Peng Liu

Developing a Solution for Multimedia Home Networking

School of Electrical Engineering

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo 25.5.2014

Thesis supervisor:

Prof. Raimo A. Kantola

Thesis advisor:

D.Sc. (Tech.) Mikko Valimaki

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Peng Liu

Title: Developing a Solution for Multimedia Home Networking

Date: 25.5.2014 Language: English Number of pages: 6+48

Department of Communications and Networking

Professorship: Networking Technology Code: S-38

Supervisor: Prof. Raimo A. Kantola

Advisor: D.Sc. (Tech.) Mikko Valimaki

This thesis studies the modern solutions for multimedia home networking(MHN), especially the popular ones who follow the Digital Living Network Alliance standard. By conducting a research on the features and implementations of these existing solutions, our team developed a suitable mobile solution for MHN which takes advantage of AirPlay, DIAL and DLNA on the Android platform.
Firstly, the thesis provides an overview of the popular streaming technologies, including AirPlay, DLNA, Miracast and Chromecast. By analyzing the features and capabilities of these streaming technologies, a universal solution is proposed for MHN in the hope of supporting multiple protocols and bridging different platforms.
Secondly, different multimedia solutions are tested and a mobile Application for home networking on Android is implemented. The corresponding system architectures, features and analysis s methodologies are also discussed .
In terms of practical contribution, an online channel proxy is made in our "Streambels" application to fulfill our target of streaming online channels such as YouTube. By implementing this online channel proxy, home networking and Internet resources are effectively bridged together.
Based on this thesis study, an Android application for Tuxera Inc has been published . Over the 6 months after launching this application on Google Play Store, we have been able to generate a statistic report on how users utilize this app. By making analysis on the collected statistics, a short summary of the user behavior is presented and some recommendations are given to help improve user experiences.
Lastly, a discussion on the possible further development of Home network is conducted to conclude this thesis study.

Keywords: Home network, Multimedia, HTTP Streaming,
UPnP, DLNA, Miracast, AirPlay

Preface

This document is my master's thesis of *Communications and Engineering Networking* at Aalto University. All research and development of this thesis was conducted at Tuxera Inc. in Helsinki from January 2013 to June 2014. Tuxera is a high-tech startup that develop kernel-level file system and multimedia solutions for leading software, hardware and electronics companies.

Duration this project I worked together with my colleagues at Tuxera, I started to work on DLNA project for the first few months during which period I learned DLNA architecture and made a research about Digital Media Server solutions. After that I worked on an Android project to develop a universal solution for multimedia home networking.

Acknowledgements

First of all, I would like to thank the Streambels team at Tuxera, whom I worked together throughout the project. I would like to thank Karthik Ramakrishna, our lead developer. Every week he helped solving out problems with the project, no matter the question is theoretical or technical, he always answered my questions. As our project manager, Oscar Santolalla helped us with organisational problems we encountered and taught us to look at things from a end user perspective as well. Sakari Tanskanen, our mobile developer helped us by integrating Chromecast and FireTV support to Stremabels. Nadir Javed, our quality assurance engineer helped us with the quality management and testing of potential bugs before releasing the product to end users. Karolina Mosiadz, our PR helped to listen to user's ton's of feedback everyday and provide unique insights in improving Streambels. Hien Le, our UX designer helped us to develop a very handy user interface. And special thanks to Mikko Valimaki and Szabolcs Szakacsits who lead the company and gave me the opportunity to participate in this great project. Without them, I would not have been able to finish this report.

I thank my university supervisor Raimo Kantola, who helped me to develop a good thesis topic based on my project and helped me with initial problem description. I got great support from him with his critique and useful advice, especially during the middle and final peroid, when I wrote the report.

Finally, I thank everybody who supported me during my graduation work, especially my family, friends and house-mates.

Otaniemi, 25.5.2014

Peng Liu

Contents

Abbreviations

MHN	multimedia home networking
DLNA	Digital Living Network Alliance
DMC	Digital Media Controller
DRM	Digital Rights Management
DMS	Digital Media Server
DMR	Digital Media Renderer
HTTP	Hyper Text Transfer Protocol
RTSP	Real Time Streaming Protocol
SDP	Session Description Protocol
UPnP	Universal Plug and Play
DIAL	DIscovery And Launch
AP	Access Point
RAOP	Remote Audio Output Protocol
GENA	General Event Notification Architecture
SOAP	Simple Object Access protocol
DRM	Digital Rights Management
DIS	DRM Interoperability Solutions
REST	Representational State Transfer
URL	Uniform resource Locater
XML	Extensible Markup Language

1 Introduction

1.1 Home networking

People's lives are being digitalized. Home multimedia devices nowadays such as digital TVs, smart phones, digital cameras, tablets, PCs, laptops and NAS (Network Attached Storage) are all being equipped with ever greater processing power and mass storage, wielding the power to record our daily lives and handle these multimedia informations. The digitalized world has also seen a rapid growing of network deployment. According to a research [1] done in 2011, in this industrialized world, networking is being rapidly adopted at homes; for example, in the U.S. in 2009, approximately 63% of homes had gained access to a broadband connection. Over 50% had even installed their own "home network", which is defined as multiple computers or devices sharing a broadband connection via either a wired or wireless connection within the home. In a typical home scenario, most of these devices are connected to a local network such as a Wi-Fi hot spot, in order to allow musics, pictures, videos and other content to be ported across different devices.

1.2 Problem description

While the adoption of home networks has steadily increased since the late 1990s and early 2000s, the home networks has indeed encountered problems and limitations [1]. For example, the usability of home-networking technologies has become a key impediment to the adoption of new applications in the home, since the home-networking technology was originally developed for research labs and enterprise networks and does not account for the unique characteristics of home usage, such as the lack of professional administrators, deep heterogeneity, and expectations of privacy.

Among all the challenges of home-networking, connecting all media devices and make them work together is getting increasingly interesting because of the rapid growth of consuming electronic market. Although there are several widely used multimedia-streaming solutions in the market, the standards employed are not compatible with each other. Furthermore, even devices using the same standard are not always compatible with each other, due to the fact that the implementation approaches could vary from device to device. These incompatibilities have caused great inconvenience to the end users.

DLNA, AirPlay, Chromecast and Miracast are the four major multimedia home network digital living solutions. AirPlay is only used between Apple products; it provides various features, including tunes play for music, AirPlay for video and photos and screen mirroring. Miracast (previously called Wi-Fi display) was proposed by Wi-Fi Alliance and has been receiving great popularity over the recent years. Since its release version 4.2.2, Android has officially added the support for Miracast. In comparison, DLNA has been the most widely deployed solution so far, with 2.2 billion installations worldwide. It was proposed by several industrial leading

electronic manufacturers and network operators including AT&T, Broadcom, Cisco, Google, Huawei, Intel, LG Electronics, Microsoft, Nokia, Panasonic, Samsung, Sony and Verizon.

As a result of different technical designs and human aspects, these standards proposed by different device manufactures naturally experience serious compatibility issues. It usually happens that end users can have several multimedia-devices, with each one using a distinctive and unique protocol, making it challenging or even impossible sometimes to share media between those devices.

These compatibility issues has really obliged us to make a study on those multimedia-streaming technologies and to develop a more easy-to-use multimedia home networking solution with more advanced technologies.

1.3 Document overview

The overview of popular home networking standards and solutions is described in chapter 2(ref required). After a short comparison of these solutions, in chapter 3(ref required) we describe a more universal solution for multimedia home networking and its implementation. Chapter 4 presents some statistics from the Google Store during the past few months. Besides, a study based on the user feedback s is also presented. In Chapter 5, the paper is concluded by giving a discussion on the further development and prospect of home networking.

2 Background

Home networking has been a hot topic for quite a few years. Thanks to the rapid development of electronics and computer science, home networking devices are becoming more affordable and more powerful than they ever were. It is now so common that a person would have several multimedia devices that can be connected to the network. In this chapter, we give an overview of current solutions of home networking that bridges these multimedia devices. By making comparison among these solutions we try to identify the challenges that face home networking.

2.1 Overview

Early researches [2] [3] [4] conducted on home networking mainly aims to find out how home networking infrastructures can be built. The subjects of these researches, including cable connection, wireless connection, optical connection etc, concerns more about the physical layer of the home network. So far, it has turned out that the IEEE 802.11 protocol stack, among all others, is the most successful and widely deployed home networking infrastructure.

Nowadays, a typical scenario of home networking is that an IEEE 802.11 supportive wireless router connecting to a Ethernet cable or optical cable from the network operator creates a local network and other user devices simply join this network. The wireless Access Point(AP) employs the 802.11 b/g/n/ac protocol, utilizing the 2.4 GHz or 5 GHz frequency channels and providing a 100+ Mbps network connection, whose bandwidth is sufficient for transmitting the popular High Definition (1080p) videos.

In terms of network and application layer technologies, different device manufactures tend to choose their preferred multimedia-sharing protocols from the pool of protocols that have been evolving and developing for a long time.

Since late 1990s, UPnP protocol had been developed for home networking usage. At that time, XML was popular and widely used by different network applications. Under such background, UPnP was designed to fully make use of XML. UPnP is independent of media types and devices and it runs on the TCP/IP stack, thus it can be easily applied to modern network infrastructures.

In June 2003, Sony and several leading consumer electronic manufacturer s established the Digital Living Network Alliance (DLNA), a nonprofit collaborative trade association. The DLNA standard is based on the widely used UPnP protocol but it added some restrictions on media formats and some compatibility requirements. A device hardware and software can be certified by DLNA organizations to prove that it can work with other devices that also passed this certification.

In 2010, Apple quit DLNA and developed its own multimedia home networking

solution, known as AirPlay. By adding screen mirroring, authentication and Remote Audio Output Protocol(RAOP) music streaming, Apple tried to forge a more advanced home network sharing system, aiming to provide a unique user experience among Apple products. Apple's solution had indeed attracted people's interest, and the user experience had proved much better than that of other similar products in the market. With its improvement over the years, Apple's solution has now been acknowledged as one of the most popular streaming solutions.

Two years later, Wi-Fi alliance released its Miracast technology, and participated in pushing new standard in wireless home networking. The Miracast uses the Wi-Fi direct technology and it does not require a wireless local network. Instead, a peer-to-peer connection is created between the sharing and receiving devices. After its release, some major software and hardware companies soon accepted this new standard. Google, for example integrated Miracast support into its Android operating system, and provided screen-mirroring feature to other Miracast receivers.

The competition in home networking rages on over the years. In 2013, Google released a 35-dollar Dongle, using its Chromecast protocol, which makes it possible to watch YouTube and Netflix video directly on TV with such a dongle device. Laptop and mobile devices with official YouTube App or Chrome browser can control Dongle through the home local network. In this solution, the home networking is pushed to the cloud, since YouTube and Netflix content are directly downloaded from Internet whereas mobile devices just act as a controller to chose interested contents.

At the same time, in September 2013, Spotify, a startup music service company also took part in making its own home networking solution, called Spotify Connect. Spotify Connect provides an interface for users at home to access its huge music database, and directly browse and stream using its mobile application. Home networking has again been pushed towards cloud and Internet services in Spotify Connect.

Since so many companies would like to develop their own devices and even their own protocols. The market is a bit messy. Devices from different companies are not compatible with other and users have to buy different device in order to access different services like Netflix and Spotify, which are provided by different companies. This has created a big demand on a solution that can connect those devices at home and make them work together friendly.

In response of this market need, our Streambels project has been initiated, aiming to fill the gap among different protocols and connect these different types of devices in the home networking environment.

2.2 Available protocols

2.2.1 UPnP [1]

UPnP device architecture
Universal Plug and Play (UPnP) is a series of networking protocols defined to work together and seamlessly discover the presence of all devices in the network , establishing functional network services for data sharing, communications, and entertainment among these discovered devices.

In most UPnP scenarios, a control point controls the operation of one or more UPnP devices. The interaction usually occurs in isolation between control point and each device. It is the control point's responsibility to coordinate the operation of each devices and the individual devices do not really interact directly with each other.

The UPnP device architecture [5] includes seven parts:

1. Addressing
   UPnP devices have a DHCP client and it needs to search for a DHCP server when connecting to the network. An UPnP device first scans for the DHCP server and then requests an IP address when the DHCP server is found. If there is no response from the DHCP server, the device uses a automatically allocated IP address, which is acquired by randomly choosing an address in the 169.254/16 range and testing it using ARP probe to determine if it is already used. The same procedure repeats until an unused address is found. After the first IP address is set, the UPnP device periodically communicates with the DHCP server, waiting for a DHCP response that provides an available IP address. At this the device stops using the address generated by Auto-IP as soon as the interaction in progress with the old Auto-IP is completed. If there is a DNS server in the network, it can also use domain names instead of the numerical IP address.

2. Discovery
   UPnP devices advertise their services to network using the UPnP discover protocol, which is based on Simple Service Discovery Protocol (SSDP). An UPnP control point searches the existence of UPnP devices in the network using SSDP. The discovery message contains a few specific attributes of a device and its services.These attributes include device type, unique identifier and a pointer to more detailed information. The device send multicast several NOTIFY messages to a pre defined address and port to advertise its availability. A control point will listen to this standard multicast address and get notifications when new devices are available in the network. An advertisement message has a lifetime, so devices in the network would periodically send the NOTIFY message before the previous message expires. When the device or

[1] Universal Plug and Play

servers becomes unavailable or when they are shut down intentionally, previous advertisements are canceled by sending cancellation messages. Otherwise, the advertisements will eventually expire on its own. Control point can search for devices actively by multicasting an SSDP Search message. Other devices in the network will respond to the search message by unicasting directly to the requesting control point.

3. Description
   The discovery message contains the URL(Uniform resource Locater) of the description information. A control point can send HTTP GET request based on this URL to get detailed UPnP description of the device. The description includes a device description and several service descriptions.

