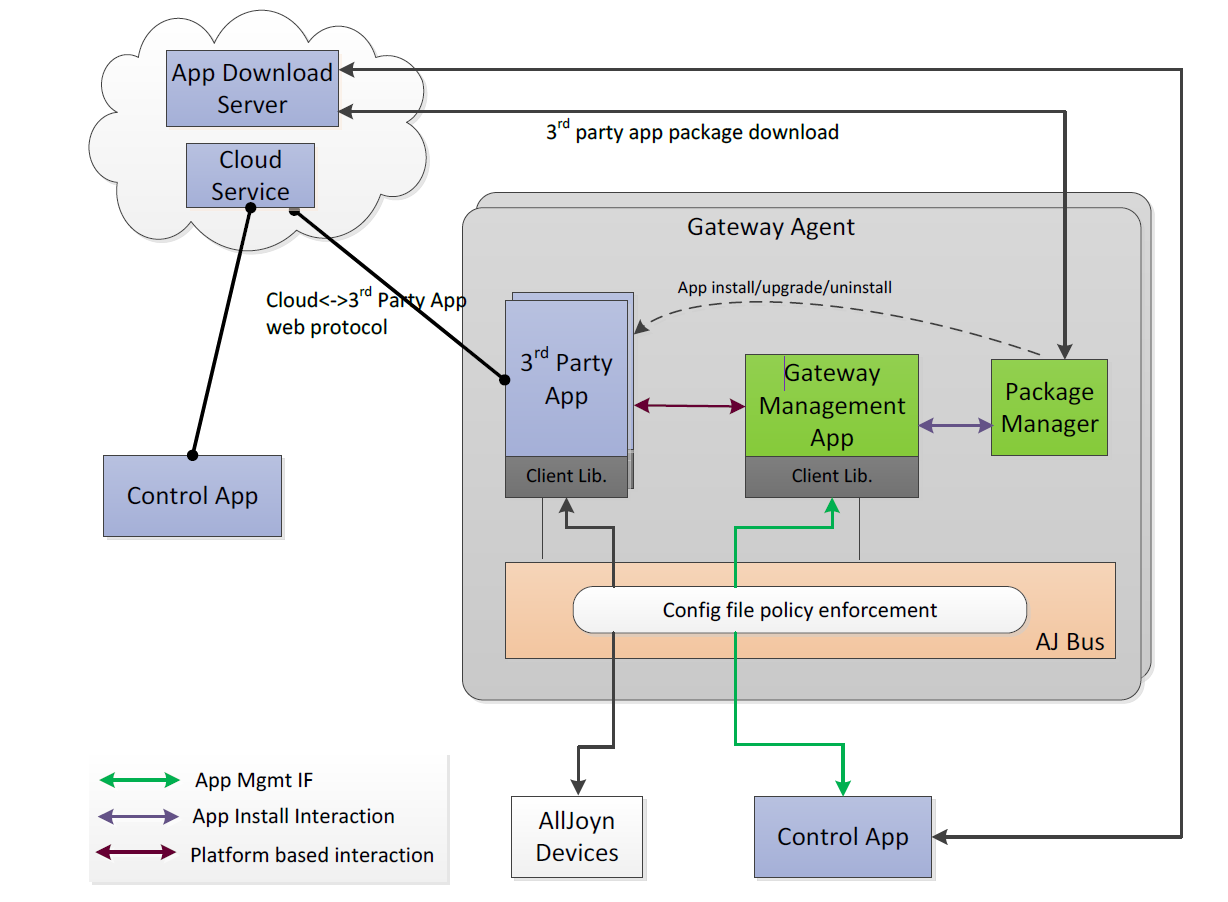
DRAFT - AllJoyn Package Format - DRAFT

Jim Howard - jhoward@affinegy.com  
10 October 2014

**1.0 Gateway Agent Overview**

The AllJoyn (AJ) Gateway Agent (GA) includes an 'App Management Function', described in the Gateway Agent High Level Design Document (HLD), Chapter 3. The GA includes a Gateway Management App (GMA) which in turn interfaces with a Package Manger (PM) component to facilitate installation of third party AJ connector applications.



