

Universal Plug and Play (UPnP)

- [Overview](#)
 - [Procedure involving in implementation of the GUPnP Client:](#)
- [UPnP Applications:](#)
 - [xdiscovery](#)
 - [XCal-Device module](#)
- [API Specification](#)

Overview

Universal Plug and Play (UPnP) is a set of networking protocols that permits networked devices, such as personal computers, printers, Internet gateways, Wi-Fi access points and mobile devices to seamlessly discover each other's presence on the network and establish functional network services for data sharing, communications and entertainment. UPnP uses -

- To discover the devices in the network.
- To gather additional information from the devices in network.(e.g receiver id, fog url, etc)
- To use the gathered additional information obtained for other running processes.

Universal Plug and Play (UPnP) is an implementation of the generic GUPnP framework for device discovery and using services from control points. GUPnP is a library for implementing both UPnP clients and services using GObject and LibSoup. It allows for fully asynchronous use without using threads and so cleanly integrates naturally into daemons, and supports all of the UPnP features.

The GUPnP framework consists of the following libraries:

- GSSDP implements resource discovery and announcement over SSDP.
- GUPnP implements the UPnP specification
 - Resource announcement and discovery.
 - Description.
 - Control.
 - Event notification.
 - Presentation (GUPnP includes basic web server functionality through libsoup).

Procedure involving in implementation of the GUPnP Client:

- **Finding Services:** Initialize GUPnP and create a control point targeting the service type. Then we connect a signal handler so that we are notified when services we are interested in are found.
- **Invoking Actions:** GUPnP has a set of methods to invoke actions where you pass a NULL-terminated varargs list of (name, GType, value) tuples for the in-arguments, then a NULL-terminated varargs list of (name, GType, return location) tuples for the out-arguments.
- **Subscribing to state variable change notifications:** It is possible to get change notifications for the service state variables that have attribute sendEvents="yes".

UPnP Applications:

xdiscovery

- The UPnP daemon xdiscovery runs in system on MoCA interface.
- While starting the xdiscovery service, we create a UPnP control point with target device type as urn:schemas-upnp-org:device:BasicDevice:1.
- Once UPNP control point is added and active in the network, It searches for the target device types in the network.
- The device-proxy-available Signal is emitted when any device which matches target types is found in the network and the event is handled in a callback routine.
- The discovery messages contains a few information about the device , Usually the device descriptions is in an XML file and it includes vendor specific information like Manufacturer, Model Name, Serial No, URL to vendor specific web sites etc. Also it lists the services as well if any.
- Each service description includes a list of commands to which the service responds with its parameters and arguments.
- The UPnP control point subscribes for the notification if any of the service variable changes during run time. After the device is discovered in the network, the UPnP control point send an action by calling the GUPnP API gupnp_service_proxy_send_action() to retrieve device info and its capabilities. After that it add or update the device in device list using its serial number.

Gateway set up : Internally UPnP uses a script /lib/rdk/gwSetup.sh to set up the gateway in client devices. Here we sets up the routing table, DNS setting etc.

UPnP Device Information in JSON format

```
{
  "sno": "PAD200067027",
  "isgateway": "yes",
  "gatewayip": "169.254.106.182",
  "gatewayipv6": "null",
  "hostMacAddress": "84:e0:58:57:73:55",
  "gatewayStbIP": "69.247.111.43",
  "ipv6Prefix": "null",
  "deviceName": "null",
  "bcastMacAddress": "84:e0:58:57:73:59",
  "recvDevType": "X1",
  "buildVersion": "66.77.33p44d5_EXP",
  "timezone": "US/Eastern",
  "rawoffset": "-18000000",
  "dstoffset": "60",
  "dstsavings": "3600000",
  "usesdaylighttime": "yes",
  "baseStreamingUrl": "http://127.0.0.1:8080/videoStreamInit?recorderId=P0118154760",
  "requiresTRM": "true",
  "baseTrmUrl": "ws://127.0.0.1:9988",
  "playbackUrl": "http://127.0.0.1:8080/hnStreamStart?deviceId=P0118154760&DTCP1HOST=127.0.0.1&DTCP1PORT=5000",
  "dnsconfig": "search hsd.tvx.comcast.net;nameserver 75.75.75.75;nameserver 75.75.76.76;nameserver 69.252.80.80;",
  "hosts": "69.247.111.43 pacexglv3;",
  "systemids": "channelMapId:1901;controllerId:3415;plantId:0;vodServerId:70001",
  "receiverid": "P0118154760"
}
```

XCal-Device module

XcCal-Device module is responsible for getting the following device discovery information.

- Read the upnp configuration details from configuration file.
- Getting the input for different gateway to populate all the service variables.
- It act like a server & whenever requested it will give the services details such as ipv6 ip address, receiver Id, etc.
- Once the xcal-device receive the services details, it will create a UPnP object and start publishing the UPnP data.

API Specification

Refer the link [UPnP API Documentation](#) for more implementation details of XCal-Device and XCal-Discovery used in RDK.