   A device description includes vender related information such as model name, serial number and manufacture name. A device may have many services. For each service, the device description lists the service type, name and URL of the detailed service description, control and eventing. A device description may also include embedded devices and a URL of a presentation page.

   A service description includes a list of actions that servers can accept, arguments of each action, and a list of state variables. The state variables reflect the device's status during runtime.

   The description follows the XML syntax and is based on standard UPnP device description template or service description template, which are defined by the UPnP forum. The template language is written in XML syntax and is derived from an XML schema language. In this sense, the template language is machine-readable and automated tools can parse it easily.

   By using description, vender has the flexibility to extend services, embed other devices and include additional UPnP services, actions or state variables. The control point can be aware of these added features by retrieving these device descriptions.

4. Control
   A control point can ask services in a device and invoke actions by sending control messages. The control process is a form of remote procedure call: a control point sends the action to device's service, and when the action has completed on the remote device, the service returns the action results or the corresponding error messages.

   The control messages are constructed in XML format using the Simple Object Access Protocol (SOAP) and conveyed though HTTP requests. Received through HTTP responses, the action results may cause the state variables to change and those changes are reflected in the eventing messages.

5. Eventing
   UPnP service description defines a list of state variables, which are updated at runtime. The service publishes those changed state variables in the form of

event messages, and a control point can subscribe to this publishing service to learn these state transitions.

A control point subscribes to the event notification by sending a subscription message to the subscription URL, which is specified in the device description. And the control point also provides a URL to receive the event messages.

Since there is no mechanism to subscribe to a subset of evented state variables, all subscribed control points will receive all event messages regardless of why the state variable changed.

When the subscription is accepted, the device gives a unique identifier for the subscription and the duration of the subscription. The device will also send an initialize event message, which includes the names and current values for all evented variables.

The event messages are General Event Notification Architecture(GENA) NOTIFY messages, sent through HTTP with a XML body, which specifies the names of one or more state variables and new values of those variables. Once the state variable changes, the event message is immediately sent to the control point, thus the control point can get a timely notification and could display a responsive user interface. The control point then send HTTP OK message to acknowledge the device that the event message is received. The event message also contains a sequence number that allows the detection of possible lost or disordered messages.

The subscription must be renewed periodically to extend its lifetime and keep it active. The renew message which contains the subscription identifier is sent to the same URL in the subscription message. When the subscription expires, the device will stop sending eventing message to the control point, and any attempt to renew the expired subscription is rejected.

A subscription can be canceled by sending an appropriate message to the subscription URL.

6. Presentation
   Many UPnP devices provide a presentation URL to "web" interface for users. Users can access the presentation URL though a standard web browser. The control point sends an HTTP GET request to the presentation URL to get a HTML page from the device, and displays the page in a web browser, providing a more user-friendly interface for controlling and viewing the status of the device.

   The presentation page, which is an HTML page, is solely specified by the device vender. The UPnP architecture does not define the details of the presentation page, however it suggests that the presentation page shall be user friendly and shall possess some basic functionalities.

UPnP A/V devices

With the general architecture of UPnP devices being introduced. We now move

on to study the UPnP A/V(audio/video) devices in home networking . The AV control point interacts with two or more UPnP devices, one of which acts as either a source or a sink. While coordinated by the AV control point, the devices themselves interact with each other using a non-UPnP communication protocol. The control point configures the devices as needed, triggers the flow of content, then gets out of the way.

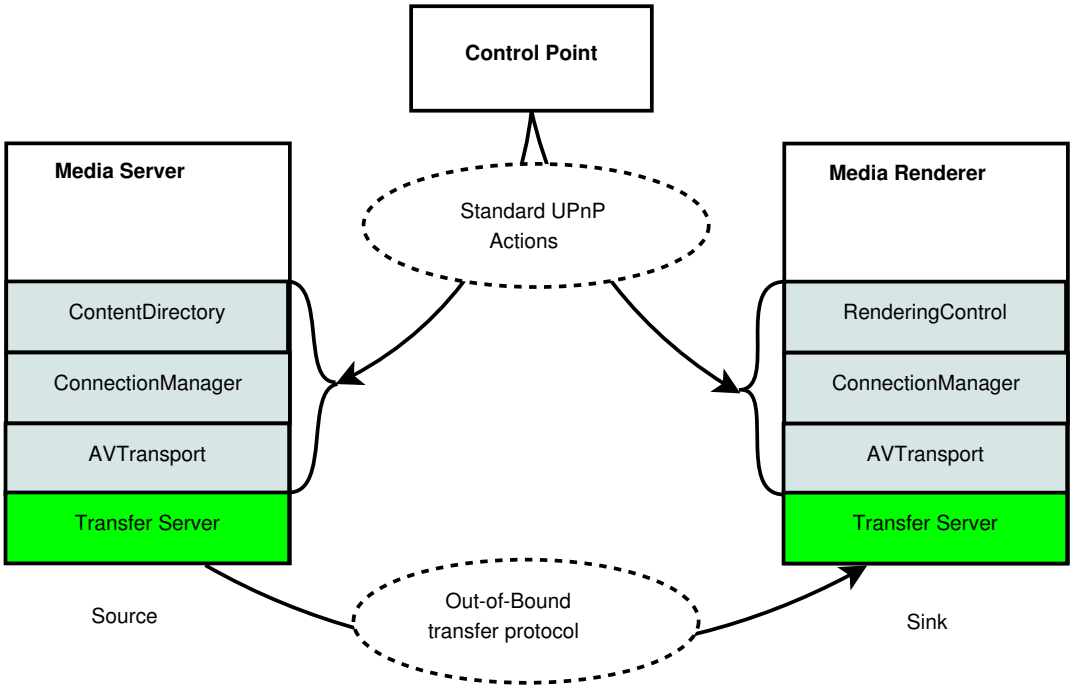

Figure 1: UPnP A/V playback architecture

1. Media Server
   The media server is used to locate available content in the home network. It's primary purpose is to allow control points to enumerate (browse or search) content items that are available for the user to render. The media server contains a ContentDirectory Service(CDS), a ConnectionManager Service(CM) , and an optional AVTransport Service(AVT) which depends on the supported transfer protocols. Some media servers are capable of transferring multiple content items at the same time.

   The ContentDirectory service is used by the control point to enumerate the content on the server. The primary action is ContentDirectroy::Browse(). After invoking this action, the control point can obtain detailed information of each item that the server can provide. This detailed information includes name, artist, date created, size and also the transfer protocols and data formats that supported by the for particular item. By parsing this detailed information, the control point is able to distinguish whether the item can be rendered by the given media renderer.

The ConnectionManager service is used to manage the connections between a control point and a device. The primary action is ConnectionManager:: PrepareForConnection(), which is invoked by the control point to prepare the server for an upcoming transfer. This action will return the instanceID of an AVTransport service that will be used later to control, say to stop, pause, seek, the flow of content. The instanceID is used to distinguish multiple instances of the AVTransport service. Since each instance is associated with a particular connection to the renderer, the instanceID enables multiple renderer support at the same time. When the control point need to disconnect the connection, it will invoke the media server's ConnectionManager::ConnectionComplete() action to release the connection. When the ConnectionManager::PrepareForConnection() action is not implemented, the control point is only able to support a single renderer at a time. In this case 0 will be used as InstanceID.

The AVTransport service is used by the control point to control the playback of the content. Operations like Stop, Pause, Seek are supported by this service. However, this service is not mandatory and the media server can choose to implement this feature according to the supported transfer protocols and data formats. If this service is supported, the InstanceID included in each AVTransport action is used to distinguish multiple instances of the service. New instances of the AVTransport service can be created by ConnectionManager's ConnectionManager::PrepareForConnection() action, and new InstanceID is allocated to each new service instance.

2. Media Renderer
   The media renderer is used to render the content obtained from home networking. Its main feature is that it can be discovered by a control point and perform content rendering according to the instructions from the control point. These instructions could control rendering settings such as brightness, contrast, volume, mute, etc. The control of flow of the content like stop, pause, seek can also be supported depending on the transfer protocol used. The media renderer provides three services including the RenderingControl service, the ConnectionManager service and an optional AVTransport service. Sometimes the rendering control and AVTransport services contain multiple independent instances so that the devices could be able to handle multiple content items at the same time. Those multiple instances can be identified by a unique InstanceID.

   The RenderingControl service is used by the control point to control how the renderer renders the incoming content. Characteristics like brightness, contrast, volume, mute etc, can be controlled by this service. The RenderingControl service supports multiple, dynamic instances, which allows a renderer to mix one or more items together. Such a dynamic instance could be a Picture-in-Picture window on a TV or a mixed audio stream. Multiple connections can be distinguished by their unique InstanceID.

The ConnectionManager service is used to manage connections associated with a device, the primary action is the ConnectionManager::GetProtocolInfo() action. The control point can invoke this action to enumerate the transfer protocols and data formats supported by the media renderer. By comparing this information with the protocol information retrieved from the media server, the control point is able to predetermine if a media server is capable of rendering a specific item from the media server. Optionally, media renderer may also implement the ConnectionManager::PrepareForConnection() action to prepare itself for an upcoming transfer. It can also assign a unique ConnectionID that can be used by a 3rd party control point to obtain information about the connections that media renderer is using. In addition, depending on the transfer protocol and data format used, this action may also return a unique AVTransport InstanceID that the control point can use to control the flow of content(stop, pause, seek, etc).

The AVTransport service is used to control the flow of streamed content. Actions like play, stop, pause and seek can be controlled depending on the transfer protocol and supported data formats. The AVTransport service can also support multiple logical instances and handle multiple simultaneous content items. The AVTransport InstanceID which is used to distinguish service instances can be allocated by ConnectionManager::PrepareForConnection().

3. Control Point
   The Control Point is used to bridge communication between a media server and a media renderer. It also provides the user interface to users. A control point does not implement UPnP services, as a result it is not visible as a device on the network. Usually the control point invokes a media server or a media renderer's services in order to complete the desired operations.

   The user control point can be used in different scenarios and usually it can perform the following functions:

   - Discover A/V devices
   - Locate desired media content
   - Get renderer's supported protocol and formats
   - Compare and match protocols and formats between media server and media renderer
   - Configure media server and media renderer
   - Select desired content
   - Start content transfer between media server and media renderer
   - Adjust rendering characteristics
   - Select next content in the list
   - Cleanup media server and media renderer

As described above, three basic functional entities are defined in the UPnP AV architecture[6], which are Media Server, Media Renderer and Control Point respectively. A physical device can consist of a combination of any of these functional entities. One typical example is that a DLNA Media player is a combination of a Control Point and a Media renderer.

A simplified UPnP Audio Video 3-box model [7] can be seen as below:

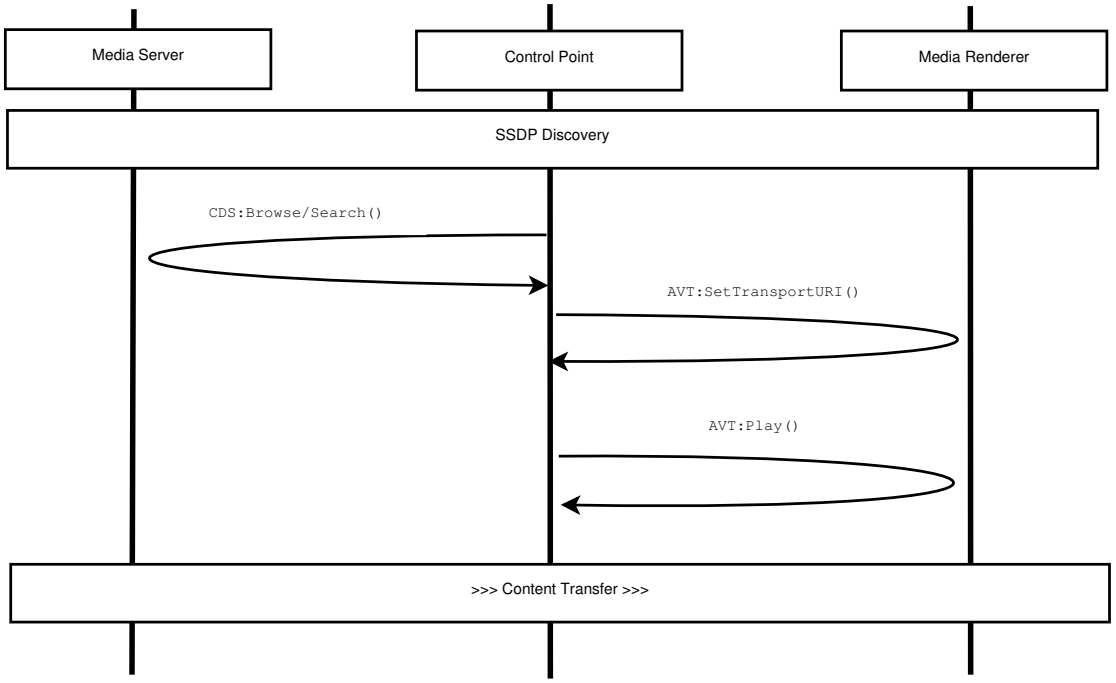

Figure 2: Typical UPnP AV use scenario

The first thing in UPnP network communication is the Simple Service Discovery Protocol(SSDP)-based device discovery. A SSDP multicast message is sent when a new device is added to the network. A control point would listen to these multicast messages. On receiving the SSDP message, the control point would send a request for the device's description and services using the location found in the SSDP discovery message. Then the control point can issue the services action command using the Simple Object Access protocol (SOAP).

