technicolor

Security review of the AllJoyn™ Security 2.0 Feature

R&I – Security and Content Protection

September 29, 2014

Revision 1.0

technicolor

Security Laboratories

CONFIDENTIAL

Table of Contents

1. Introduction

1.1 Purpose and scope

This document is a review of the AllJoyn™ Security 2.0 Feature High-Level Design Document [REF_01]. It is still a work in progress as [REF_01] is also still a draft.

Without any prior knowledge on AllJoyn™, the Collaboration Meeting Notes [REF_02] on AllSeen Alliance wiki definitely helped us fill the gaps. This additional source of information gives insights about the designer's views and choices and also an idea about the evolutions that could come up in next revisions of the main document.

1.2 Revision history

Revision	Author	Date	Change Log
1.0	MR	Sept 25, 2014	Initial creation The review is based on: - AllJoyn™ Security 2.0 Feature High-Level Design Document Rev 1 Update 2 [REF_01] - Allseen Alliance CORE working group - Security design discussion [REF_02]

1.3 Authors and Reviewers

Name	Initials	Function
Marc Rivière	MR	R&I Security
Charles Salmon Legagneur	CSL	R&I Security
Antoine Monsifrot	AM	R&I Security
Alain Durand	AD	R&I Security

1.4 Acronyms

Acronym	Definition
CA	Certificate Authority
GUID	Globally Unique Identifier
ISO	International Standards Organization
N/A	Not-Applicable
OID	Object Identifier
RTC	Real Time Clock
TEE	Trusted Execution Environment (GlobalPlatform™ terminology)

1.5 References

Reference and Link	Date	Version	Title
[REF_01]	Sept 8, 2014	Rev 1 Update 2	AllJoyn™ Security 2.0 Feature High-Level Design Document Rev 2 Update 2
[REF_02]	Aug 12-14, 2014	N/A	Allseen Alliance CORE working group - Security design discussion
[REF_03]	Sept 4, 2013	N/A	Hundreds of Camera Feeds for Home Security, Baby Monitoring Were Hacked
[REF_04]	July 3, 2012	N/A	Webinar: AllJoyn for Game Developers - YouTube

2. AllJoyn™ Security 2.0 Feature High-Level Design review

This chapter lists issues, remarks and questions about the AllJoyn™ Security 2.0 Feature High-Level Design Document [REF_01]. A distinction has been made between security issues and other issues.

2.1 Security issues

All security issues and concerns have been categorized into three levels of importance (critical, major and minor) and in three types (unclear, missing and extra). An issues is considered as "unclear" category when the subject is discussed but contradictions remains, unclear terminology is used or more details are needed. "Missing" issues are issues that concern subjects that should be developed in the document. The "extra" category is used for issues that would have more to do with a low-level document if such a document existed (future improvements, additional features, implementation considerations, team organization).

2.1.1 Critical security issues

2.1.1.1 Critical security issues – Unclear

2.1.1.1.1 No compliance commitment to security 2.0 feature

[ISS_SEC_CRI_UNC_01] **The document shall state that compliance to the Security 2.0 Feature is mandatory for any AllJoyn™ applications even when they run on small devices with limited resources. A simplified mode of operation shall be defined for small devices (cf [ISS_SEC_CRI_MIS_04] and [ISS_SEC_CRI_MIS_05]).**

This breaks the compatibility with existing device and applications but it has to be done. The Internet of Things/Everything (IoT/IoE) protocols are often criticized for not paying enough attention to the security challenge they are facing. Today device makers and application developers expect that AllJoyn™ framework will resolve connectivity issues. However, in the future, they will also want it to guarantee the confidentiality and privacy of their clients because their responsibility is engaged with penalty at stake [REF_03].

