

Prácticas Fundamentos de Redes

Chromecast

Francisco Javier Morales Piqueras
Rubén Morales Pérez

11 de diciembre de 2016

Índice

1. Introducción	2
2. Hardware	2
3. Software	4
3.1. Protocolos que usa Google Cast	7
3.1.1. DIAL	7
3.1.2. mDNS (multicast Domain Name System)	8
3.1.3. Modo invitado vía ultrasonidos	9
4. Otros protocolos multimedia	10
4.1. Miracast	10
5. Código de ejemplo de una aplicación	11
Referencias	13

1. Introducción

Google Chromecast es un dispositivo de reproducción multimedia fabricado por Google y comercializado a partir de Julio de 2013. Reproduce contenido multimedia conectado a una televisión o monitor vía HDMI haciendo streaming mediante Wi-Fi.

Para hacer streaming utiliza el software Google Cast, un protocolo propietario de Google que permite controlar la reproducción multimedia de un receptor desde un dispositivo local. Google Cast dispone de librerías para las últimas versiones de Android y iOS, así como para Chrome OS y aplicaciones de Google Chrome.

Chromecast permite reproducir contenido almacenado en un dispositivo conectado a una red local o en un servidor externo. El control de la reproducción se realiza en ambos casos desde uno o varios dispositivos locales compatibles con la tecnología Google Cast. Cuando no hay contenido en streaming reproduce un contenido personalizable de fondo, puede incluir fotos personales, de satélite, noticias, etc. Por defecto muestra imágenes aleatorias seleccionadas por Google.

Para reproducción exclusiva de sonido existe un Chromecast Audio pensado para conectarse a altavoces. Hay otras tres versiones del Chromecast: la primera generación, la segunda y el Chromecast Ultra. La principal mejora de la segunda generación fue la compatibilidad con redes Wi-Fi a 5GHz. El Chromecast Ultra incluye como novedad la posibilidad de reproducir vídeo 4K.

Su principal competidor es el servicio AirPlay de Apple, que permite streaming inalámbrico entre dispositivos iPhone, iPad o Mac para audio, vídeo, fotos, etc.

2. Hardware



Figura 1: Chromecast de primera generación

El Chromecast de primera generación incluye un decodificador de VP8 y H.264 para formatos de compresión de vídeo, 512 MB de Micron DDR3L RAM y 2 GB o 4 GB de memoria flash según la fuente (Google no ha publicado las especificaciones del dispositivo). Se conoce que tiene un chip a 700 MHz single core. Tiene una salida HDMI, una entrada micro USB para la alimentación, un LED que

indica el estado del dispositivo y un botón de reset. Sus estándares de conexión son Wi-Fi 802.11 b/g/n (solo 802.11n a 2,4 GHz).

El de segunda generación tiene 512 MB de Samsung DDR3L RAM y 256 MB de memoria flash. El dispositivo tiene un cable flexible en cuyo extremo se encuentra la salida HDMI, usa un procesador dual core ARM Cortex-A7 con 1.3 GHz de frecuencia y tiene tres antenas adaptativas para mejorar la conexión con el router. Sus estándares de conexión son 802.11 b/g/n/ac Wi-Fi (2,4 GHz/5 GHz).



Figura 2: Chromecast de segunda generación

El Chromecast Audio es externamente igual que el de segunda generación, pero tiene una salida Minijack de 3,5mm en lugar del HDMI y un chip de procesamiento de audio.



Figura 3: Chromecast Audio

Acerca del hardware del Chromecast Ultra se conoce poco de manera oficial. La mayor diferencia es, aparte de mayor potencia de procesamiento para reproducir vídeo 4K, una entrada Ethernet en el adaptador de corriente para conectarlo a Internet como alternativa al Wi-Fi.