In media sharing scenarios, the control point would browse the information about the Content Directory Service (CDS) provided by the Media Server. A browse/Search action can be invoked to navigate through the content stored in the Media Server device. After the control point has selected the media content from Media Server, a Media Renderer AVTransport::SetAVTransportURI would be sent by the control point to the Media Renderer. Finally, the Play command is invoked by the control point to instruct the Media Renderer. Afterward, the transfer begins. The media stream travels directly between the Media Server and the Media Renderer, through HTTP, RTP or other streaming protocols.

The media playback control actions can also be invoked by the control point. Methods supported include volume control, seek, pause etc.

2.2.2 DLNA[2]

DLNA is a relatively old industry standard compared with other home networking solutions. It is mainly based on the UPnP Audio/Video architecture, which is discussed in 2.2.1. As a result it is widely used by many manufactures. Newer home networking solutions are also influenced by DLNA and they follow similar technologies used in DLNA. In this paper the DLNA and UPnP standard architectures are studied to help us gain a grasp of how a home networking solution could look like.

An overview of the DLNA architecture [8] is described below:

1. Architectures and Protocols

Table 1: Key Technology Ingredients

Functional Components	Technology Ingredients
Connectivity	Ethernet, 802.11 (including Wi-Fi Direct) MoCA, HPNA and Bluetooth
Networking	IPv4 Suite
Device Discovery and Control	UPnP* Device Architecture v1.0 2.2.1
Media Management and Control	UPnP AV and UPnP Printer:1 2.2.1
Media Formats	Required and Optional Format Profiles
Media Transport	Media Transport
Remote User Interfaces	CEA-2014-A

   The DLNA architecture is built upon the UPnP protocol, which is discussed in in 2.2.1. In the network layer DLNA uses the IPv4 suite. On top of the network layer, the UPnP device architecture and UPnP AV architecture is used to control and manage media devices. The DLNA guideline also addresses the media format compatibility and media transport interoperability issues in support of interoperability among devices.

2. Media Format Profiles
   DLNA defines the media formats used by the DLNA home networking standard. There are three types of media in DLNA: music, video and photo.

   (a) Music
   Minimal requirement is the LPCM format. Used by PCM raw data, this format is not compressed and it does not require heavy CPU usage.

[2]Digital Living Network Alliance

However the bandwidth consumption is considerably bigger than other formats.

MP3 is the most popular music format. It is a compressed format and requires some CPU power for encoding or decoding. Compared with LPCM, the bandwidth consumption of MP3 is less, making it suitable for low bandwidth networking.

AAC is another kind of compressed audio format and it becomes popular since it is the default media format of iTunes. It has similar characteristics of MP3.

(b) Photo
The minimal requirement in the DLNA guideline is the JPEG format. In many occasions JPEG is the only suggested format due to its proven quality and compress ratio.

(c) Video
The minimal requirement in DLNA guideline is the MP4 format. The detailed audio and video codecs are also specified in DLNA media format guidelines.

In a device-to-device scenario, the media server may store a huge amount of differently formatted media. The communication between two devices should follow the same encoding mechanism. Normally the media server takes the responsibility to transcode the media to a certain format defined by the DLNA media format profile guideline.

3. Link Protection
DLNA Link Protection is defined as the protection of a content stream between two devices on a DLNA network against illegitimate observation or interception.

Content protection is an important mechanism to ensure that commercial content is protected from piracy and illegitimate redistribution. Link Protection is a technique that enables distribution of protected commercial content on a home network. It provides protection for copyright holders and content providers without sacrificing consumer flexibility.

4. Digital rights management (DRM) Interoperability Solutions (DIS)
DIS is intended to be used to enable the secure transfer and use of protected commercial content among different implementations on network media devices. The content could be protected by different content protection technologies, which are described as DRMs in short.

5. Device Profiles
A Device Profile is a collection of DLNA capabilities and features within a DLNA device. For a device to be compliant with a Device Profile, it has to conform to all of the guidelines listed for that Device Profile.

In practice, Device Profiles reference existing optional or recommended DLNA guidelines that enable certain features, and makes those DLNA guidelines mandatory within the context of a Device Profile. A Device Profile can also provide some additional guidelines that complement or modify existing DLNA guidelines for a feature.

A particular type of DLNA Device Profile is the Commercial Video Profile (CVP). A CVP Device Profile is an extension of the DLNA guidelines that will allow content from service providers and multichannel video programming distributers to be distributed on the DLNA network. DLNA Commercial Video Profiles (CVPs) are defined as Device Profiles that consistently enable commercial content that enters the home network through a gateway device via an interface to a commercial content service provider. Since different regions of the world have different requirements for commercial content, there are multiple CVPs defined.

2.2.3 AirPlay

AirPlay is Apple Inc's home networking solution. It is a family of protocols used to display different types of media content on Apple TV from other iOS devices. AirPlay support multiple functions, including displaying photos and sideshows from iOS devices, streaming audio from iOS devices or iTunes, as well as displaying videos from iOS device and showing the whole screen on Apple TV, which is known as AirPlay Mirroring.

AirPlay's specification is not open to public. However unofficial specifications have been made by some hackers through reverse engineering the protocol stack [9]. These unofficial specifications could be found on the Internet [9]. The specification includes 6 parts, which are described below:

1. Service Discovery
   The service discovery of AirPlay stems from the IETF Zeroconf Working Group, who is dedicated to improve the ease-of-use (Zero Configuration) of networks. The Zeroconf working group has made it possible to make two devices in the network to communicate effectively using IP, without requiring a specialist to manually configure the network.

   AirPlay's service discovery is based on Multicast DNS [10], which fulfills the Zeroconf requirement. Multicast DNS is a way of using familiar DNS programming interfaces, packet formats and operating semantics, in a small network where no conventional DNS server has been installed. The requirements for Zeroconf name resolution could be met by designing an entirely new protocol, since it is better to provide this functionality by making minimal changes to the current standard DNS protocol. By using Multicast DNS, most current applications need no changes at all to work correctly using mDNS in a Zeroconf network. Besides, engineers do not have to learn an entirely new protocol.

Moreover current network packet capture tools are already capable of decoding and displaying the DNS packets. Thus they do not have to be updated to understand new packet formats.

An AirPlay device such as the Apple TV publishes two services. The first one is RAOP (Remote Audio Output Protocol), used for audio streaming. The second one is the AirPlay service, used for photos and video content.

The AirPlay server is an HTTP server (RFC 2616). Two connections are made to this server, with the second one being used as a reverse HTTP connection. This allows a client to receive asynchronous events, such as playback status changes, from a server.

2. Video streaming
   The video streaming uses typical HTTP streaming technology, the controller sets the streaming URL to Apple TV or other AirPlay receivers. While the URL is set, Apple TV starts to download video from the server using the URL and starts playing when enough data is buffered. The control messages can be seen in table 2.

   Note that Apple TV does not support Video Volume Control.

Table 2: AirPlay Video Control HTTP requests

Method	Request	Description
GET	/server-info	Fetch general informations about the AirPlay server
POST	/play	Start video playback
POST	/scrub	Seek at an arbitrary location in the video
POST	/rate	Change the playback rate,0 is paused, 1 is normal
POST	/stop	Stop playback
GET	/scrub	Retrieve the current playback position
GET	/playback-info	Retrieve playback informations like position, duration...
PUT	/setProperty	Set playback property
GET	/getProperty	Get playback property

3. Photo streaming
   Image streaming uses the HTTP put message to send raw image data to Apple TV or other devices. After the whole image is received, the image is then rendered on screen. AirPlay also supports slide show, the control message can be seen in table 3.

4. Music streaming
   AirPlay music streaming is a bit different from video and image streaming. The technology used is the RTSP streaming protocol, which is more of a "push

Table 3: AirPlay Photo Control HTTP requests

Method	Request	Description
GET	/slideshow-features	Fetch the list of available transitions for slideshows
PUT	/photo	Send a JPEG picture to the server
PUT	/slideshows/1	Start or stop a slideshow session
POST	/stop	Stop a photo or slideshow session

like" protocol. Different than HTTP streaming where the server responds a request, The RTSP streaming server actively pushes UDP packets to the receiver . However Apple does not use the standard RTSP but instead uses its own implementation of RTSP, which is called RAOP (Remote Audio Output Protocol). The control messages of RAOP can be seen in table 4.

Table 4: AirPlay Audio Control RTSP requests

RTSP request	Description
OPTIONS	Ask the RTSP server for its supported methods
ANNOUNCE	Tell the RTSP server about stream properties using SDP
SETUP	Initialize a record session
RECORD	Start the audio streaming
FLUSH	Stop the streaming
TEARDOWN	End the RTSP session

5. Screen mirroring
   AirPlay screen mirroring is achieved by transmitting H.264 encoded video stream over a TCP connection. The stream is packeted with a 128-byte header. The audio uses the the AAC-ELD format and is sent using AirTunes protocol. The NTP(Network time protocol) protocol is used for synchronization and the synchronization takes place on UDP port 7010(client) and 7011(server). The AirPlay server runs a NTP client. Requests are sent to an AirPlay client every 3 seconds. The reference time stamp marks the beginning of the mirroring session. The control messages can be seen in table 5.

6. Authentication
   An AirPlay server can require a password for displaying any content from the network. It is implemented using standard HTTP Digest Authentication (RFC 2617), over RTSP for AirTunes, and HTTP for everything else. The digest realms and usernames accepted by Apple TV are described in table 6.

Table 5: AirPlay Mirroring Control HTTP requests

Method	Request	Description
GET	/stream.xml	Retrieve information about the server capabilities
POST	/stream	Start the live video transmission

Table 6: AirPlay HTTP Digest Authentication

Service	Realm	Username
AirTunes	roap	iTunes
AirPlay	AirPlay	AirPlay

2.2.4 DIAL

Chromecast and FireTV use the DIAL [11] (DIscovery And Launch) protocol, co-developed by Netflix and YouTube, to search for available devices on a Wi-Fi network. Once a device is discovered, the protocol synchronizes information on how to connect to the device. The protocol is proposed by Google and Netflix and consequently YouTube and Netflix have already had their build in DIAL device discovery functionality. The streaming part uses HTTP streaming, which means a controller can directly set the streaming URL and the receiver will start downloading automatically.

The DIAL protocol has two components: DIAL Service Discovery and the DIAL Representational State Transfer (REST) Service. DIAL Service Discovery enables a DIAL client device to discover DIAL servers on its local network and gain access to the DIAL REST Service on those devices. The DIAL REST Service enables a DIAL client to query, launch, and optionally stop applications on a DIAL server device.

1. DIAL Service Discovery
   The DIAL Service Discovery protocol is based on Simple Service Discovery Protocol (SSDP), which is defined as part of UPnP device architecture discussed in 2.2.1.

   A DIAL client will first send an M-Search request, which includes the Search Target header, over UDP to the IPv4 multicast address 239.255.255.250 and UDP port 1900. After that the SSDP/UPnP server responds with a response message including a LOCATION header containing an absolute HTTP URL for the UPnP description of the root device. When receiving the M-SEARCH response, the DIAL client then sends an HTTP GET request to the URL found in the LOCATION header of the M-SEARCH response, to get a prospected HTTP response message containing a XML format UPnP device description. The overall flow of DIAL discovery is shown in figure 3.

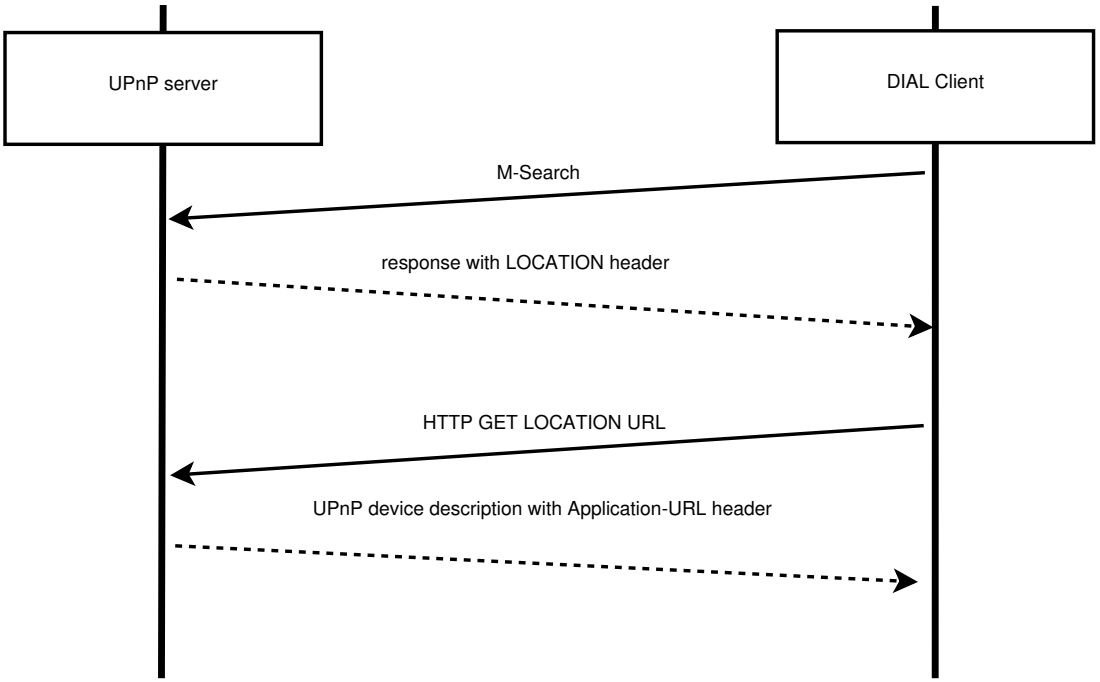

Figure 3: DIAL Discovery

2. DIAL REST Service
   The DIAL REST service allocates URLs for different resource applications such as YouTube and Netflix. Then the application can be controlled by issuing HTTP requests against the URL for that application. The Application resource URL is constructed by concatenating the DIAL REST service URL, a single slash character ('/') and the application name. The application name must be registered in DIAL Registry to be used.