The opportunity to make AllJoyn™ secure by default shall not be missed. This will avoid the adverse publicity generated when hundreds of AllJoyn™ devices get hacked and allow to use the high level of security of AllJoyn™ as an asset against competitors.2.1.1.1.2

[ISS_SEC_CRI_UNC_02] **To enforce the security on the bus, unclaimed devices must not use or provide any service or data to any other application. They must only accept or initiate connection in order to be claimed by a Security Manager. Standard Applications shall reject any connection attempt from unclaimed devices.**

This is the condition to make AllJoyn™ connections secure by default.

[ISS_SEC_CRI_UNC_03] **The specification shall explain how an unclaimed application is discovered as such by other applications.**

[ISS_SEC_CRI_UNC_04] **In the Section 2.1 (Overview), the following sentences shall be clarified.**

> Section 2.1 Overview
>
> *"The AllJoyn™ Core Permission Management component does all the enforcement including the concept of mutual authorization before any message action can be taken."*

This is actually true only if all the applications strictly implement the Security 2.0 feature and if a Security Manager has previously claimed the applications and defined some policies.

> Section 2.1 Overview
>
> *"The Security Manager is optional service …*
>
> …
>
> *The Security Manager is optional because the permissions can be installed directly into the application"*

The Security Manager is required, at least at some point to set up the permissions. When permissions are configured, applications enforce the security on the bus even if the Security Manager goes offline. It does not make it optional.

The Security Manager is more a configuration wizard that the end-user cannot skip than an optional service.

With such sentences, one may think permissions get "automagically" installed in applications or, even worse, that the whole Security 2.0 is optional.

[ISS_SEC_CRI_UNC_05] **A trust model is required to validate AllJoyn™ applications**

This specification must explain how device makers and applications developers will test the compliance of their product with the Security Feature 2.0 and how it will be ensured that only these compliant devices will obtain certificates from Security Managers.

2.1.1.2 Critical security issues – Missing

2.1.1.2.1 Claiming a factory-reset device

Section 2.3.1 Claim a factory-reset device

"The procedure to make the device to become claimable again is manufacturer's specific."

[ISS_SEC_CRI_MIS_01] **An action from the end-user shall be required to make a device claimable again.**

A power-on reset is not enough. Nobody would expect that a front door access device become claimable after a power failure. As a consequence, it shall be clearly stated that an embedded device with limited resources must have a persistent storage to comply with this specification.

[ISS_SEC_CRI_MIS_02] **AllJoyn™ compliant devices and applications must have access to a persistent storage to store policies, certificates and claimed status.**

Section 2.3.1 Claim a factory-reset device

"A user can claim any factory reset device during the claiming interval."

[ISS_SEC_CRI_MIS_03] **With the security enforcement defined in [ISS_SEC_CRI_UNC_02], there is no need to restrict the claiming interval since unclaimed device does not interact with other applications on the AllJoyn™ bus.**

Otherwise, the claiming interval has to be specified to avoid heterogeneous and too long claiming interval across all AllJoyn™ devices and applications.

2.1.1.2.2 Small devices

Section 2.6 Certificates

"validityFrom: Validity period. Subfield Valid From. It's represented in seconds since EPOCH Jan 1, 1970."

[ISS_SEC_CRI_MIS_04] **In certificate, configuration revisions shall be used instead of or in addition to POSIX time for certificate validity period.**

Many small devices (ex: smoke detector) do not have a RTC and do not synchronize with a time server. Such devices would not be able to determine if a certificate has expired.

The Collaboration Meeting Notes for August 12-14, 2014 [REF_02] record the decision that there is no need for certificates on small devices (256K to 512K RAM device). This choice will probably show up in a future revision of the high-level design document. It means that the security will not be enforced for all AllJoyn™ applications. In other words, the security 2.0 feature will be optional for all applications. This decision is very bad from a security point of view, as already stated in [ISS_SEC_CRI_UNC_02], and a specific mode of execution for small device has to be proposed instead:

[ISS_SEC_CRI_MIS_05]	**A simplified security enforcement mode of operation is needed for small devices. Such device will only accept ECDHE_ECDSA sessions from and only from the application that claimed it. The claimer will act as a proxy to other standard applications.**

This mode of operation is already compatible with the use cases of the AllJoyn™ Security 2.0 Feature High-Level Design Document [REF_01]. There is only a need to describe how such a proxy application would work.

