AllJoyn Security Manager Sample Application in Windows

January 13th, 2015

Contents

1 Introduction

1.1 Purpose

This document is the design document, accompanying the delivery of a sample AllJoyn Security Manager in Windows and associated files and source code. This document is closely aligned with the Technicolor's AllJoyn Security Manager sample app documentation (written on December 19, 2014).

1.2 Scope

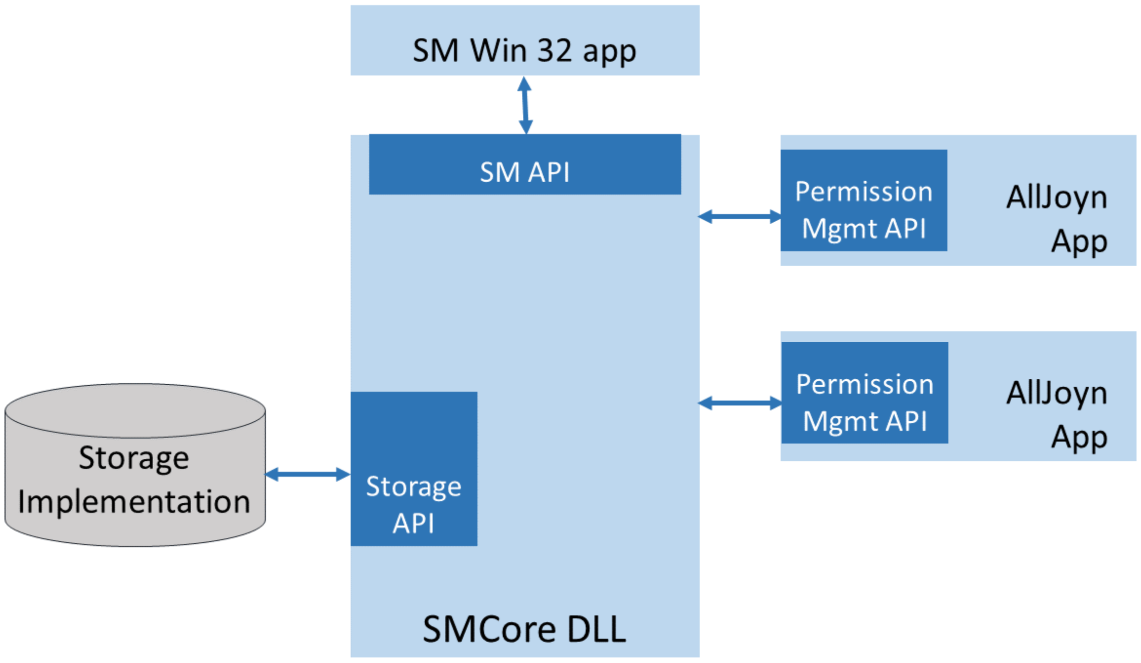
This document is intended for third-party developers who are interested in developing an AllJoyn Security Manager in Windows[1].

[1] Windows operating systems by Microsoft

2 Security Manager Architecture

2.1 Architecture & Description

The Security Manager app (SM app) is designed to run on Windows 7 devices. The diagram below shows how key modules interact through which API.



SM app is built on top of *SMCore DLL*. This DLL handles all the core security tasks defined in AllJoyn Security 2.0 HLD Update 1 Revision 4. In particular, the DLL performs two key tasks:

- **Perform security tasks with *on-line* AllJoyn applications:** The DLL supports claiming an application, installing, updating, and removing policies and certificates from applications. These tasks are accomplished via respective calls to the AllJoyn PermissionMgmt interface. Note that the DLL currently doesn't support tasks that involve an application that is not online at the moment. We envision that there needs to be a service component that ensures the delivery of policies and certificates to respective destinations in a secure manner.

- **Persist application states and security data:** The DLL will persist the state of the applications it manages and the security data including the information needed to authenticate the admin and the keys used to generate certificates. Similarly to the Security Manager Library contributed by Technicolor, the DLL will also provide a Storage API in order to abstract how and where the data is stored by the Manager. In the current implementation, SQLite is employed but we envision that this will be replaced with the Windows Secure Key Store, which is currently being implemented by Microsoft.