   A DIAL client sends an HTTP GET request to the application resource URL. The server receiving the request then extract the application name and check if the application is installed or not. If the application is not recognized, the server will either return 404 Not Found or trigger the installation of this specific application. If the application has been installed, the DIAL server return an HTTP response with 200 OK and a body contains MIME type in XML.

   The client then sends a HTTP POST request to the application resource URL to launch the desired application. On receipt of a valid POST request, the DIAL server will extract the application name, run the application, and then send a HTTP response with the LOCATION header, to inform the absolute HTTP URL, which identifies the running instance of the application.

   The first-screen application can also send small amount of data to the DIAL server, and then DIAL server can send the information to DIAL clients. After the application is launched and communication is established, the DIAL client can communicate directly with the application. The flow chart of the DIAL REST service is shown as figure 4.

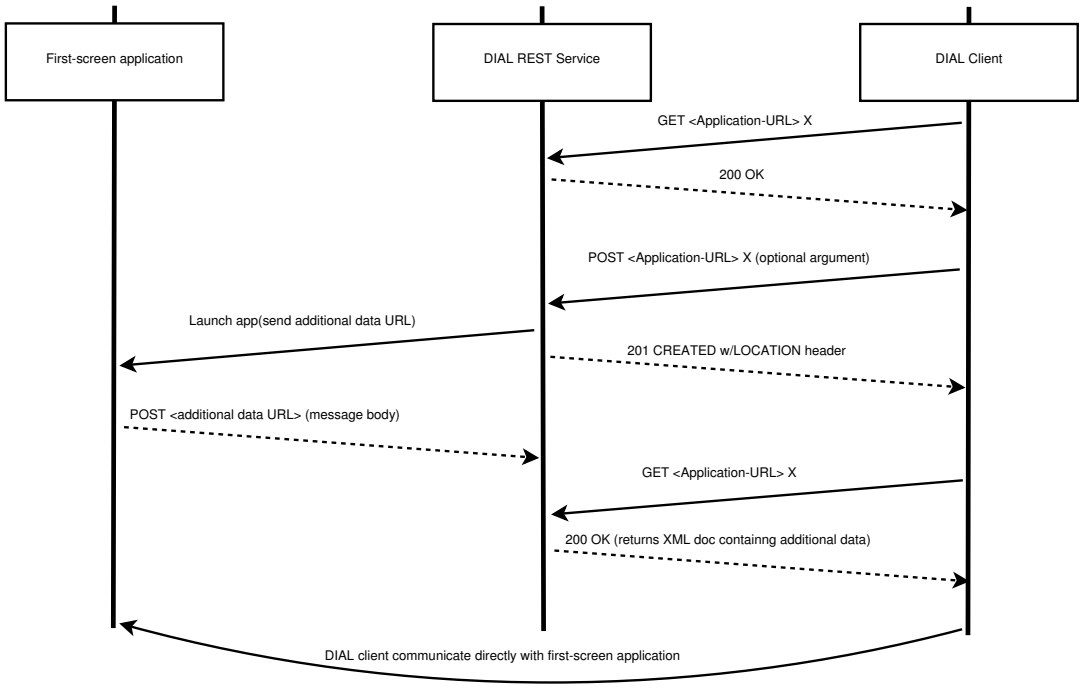

Figure 4: DIAL REST service: application launch

2.2.5 Miracast

Miracast [12] is very different on technology perspective. Devices which utilizing Miracast technology are not necessarily connected to the same local network, Wi-Fi peer to peer connection will be created when needed. This makes Miracast more adaptive than other technologies, in other words, Miracast is not limited to the pre-configured network infrastructure.

Another point worth mentioning is that Miracast utilizes many Wi-Fi alliance building blocks that is constantly developed over the years. These components include Wi-Fi CERTIFIED n (improved throughput and coverage), Wi-Fi Direct (device-to-device connectivity), Wi-Fi Protected Access 2 (WPA2) (security), Wi-Fi Multimedia (WMM) (traffic management) and Wi-Fi Protected Setup. These technologies have enriched the user experience and increased user's trust in Wi-Fi.

Not limited to Wi-Fi direct connection, some Miracast devices also support Tunneled Direct Link Setup (TDLS), which allows devices to connect via an infrastructure network. TDLS enables more efficient data transfer and keeps the advantage of more advanced Wi-Fi capabilities at the same time.

In most cases, Miracast connections are expected to be predominantly established between Wi-Fi devices connected with each other directly, without an AP acting as an intermediary. When two devices connect with each other directly, one fulfills its role as the source(the transmitting device) and the other functions as a display(the

device receiving and rendering the content to the user).

There are 4 typical topologies that are supported by Miracast. The Source could directly connect to Display without AP present, or the Source with access to AP and direct connect to Display, or the source could directly connect to Display with AP present, but not connected, or the Source and Display could connect to each other and connect to AP at the same time. These four topologies can be described as in Figure 5.

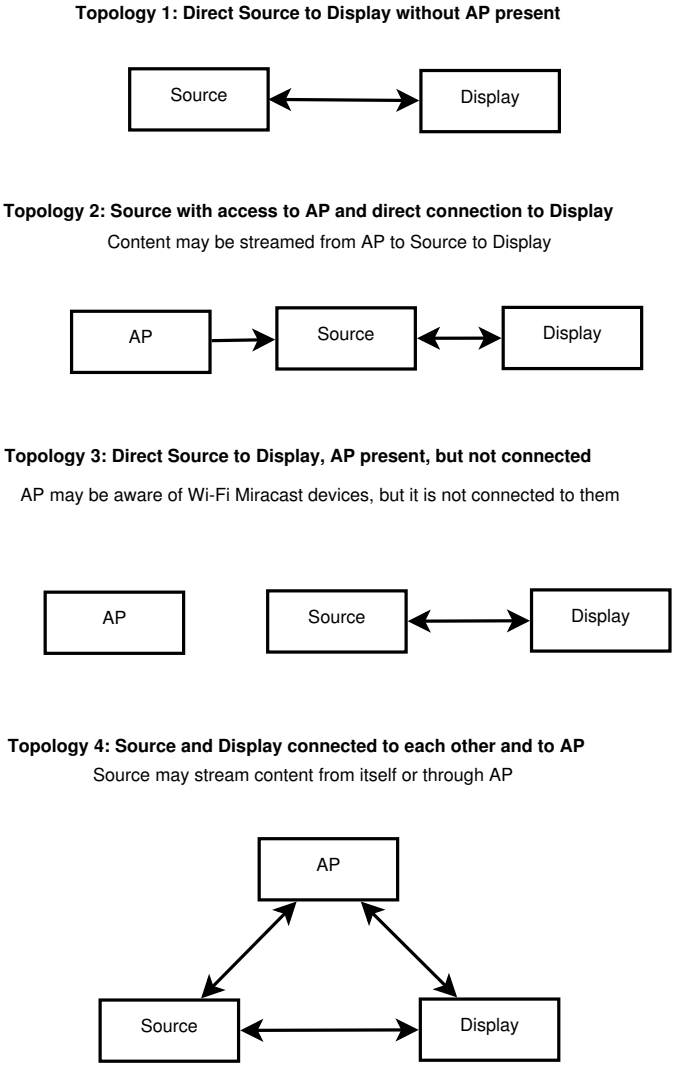

Figure 5: Miracast topologies

On technology perspective, as mentioned previously, Miracast is built upon many different Wi-Fi technologies. These technologies are built together in an architecture that can be described by Figure 6.

The technology component can be described in 6 different aspects:

1. Connectivity

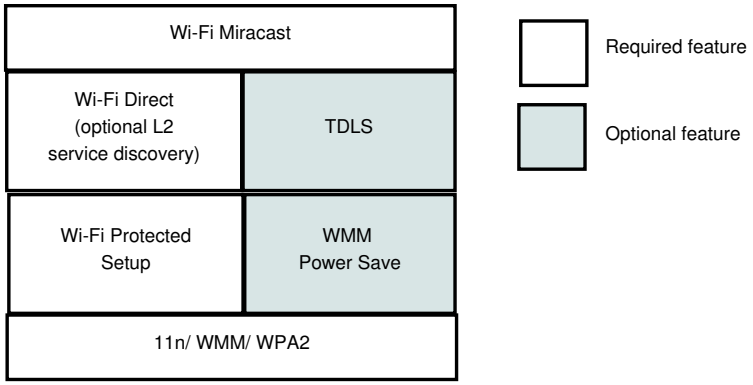

Figure 6: Miracast technology architecture

Wi-Fi CERTIFIED n provides a transmission channel designed to support multimedia content.

2. Device-to-device connectivity
   Wi-Fi Direct allows devices to connect directly to each other easily, without the need for a Wi-Fi AP. TDLS allows devices that are associated to the same Wi-Fi network to establish a direct link with each other.

3. Security
   WPA2 encrypts the transportation between the source and the display, ensures the safety of multimedia content.

4. Quality of service (QoS)
   Wi-Fi Multimedia (WMM) gives real-time content priority, which is appropriate over best-effort traffic. This brings better user experience for multimedia content such as video and audio.

5. Battery life
   WMM Power Save extends the battery life of mobile devices by minimizing the time the device is actively connected to the AP during idle time. Power save mechanisms in Wi-Fi Direct also provide similar benefits when connecting devices without an AP.

6. Ease of installation
   Wi-Fi Protected Setup helps users to automatically configure Wi-Fi networks, enable WPA2 security, and add new devices.

The whole Miracast session can be described as following steps:

Device Discovery
Source and display devices discover each other prior to connection setup. The Device discovery mechanism is defined in the Wi-Fi Peer-to-Peer (P2P) Specification.

Service Discovery

Source and display devices discover each other's Miracast capabilities prior to connection setup. The Service discovery mechanism is defined in the Wi-Fi P2P specification.

Device selection
A remote device is selected for connection setup. User input and local policies may be used to decide which device is a display and which is a source.

Connection setup
Connection setup selects a method (Wi-Fi Direct or TDLS) to manage the connection. Wi-Fi Direct sets up a group owner and client to initiate a device-to-device link. A WPA2 single-hop link with selected devices is established. Upon the establishment of connectivity between the source and display devices, the display initiates a Transmission Control Protocol (TCP) connection, with a control port using Real-Time Streaming Protocol (RTSP) to create and manage the sessions between source and display devices.

Capability negotiation
Source and display devices determine the parameters for the Miracast session.

Content protection setup (optional)
If the devices support content protection and their streaming content requires protection, the session keys for link content protection will be derived using High-bandwidth Digital Content Protection (HDCP) 2.0/2.1. HDCP session keys will be established before the RTP session is initiated. This feature is designed to protect the digital rights of content owners and to encourage the content owner's efforts to make their content available.

Session establishment and streaming
Upon completion of capability negotiation, the source and display devices setup the Miracast session prior to streaming content. The audio and video content available on the source device is packetized using Moving Picture Experts Group 2 Transport Stream (MPEG2-TS) coding and encapsulated by Real-Time Protocol (RTP) User Datagram Protocol (UDP) and Internet Protocol (IP). Finally, IEEE 802.11 packetization enables the source device to send content to the display device.

User input back channel setup (optional)
A User Interface Back Channel(UIBC) for transmitting control and data information related to user interaction with the user interface is set up. User inputs at a display are packetized using a UIBC packet header and transported using Transmission Control Protocol/Internet Protocol (TCP/IP).

Payload control
When the payload transfer starts, devices may adapt transmission parameters on the basis of channel conditions and power consumption. Adaptation can be achieved

by: Compression ratio change and macroblock skipping (using the H.264 standard); Frame skipping (if the display device supports this functionality, the source device may skip some of the frames to be transmitted according to the current resolution); Format change.

Display session teardown
Either the source or the display terminates the Miracast session.

2.2.6 Other protocols

Apart from all the mentioned standards above, many other companies or associations also developed their own proposals, such as SonosNet [13] that is based on peer to peer network and Spotify Connect [14]. With all these standards and proposals competing the market, the war of standardization on home networking, however, is still not over.

2.3 Comparison of existing solutions

2.3.1 History

- DLNA is proposed by several leading consumer electronic manufactures based on the UPnP technology. From early 2000s on, over 2.2 billion devices with DLNA have been shipped, making it possible to share audio and video seamlessly among different smart devices. Moreover, the DLNA alliance had been holding two meetings annually to discuss the marketing and development related issues, making DLNA a more and more accomplished standard.

- AirPlay, on the other hand, is proposed by Apple Inc. After departing the DLNA alliance in 2010, Apple proposed AirPlay, which brought more advanced features such as screen mirroring, RAOP audio streaming and some authentication functionalities.

- Miracast is a most recent technology. It was formerly known as Wi-Fi Display, which was originally proposed in 2012 by the Wi-Fi alliance. Different than AirPlay and DLNA, it is not based on home AP but Wi-Fi direct instead. It provides a screen-mirroring feature that resembles Apple's AirPlay Mirroring. Now it has gained great popularity among manufactures and software ventures alike. For instance, Google has launched its Android 4.2 with native support for Miracast. The latest Kitkat Android 4.4 has even been certified as Miracast compatible, by the Wi-Fi alliance according to the Wi-Fi Display Specification. It is now commonly acknowledged that this standard will soon become very popular in multi-screen sharing market.

- Chromecast or Google cast is another new technology in market. Released in 2013, a piece of 2.83-inch (72 mm) dongle hardware, which utilizes the Google cast standard, has become a hot topic recently. With a 35$ price tag, it has been ranked as the most popular device of its kind. The Google cast

standard was proposed with the joint effort of Google and Netflix. Since they are Internet companies, this standard has been designed with Cloud in mind. With the support of Cloud, the content is directly streamed from YouTube or Netflix servers to the Chromecast dongle. One thing worth mentioning is that when using dangle, any applications running on mobile platforms are just acting as a control points. Dangle also provides features like browser mirroring. For example, with a Chromecast plugin, a Chrome browser can stream its web tab to dangle who transfer the signal to a big screen TV. In a foreseeable future, the Google cast standard could become more and more popular.