> Section 2.7.6- Figure 2-20 Use case - Son can control different TVs in the house
>
> *"control is allowed since peer is admin"*

This simplified mode of operation would limit to the strict minimum the number of certificates and policies that a small device would have to store on its persistent storage.

2.1.1.2.3 Revocation (Security Manager, certificates, policies updates)

There are several revocations scenarios that need to be taken into consideration:

1. My phone is stolen. I have exclusively used the Security Manager application/service installed on this phone to setup the permissions of my AllJoyn™ devices/applications. I need to install a new Security Manager on my new phone, prove my identity, revoke the old Security Manager and retrieve any configuration I have done so far.
2. I allow my son to access one of my device but later on I change my mind

It is not easy to get it right …

- … without some sort of always-on proxy and if the applications do not synchronize their policies, certificate and revocation data when they connect to each other.
- … with Security Managers that appear and disappear all the time
- … with only unicast sessions and unicast security model
- … when Security Managers are not "linked" to some identity providers (Google, Facebook, …).

2.1.1.2.3.1 Security Manager revocation

[ISS_SEC_CRI_MIS_06] **The revocation of Security Manager applications is clearly a critical point but is not addressed.**

There is a high probability that a malicious application compromises the Security Manager on Android devices and that the Security Manager private key gets lost (stolen phone for instance). Therefore, Security Manager revocation is clearly mandatory. However at the time of writing, this point is not addressed yet.

There are two, trivial but not satisfactory, options:

- Manually reset all the AllJoyn™ devices to the unclaimed status.
- Accept revocation only with an out-of-band confirmation (button on headless devices) and on a per device basis.

In both cases, the current policies, guilds, certificates would be lost, what is not acceptable for the end-user.

Applications and security managers will be offline most of the time but some devices will probably be online most of the time. This can be anticipated. Hence revocation data (as well as policies and certificate updates) could be sent to the currently online applications and these applications could forward the revocation to other applications as they show up.

The Security Manager may have a UI that would let the user probe and display the propagation status of revocation material for all the known application and for all the guilds managed by this Security Manager.

[ISS_SEC_CRI_MIS_07] **A distinct uncompromised CA shall protect revocation metadata.**

This could be done with a third party public CA in the cloud but we believe that nobody is willing to pay the bill for such a service.

This could also be done by a private revocation CA created by Security Managers themselves. In this case, there must be a strong isolation between the initial CA and the revocation CA so that the impersonation of the first CA does not compromise the revocation CA. This may be implemented in this way:

- A Security Manager generates not only one key pair but two key pairs.
- The first key pair is used as described in the High-Level Design document [REF_01].
- The second key pair is used to create the revocation data and revocation identity certificate. Only the revocation public key is stored on a persistent storage by the Security Manager.
- The revocation identity certificate includes the revocation public key and is signed by the first private key. It is used by application to check the validity of revocation data.
- The revocation data include the first public key, the revocation key pair, the whole being signed by the first private key and encrypted with a symmetric key.
- This symmetric key is derived from a password provided by the end-user or derived from an OAuth token.
- A Security Manager sends the revocation data as well as a revocation identity certificate to applications in the claiming process.
- Applications keep revocation data and revocation identity certificates in a persistent storage and forward them to any application that requests it.
- A new Security Manager that is able to derive the symmetric key gets access to the decrypted revocation key pair.

- The revocation private key can be used by the new Security Manager to emit a revocation certificate (signed by the revocation private key and targeting the old Security Manager) to take ownership on applications.
- Applications can verify the revocation certificate with the revocation identity certificate obtained during the claiming process.
- Applications must keep the revocation data and the old Security Manager public key to identify other applications claimed by the old Security Manager and forward the revocation data to them.

