL devices have only been evaluated under local networking conditions such as in P2P applications—MultiPath TCP may still be useful for security reasons though (*see next section*). Higgens et al. [12] present *Intentional Networking* whereby mobile nodes can express intent behind network messages and the system attempts to identify the best available interface for communication. This overall vision is desireable for PEARL, however, PEARL is not constrained to Wi-Fi and cellular network interfaces for communications. In [15], authors evaluate the advantages and disadvantages for using MultiPath TCP to communicate on multiple interfaces simultaneously. Their findings suggest that throughput only increases marginally, while CPU activity and overall power consumption are significantly increased. These are relevant concerns for PEARL that we intend to evaluate in future works.

We would also like to do more fine-grained sensing of channel and environmental conditions so that PEARL can be smart enough to factor this information into the decisions it makes to adaptively select the correct communication interface(s) for communication. Along with channel quality, we intend to utilize other context that is made available by the platform. Specifically, a primary target platform for a PEARL-like system is sensing-capable device such as a smartphone or modern smartwatch/fitness tracker that may have accelerometers, gyroscopes, light sensors, microphones, temperatures sensors, and so forth. If this is the case then PEARL could, as [20] discuss, leverage sensor inputs from these resources to infer other context such as movement and lighting conditions, as well as other factors like noise—all of which are factors that can impact the reliability of communication between local devices attempting to communication on various channels.

Secure Communications. Previous work has been done to demonstrate the seriousness—and ease of carrying out attacks against—vulnerabilities in certain types of communications such as radio-based communications [8], for example. Most issues come down to an adversary being able to eavesdrop sensitive communications from a distance, allowing their presence to go unnoticed while they steal information. In response to this vulnerability (and others), researchers have explored novel communication methods that have nice security properties such as encoding data in signals that cannot travel through walls or even in signals that are extremely difficult (or impossible) to observe unless a receiver is very close in distance (i.e., a few centimeters).

It should be understood that new communication technologies are not as perfect at securing communications as some might have you think. For example, in [9] the authors demonstrated that there are multiple ways to eavesdrop on visible light communications without being physically present in the same room as the devices communicating over VLC, and in [14] the authors discuss multiple viable attacks on ultrasonic communications. While this methods aren't perfect solutions, they certainly "raise the bar" for attackers seeking to eavesdrop secret information. We believe PEARL can make it even more challenging for an adversary to observe secret information by utilizing mechanisms such as MultiPath TCP [3] to send portions of sensitive messages over multiple interfaces.

VIII. Conclusion

In this paper, we discussed the design, implementation, and evaluation of our proof-of-concept PEARL communication system, a first of its kind communication system that enables Peer-to-peer Effective Audio, Radio, and visible Light communication capabilities that all co-exist on a single platform. PEARL integrates diverse communication technologies under a single system and defines a socket-like interface which applications can use to communicate, allowing them to focus on the data they wish to communicate, and delegating the task of figuring out how best to communicate with other devices to the system.

PEARL, though in the early stages, shows great promise towards enabling easier, more reliable, and more secure communications to applications and systems in the future.

Acknowledgments

The authors thank Xia Zhou, our awesome professor from Dartmouth College, for her help in refining the vision of our work and helping us to better understand proper techniques for evaluating and demonstrating the effectiveness of our prototype system. The authors also thank Qing Wang and Domenico Giustiniano from IMDEA Networks Institute for providing early access to their OpenVLC hardware and software, as well as providing help with setting up and working with OpenVLC. Last, but certainly not least, the authors thank Jefferson Pita and the team at NearBytes for providing us access to their SDK along with their support.

References

[1] BeagleBone Black. http://beagleboard.org/black.

[2] Bluetooth. https://en.wikipedia.org/wiki/Bluetooth.

[3] MultiPath TCP. http://multipath-tcp.org/.

[4] NearBytes. http://nearbytes.com/.

[5] Nexus 5X (Google). https://www.google.com/nexus/5x/.

[6] OpenVLC. http://openvlc.org/.

[7] PyUSB. https://github.com/walac/pyusb.

[8] WiFi jamming attacks more simple and cheaper than ever.

[9] Classen, J., Chen, J., Steinmetzer, D., Hollick, M., and Knightly, E. The Spy Next Door: Eavesdropping on High Throughput Visible Light Communications. In *Proceedings of the 2Nd International Workshop on Visible Light Communications Systems* (New York, NY, USA, 2015), VLCS '15, ACM, pp. 9–14.

[10] Deng, S., Netravali, R., Sivaraman, A., and Balakrishnan, H. WiFi, LTE, or Both?: Measuring Multi-Homed Wireless Internet Performance. In *Proceedings of the 2014 Conference on Internet Measurement Conference* (New York, NY, USA, 2014), IMC '14, ACM, pp. 181–194.

[11] GROSSE-PUPPENDAHL, T., HERBER, S., WIMMER, R., ENGLERT, F., BECK, S., VON WILMSDORFF, J., WICHERT, R., AND KUIJPER, A. Capacitive Near-field Communication for Ubiquitous Interaction and Perception. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (New York, NY, USA, 2014), UbiComp '14, ACM, pp. 231–242.

[12] HIGGINS, B. D., REDA, A., ALPEROVICH, T., FLINN, J., GIULI, T. J., NOBLE, B., AND WATSON, D. Intentional Networking: Opportunistic Exploitation of Mobile Network Diversity. In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking* (New York, NY, USA, 2010), MobiCom '10, ACM, pp. 73–84.

[13] KAHN, J., AND BARRY, J. Wireless infrared communications. *Proceedings of the IEEE 85*, 2 (Feb 1997), 265–298.

[14] MAYRHOFER, R., AND GELLERSEN, H. On the Security of Ultrasound as Out-of-band Channel. In *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International* (March 2007), pp. 1–6.

[15] NIKA, A., ZHU, Y., DING, N., JINDAL, A., HU, Y. C., ZHOU, X., ZHAO, B. Y., AND ZHENG, H. Energy and Performance of Smartphone Radio Bundling in Outdoor Environments. In *Proceedings of the 24th International Conference on World Wide Web* (Republic and Canton of Geneva, Switzerland, 2015), WWW '15, International World Wide Web Conferences Steering Committee, pp. 809–819.

[16] ROY, N., GOWDA, M., AND CHOUDHURY, R. R. Ripple: Communicating through Physical Vibration. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)* (Oakland, CA, May 2015), USENIX Association, pp. 265–278.

[17] SANTAGATI, G. E., AND MELODIA, T. U-Wear: Software-Defined Ultrasonic Networking for Wearable Devices. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services* (New York, NY, USA, 2015), MobiSys '15, ACM, pp. 241–256.

[18] WANG, Q., GIUSTINIANO, D., AND GNAWALI, O. Low-Cost, Flexible and Open Platform for Visible Light Communication Networks. In *Proceedings of the 2Nd International Workshop on Hot Topics in Wireless* (New York, NY, USA, 2015), HotWireless '15, ACM, pp. 31–35.

[19] ZHANG, B., ZHAN, Q., CHEN, S., LI, M., REN, K., WANG, C., AND MA, D. ssrPriWhisper : Enabling Keyless Secure Acoustic Communication for Smartphones. *Internet of Things Journal, IEEE 1*, 1 (Feb 2014), 33–45.

[20] ZHOU, X., AND CAMPBELL, A. T. Visible Light Networking and Sensing. In *Proceedings of the 1st ACM Workshop on Hot Topics in Wireless* (New York, NY, USA, 2014), HotWireless '14, ACM, pp. 55–60.