Figura 4: Chromecast Ultra

3. Software

El Google Chromecast no es más que un dispositivo compatible con el protocolo propietario Google Cast que actúa como receptor. Este protocolo fue lanzado en julio de 2013 en exclusividad para YouTube, Google Play Music, Google Play Movies & TV y Netflix, usando como receptor el Chromecast de primera generación.

En febrero de 2014 pusieron el SDK a disposición de todos los desarrolladores para usarlo en sus propias aplicaciones. En mayo de 2015 había más de 20.000 aplicaciones de terceros compatibles con esta tecnología según reconoce Google.

Para iniciar la reproducción de un contenido pulsamos el botón de *cast*, que aparecerá automáticamente si Google Cast está integrado con la aplicación. En ese momento aparecen los dispositivos Chromecast conectados a la red local y se elige aquel donde se quiere emitir el contenido. Si el receptor es una televisión cuyo HDMI dispone de Consumer Electronics Control (CEC) se encenderá automáticamente.

Google Cast usa dos modos de funcionamiento. El primero consiste en usar el dispositivo desde el que solicitamos el streaming (emisor) para controlar la reproducción: pausar un vídeo, subir el volumen, crear o modificar una cola de reproducción, etc. El dispositivo receptor (por ejemplo un Chromecast) se encarga de descargarlo y comunicarse con el servidor de contenido, liberando al dispositivo emisor de esta tarea. Esto garantiza una carga de trabajo muy baja para el emisor y le permite estar bloqueado o en otra aplicación mientras la reproducción está teniendo lugar. Ejemplos de aplicaciones que usan este modo serían Netflix o YouTube. Las aplicaciones emisoras de este tipo deben ser compatibles con Android 4.1, iOS 7.0 o versiones superiores si son aplicaciones móviles y con Windows 7, macOS 10.7, Chrome OS 28 o versiones superiores si son aplicaciones Chrome (con la extensión Cast instalada).

El otro modo está pensado para enviar contenido del dispositivo emisor, como cuando hacemos mirroring o usamos la televisión como segunda pantalla (por ejemplo en juegos que hacen uso de la Game

Manager API). El mirroring está soportado de manera nativa en el Chromecast (es decir, no hace falta descargar una aplicación), para pestañas de Chrome, el escritorio de un ordenador con Chrome o dispositivos con Android 4.4 o superior. La calidad del streaming en este caso varía ampliamente según la potencia de procesamiento del emisor. En caso de hacerse desde un smartphone, la calidad de las imágenes normalmente se deteriora debido al escalado.

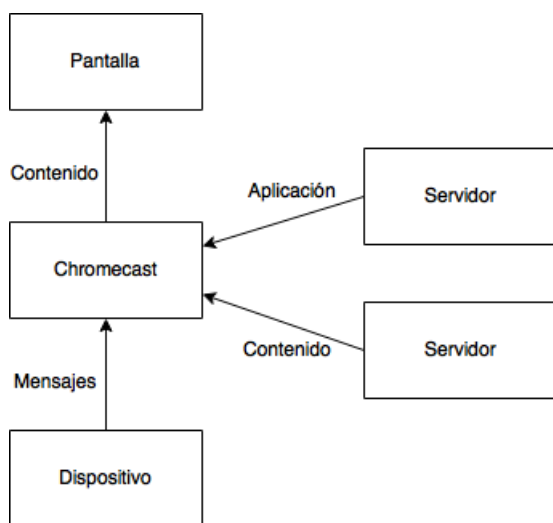


Figura 5: Primer modo

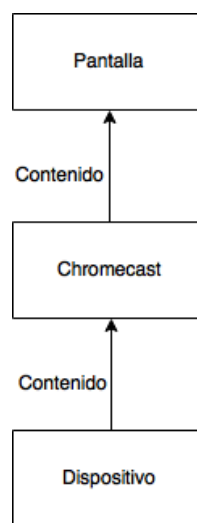


Figura 6: Segundo modo

Como hemos adelantado, la API de Google Cast implementa el paradigma del productor-consumidor. Para implementar el protocolo hacen falta dos aplicaciones:

- La aplicación emisora
- La aplicación receptora

La aplicación emisora se encarga de proveer al usuario la capacidad de controlar la reproducción y elegir el dispositivo donde se emite el contenido. Esta aplicación crea un canal seguro con la aplicación receptora para el intercambio de mensajes.