2.1.1.2.3.2 Certificate revocation and policies update

[ISS_SEC_CRI_MIS_08] **The Security Manager on a mobile device cannot provide a reliable revocation service.**

Section 2.3.10 Certificate Revocation

"The application will validate the certificate using a revocation service provided by the Security Manager

The Certificate Revocation Service is expected to provide a method call that takes in the certificate basic information and return whether the given certificate is revoked."

If the Security Manager is a tablet or phone application, it will be most of the time offline. Such service would not be stable enough.

Section 2.3.11 Distribution of policy updates and membership certificates

"This Distribution Service is a service provided by the Security Appliance or the Security Manager. This service provides persistent storage and high availability to allow for push and pull strategy to distribute updates to applications."

[ISS_SEC_CRI_MIS_09] **The role and the form of the Security Appliance is not clearly defined. In addition, there would be a single point of failure if all the Security Managers are offline. Finally, centralized services go against the architecture principles of AllJoyn™.**

There are typically several Security Managers in a home networking domain. The Security Appliance would need to aggregate all the data from all these Security Managers. This would therefore be a centralized service that the Security 2.0 feature model has been trying to avoid since the beginning. For confidentiality, such a service would be only accessible to the members of the guilds created by all the assisted Security Managers in the home networking domain.

Applications may also quickly check and synchronize their policies, certificate and revocation data each time a session is established. This would avoid the need for a centralized service.

2.1.2 Major security issues

2.1.2.1 Major security issues - Missing

2.1.2.1.1 Broadcast and multipoint sessions

Section 2.3.11 Distribution of policy updates and membership certificates

"The Distribution service broadcasts that updates are available so the applications can connect to it in order to retrieve the updates."

Section 4.1 Broadcast signals and multipoint sessions

<*empty*>

[ISS_SEC_MAJ_MIS_01] **This part of the documents is empty.**

Broadcast is definitely not what AllJoyn™ is made for, so broadcast signals must be implemented only if they are absolutely required.

The sessions described in this document are unicast sessions and AllJoyn™ has been built around unicast peer-to-peer sessions. n devices need to establish $\frac{n\cdot(n-1)}{2} = O(n^2)$ unicast connections to connect to each other. Multipoint sessions have been proposed to solve this so-called n^2 issue. In this case and in the current implementation, the application with the highest GUID become the host. The other applications only establish n-1 unicast connections to the host and the host forward the traffic to the right application [REF_04].

[ISS_SEC_MAJ_MIS_02] **Multipoint sessions must offer the same level of security as unicast sessions.**

Before joining a multipoint session, applications could set up standard ECDHE_ECDSA unicast sessions with other applications for mutual authentication and confidentiality. Afterwards, each application would only keep the connection with the host application and encapsulate the traffic for other applications in an encrypted message to the host (dual encryption). A part of the initial AllJoyn™ message headers needs to remain accessible to the host application to let it know how to redirect the messages. Defining the encapsulation format for such messages goes beyond the scope of this review.

The above solution does not scale well when a given message need to be sent to many applications at the same time in an efficient way. If there is a strong requirement to handle this case, a n-party key exchange between applications of the same guild will have to be proposed and reviewed.

2.1.2.1.2 Confidentiality and peer discovery and network topology

[ISS_SEC_MAJ_MIS_03] **The confidentiality of the peer discovery process shall be examined for unicast and multipoint sessions**

The document does not say anything about service discovery. Attackers sniffing network traffic should not be able to learn too much about a AllJoyn™ front door device (presence, model). The security of the discovery process will probably depend on the security of the local networks in use (Wifi, Bluetooth). This has to be confirmed. This is an important matter, especially if WAN distribution is envisioned (ex: multi-home, control device from a smartphone at work).

When an application does not have the rights to access a service, it should probably not know that this service is running. That means that peer discovery would be done without any security in an anonymous way but the services and data exposed by the peers would only be shared once ECDHE_DSA sessions are established. This would improve the protection against an attacker who has gained access to the network.

