

# Biblioteca para comunicación directa entre dispositivos basada en tecnologías P2P

Library for direct communication between devices based on P2P technologies

Emilio Cobos Álvarez

Grado en Ingeniería Informática

Universidad de Salamanca

Septiembre de 2025

# Introducción

Todos tenemos una radio portátil.


# Introducción

Todos tenemos una radio portátil.


**¡Pero no la usamos!**

# Ventajas I: Privacidad y resiliencia

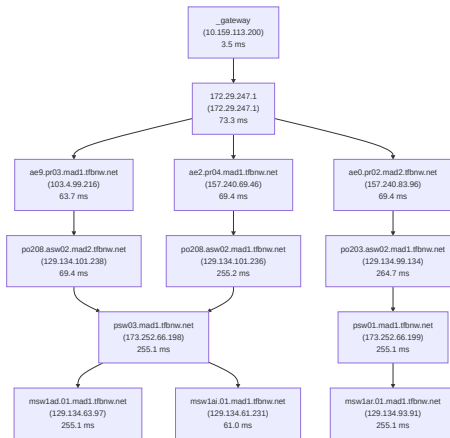
## Privacidad

- Tus datos no tienen que pasar por servidores centralizados
- Immune a censura gubernamental y judicial como puede pasar en Irán, China... O en España si hay Liga 

## No necesita infraestructura

- Usable en ubicaciones remotas
- En caso de emergencia
- O durante apagones 

# Ventajas II: Eficiencia



Traceroute a web.whatsapp.com

# La tecnología existe!

- Bluetooth/LE

# La tecnología existe!

- Bluetooth/LE
- WiFi Aware o Neighbor Awareness Networking (NAN)

# La tecnología existe!

- Bluetooth/LE
- WiFi Aware o Neighbor Awareness Networking (NAN)
- WiFi Direct



# La tecnología existe!

- Bluetooth/LE
- WiFi Aware o Neighbor Awareness Networking (NAN)
- WiFi Direct
- Apple Wireless Device Link (AWDL)

# La tecnología existe!

- Bluetooth/LE
- WiFi Aware o Neighbor Awareness Networking (NAN)
- WiFi Direct
- Apple Wireless Device Link (AWDL)
- Sparklink (Huawei)

# Usos actuales

- Apps de *contact tracing* durante la pandemia usaban Bluetooth LE

# Usos actuales

- Apps de *contact tracing* durante la pandemia usaban Bluetooth LE
- *FireChat* en las protestas de Hong Kong de 2014

# Usos actuales

- Apps de *contact tracing* durante la pandemia usaban Bluetooth LE
- *FireChat* en las protestas de Hong Kong de 2014
- *Nearby Share* en Android usa WiFi Direct

# Usos actuales

- Apps de *contact tracing* durante la pandemia usaban Bluetooth LE
- *FireChat* en las protestas de Hong Kong de 2014
- *Nearby Share* en Android usa WiFi Direct
- *AirDrop* usa AWDL

# Usos actuales

- Apps de *contact tracing* durante la pandemia usaban Bluetooth LE
- *FireChat* en las protestas de Hong Kong de 2014
- *Nearby Share* en Android usa WiFi Direct
- *AirDrop* usa AWDL
- MANETS de uso militar

# Usos actuales

- Apps de *contact tracing* durante la pandemia usaban Bluetooth LE
- *FireChat* en las protestas de Hong Kong de 2014
- *Nearby Share* en Android usa WiFi Direct
- *AirDrop* usa AWDL
- MANETS de uso militar
- Meshtastic



# Hipótesis

Se hipotetiza que la baja adopción de este tipo de es por:

- Dificultad de desarrollo

# Hipótesis

Se hipotetiza que la baja adopción de este tipo de es por:

- Dificultad de desarrollo
- Soporte para hardware variable

# Hipótesis

Se hipotetiza que la baja adopción de este tipo de es por:

- Dificultad de desarrollo
- Soporte para hardware variable
- Poca interoperabilidad entre plataformas

# Hipótesis

Se hipotetiza que la baja adopción de este tipo de es por:

- Dificultad de desarrollo
- Soporte para hardware variable
- Poca interoperabilidad entre plataformas
- Intereses económicos

# Propuesta

Se propone crear una biblioteca para facilitar el desarrollo de aplicaciones P2P que:

- Abstraiga la capa de transporte

# Propuesta

Se propone crear una biblioteca para facilitar el desarrollo de aplicaciones P2P que:

- Abstraiga la capa de transporte
- Sea multi-plataforma

# Propuesta

Se propone crear una biblioteca para facilitar el desarrollo de aplicaciones P2P que:

- Abstraiga la capa de transporte
- Sea multi-plataforma
- Funcione en dispositivos de consumo

# Propuesta

Se propone crear una biblioteca para facilitar el desarrollo de aplicaciones P2P que:

- Abstraiga la capa de transporte
- Sea multi-plataforma
- Funcione en dispositivos de consumo
- Soporte autenticación



# Propuesta

Se propone crear una biblioteca para facilitar el desarrollo de aplicaciones P2P que:

- Abstraiga la capa de transporte
- Sea multi-plataforma
- Funcione en dispositivos de consumo
- Soporte autenticación
- Cifre mensajes independiente de la capa de transporte

# Propuesta

Se propone crear una biblioteca para facilitar el desarrollo de aplicaciones P2P que:

- Abstraiga la capa de transporte
- Sea multi-plataforma
- Funcione en dispositivos de consumo
- Soporte autenticación
- Cifre mensajes independiente de la capa de transporte

# Propuesta

Se propone crear una biblioteca para facilitar el desarrollo de aplicaciones P2P que:

- Abstraiga la capa de transporte
- Sea multi-plataforma
- Funcione en dispositivos de consumo
- Soporte autenticación
- Cifre mensajes independiente de la capa de transporte

Y una aplicación demostrativa.

# Herramientas

- Lenguajes: Rust, Java, Kotlin, C

# Herramientas

- Lenguajes: Rust, Java, Kotlin, C
- Control de versiones: Git

# Herramientas

- Lenguajes: Rust, Java, Kotlin, C
- Control de versiones: Git
- UI: GTK, Jetpack Compose

# Herramientas

- Lenguajes: Rust, Java, Kotlin, C
- Control de versiones: Git
- UI: GTK, Jetpack Compose
- Depuración: rr

# Herramientas

- Lenguajes: Rust, Java, Kotlin, C
- Control de versiones: Git
- UI: GTK, Jetpack Compose
- Depuración: rr
- Documentación: pandoc,  $\text{\LaTeX}$ , Mermaid, rustdoc



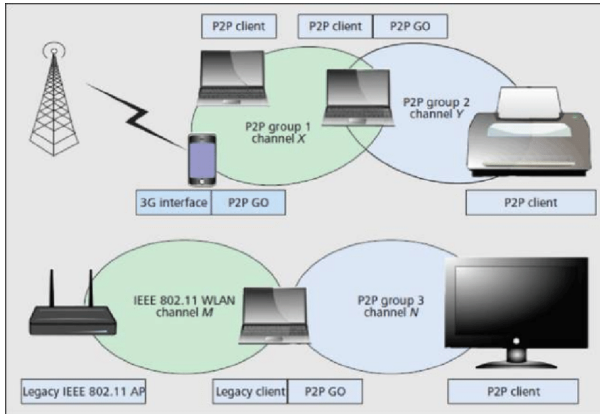
# Metodología

Se ha elegido *Scrum* con sprints semanales como metodología de desarrollo ágil (con algunas licencias para acomodar las restricciones existentes).

# Interfaz principal

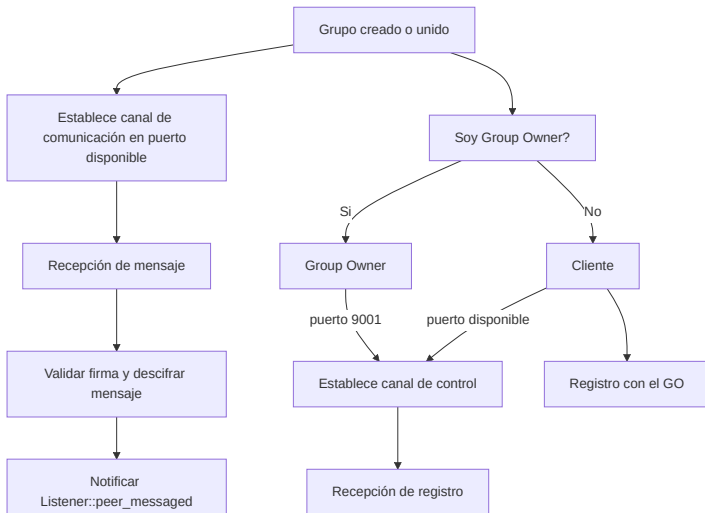
```
pub trait P2PSession: ... {  
    fn new(..., listener) -> Result<Self>;  
    fn discover_peers(&self) -> Result;  
    fn connect_to_peer(&self, id: PeerId) -> Result;  
    fn message_peer(&self, id: PeerId, msg: &[u8]) -> Result;  
}
```

# WiFi Direct



Source: Camps-Mur *et al.* (2013)

