# Data over Sound Technology

Device-to-device communications & pairing without wireless radio networks

Djalma L. S. Silva

# Disruptive Analysis

This presentation is based on document that has been  prepared by independent research firm Disruptive Analysis, and commissioned by Chirp Ltd.

The document is available at https://chirp.io/docs/data-over-sound-whitepaper.pdf

# Agenda

- Introduction
- Why Data-over-Sound useful?
- Key technology enablers of DoS
- Software Development Kits
- Chirp.io SDK

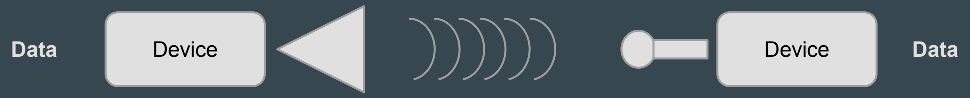
# Introduction

The use of sound waves, or audio-based electronic systems to send data (as well as speech) is not new.

- Humans have used drums and other percussion instruments to send messages, as well as create music, for thousands of years.
- Other animals use audio waves as well – such as bats' ultrasound or dolphins' sonar for echo-location – as well as more-normal communications between species' members.
- For decades, telegraph and telecom networks have used sound technology.
    - such as touch-tone (DTMF, dual-tone multi-frequency signalling) controls on handsets
    - modems' encoding of data onto phone lines, the latter invented by IBM in the 1940s.
    - others have transmitted data in acoustic form, through the air – or even through water or solid objects.

# Introduction

- Data-over-sound technology encodes information into audio format:
  - either audibly as "bleeps and tones"
  - inaudibly above human hearing range
  - or "hidden" as imperceptible modifications of existing speech or music.
- It is received through a microphone, and decoded on another device.




Depending on the implementation, closeness of the devices and environment, it can operate at data-rates of around 50-100bit/s, usually within a room.

# Why Data-over-Sound useful?

- It is reasonable to question why a relatively slow, and seemingly-primitive form of communication like DoS is needed today at all.
- We are constantly surrounded by multimegabit (and soon gigabit) per-second wireless transmissions of data using WiFi, 4G and various other technologies.

Yet despite these innovations, there are still various reasons why they cannot cover all eventualities and use-cases.

Sound-based data transmission has a range of important characteristics and strengths that cannot easily (or reliably) be replicated with other mechanisms.

# Why Data-over-Sound useful?

- Quiet RF Environments
- Usability and ubiquity vs. alternative wireless options
- IoT and legacy support
- Physical/environmental considerations

# Quiet RF Environments

Perhaps the easiest and most-understandable reason to use sound is that some locations cannot use radio.

For reasons of safety or interference with other systems, various locations are "quiet".

There are contexts where there are strict prohibitions or practical constraints:

- In hospitals, especially intensive-care units or near X-ray machines or other scanners
- Industrial facilities where welding or other equipment can blast out huge amounts of RF interference
- Mining, military and oil/gas installations which store and use explosives on-site, and which often prohibit RF use
- Sensitive facilities such as military or nuclear sites, where radios are strictly limited
- Scientific laboratories with delicate instruments, or indeed conducting research into RF phenomena themselves
- Buildings where wire meshes or metalwork can act as RF screens or filters

While some applications can obviously use wired or fibre connections, this is not possible where endpoints are mobile.

# Usability and ubiquity vs. alternative wireless options

Many applications, especially on smartphones, need "proximity" connectivity.
There are multiple ways for short-range connections to be set up to transfer data:

- Bluetooth (including the BLE low-energy variant)
- NFC contactless chips
- QR-codes displayed on screens
- WiFi in P2P mode
- Proprietary methods like Apple AirPlay or AirDrop, and so on.

While these approaches all have their uses, there is often either "friction" in setting them up initially, or a problem with accessibility for all relevant users.

Using sound as a primary (or secondary) approach to data-transfer may work better.

- More devices have microphones and speakers
- There is less dependency on operating systems or OS version.
- Devices can be distances of a few metres or more apart.
- Legacy products can be supported more easily.

One further advantage with DoS compared with NFC is that it is full-duplex: on smartphones and most other devices, the microphone and speaker can send and receive data simultaneously.

# IoT and legacy support

A critical benefit of DoS is that its use can be extended more-easily to a variety of existing devices and systems than other signalling/data-transmission methods. Far more objects have microphone and/or speaker support already such as

- Door entry-phones
- music systems
- industrial machines
- point-of-sale terminals
- walkie-talkies
- street-furniture
- Vehicles
- old desk-phones

While they may not have sophisticated computing capabilities or audio-processing, they may still be “hackable” by developers, to bridge them with other devices that can offer those functions.

# Physical/environmental considerations

Audible and near-audible sound waves have a number of important characteristics, that can both make DoS especially useful, or make it impossible.

Among the key considerations are:

- Limited transmission through building materials and walls. This means that sound is (mostly) confined to one room in a building, assuming doors and windows are shut and the volume levels are limited.
- Slow speed of sound waves. This is one of the factors limiting the data rate of the technology.
- In some circumstances, ultrasound signals may have a longer latency than audible ones.
  - Meaning that they need to be repeated in a loop several times, for reliable reception by the microphone.
  - Also, ultrasound data rates are lower, as they can only use about 2kHz (between about 18-20kHz), rather than the effective 15kHz available in audible range.
- Sound transmission can be affected by local furnishings and decorations in a room.