La aplicación receptora es una web app ejecutándose en una versión adaptada del navegador Chrome con una interfaz gráfica en CSS. Esta puede tener una complejidad muy variable, pudiendo ir desde limitarse a reproducir contenido HTML5 hasta soportar protocolos de streaming como MPEG-DASH, HTTP Live Streaming o el Microsoft Smooth Streaming Protocol [1].

Al ser una web app, el código de la misma debe estar alojado en un servidor, ya que el Chromecast no almacena aplicaciones. Una consecuencia de esto último es que, aunque el contenido final esté alojado en un dispositivo de la red local, seguirá siendo necesaria una conexión a internet para cargar la web app.

Sería posible omitir la conexión a internet si el servidor local (por ejemplo, un móvil u ordenador) alojase la aplicación web, de esta forma solo se necesitaría una red local. En cualquier caso esto solo tiene sentido para aplicaciones que desarrolle el propio usuario.

Los formatos multimedia a los que Google Cast da soporte son los siguientes:

- Imágenes en formato BMP, GIF, JPEG, PNG y WEBP, con un límite de 1280x720 píxeles de resolución.
- Los codecs de audio HE-AAC, LC-AAC, MP3, Vorbis, WAV (LPCM) y FLAC. AC-3 (Dolby Digital) y E-AC-3 (EC-3, Dolby Digital Plus) están disponibles para passthrough de audio.
- Los codecs de vídeo H.264 High Profile Level 4.1 (decodificación hasta 720/60 o 1080/30) y VP8.

En el CES de 2015, Google anunció una expansión de Google Cast centrada en la reproducción de audio. La idea era que los fabricantes de altavoces integraran la tecnología Google Cast sin necesidad de depender de un Chromecast. Está disponible en varios modelos de LG y Sony.

En mayo de 2015, Google lanzó nuevas APIs dirigidas a utilizar el televisor como segunda pantalla que muestre un contenido distinto del de la aplicación emisora. Esto, junto con las Game Manager APIs, permite, por ejemplo, usar varios dispositivos como mandos en una partida de un videojuego y una pantalla común que proyecte la partida. Uno de esos dispositivos sería el que controlara el estado de la partida y se sincronizarían entre ellos intercambiando mensajes con un Chromecast u otro dispositivo receptor.



Figura 7: Ilustración de Google para explicar el potencial de su Game Manager API



Figura 8: Ilustración de Google como ejemplo de uso de una pantalla externa a través de Google Cast



Figura 9: Ejemplo de uso de una pantalla externa para videojuegos

3.1. Protocolos que usa Google Cast

El funcionamiento de Google Cast se descompone en dos pasos: primero, la detección de dispositivos receptores desde la aplicación emisora y, segundo, el intercambio de mensajes entre receptor y emisor. Las primeras versiones del Chromecast usaban el protocolo DIAL que englobaba ambos pasos. En las últimas versiones, para el primer paso se utiliza el protocolo mDNS cuando los dispositivos se encuentran en la misma red local y un protocolo especial basado en ultrasonidos para el modo invitado. Para el segundo paso, Google no ha hecho público qué protocolo se usa. Solo ha trascendido [que no usa WebSockets y que se establece un canal binario seguro entre ambos extremos](#).

Hasta diciembre de 2014, el dispositivo emisor y receptor debían estar conectados a la misma red Wi-Fi, en las versiones posteriores no es necesario al haber añadido un modo invitado. En este modo, el receptor emite ultrasonidos a través de los altavoces y el emisor es capaz de localizarlo usando el micrófono. También puede usarse un PIN de cuatro dígitos que aparece en pantalla. El modo invitado está disponible para todos los Chromecast con un dispositivo Android como emisor y para los Chromecast a partir de la segunda generación con dispositivos iOS como emisor.