# Conexión (simplificada)



- Identidad y firma de mensajes usando clave Ed25519

# Seguridad

- Identidad y firma de mensajes usando clave Ed25519
- Generación de secreto usando ECDH X25519

# Seguridad

- Identidad y firma de mensajes usando clave Ed25519
- Generación de secreto usando ECDH X25519
- Cifrado usando AES-256-GCM

# Obstáculos I: Asignación de direcciones

- Android usa IPv4 + DHCP por defecto



# Obstáculos I: Asignación de direcciones

- Android usa IPv4 + DHCP por defecto
- IPv6 Neighbor discovery (ICMPv6): Requiere CAP\_NET\_RAW en Linux, imposible en Android

# Obstáculos I: Asignación de direcciones

- Android usa IPv4 + DHCP por defecto
- IPv6 Neighbor discovery (ICMPv6): Requiere CAP\_NET\_RAW en Linux, imposible en Android
- IPv6 Link Local Address

# Obstáculos I: Asignación de direcciones

- Android usa IPv4 + DHCP por defecto
- IPv6 Neighbor discovery (ICMPv6): Requiere CAP\_NET\_RAW en Linux, imposible en Android
- IPv6 Link Local Address
  - Depende de la configuración del dhcp del GO

# Obstáculos I: Asignación de direcciones

- Android usa IPv4 + DHCP por defecto
- IPv6 Neighbor discovery (ICMPv6): Requiere CAP\_NET\_RAW en Linux, imposible en Android
- IPv6 Link Local Address
  - Depende de la configuración del dhcp del GO
  - Android no expone la dirección MAC de la interfaz

# Obstáculos I: Asignación de direcciones

- Android usa IPv4 + DHCP por defecto
- IPv6 Neighbor discovery (ICMPv6): Requiere CAP\_NET\_RAW en Linux, imposible en Android
- IPv6 Link Local Address
  - Depende de la configuración del dhcp del GO
  - Android no expone la dirección MAC de la interfaz
  - Linux no expone la MAC del GO

# Obstáculos II: Linux

- Permisos necesarios para interactuar con `wpa_supplicant`

## Obstáculos II: Linux


- Permisos necesarios para interactuar con `wpa_supplicant`
- Interacción entre `NetworkManager` y `wpa_supplicant` (issue reportada)

# Obstáculos II: Linux



- Permisos necesarios para interactuar con `wpa_supplicant`
- Interacción entre `NetworkManager` y `wpa_supplicant` (issue reportada)
- API de D-Bus de `wpa_supplicant` subóptima:





## Obstáculos II: Linux

- Permisos necesarios para interactuar con `wpa_supplicant`
- Interacción entre `NetworkManager` y `wpa_supplicant` (issue reportada)
- API de D-Bus de `wpa_supplicant` subóptima:
  - Gestión de errores pobre (fix enviado y aceptado )



## Obstáculos II: Linux

- Permisos necesarios para interactuar con `wpa_supplicant`
- Interacción entre `NetworkManager` y `wpa_supplicant` (issue reportada)
- API de D-Bus de `wpa_supplicant` subóptima:
  - Gestión de errores pobre (fix enviado y aceptado )
  - No soporta auto-join (fix enviado y aceptado )




# Obstáculos II: Linux

- Permisos necesarios para interactuar con `wpa_supplicant`
- Interacción entre `NetworkManager` y `wpa_supplicant` (issue reportada)
- API de D-Bus de `wpa_supplicant` subóptima:
  - Gestión de errores pobre (fix enviado y aceptado )
  - No soporta auto-join (fix enviado y aceptado )
  - No expone la MAC del dispositivo propio (fix enviado, pendiente)





## Obstáculos II: Linux

- Permisos necesarios para interactuar con `wpa_supplicant`
- Interacción entre `NetworkManager` y `wpa_supplicant` (issue reportada)
- API de D-Bus de `wpa_supplicant` subóptima:
  - Gestión de errores pobre (fix enviado y aceptado )
  - No soporta auto-join (fix enviado y aceptado )
  - No expone la MAC del dispositivo propio (fix enviado, pendiente)
  - No expone la MAC de la interfaz del GO (fix enviado, pendiente)

# Obstáculos II: Linux

- Permisos necesarios para interactuar con `wpa_supplicant`
- Interacción entre `NetworkManager` y `wpa_supplicant` (issue reportada)
- API de D-Bus de `wpa_supplicant` subóptima:
  - Gestión de errores pobre (fix enviado y aceptado )
  - No soporta auto-join (fix enviado y aceptado )
  - No expone la MAC del dispositivo propio (fix enviado, pendiente)
  - No expone la MAC de la interfaz del GO (fix enviado, pendiente)
- Configuración de `dhcp` (issue reportada y arreglada por upstream )