2.1.2.2 Major security issues - Extra

2.1.2.2.1 Secrets protection on mobile OS

[ISS_SEC_MAJ_EXT_01] **The Keychain Service API on iOS and the KeyChain and KeyStore APIs on Android shall be used to protect all private keys**

If someone steals a device, he should not be able to extract too easily the private key of the Security Manager application. Otherwise, he would be able to impersonate a legitimate user and control his AllJoyn™ devices.

Therefore, secure API should be used on mobile devices to protect the private keys of the AllJoyn™ applications.

Allseen Alliance CORE working group - Security design discussion [REF_02]

TPM support in the certificate?

- *This could be implemented in the trust zone*
- *Consider putting attestation in the certificate*
  - *Note: TPM key attestation is the ability of the user who is requesting a certificate to cryptographically prove to a CA that the RSA key in the certificate request is protected by either "a" or "the" TPM that the CA trusts.*
- *Need to investigate - action item created*

This item can be closed. On Android, the higher level of security offered by a platform is basically reached by using the standard KeyChain and KeyStore API.

Many Android devices (ex: devices based on Qualcomm SoCs, Samsung devices) take advantage of a Trusted Execution Environment (TEE) implemented on top of the ARM Trustzone technology. For the Android devices that support it, the KeyChain and KeyStore API actually make use of the Trustzone secure world for cryptographic operations and to protect application keys. Therefore, we should simply use the standard APIs.

2.1.3 Minor security issues

2.1.3.1 Minor security issues - Missing

2.1.3.1.1 User equivalence certificates

[ISS_SEC_MIN_MIS_01] **A description and a sequence diagram for user equivalence certificate shall be added in section 2.3 (Typical operations).**

Section 2.6.3 User equivalence certificate

"Table 2-4 lists the user equivalence certificate fields. The subject will have the same privileges as the issuer."

The fields of user equivalence certificates are specified in section 2.6 (Certificates) but there is nothing in section 2.3 (Typical operation) about this type of certificate.

2.1.3.1.2 Undefined value for ANY-USER ID

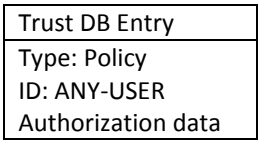
[ISS_SEC_MIN_MIS_02] **The value for the ANY-USER ID is not specified.**

[ISS_SEC_MIN_MIS_03] **This value must be out of the range of the pseudo unique IDs used to identify applications.**

A malicious application could try to set its GUID to the value of ANY-USER expecting that other applications will grant illicit access to their resources.

Section 2.3.4 Install an ANY-USER policy

Figure 2.4 Install an ANY-USER policy



2.1.3.1.3 Pseudo-unique ID and key collision

Collision in pseudo-unique ID and keys are very unlikely but could happen between similar devices with no source of entropy, no RTC, badly provisioned or with a bad implementation of the GUID generator. Attackers may try to guess device private key or a guild ID, and then try to provoke and exploit such corner cases.

[ISS_SEC_MIN_MIS_04] **GUID acronym (globally unique identifier) and GUID properties (generation algorithm, size) are not defined.**

2.1.1.2.3.3 Application identity

[ISS_SEC_MIN_MIS_05] **The identity of an application shall be check against the couple (verified issuer public key, application public key) and not only against the application public key.**

When two applications with the same GUID but claimed by different Security Managers appear on the same AllJoyn™ network, identity certificates need to be crosschecked by each application. Since applications belong to different owners, each application have to look for a membership certificate with delegation or a user equivalence certificate that would reveal a legitimate peering between both owners.

2.1.1.2.3.4 Guild ID

The same apply to guild IDs.

[ISS_SEC_MIN_MIS_06] **Two applications are in the same guild if each application has been targeted by a valid membership certificate issued by the same owner (Security Manager) for the same guild GUID.**

Otherwise, a malicious Security Manager that knows the GUID of a guild could deliver a membership certificate to a malicious application for the targeted guild and the others applications of the guild would accept it.