2.2 The Security Manager API (SM API)

Similarly to the Security Manager Library from Technicolor, the DLL exposes the following nine key API:

- GetApps(): Return an iterator of a collection of applications and their states
- ClaimApplication(): Claim an application for a given bus name. This requires that (1) the application is claimable and (2) the admin accepting the application's manifest. If these requirements are met, the SM DLL issues the identity certificate of the app (using the app's public key and GUID) and sends it to the application to store. Once claimed, the security information of the app can only be manipulated by the admin (who is a certificate authority of the app). Factory reset will, however, erase any security information on an application.
- CreateGuild(): Create a new guild with a given ID (and optionally a description)
- RemoveGuild(): Delete a guild that corresponds to a given guild ID
- GetGuilds(): Return an iterator of a collection of guilds and their states
- AddMemberToGuild(): Install a membership certificate of a given guild on the application identified by a given bus name. The membership certificate includes the permissions, which should be checked by the Security Manager. An application will not get more rights within a specific guild than allowed by its own manifest.
- DeleteMemberFromGuild(): Delete a membership certificate of a given guild from the application identified by a given bus name
- InstallPolicyToApp(): Install a policy to the application identified by a given bus name
- GetPolicyOfApps(): Get the policy from the application identified by a given bus name

3 Security Manager Sample App

3.1 Supported Functionalities

Although the SM DLL handles tasks of generating membership certificates and installing and revoking them, it is up to the admin to decide which applications can join a guild and which policies should be applied to members of a guild. A person who first claims a security manager app can become the admin of the security manager app. A security manager app provides UI through which an admin can manage AllJoyn applications' security through guilds. The security manager sample app in Windows supports the following eleven functionalities similar to Technicolor's security manager sample app:

- List all claimable AllJoyn apps
- Claim an app: In the current implementation, we only support the method 2.3.1.1 in the Security 2.0 HLD ("claim factory-reset device without out-of-band registration data").
- List all claimed app
- List all active apps
- Create a guild: The admin supplies a guild name and the security manager app creates a GUID and binds the GUID with the guild name.
- Remove a guild
- List all guilds
- Install a membership certificate of a given guild and bus name: In the current implementation, we do *not* support enabling delegate flag on.
- Delete a membership certificate of a given guild and bus name
- Install a policy of a given bus name
- Get the policy of a given bus name

Not supported functionalities are

- Add a guild equivalent certificate to an application (2.3.8 in the Security 2.0 HLD)
- List revoked certificates (2.3.9 in the Security 2.0 HLD)

3.2 Security Assumptions

Although the SM DLL handles various tasks of generating membership certificates and installing and revoking them, it is up to the admin to decide which applications can join a guild[2] and which policies should be applied to the members of a guild. In the current version of the security manager app in Windows, we make the following assumptions (similar to Technicolor's Security Manager Library):

- One admin: We assume that there is only one admin authenticated by a shared secret (e.g., password) to the security manager app. The chosen secret is used to generate a pair of a public key and a private key used by the app to generate membership certificates.

- One guild: All applications claimed by the security manager app are put into one guild after it is created by the admin. Although allowing an app to join multiple guilds provides more flexibility, this requires the security manager app to handle policy conflicts amongst multiple guilds and this is not implemented in the current version. However, the current design has a stub that can be used to plug in a policy checker to support multiple guilds in the future.

- No policy manipulation: A default policy is used, whereby all claimed applications are allowed to access all signals and interfaces on the Bus.
- All-or-nothing permissions. If the administrator accepts the manifest of an application during the claiming process, the admin will have access to all the signals and interfaces it requested in its manifest. If the administrator rejects the manifest, the claiming process will be aborted.

3.3 Dates

- Design doc complete: 1/23/2015
- Code check-in: 2/18/2015
- Code Complete: 2/27/2015

3.4 Installing the Sample App

TBA

[2] Guild is defined as a logical grouping of devices, applications, and users per Security HLD version 1.4.