The HLD provides an overview of the GA's application management design:

*The Control app provides a UI for initiating the app install/upgrade and provides*

*download URL to the Gateway Management app for downloading app packages*

*from the App Download server.*

*The Gateway Management app exposes an App Management interface to enable*

*the Control app to manage third-party apps*

*...*

*The Gateway Management app and the Package Manager app will be packaged as*

*OpenWRT packages and will be installable using the OpenWRT package manager*

*(opkg). A Gateway Agent device may be packaged only with Gateway Management app*

*and Package Manager app and may not have any third-party apps installed initially. In*

*this case, all third-party apps will get installed via the App Management interface.* (HLD 3.1)

The HLD provides this description of the function of the package manager:

*Due to security concerns, it has been decided to not use OpenWRT opkg package*

*manager for installing third-party apps. A separate Package Manager (PM) application is*

*supported on the Gateway Agent for installing/upgrading/uninstalling third-party apps.*

*The Packet Manager app performs the download of the app package and verifies the*

*signature before installing the package. The exact nature of the Package Manager app*

*(AllJoyn vs. nonAllJoyn app) is considered implementation detail and is outside of this*

*HLD scope.* (HLD 3.1)

**2.0 Package Manager Functionality**

**2.1 Overview**

HLD figure 3.2.1 summarizes the desired PM third party app install functionality as three steps:

1) Download [*third party connector*] app package using HTTPS GET.

2) Verify the signature of the application package.

3) Install the new app package.

App upgrade and app uninstall functions are similar.

Installed AJ applications are each assigned a unique Unix user Id by the GM. All installed AJ applications share the same Unix group id (HLD 3.3.2.1).

**2.2 Package Manager Interfaces**

The PM exposes two functions to the App Manager:

*InstallApp: Results in the download of the third-party app package from the URL provided*

*and installs the package on the Gateway Agent in a new user account. The*

*package signature is verified before installing the app. Also used for upgrading an*

*existing app to a newer version.*

*UninstallApp: Results in uninstalling the specified third-party app from the Gateway Agent.* (HLD 3.3.2)

The communications channel between the Application Manager and the PM are not defined by the HLD. The PM is required to have a command line interface (CLI) (HLD 3.1.2).

The desired communications channel between the GM and PM is via an Alljoyn Bus, the details of which are to be determined.

**2.3 Package Manager Limitations**

Note that the PM merely extracts files from a compressed archive and places them in a well known device directory designated by the Gateway Manager Application, currently /opt/alljoyn/apps.

The PM will **not** execute any code or scripts contained in the downloaded package.

**2.4 Package Download Security**

Downloads will be allowed only from HTTPS web servers using web server certificates stored in the Gateway device certificate store.

Downloaded AJ applications must be digitally signed by a trusted authority's private RSA signing key, digital signatures must be verifiable with a corresponding public RSA key.

**3.0 AJ Package Format Requirements**

The HLD directs that an AJ package format be developed:

*Third-party app packages should be made available in an AllJoyn package format. The*

*app package file should include following in a single binary:*

*- Manifest file providing metadata for the app*

*- App package files* (HLD 3.3.3)

3.1 Package Components

The AJ Package should contain at a minimum the following components (see HLD 3.3.3):

Application binaries and resources  
required libraries not known to be present on standard device installation  
 AJ Manifest

All components should be contained in an application folder. The application manifest should be placed in the top level of this folder.

3.2 AJ Package Format

**3.2.1 Assumptions**

a) The public key of a trusted signing certificate will be installed in the OpenWRT hardware, either prior to shipment of the hardware or installed along with the initial AJ Gateway Agent (see HLD 3.1).

b) Only trusted entities will be able to sign Alljoyn packages downloadable through the PM.

c) The Package Manger verifies authenticity of downloaded third party AJ applications using RSA signature verification. It does **not** provide any malware scanning capability of downloaded packages. In particular, the PM cannot determine if the third party application itself might download content from untrustworthy sources.

c) The PM implementation creates the Unix user specified in the InstallApp interface, and deletes this user in during the UninstallApp function. (HLD 3.3.2)

d) The package manager must have sufficient operating system privileges to perform HTTPS downloads, create and delete operating system users, create directories in the Gateway application directory, and assign execute permissions to the downloaded application.

e) The GM will stop the application processes before performing an application upgrade or uninstall.