## Obstáculos II: Linux

- Permisos necesarios para interactuar con `wpa_supplicant`
- Interacción entre `NetworkManager` y `wpa_supplicant` (issue reportada)
- API de D-Bus de `wpa_supplicant` subóptima:
  - Gestión de errores pobre (fix enviado y aceptado )
  - No soporta auto-join (fix enviado y aceptado )
  - No expone la MAC del dispositivo propio (fix enviado, pendiente)
  - No expone la MAC de la interfaz del GO (fix enviado, pendiente)
- Configuración de `dhcp` (issue reportada y arreglada por upstream )
- Mejoras de rendimiento en `zbus` aceptadas 

# Obstáculos III: Android

- Excesivos permisos necesarios

# Obstáculos III: Android

- Excesivos permisos necesarios
- Interacción de usuario requerida



# Obstáculos III: Android

- Excesivos permisos necesarios
- Interacción de usuario requerida
- Soporte sólo para un grupo físico

# Obstáculos III: Android

- Excesivos permisos necesarios
- Interacción de usuario requerida
- Soporte sólo para un grupo físico
- Servicios de ubicación activados necesario

## Obstáculos III: Android

- Excesivos permisos necesarios
- Interacción de usuario requerida
- Soporte sólo para un grupo físico
- Servicios de ubicación activados necesario
- No expone MAC del dispositivo propio

## Obstáculos III: Android

- Excesivos permisos necesarios
- Interacción de usuario requerida
- Soporte sólo para un grupo físico
- Servicios de ubicación activados necesario
- No expone MAC del dispositivo propio
- No expone MAC de la interfaz propia ni del GO

## Obstáculos III: Android

- Excesivos permisos necesarios
- Interacción de usuario requerida
- Soporte sólo para un grupo físico
- Servicios de ubicación activados necesario
- No expone MAC del dispositivo propio
- No expone MAC de la interfaz propia ni del GO
- Grupos previos almacenados global e indefinidamente

# Obstáculos IV: Pruebas

- Imposible testear en un emulador Android

# Obstáculos IV: Pruebas

- Imposible testear en un emulador Android
- Testear en Linux requiere:

# Obstáculos IV: Pruebas

- Imposible testear en un emulador Android
- Testear en Linux requiere:
  - Desconectar `NetworkManager`



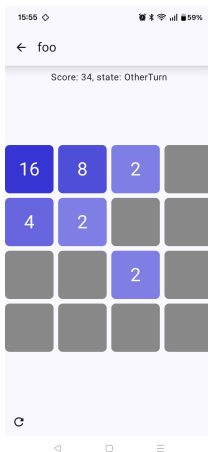
# Obstáculos IV: Pruebas

- Imposible testear en un emulador Android
- Testear en Linux requiere:
  - Desconectar `NetworkManager`
  - Desconectar `wpa_supplicant`

# Obstáculos IV: Pruebas

- Imposible testear en un emulador Android
- Testear en Linux requiere:
  - Desconectar NetworkManager
  - Desconectar wpa\_supplicant
  - Una instancia de wpa\_supplicant, dbus-daemon, y mac80211\_hwsim por cada nodo a controlar

# Demostración: Juego multijugador off-line



# Conclusiones I

- Creo que hay hueco / demanda para una biblioteca como la propuesta, si bien requiere mucho más trabajo de implementación (Windows, Bluetooth, WiFi Aware...)

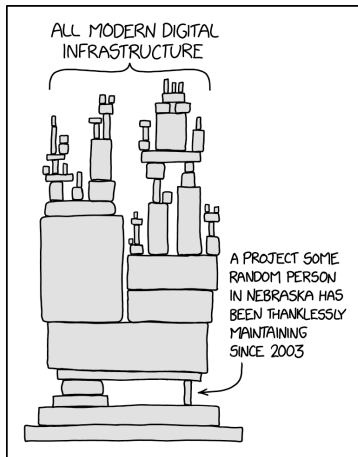
# Conclusiones I

- Creo que hay hueco / demanda para una biblioteca como la propuesta, si bien requiere mucho más trabajo de implementación (Windows, Bluetooth, WiFi Aware...)
- Hay mucho por hacer a nivel de plataforma e interoperabilidad también

# Conclusiones I

- Creo que hay hueco / demanda para una biblioteca como la propuesta, si bien requiere mucho más trabajo de implementación (Windows, Bluetooth, WiFi Aware...)
- Hay mucho por hacer a nivel de plataforma e interoperabilidad también
- La presión regulatoria de la DMA puede mejorar la situación

## Conclusiones II



XKCD #2347: Dependency