2.1.3.1.4 Guild-specific certificates and policies

[ISS_SEC_MIN_MIS_07] **The way guild-specific policies are securely delivered to applications is not completely described.**

The section 2.3.5 describes how a guild-specific policy is installed. In the figure 2.5, a “guildCert” is sent from the “admin user” to the “device”. However, in section 2.6 (Certificates), “guild certificates” are not specified. On the other hand, the “policy certificate” described in section 2.6 is not used anywhere. It cannot be the “guild certificate” we are looking for since it does not even have any filed named “guild GUID”.

A pseudo unique ID is not enough to protect a guild. It could be leaked or guessed by an attacker, moreover if the Security 2.0 is an optional feature.

2.1.3.2 Minor security issues - Extra

2.1.3.2.1 Identity provider

[ISS_SEC_MIN_EXT_01] **A provable identity could be derived from OAuth2 permission token. This could be used for Security Manager revocation.**

We need to investigate how user account (Google, Facebook, …) could be used to resolve the Security Manager, certificate and policy revocation challenges.

2.2 General issues

The general issues and concerns have been categorized (in the same way as security issues) into three levels of importance (critical, major and minor) and in three types (unclear, missing and extra).

2.2.1 Major general issues

2.2.1.1 Major general issues - Unclear

2.2.1.1.1 Definition of an "admin" and admin management need further development

Section 2.2 Premises:

Definition of an "admin": *"An admin (or administrator) is a peer with administrator privilege for the application"*

Definition of a "Security Manager": *"<nothing yet>"*

[ISS_GEN_MAJ_UNC_01] **The admin definition does not make a lot of sense to us.**

Does this means that an admin is a real user interacting with the application or an application (driven by a real user) that claimed some other application?

[ISS_GEN_MAJ_UNC_02] **The link between an "admin" and a "Security Manager" shall be clearly defined.**

It appears that both terms refer to very close concepts.

Section 2.2 Premises:

"A admin can add/remove another admin"

At the time of writing, the document does not fully explain how it can be implemented. The revocation of another admin (cf [ISS_SEC_CRI_MIS_06] [ISS_SEC_CRI_MIS_07]), as well as the revocation of certificates and policy updates are still works in progress (cf [ISS_SEC_CRI_MIS_08] [ISS_SEC_CRI_MIS_09]).

[ISS_GEN_MAJ_UNC_03] **The document shall explain how and why an admin would "add" an admin. A security Manager application does not need to be blessed by another Security Manager to take ownership on unclaimed devices.**

What happens when an admin "adds" another admin? Does it involve a membership certificate with the delegate flag set, a guild equivalence certificate?

2.2.1.1.2 Terminology consistency "User vs Admin vs Application", "guild authority"

[ISS_GEN_MAJ_UNC_04] **Since AllJoyn™ is application-centric, "users" are "applications" and the difference between the two operations described in sections 2.3.6 and 2.3.7 in not clear.**

Section 2.3.6 Add an application to a guild

"An admin signs a membership certificate with the given guild ID and installs in the application"

Section 2.3.7 Add a user to a guild

"The guild authority uses the Security Manager to generate the membership certificate for the user for a given guild ID"

[ISS_GEN_MAJ_UNC_05] **The terms "user" and "admin", used all over the document, are describing two types of applications or perhaps the same application used in different contexts. Applications shall not be personified.**

"User" should be used for a real user, the one who may link (or not) his Facebook or Google account to a Security Manager.

At least, the terms should be unambiguously defined in Section 2.2 (Premises).

[ISS_GEN_MAJ_UNC_06] **The terms "user", "admin", "application", "device", "controller" shall be more consistently used all over the sequence diagrams of section 2.3.**

The following definitions are proposed:

Term	Definition
Admin	Security Manager application (implicitly driven by a human being)
User	Context dependent: - Sometime a Security Manager application implicitly driven by a human being (ex: Section 2.3.1.1 – Figure 2-2 or Section 2.3.8 - Figure 2-8) - Sometime a simple application driven or not by a human being and claimed by a Security Manager (ex: Section 2.3.7 - Figure 2-7)

[ISS_GEN_MAJ_UNC_07] **A guild authority cannot use a Security Manager because a guild authorities is a Security Manager.**

Section 2.3.7 Add a user to a guild

"The guild authority uses the Security Manager to generate the membership certificate for the user for a given guild ID"

AllJoyn™ is application-centric. A guild authority is necessarily a Security Manager.