3.1.1. DIAL

DIAL (DIsccovery And Launch) es el antiguo protocolo de comunicación que usaba Chromecast, desarrollado con Netflix y Youtube. Se basa en Universal Plug and Play (UPnP), Simple Service Discovery Protocol (SSDP) y protocolos HTTP. SSDP es un protocolo que sirve para la búsqueda de dispositivos UPnP en una red. Utiliza UDP en unicast o multicast en el puerto 1900 para anunciar los servicios de un dispositivo. Si el receptor ofrece el servicio deseado devuelve el mensaje '200 OK' con HTTP.

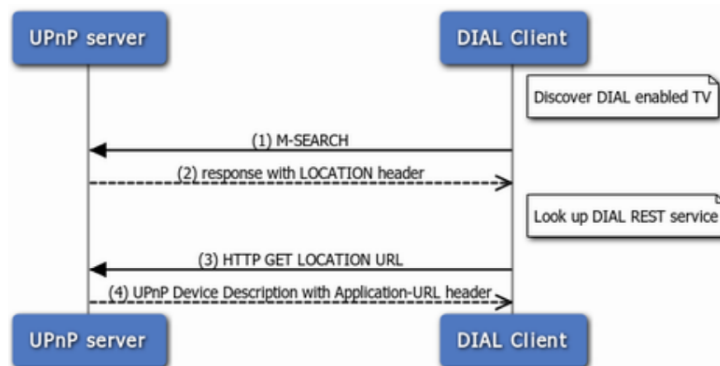


Figura 10: Funcionamiento general DIAL

DIAL permite a dispositivos en segundo plano, como tablets o móviles, enviar contenido a dispositivos en primer plano, por ejemplo televisiones. Los dispositivos en segundo plano almacenarán el cliente, los de primer plano ejecutarán el servidor. Para evitar conflictos con los nombres de aplicaciones permitidas DIAL tiene un registro con las aplicaciones que soporta. En el caso de que la aplicación deseada no esté en esa lista puede que tenga un prefijo que esté registrado, para así ofrecer mayor flexibilidad.

El protocolo DIAL tiene dos componentes, DIAL Service Discovery y DIAL REST Service. El primero busca el dispositivo en primer plano dentro de una red local y permite establecer conexión. El segundo permite al cliente mandar contenido, hacer consultas como subir o bajar volumen, etc. al servidor. El servidor puede procesar mensajes de hasta 4KB.

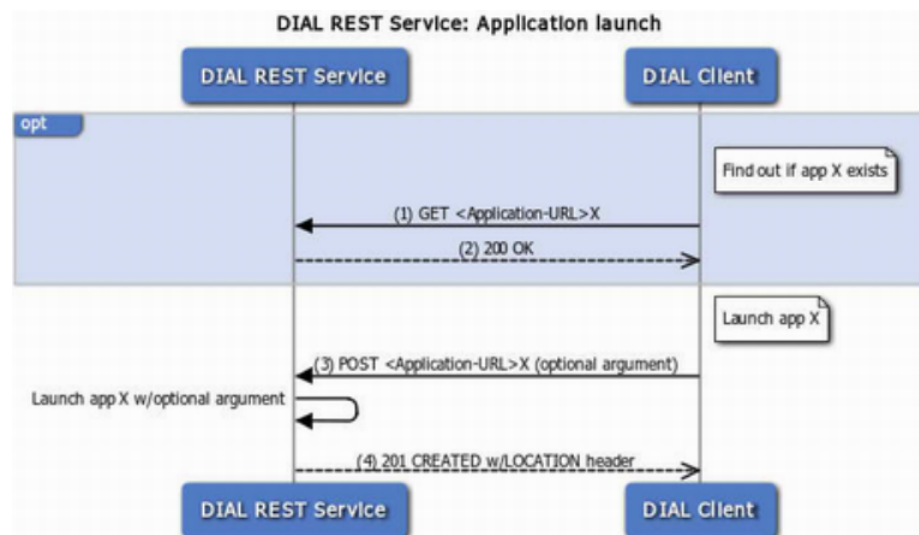


Figura 11: Funcionamiento DIAL REST service

Las respuestas específicas y otros detalles como manejo de excepciones se pueden encontrar en [2]