2.3.2 Market

– DLNA is one of the first proposed solutions for multimedia home networking, thus it is so far the most accepted one. Figure 7 shows the history and prediction of the DLNA-certified device sales. In 2018, the sales will reach 7.32 billion, nearly the same as the human population on earth.

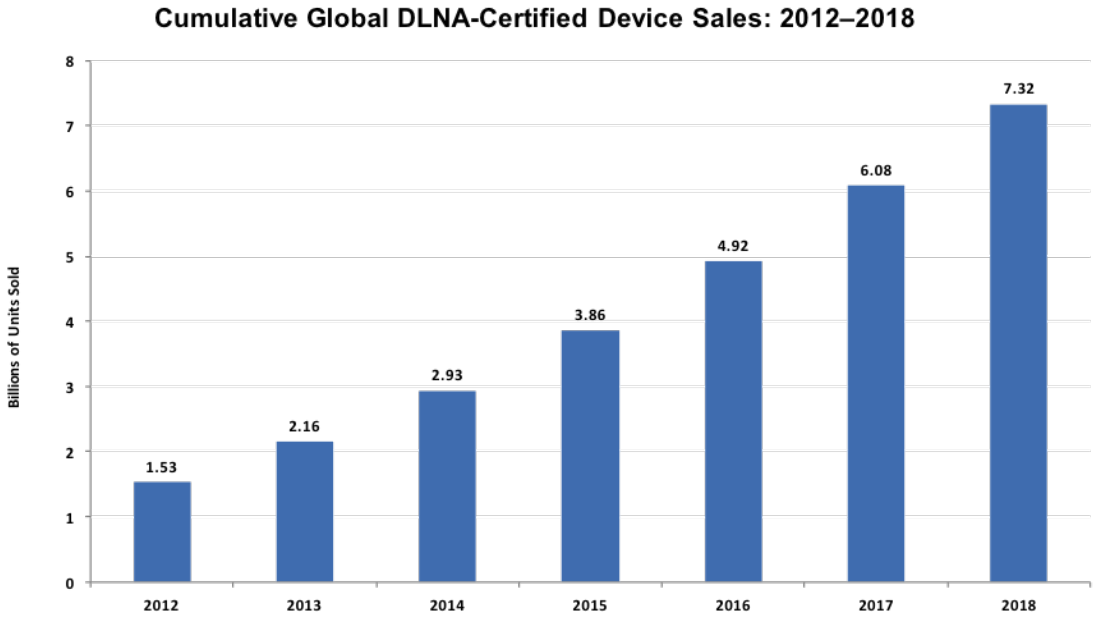

Figure 7: Cumulative Global DLNA-Certified Device sales

– AirPlay is bundled with Apple products. With great sales of Apple TV, Airport Express, Mac, iPhone, iPad, iPod, many families have become accustomed to use Apple's product for everything. In this sense AirPlay has become the easiest way to build home networking solutions. Moreover, it could be the only solution for Apple users, since a lot of speaker manufactures implement their own AirPlay receiver on their AirPlay compatible speakers. And indeed AirPlay provides enough easy to use features for daily usage.

– Bundled with Android operating system, Miracast has experienced a fast growth in the past two years. Many TVs have been built with Miracast support, to accept peer-to-peer Wi-Fi direct connection.

– Chromecast dongle is a cheap device that everyone wants to try. It can be used to easily upgrade an old TV to a "Smart TV". What's more, since Google has provided good content support for Chromecast dongle, it has soon been accepted by huge amount of users.

2.3.3 Technology feature

1. Media format support
   AirPlay and Chromecast has very limited media format support, since there are only limited Apple device types. For example, Chromecast has only released 2 devices so far and there is not much change in the media format. Similarly Apple TV, even in its third generation, has merely seen the improvements in its high definition supports, rather than the changes in media format support.

   In contrast, DLNA not only specified mandatory media formats such as LPCM, JPEG and MP4, but provided a lot more optional media formats in its specifications as well.

   Since Miracast is a screen mirroring technology, all formats that can be played on a device are supported in Miracast streaming. Consequently, there is no mandate on media format in Miracast.

2. Networking technologies used

   A short technology specification comparison is made to help better understand the existing solutions. Table 7 below shows the main technology used in different popular solutions.

Table 7: Technology used comparison

	Device discovery	Control Protocol	Streaming protocol
DLNA	SSDP	UPnP	HTTP
AirPlay	Multicast DNS	HTTP	HTTP/RTSP
DIAL	SSDP	Chromecast	HTTP
Miracast	Wi-Fi direct		Wi-Fi direct

Compared to some standards which only provide basic features, others also offer advanced features, including screen mirroring. Table 8 below shows the advanced features provided by different solutions

Table 8: Advanced feature comparison

	DLNA	AirPlay	Chromecast	Miracast
Screen mirroring	No	Yes	No	Yes
Multiple connection	Yes	No	No	No
Authentication	No	Yes	No	Yes

According to the comparison, each standard has its own features and uses different protocols to communicate. There are, however, many common features and protocols that are supported by most standards. For example, the HTTP protocol is frequently used to handle video and photo streaming, by many streaming solutions. Another example is that the UPnP device discovery protocol SSDP is commonly used for device discovery.

Since multiple standards share the commonly used protocols in their implementations, it is possible to make an application that is aware of all these common protocols. In this sense, making such a mobile application to connect multiple types of devices in a home network can be a good solution to home-networking interoperability.

3 Developing a solution for multimedia home networking

To fulfill the need of interoperability among devices in home networking, Tuxera Inc. started a project named Streambels(later renamed as AllConnect). The project aims to solve the interoperability issue in multimedia home networking by making a universal solution that can connect all available devices at home and make them work together regardless of what protocol they use.

Most devices at home are embedded solutions and have their own firmwares, it is hard to update or even impossible to upgrade the software running on these devices. On the other hand, most home network users are not knowledgeable enough to to manually set up the more advanced network features to achieve certain degree of device interoperability. Plus most of these network infrastructures are not designed to be easily configured. Due to these reasons, building interoperability among different devices through a mobile device seems to be the most straightforward solution, for mobile devices can serve as a very flexible and programmable portal for home-networking. Other advantages of mobile devices include their great processing power, networking capabilities and their wide adoption and availability. Through the available platforms and tools, a mobile application could possibly be developed to control all multimedia streaming data flows and act as a personal access portal for home networking.

After a year of development, our team have built up an Android application that can be used to control and connect every known type of multimedia device at home. Encouragingly, the number of our application users have grown to nearly one million so far, providing a strong proof of the effectiveness of our solution.

3.1 Architecture overview

In order to solve the multimedia home networking interoperability problem, the system should be designed to control media playback sessions. Consequently, content navigation, manage receiver device, and media playback should be the most important three components. In our solution, the system architecture consists of three major parts: device discovery, content management and streaming.

Discovery component is responsible for device discovery. As discussed in2.2.1, UPnP devices and DIAL devices use Simple Service Discovery Protocol for device discovery. An application firstly send a M-Search request over UDP to the IPv4 multicast address 239.255.255.250 and UDP port 1900. Then the application listens to other devices' response. A DIAL device will return a response with Application URL header, while the UPnP/DLNA devices will return a message with a XML body, which gives detailed service URL and description URL. On the other hand, Apple's products use Multicast DNS is for discovery.

For the three protocols we are planning to support, we need to implement two kinds of discovery mechanism: Simple Service Discovery Protocol(SSDP) and Multicast DNS protocol.

Content management component is responsible for organizing and navigating multimedia contents which can be found in the home network. In our solution, these content sources include both phone's local storage and DLNA digital media servers that are connected to home network. Content from these sources could be streamed using all of the three protocols that we support.

Streaming component is responsible for streaming multimedia content to the selected multimedia receivers, such as TVs, wireless speakers, set top boxes. Since DLNA, AirPlay video/ photo and Chromecast all use HTTP streaming, while AirPlay music uses Remote Audio Output Protocol (RAOP). In our solution, two types of media server are built inside our application. A RAOP server will handle the AirPlay music streaming and a HTTP media server will handle the streaming of all the other solutions.

3.2 Implementation

Since the application is built upon Android platform, we studied Android system architecture and Android Software Development Kit (SDK). Fortunately, Android SDK provides many useful Application Programming Interfaces (APIs) and gives crucial permissions to access hardware features, such as accessing Internet, accessing WiFi state change, allowing WiFi multicast, reading phone storage. The programming language used in developing Android Application is Java, however some CPU-intensive work such as transcoding have to be implemented in C and embed to the application using Android Native development kit (NDK).

While Apple has not provided its official specification of AirPlay, the implementation of AirPlay is mostly written following the unofficial guidelines. However, Apple provide its official Multicast DNS (mDNS) implementation. This piece of code is reused and compiled as a shared native library. Remote Audio Output Protocol (RAOP) is implemented according to the unofficial guidelines. The implementation includes a UDP server and TCP control channel.

DLNA has open specification only for its members. Tuxera is a member of DLNA so we got detailed specification and test tools. Since DLNA is a popular standard, there are a lot of open source UPnP/DLNA libraries. In our implementation, we used a library called "cling"[15], it has minimal implementation of UPnP device discovery, description information parsing, and basic message handling. More code is written by us to ensure the media format compatibility across different devices.

Google has officially provided Chromecast SDK for different mobile platforms, so integration with Chromecast becomes trivial. In Streambels project, the Chrome-

cast integration is built upon Chromecast API.

When it comes to media server implementation, according to the DLNA guideline, certain additional headers need to be implemented in the stream. This requirement demand the HTTP server to be flexible to add DLNA specific headers. Another requirement is that "Seek" action needs to be supported on the server side, in the implementation we need to enable byte based seeking operation.

For receivers other than DLNA standard, a basic file server with byte range support will be sufficient to do the work.

In order to serve media for all the receivers from both online and local storage, a separate media proxy also needs to be implemented.

After investigating and comparing multiple server implementations on Android, we concluded that NanoHTTPD is the ideal solution for our need. It is easy to use, Apache licensed, very tiny and efficient implementation. Because it is minimal implemented, it is also easy to be modified. Additional headers can be easily added to be compatible with DLNA receivers. Besides, a proxy is also easy to integrate with the NanoHTTPD.

Taking all these technical details into consideration, we concluded a simplified architecture of our implementation, which is shown in the Figure 8.

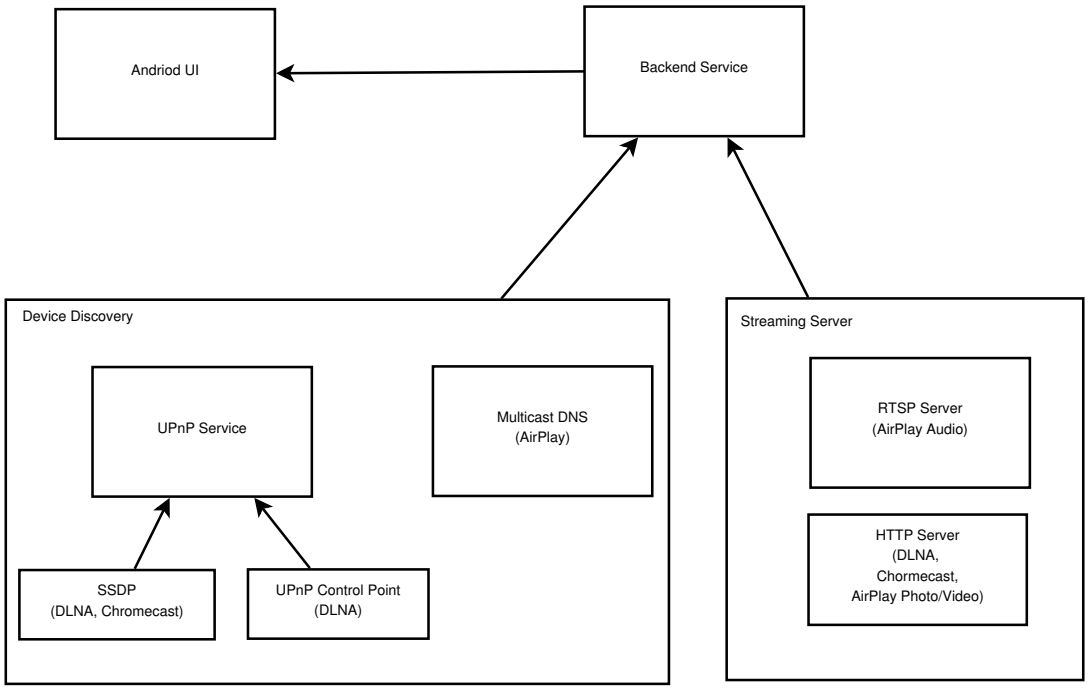

Figure 8: Simplified application architecture

When streaming content, the data flow can be described in three different mod-

els. Figure 9 shows the following three use scenarios:

If the content is stored in mobile phone, a streaming server in the application will be used to stream the content from phone to selected receiver.

Otherwise, if the content is located on Internet and the receiver is a DLNA Media Renderer, a proxy will be needed. The proxy will firstly download the resource stream, then add the required headers by DLNA specification and finally stream the modified content to the selected DLNA Media Renderer.

Finally, if the streamed content locates in a DLNA Digital Media Server, then the source can be used directly by all receivers. In this case, the streaming proceeds directly from media server to receivers. Application will only be used as a control point and do not participate in the media transmission in this scenario.