[ISS_GEN_MAJ_UNC_08] **The term "guild authority" is not defined either in section 2.2 (Premises).**

The following definition is suggested:

Term	Definition
Guild authority	Security Manager application that have originally created the guild by giving it a pseudo-unique ID and by issuing guild-specific certificates and policies to at least one application

2.2.1.1 Major general issues - Missing

2.2.1.1.1 Guild ID and common names

Section 2.3.2 define a guild:

"When the user specifies a guild name, the Security Manager creates the guild ID (typically a GUID value)"

[ISS_GEN_MAJ_MIS_01] **The documents shall describe how the common names provided by the user for guilds (ex: "Living room") and applications (ex: "iRemote - Bob") are shared between applications.**

Users (human beings) need to see and manipulate common names in Security Manager applications.

2.2.1.1.2 Mass storage considerations

[ISS_GEN_MAJ_MIS_02] **The typical amount of storage recommended for both types of compliant devices/applications (small devices and normal devices) shall be defined**

This way, device makers will not discover that a product in late-stage development has not been correctly dimensioned.

2.2.1.2 Major general issues - Extra

2.2.1.2.1 Data Driven API (DDAPI) and battery powered devices

People working on the DDAPI rely on stable connections. This can be challenging on mobile platforms as explain commented in the section 2.2.1.3 of this document.

[ISS_GEN_MAJ_EXT_01] **People working on the DDAPI have specific needs that have to be taken into account in the Security 2.0 Feature High-Level Design Document.**

We do not want to notice in 6 months that Security Enhancement and DDAPI projects took incompatible design decisions.

2.2.2 Minor general issues

2.2.2.1 Minor general issues - Unclear

2.2.2.1.1 Use a standard naming convention for the authentication algorithm

[ISS_GEN_MIN_UNC_01] **ECDHE_DSA and DSA should be respectively replaced everywhere by ECDHE_ECDSA and ECDSA.**

Section 2.6 Certificates

"DSA signature, which is computed over the fields from subject field to…"

All over the document, ECDHE_DSA is used as the key exchange algorithm for authenticated sessions. In Section 2.4.6 – Figure 2-14, ECDHE_ECDSA is used instead. In Section 2.6 (Certificates), DSA algorithm is defined as ECC NIST p-256 DSA.

Elliptic Curve Cryptography has clearly been chosen over RSA or DSA because it offers the same security strength with smaller key sizes and better computational efficiency. Therefore, Elliptic Curve Diffie-Hellman Ephemeral key agreement will be used in a TLS handshake and Elliptic Curve Digital Signature Algorithm will be used for authentication. ECDHE_ECDSA is the name used in TLS specifications for that.

2.2.2.2 Minor general issues - Extra

2.2.2.2.1 Improvement of certificates fields representation

[ISS_GEN_MIN_EXT_01] **An ASN1 syntax to represent the certificate fields and how X.509 v3 Certificate Extensions are used would be appreciated.**

Section 2.6 Certificates

"The certificate format is X.509 v3. The certificate lifetime will be considered in order to avoid having to revoke the certificate. The subsections will be updated with the X.509 required fields."

A certificate extension is identified by an object identifier (OID). OIDs are a string of numbers identifying unique objects and are controlled by the International Standards Organization (ISO) registration authority.

3. Conclusion

A few security issues have been found in the AllJoyn™ Security 2.0 Feature High-Level Design document. The most critical issue for us is that AllJoyn™ will remain unsecure by default according to this document. It should finally be noted that the document will evolve with the specifications and should not be seen as a final version.