3.1.2. mDNS (multicast Domain Name System)

Multicast Domain Name System (mDNS, [RFC 6763](#)) es el protocolo usado para encontrar los dispositivos a los que conectarnos.

mDNS transforma nombres de dominio en direcciones IP dentro de su registro de direcciones IPv4 o IPv6 usando el puerto 5353. Está pensado para redes locales sin tener que incluir un servidor DNS. Tiene las mismas interfaces de programación, formatos y restricciones semánticas que DNS, pero permite designar una porción del espacio de nombres de DNS a nuestro gusto, sin tener que pagar anualmente.

Usa UDP en multicast y sus ventajas son:

- Necesita poca configuración para activarse
- Funciona cuando no hay infraestructura
- Soporta fallos en la infraestructura

En el primer caso envía a todos los dispositivos una solicitud para identificarse.

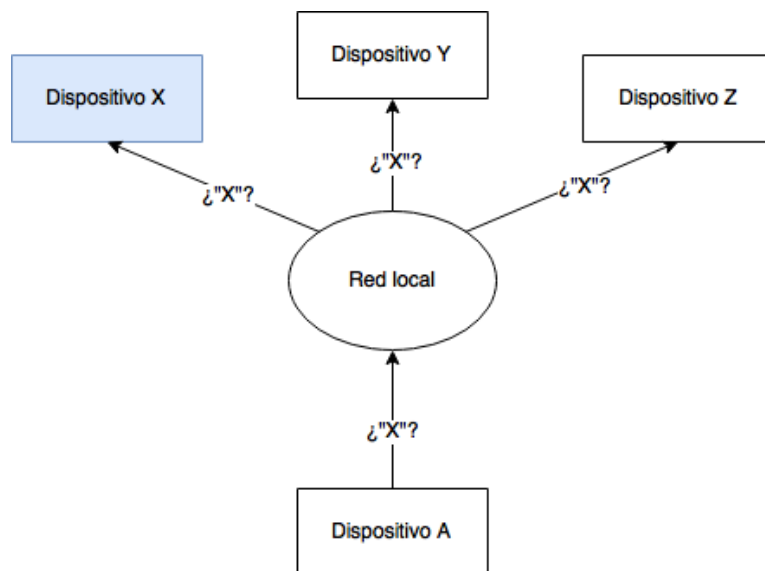


Figura 12: Primer paso del protocolo mDNS

Como respuesta se devuelve la dirección IP con una variable TTL (Time To Live) para saber cuánto tiempo durará vigente la IP enviada.

Si un dispositivo quiere rechazar la petición manda una IP con TTL cero.