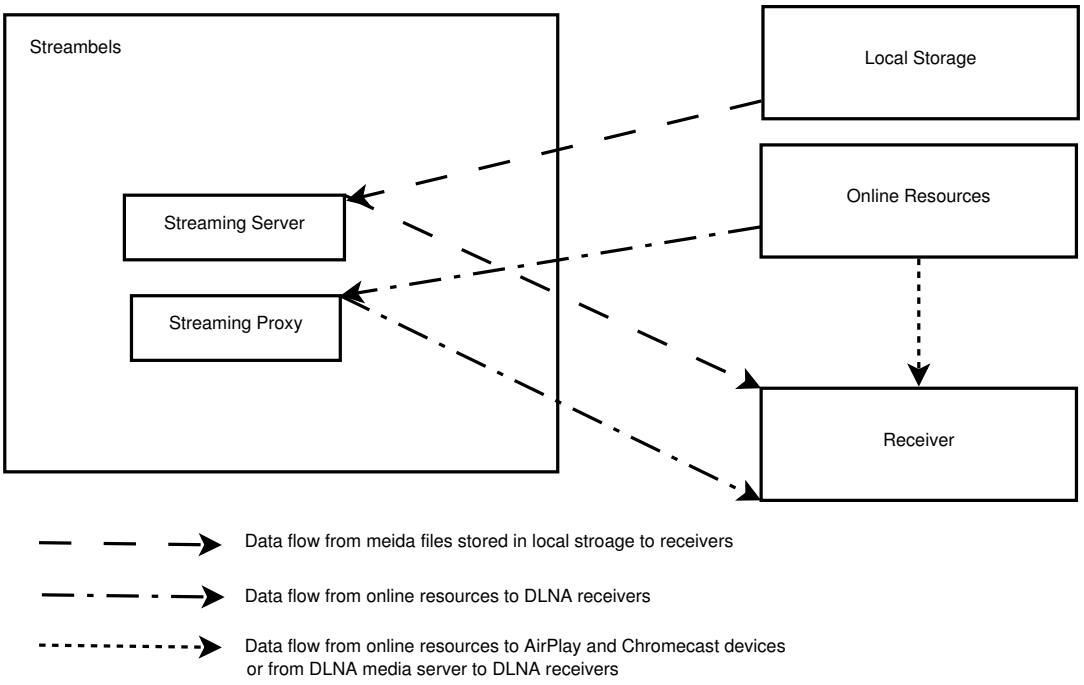

Figure 9: Simplified data flow

3.3 UX design

In terms of UI and UX, the application should be simple enough to be used by everyone. Users should be able to locate the media, browse content on different sources, and follow the data flow between devices without any difficulty. The control of different devices should also be intuitive so that the interoperation between different devices goes seamlessly.

A multimedia home networking solution should be content centric so that user can easily navigate through different sources. The application is designed similar to a multimedia player. A cast button is added at the top of the application to make it easier to select cast devices. The content is categorized into 4 sections: music, video, photo and online sources. In Android, since the system provides share intent method for inter-activity communication, an interface is also made to manage share intents from other activities. The selected receiver is designed to be visible to user from everywhere inside the application.

Having all these considerations in mind, the final appearance of the application becomes simple but effective. Figure 10 shows the final design of our application.

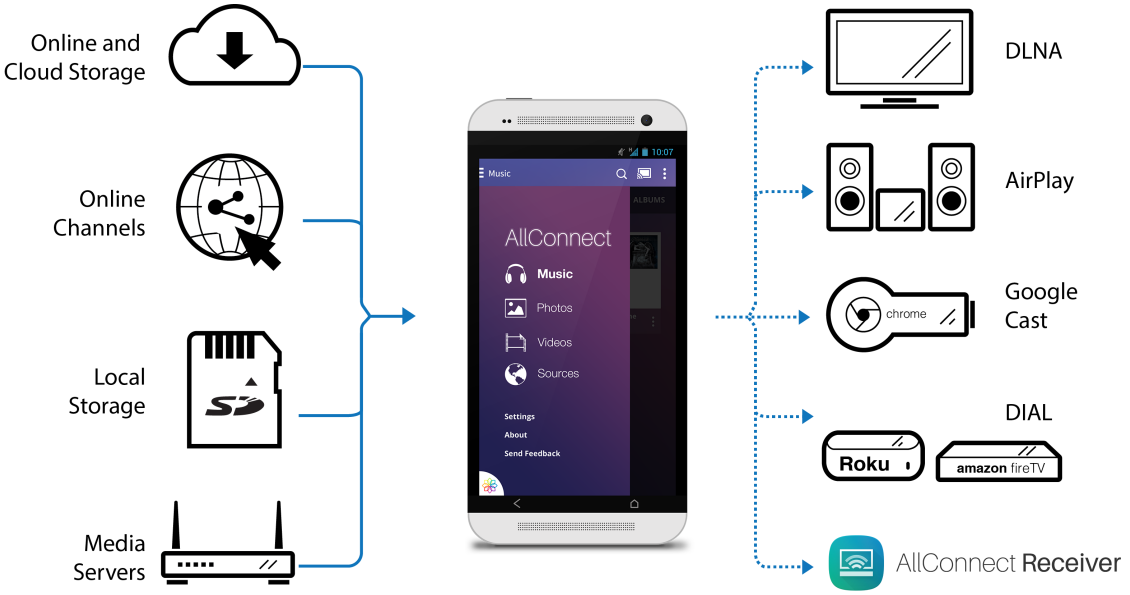

Figure 10: Application UX design

3.4 Features

The Android application we developed can handle most multimedia devices in a typical home networking. It has various features that make it a useful and universal solution for multimedia home networking.

Firstly, the app itself is a multimedia player. All the media stored locally on the phone storage, all the media located in the DLNA media servers can be browsed and played on the phone.

Secondly, the application is fully compatible with AirPlay, DLNA, Chromecast and FireTV receiver devices. All devices can be discovered as renderer devices.

Thirdly and most importantly, the application make the DLNA media server works together with all kind of receivers regardless of protocol used. The app served as a bridge of different multimedia receivers and media sources.

Last, YouTube and other on-line channels like Vimeo and Facebook are supported as media source. These content can be streamed regardless of protocol to all supported receivers that are connected to home network.

3.5 Extensibility

Streambels has embedded a media streaming server for local files and streaming proxy for bridging the gap between online resources and home networking. By using built-in proxy, Streambels is able to share on-line resources from Internet to devices in home networking environment.

New service providers and content providers can integrate home networking support to their product easily by just sending formatted intent to our application following our guideline. The proxy will automatically bridge the gap between internet and home netowrk.

The proxy system enables a huge extensibility possibility, which makes it possible to connect home networking to Internet or Cloud Services.

In the future we could also develop a Software Development Kit (SDK) to make this technology even directly be used by other application builders.

3.6 Test methodology

Software testing is extremely important for a modern IT project. Buggy implementation may kill the product in the very beginning. Through testing we could assure the performance and stability of our product. Before the final releasing to App store, the application needs to be thoroughly tested.

These tests include unit test, integration test and functional test. Unit tests were written while coding, when each class was finished, unit tests would be written for each method. We also set up an continuous integration server so that each time we commit anything to the git repository, full set of unit tests will be executed. If there were any failure in the unit tests, developer will be immediately notified. Integration tests were done in a way that we ensure each function module should work together with other modules in the system. Last, we listed all the possible use cases in paper and prepared a huge media base which contains all kind of media files. With all these preparations ready, manual tests were conducted before the app is finally released in the market.

3.7 Evaluation methodology

The evaluation of this application can be described in 2 categorizes. Experiment should be conducted to evaluate the performance of different standards. Another aspect worth studying is the users' usage and feedback.

3.7.1 Experimental setup

The test environment is set up as shown in Figure **??**. XBMC media receiver is running on MacBook A and Streambels is running on a rooted Android phone. Both A and B are connected to a router C using 802.11 g wirelessly.

With support of both AirPlay and DLNA protocols, XBMC is an open source media center software that can run on different platforms. It is a popular software solution that is widely used in home networks, it has been ported to different platforms including Raspberry PI.

Network Link Conditioner is an application provided by Xcode that runs on Mac to emulate network conditions, such as packet loss, network delay and bandwidth. By changing the configurations, different network conditions can be set to decide the boundary network conditions.

Wireshark is an application an useful developer tool to capture and analyze network traffic. It can be installed on both the sender and receiver side, however, it can only be installed on rooted Android phones to gain the access to network interface on Android. In our setup, Wireshark is installed on MacBook, since we need to adjust network condition on receiver side, and there might be packet loss since some protocols may use UDP protocol.

When testing starts, same content will be streamed to receiver using two different standards, AirPlay and DLNA. The same test will be run several times to get the average statistics. Different parameters will also be tested by using different network condition configurations.
3 parameters are taken into consideration when conducting the tests: bandwidth, packet loss and delay.

3.7.2 User study and feedback collection

Since the product is targeted to Android market and is directly used by end users, user feedback is really important to us for improving the product continuously. Email is used for normal communication between user and our support. A submission window is built inside the application, so that users can easily send feedback to us directly by Email.

There is no perfect program, crashes sometimes happen. Thanks to Google, all crash reports are collected and showed inside developer console. This makes it a lot

easier to track and debug our application.

Inside Streambels, we also integrated Google's Analytics API. The API provided great convenience for us to collect number of users and sessions every day. Other information such as version of operating system, application version, active users helped us to achieve better insights to our users and helped us to market our application to more people in the world.

It is also interesting to see what kind of technologies are most used in their daily life. With the analytics SDK, we could trigger events when user select their receivers. After months of statistics, we have figured out the most popular standards and most popular online channels that user uses.

These valuable information will in turn help our decision making on how we should improve our application. Some of these result will be discussed in chapter 4.

4 Results

After designing, implementing and testing the application. We started several evaluation tests on the streaming performance. Finally we had released the application in Google Play Store in Nov 2013. So far, we have improved the application in many aspects and also brought many new features to it. During the past one year, we have accumulated many useful data and interesting results. In this chapter we will present and analyze those results.

4.1 Performance

In terms of streaming, our solution include two major streaming components. It would be helpful to study and compare which streaming protocol have the better performance while streaming multimedia contents. Moreover, by comparing different streaming types of media, we could investigate which protocol is best suitable for certain type of media.

Two major streaming technology we used in our solution are HTTP streaming and RTSP streaming.

Hypertext Transfer Protocol (HTTP) refers to the protocol used to deliver web pages and images across the Internet worldwide. HTTP is an adopted, open standard and the most ubiquitous mode of delivery on-line. HTTP can be delivered by a variety of web servers, both commercial and open source.

Real Time Streaming Protocol (RTSP) is a network control protocol used in entertainment and communications systems to control streaming media servers. RTSP is used to establish and control media sessions between two points, usually server and player client. Clients of media servers issue VCR-like commands, such as PLAY and PAUSE, to facilitate real-time control of playback of media files from the server. AirPlay uses RAOP, a RTSP-like streaming protocol, for the streaming of iTunes music.

Since we have both protocols implemented in our application. We could compare the performance by streaming the same content to two receivers using different protocol. We selected a mp3 music file and try to stream it to an AirPlay Speaker and an DLNA Speaker, and we used Wireshark running on rooted Android phone to capture the packets in the network.

The result is presented below, the initial packet count is relatively high. This is because there is a lot multicast messages in the network for device discovery. After that, streaming graph shows that after an initial increase in the traffic, network traffic enters a relatively stable state. This is because the TCP protocol reaches the best optimized transmission speed.

Next, we added packet loss in the network and compare the influence to the two

streaming process. The result is shown in Figure below:

4.2 Statistics

Through one year's releasing, our application have reached 895000 downloads from 223 countries all around the world, with around 10000 daily active users. So far, the average rating of our app is 3.9 out of 9934 ratings. The application turns out to be popular in countries like France, United States, Germany, United Kingdom and Brazil.

- Totally more than 895000 downloads
- Used by people from 223 countries
- 10000+ daily active users
- 1,241,074 visits to home page
- Average rating 3.9, 9934 user gave rates
- On-line for 16 months

Table 9: Receiver type statistic

Device type	Total selection	Percentage
AirPlay device	637446	39.72
DLNA device	460139	28.67
Phone speaker	353474	22.03
Chromecast device	147333	9.18
FireTV device	6368	0.40

4.3 User study

- What information we can get back from users
- User behavior/ statistics
- Improve the application accordingly
- Strategies for decision making

Write result here.