**3.2.2 Package Format**

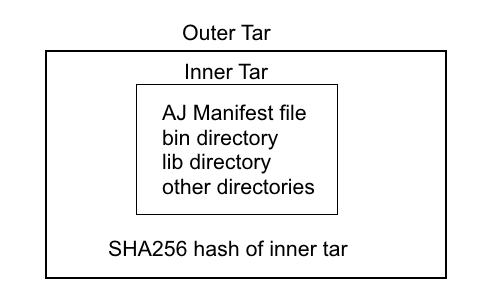
The AJ package format consists of two components, an 'inner tar' and an 'outer tar'

The top directory level of an AJ third party application should contain only the application manifest. Other application components should be placed in subdirectories.

Table 1 - Inner Tar Contents

|  |  |
| --- | --- |
| Component | Remarks |
| Application Manifest file | HLD 3.3.3.1 |
| 'bin' directory | Application executables |
| 'lib' directory | Library files |
| Other sub directories as required |  |

The inner tar is compressed and then signed with a trusted private key. The inner tar is then created by creating an 'outer tar' archive combining the inner tar and the digital signature. This 'outer tar' is the downloadable AJ package.



**3.2.3 Preparing an Alljoyn Package**

For this discussion assume we desire to create an AJ package for an application called 'ExampleApp'. As the package creator we must have access to the private key of the trusted signing certificate of entity distributing this application. The AJ Gateway device must have the corresponding public key.

**3.2.3.1 Create the inner tar archive**

Create a directory with the Application Name that will be passed to the PM by the GM as the 'App package name' (HLD table 3-12), in this case 'ExampleApp'. This directory must contain the components listed in Table 1.

Compress this folder as a Linux tar gz archive with permissions preserved, giving it the 'App package name'.tar.gz (HLD 3.3.2.1).

tar cvzpf ExampleApp.tar.gz ExampleApp

**3.2.3.2 Compute the digital signature of the inner tar archive.**

Compute a SHA256 digital signature of the inner tar file, signing using a closely held private key of the signing entity. The OpenSsl command line with a private key called ourkey.private.pem, the command to sign the inner tar is :

openssl dgst -sha256 -sign ourkey.private.pem \  
-out ExampleApp.tar.gz.sha256 ExampleApp.tar.gz

The Openssl command line to verify that ExampleApp.tar.gz.sha256 was computed correctly is:

openssl dgst -sha256 -verify ourkey.public.pem -signature \ examplepackage257.tar.gz.sha256 examplepackage257.tar.gz

**3.2.3.3 Creating the outer tar archive**

The working directory now contains two files of interest, the inner tar and the signature file: ExampleApp.tar.gz and ExampleApp.tar.gz.sha256 .

To create the final downloadable package combine these two files into a tar archive (no compression needed):

tar cpf ExampleApp.tar ExampleApp.tar.gz \ ExampleApp.tar.gz.sha256

ExampleApp.tar is now ready for download by the Package Manager.

**4.0 Package Manager Execution Flow**

**4.1 Install/upgrade**

The PM performs the following steps in executing the InstallApp function (HLD 3.2.1):

a) Create a temporary directory in /tmp/aj

b) Downloads package into temporary directory using HTTPS

c) Extracts the inner tar and signature hash, preserving permissions. Performs RSA verification of the compressed application directory.

c) If upgrade flag set delete the current application directory.

d) If not an upgrade, create the an Operating system user with the ID provided by the GM, the common group ID, and set the user name to the application name.

e) Extract the inner tar archive to the Gateway Apps Directory. Change the ownership of the downloaded app directory to the application Id.

**4.2 Uninstall**

The PM performs the following steps when uninstalling an application(HLD 3.2.3):

a) Deletes the application operating system user

b) Deletes the application directory

**5.0 Rollbacks of failures - TBD**