# Key technology enablers of DoS

- Watermarking vs. Modulating
- Microphones & speakers
- Integration with voice / audio systems
- Acoustics and encoding methods
- The role of standards

# Key technology enablers of DoS

## Watermarking vs. Modulating

There are two basic ways to encode data into sound waves:

- **Modulating:**
    - This method takes an existing source of data, and translates it into tones or pulses, based on a variety of possible coding schemes.
    - As it does not require any other ongoing audio track, it tends to be more useful for ad-hoc and real-time communications between devices.
- **Watermarking (also steganography or audio-hiding):**
    - This involves changing an existing audio signal such as music, by putting additional (inaudible) data into the stream.
    - This is often used for purposes such as copyright protection, or putting extra meta-data into a particular source (eg artist or track-title).
    - Because a host audio stream is required, this type of approach is often used for one-to-many or broadcast purposes, rather than ad-hoc device-to-device communication.

# Key technology enablers of DoS

## Microphones & speakers

Most adults have a hearing range from around 20Hz (deep bass) up to around 16-17kHz, although children can sometimes hear up to 21-22kHz, and cats and dogs can hear up to 40kHz ultrasonics (hence inaudible dog-whistles).

- Most standard audio components in common personal devices work well up to around 20kHz, and some have usable ranges considerably higher.
- This means that use of quasi-ultrasound in the 18-19kHz range is feasible for many use-cases, notably those involving smartphones.

Not all legacy or larger-scale audio systems can handle those near-ultrasound frequency ranges.

- Some speakers such as those in cheaper phones, venue publicaddress (PA) systems, older TVs, door-entry systems and others products have more limited frequency ranges.
- Other devices may have microphones or speakers inside an enclosure of some sort (for example, a soft toy) which limits their audio capabilities.

# Key technology enablers of DoS

## Integration with voice / audio systems

In some circumstances, it is possible to combine short-range data-over-sound technologies with other communications or audio systems.

This can extend the range locally, provide broadcast capability, or allow hybrid services The main platforms to consider are:

- Public phone networks (also called PSTN)
- Business phone and conferencing systems
- Broadcast TV and radio
- Web audio/video/game streaming
- Voice and video chat/conferencing features built-into websites and mobile apps

There are numerous subtleties to consider:

- Many existing audio and telecom systems are intended just for a subset of audible frequencies
- Many boxes in the middle of telecom networks (eg transcoders) strip out ultrasound frequencies
- Streamed video / audio services such as Netflix and Spotify reduce the quality and amounts of data, based on the available network capacity and speed – these too, are unsuitable for ultrasound.

# Key technology enablers of DoS

Acoustic data communications requires specific ways of setting up links and encoding information (protocols).

In common with other forms of communications, there is a “stack” of different protocols that cover

- signalling
- identity
- error-correction
- security/encryption
- and all the other “machinery” needed to get the overall system to function well.

## Acoustics and encoding methods

There are numerous trade-offs for speed vs. accuracy, and the impact of these balances varies depending

- whether sound is audible/inaudible
- physical environment
- the quality of the microphones and speakers used.

# Key technology enablers of DoS

## The role of standards

At the moment (2017), most DoS systems are highly proprietary, either customised for particular products, or provided through a specific vendor's SDKs.

While that makes sense in the current phase of market evolution, Disruptive Analysis believes that standardisation will be needed, if DoS is to be truly "democratised" and used routinely in a similar fashion to Bluetooth or QR-codes for device-to-device communication.

As well as technical standards allowing interoperability of different equipment, there may also need to be implementation guidelines or industry codes-of-conduct, for example around privacy and consent.

# Key technology enablers of DoS

## The role of standards

SoniTalk - Towards a first open standard for data-over-sound

During this past year we have done extensive research and development on ultrasonic communication. One of the main outcome is the proposal of a new protocol for communication via sound (and in particular via near-ultrasound) that is simple enough to be implemented on devices with limited computational resources, such as Internet-of-Things (IoT) devices.

We submitted a first version to the IETF that you can see here:
**https://tools.ietf.org/html/draft-zeppelzauer-data-over-sound-00**

We are now in contact with the IETF editors to get feedback and propose this specification as an Experimental RFC document:

# Software Development Kits

Proprietary

- Chirp
  https://developers.chirp.io/
- LISNR
  https://lisnr.com/resources/developers/
- SONARAX
  https://www.sonarax.com/#SDK

Opensource

- Audio Network ( *written in TypeScript* )
  https://github.com/robertrypula/AudioNetwork
- Quiet Modem Project
  https://github.com/quiet/quiet
- Quiet.js
  https://quiet.github.io/quiet-js/

# chirp.io SDK

Getting Started

https://developers.chirp.io/docs

Using Chirp

https://developers.chirp.io/docs/using-chirp

# chirp.io

*WebAssembly*

## Sending data

Import the ChirpSDK into the JavaScript file where you would like to use Chirp, and instantiate with your `app_key`.

```
const { Chirp } = ChirpConnectSDK;

Chirp({ key: 'CHIRP_APP_KEY' }).then(sdk => {
  sdk.send('hello')
}).catch(console.error)
```

# chirp.io

*WebAssembly*

## Receiving data

```
const { Chirp, toAscii } = ChirpConnectSDK;

Chirp({
  key: 'CHIRP_APP_KEY',
  onReceived: data => {
    if (data.length > 0) {
      console.log(toAscii(data))
    } else {
      console.error('Decode failed')
    }
  }
}).then(sdk => {
  sdk.send('hello');
}).catch(console.error)
```

https://messenger.chirp.io

chirp.io

[Alexa and Beat Bugs]

[Let there be Light]

https://vimeo.com/channels/chirp

# Example use-cases and applications for DoS

Chirp

https://chirp.io/solutions/

https://vimeo.com/channels/chirp

LINSR

https://lisnr.com/solutions/payments/

https://lisnr.com/solutions/retail/

https://lisnr.com/solutions/mobility/

SONARAX

https://www.sonarax.com/#Solutions