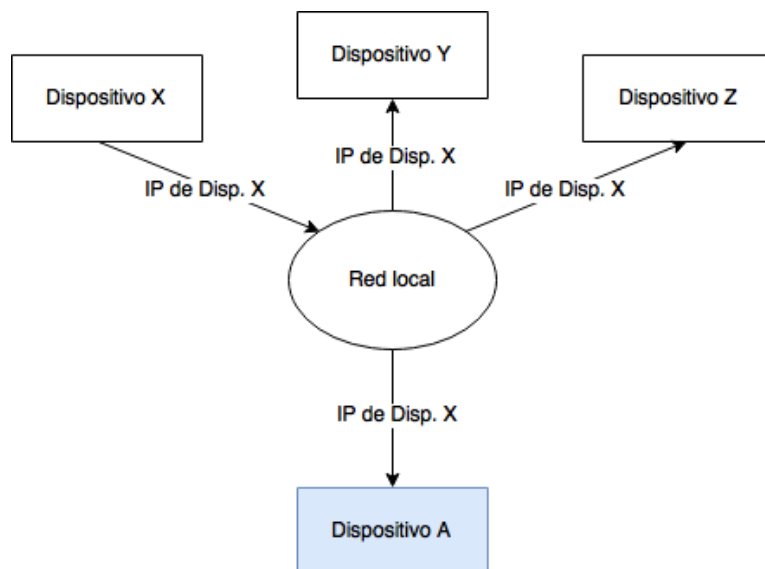


Figura 13: Segundo paso del protocolo mDNS

3.1.3. Modo invitado vía ultrasonidos

Si se usa un dispositivo cercano a un Chromecast, este puede detectarlo y sincronizarse vía ultrasonidos. El Chromecast emite sonidos a muy alta frecuencia (inaudibles para un humano) a través del altavoz de la televisión. Estos sonidos son la codificación de un mensaje determinado. A continuación, el emisor escucha este sondo a través del micrófono y lo transforma a un mensaje interpretable

por la aplicación. Este procedimiento está limitado por especificaciones del dispositivo: a menudo, el micrófono no es capaz de detectar sonidos a frecuencias tan altas, bien por limitaciones de hardware o de firmware. Para entender mejor el funcionamiento del intercambio de información a través de ultrasonidos, se explica con más detalle en [el blog del ingeniero de Google Boris Smus](#).

4. Otros protocolos multimedia

4.1. Miracast

Miracast es un protocolo multimedia para hacer streaming a un monitor desde un dispositivo local. Para que nuestra smartTV sea capaz de usarlo, necesita soportar Wi-Fi Direct. Wi-Fi Direct es una norma que permite la conexión entre dos dispositivos Wi-Fi sin necesidad de un intermediario. Los dispositivos que envían y reciben información tienen que estar certificados para Miracast, pero existe un plug para dispositivos no certificados. Miracast está disponible de manera nativa para dispositivos con versiones de Android 4.2 y Android 6.0.

La conexión está creada vía Wi-Fi Protected Setup (WPS), mecanismos para facilitar la configuración de una red WLAN con seguridad WPA2. WPS contempla cuatro configuraciones para el intercambio de credenciales: PIN (Personal Identification Number), PBC (Push Button Configuration), NFC (Near Field Communications) y USB (Universal Serial Bus). La configuración PIN no es recomendable por su debilidad ante ataques de fuerza bruta.



Para la capa de internet usa IPv4; para la de transporte, TCP/UDP, y para la de aplicación, RTSP y RTP, que se encargan de controlar el streaming.

A partir de Android 6.0, Google ha dejado de dar soporte nativo a Miracast en favor de su propio Google Cast. Con Miracast el dispositivo receptor es dependiente de que el dispositivo Android emisor se mantenga activo [3]: si se bloquea también bloqueará la reproducción en el receptor. Esto implica una mayor carga de trabajo y consumo de batería respecto a Google Cast, que solo se encarga de enviar señales para el control de la reproducción.

Existe una alternativa de código abierto a Miracast llamada MiracleCast. El nombre viene por la dificultad de crear una red Wifi-P2P estable (basado en wpa_supplicant).

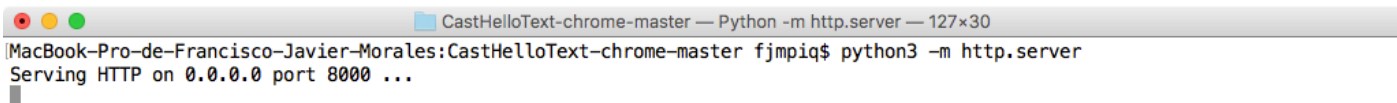
El núcleo de MiracleCast es un demonio llamado miracled [4], que controla links locales, las peticiones de conexión, se encarga de la codificación del protocolo y el parsing. Su línea de comandos puede ser usada para controlar el demonio, crear nuevas conexiones, modificar parámetros, etc. Soporta un modo interactivo que muestra las peticiones de conexión y permite al usuario aceptarlas o no.