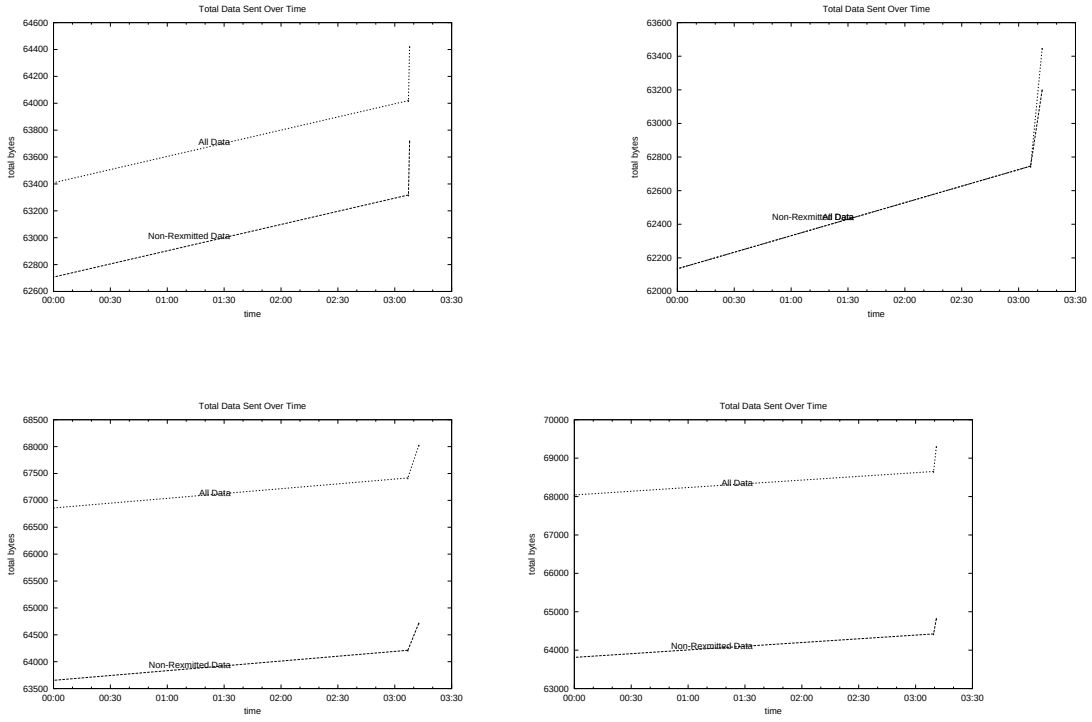

Figure 11: AirPlay streaming performance in terms of packet loss

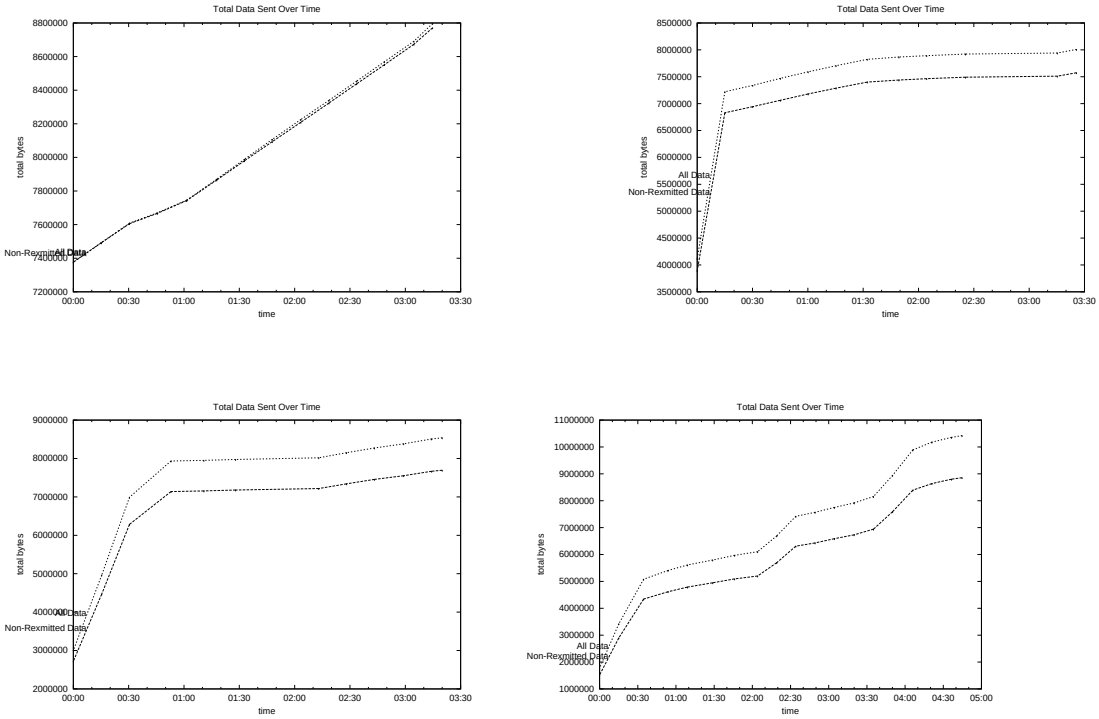

Figure 12: DLNA streaming performance in terms of packet loss

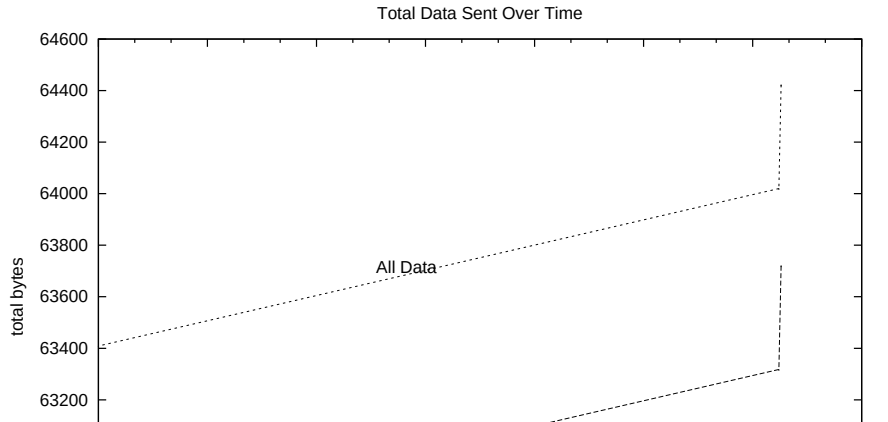

Figure 13: AirPlay traffic data chart

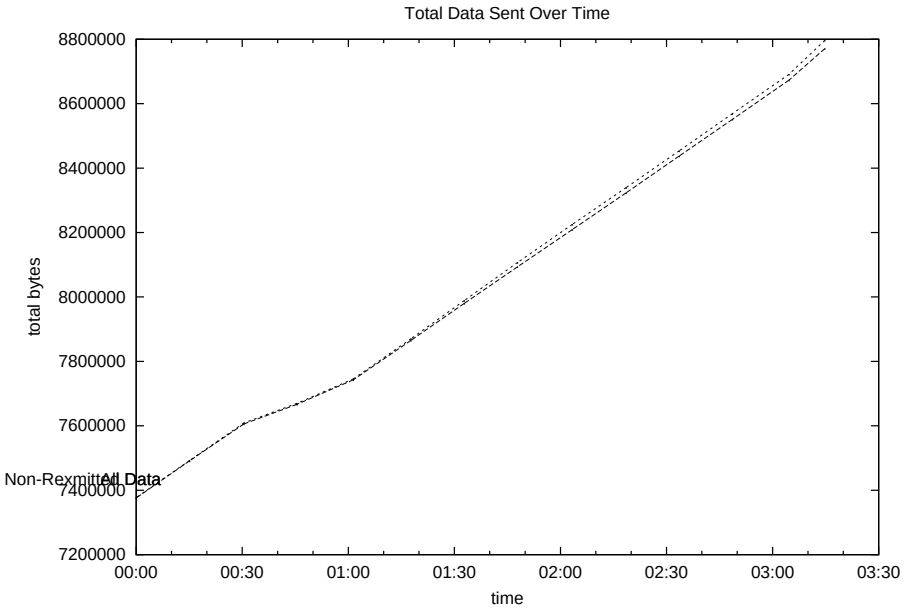

Figure 14: DLNA traffic data chart

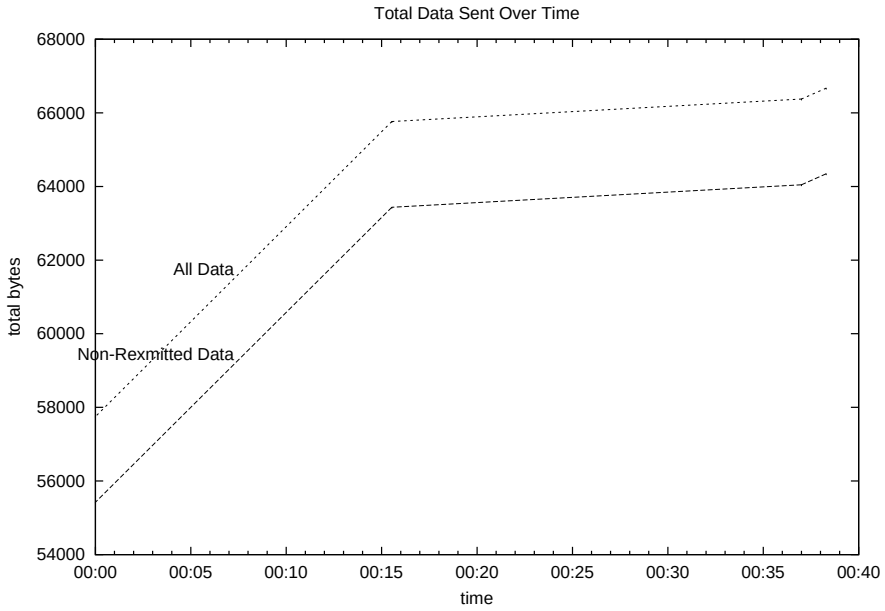

Figure 15: AirPlay bad network traffic data chart

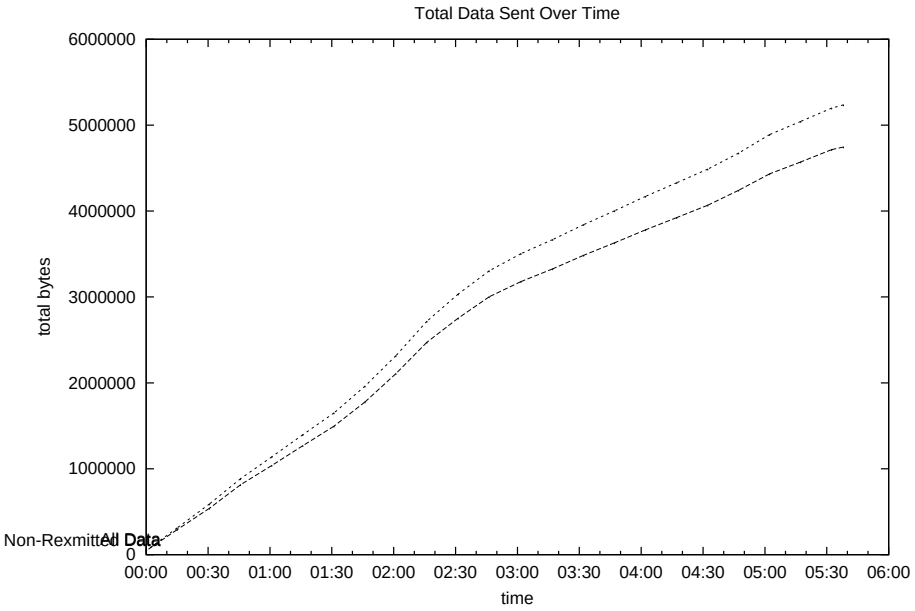

Figure 16: DLNA bad network traffic data chart

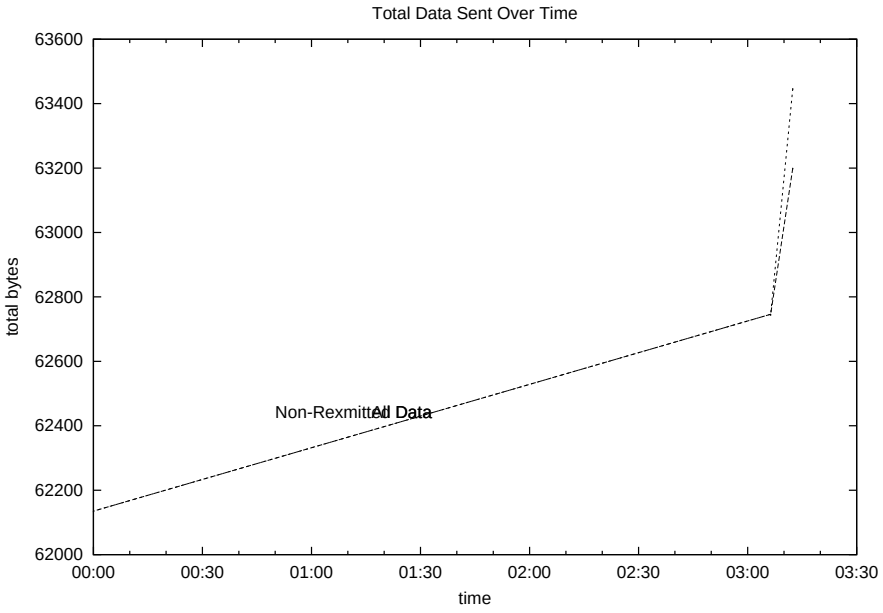

Figure 17: AirPlay bad network traffic data chart

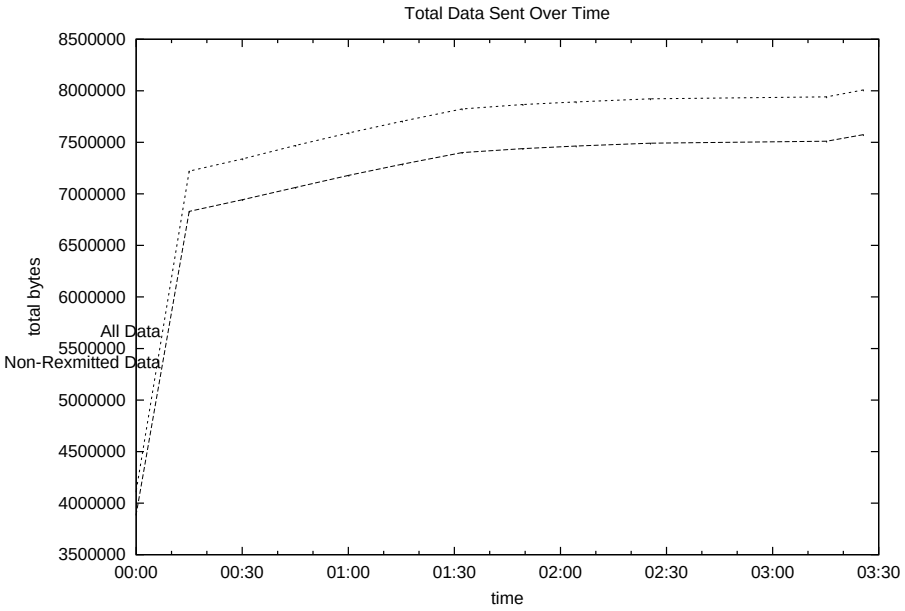

Figure 18: DLNA bad network traffic data chart

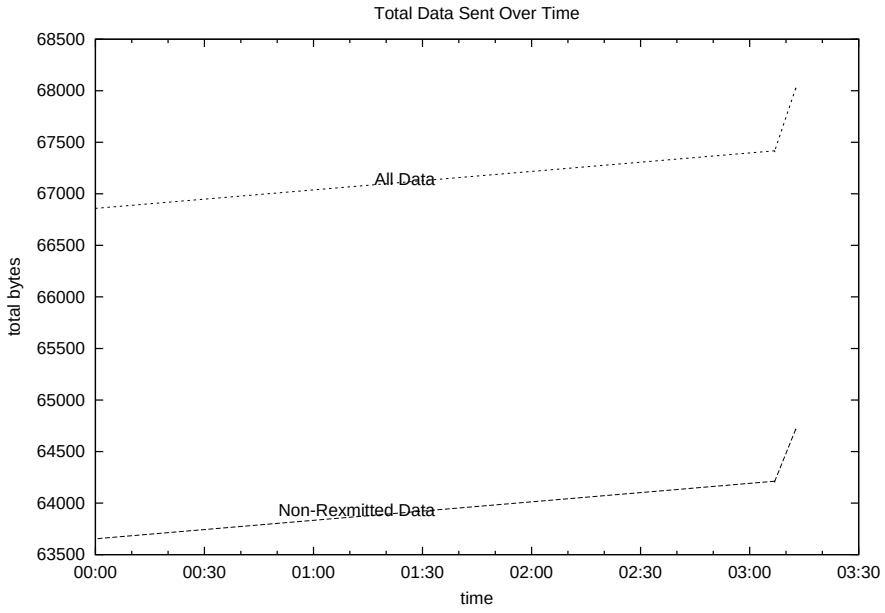

Figure 19: AirPlay bad network traffic data chart

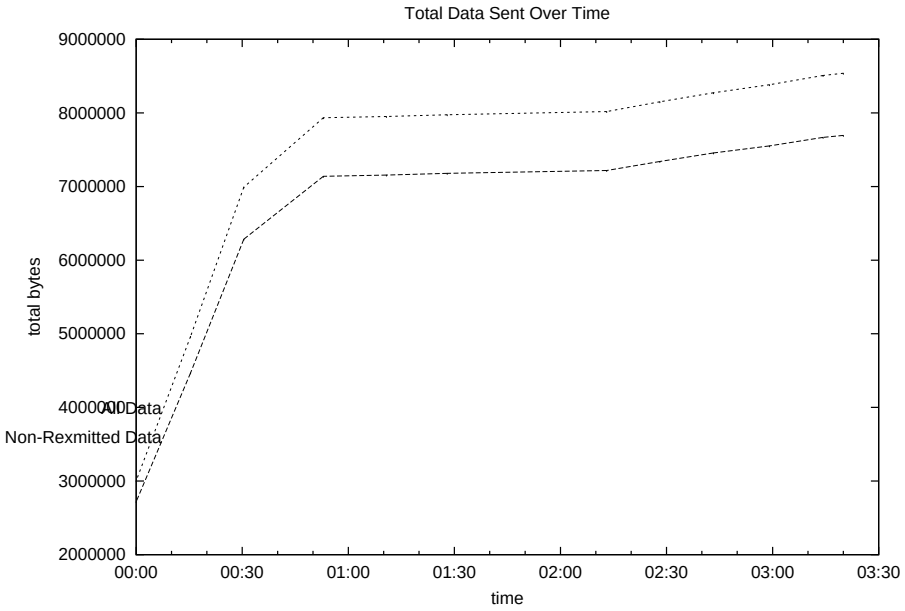

Figure 20: DLNA bad network traffic data chart

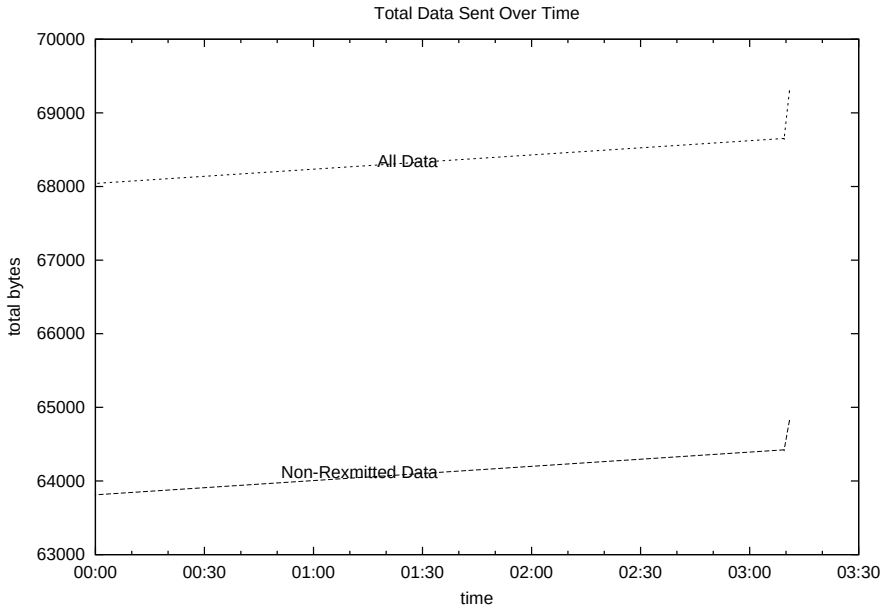

Figure 21: AirPlay bad network traffic data chart

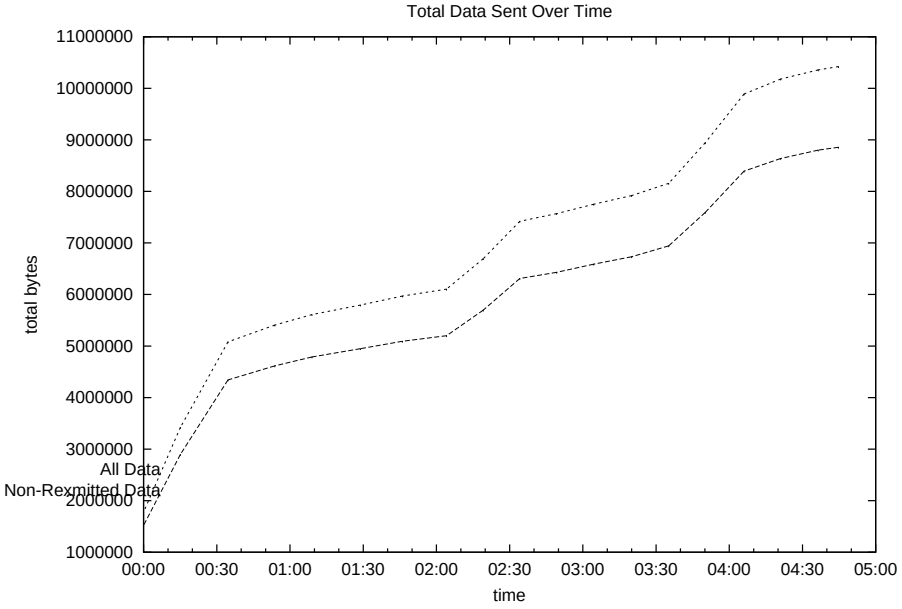

Figure 22: DLNA bad network traffic data chart

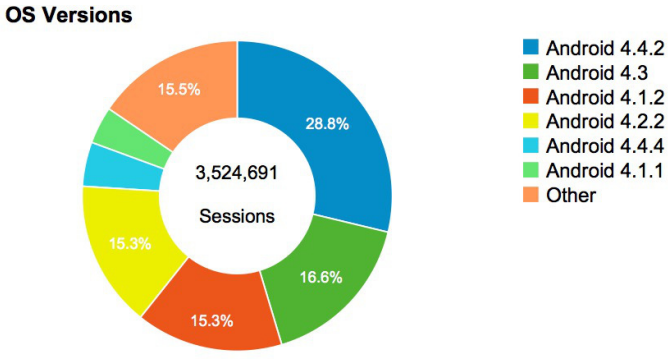

Figure 23: Android versions

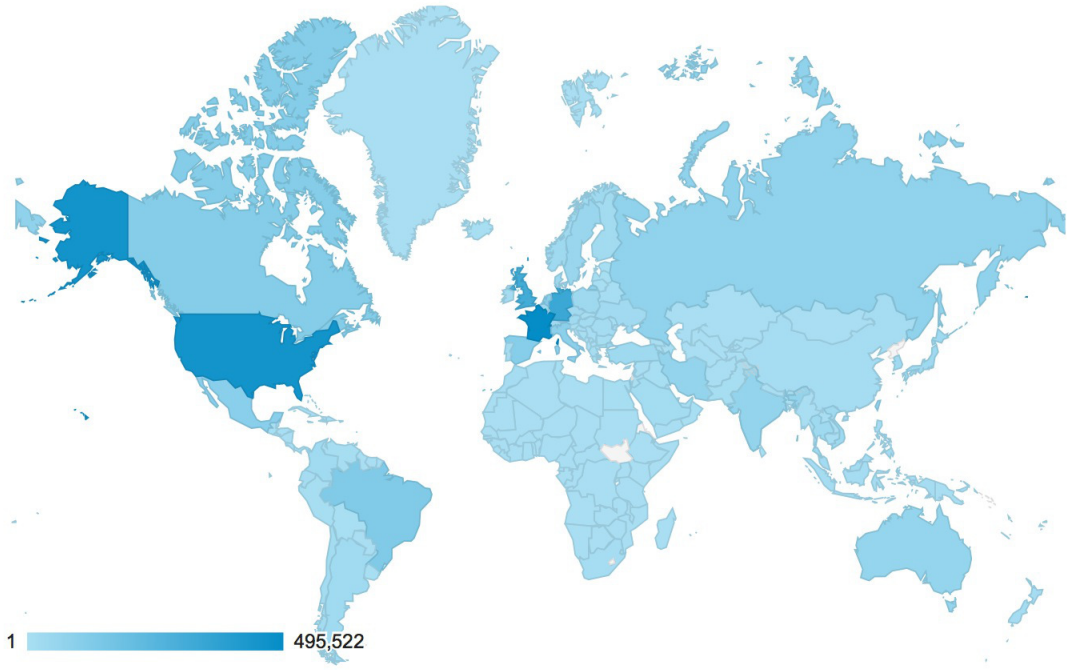

Figure 24: User world map

5 Discussion

Write discussion here.

5.1 Further development

Development of a universal streaming SDK is essential for different manufacturers and on-line content providers. (Connect SDK by LG)

Multi-room and Multi-Session support, stream video and audio to different devices with synchronization.

Kernel level optimization, multimedia transcoding, file system and network layer optimization.

More platforms.

5.2 Future of multimedia home networking

- Multiple standards may still exist for quite a long time
- More devices will support streaming technologies, there will be more interactions between devices like mobile phone and TVs
- Cloud and local storage will both work and make it easy for users to access their multimedia content
- Internet of things, smart home

[1] [9] [16] [17] [5] [18] [7] [3] [19]

References

[1] Ratul Mahajan David Wetherall W. Keith Edwards, Rebecca E. Grinter. Advancing the state of home networking. *Communications of the ACM*, 54(6):63–71, June 2011.