El código fuente se puede encontrar en [GitHub](#).

5. Código de ejemplo de una aplicación

A continuación exponemos brevemente el código de una aplicación que se encuentra en el repositorio en GitHub de aplicaciones plantilla para Google Cast. Esta aplicación sirve para enviar texto desde una pestaña de Google Chrome y mostrarlo en una pantalla conectada a un Chromecast.

Esta aplicación consta de dos partes: la del emisor (Google Chrome) y la del receptor (Chromecast). Ambas son web apps en HTML que cargan un código JavaScript. Para ejecutarla debemos alojarla en un servidor.

A screenshot of a macOS terminal window. The title bar shows the window name 'CastHelloText-chrome-master' and the command 'Python -m http.server' with window dimensions '127x30'. The terminal text shows the prompt 'MacBook-Pro-de-Francisco-Javier-Morales:CastHelloText-chrome-master fjmpiq\$' followed by the command 'python3 -m http.server'. The output is 'Serving HTTP on 0.0.0.0 port 8000 ...' followed by a cursor.

```
MacBook-Pro-de-Francisco-Javier-Morales:CastHelloText-chrome-master fjmpiq$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 ...
```

Figura 14: Creando un servidor en el puerto 8000

Desde Chrome, accedemos a la aplicación emisora ([chromehellotext.html](#)).



Figura 15: Listado del directorio del servidor que acabamos de crear



Figura 16: Aplicación emisora

A continuación, desde la extensión de Google Cast, seleccionamos enviar el contenido de la aplicación a nuestro Chromecast.

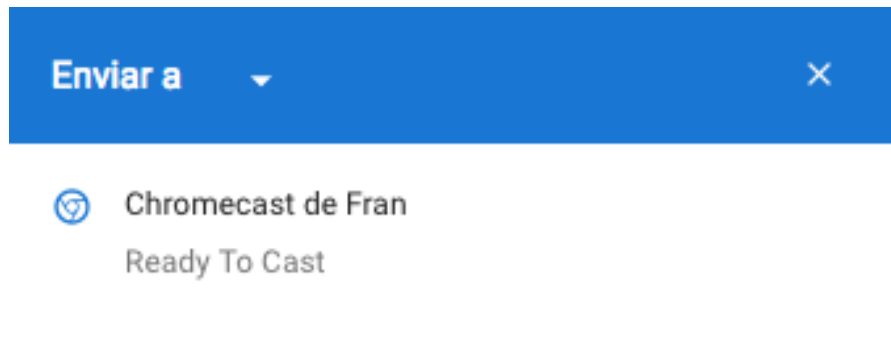


Figura 17: Selección del Chromecast

En ese momento, debería cargar en el Chromecast la aplicación receptora (receiver.html). Ya podemos escribir texto en el emisor y, al pulsar intro, debería aparecer en el televisor.

Referencias

- [1] John Affaki. Ready to cast: Chromecast now open to developers with the google cast sdk. <https://developers.googleblog.com/2014/02/ready-to-cast-chromecast-now-open-to.html>, 2014.

- [2] Netflix. Dial-2ndscreenprotocol. <https://community.arubanetworks.com/aruba/attachments/aruba/unified-wired-wireless-access/26971/1/DIAL-2ndScreenProtocol-1.6.4.pdf>, 2012.
- [3] Sharon Profis. Miracast: Everything to know about mirroring android. <https://www.cnet.com/how-to/miracast-everything-to-know-about-mirroring-android/>, 2013.
- [4] Patrick Herrmann. On wifi-display, democratic republics and miracles. 2014.