[2] Xinhua Feng. Home networking. Technical report, Ohio State University, 2000.

[3] Sandy Teger and System Dynamics Inc. David J. Wakes. End-user perspectives on home networking. *IEEE Communications Magzine*, pages 114–119, April 2002.

[4] Bruce Nordman. Summary of analysis of communication link technologies. Technical report, National Lab Buildings Energy Efficiency Research Projects, September 2012.

[5] Charlie Tai Mark D. Wood Brent A. Miler, Toby Nixon. Home networking with universal plug and play. *IEEE Communications Magzine*, pages 104–109, December 2001.

[6] Wouter van der Beek Jeffrey Kang John Ritchie, Thomas Kuehnel. Upnp av architecture. Technical report, UPnP Forum, December 2010.

[7] Eui-Hyun Paik Kwang-Rog Park Yeon-Joo Oh, Jung-Tae Kim. The dlna proxy system architecture for sharing in-home media contents via internet. *ICACT2006*, pages 1855–1858, 2006.

[8] Digital Living Network Alliance. Dlna guidelines. Technical report, Digital Living Network Alliance., March 2014.

[9] Airplay specification.

[10] Marc Krochmal Stuart Cheshire. Multicast dns, February 2013.

[11] Inc. Netflix. Discovery and launch protocol specification. Technical report, Netflix, Inc., 2014.

[12] WiFi Alliance. Wi-fi certified miracast: Extending the wi-fi® experience to seamless video display. Technical report, WiFi Alliance, September 2012.

[13] 11 2013.

[14] 1 2014.

[15] Cling - java/android upnp library and tools.

[16] Takayoshi Okodo Yohei Tsuchida Tomoyuki Yamaguchi-Takahiro Murooka Kimio Oguchi, Kunio Tojo. Next generation home networking and relevant technologies. *Proc. of SPIE*, 5626, 2005.

[17] Hoon-Ki Lee Eui-Hyun Paik Kwang-Roh Park Jung-Tae Kim, Yeon-Joo Oh. Implementation of the dlna proxy system for sharing home media contents. *IEEE Transactions on Consumer Electronics*, 53(1):139–144, February 2007.

[18] Bill Rose. Home networks: A standards perspective. *IEEE Communications Magzine*, pages 78–85, December 2001.

[19] Yueh-Min Huang Chung-Sheng Li. Upnp ipv4/ipv6 bridge for home networking environment. *IEEE Contributed Paper*, August 2008.