Car Connectivity Consortium

MirrorLink®

Device Attestation Protocol

Version 1.1.6
(CCC-TS-014)

VERSION HISTORY

Version	Date	Comment
1.0	06 October 2010	Public release
1.0.1	26 June 2011	Approved Version
1.1	31 March 2012	Approved Version
1.1.1	24 September 2012	Approved Errata Version
1.1.2	05 December 2012	Approved Errata Version
1.1.3	05 March 2013	Approved Errata Version
1.1.4	10 April 2013	Approved Errata Version
1.1.5	24 July 2013	Approved Errata Version
1.1.6	05 November 2013	Approved Errata Version

LIST OF CONTRIBUTORS

Beckmann, Mark	Volkswagen AG
Benesch, Matthias	Daimler AG
Bose, Raja,	Nokia Corporation
Brakensiek, Jörg (Editor)	Nokia Corporation
Fernahl, Dennis	Carmeq (for Volkswagen AG)
Kostiainen, Kari	Nokia Corporation
Lehner, Martin	Jambit
Park, Keun-Young	Nokia Corporation
Wolf, Michael	Daimler AG

LEGAL NOTICE

The copyright in this Specification is owned by the Car Connectivity Consortium LLC ("CCC LLC"). Use of this Specification and any related intellectual property (collectively, the "Specification"), is governed by these license terms and the CCC LLC Limited Liability Company Agreement (the "Agreement").

Use of the Specification by anyone who is not a member of CCC LLC (each such person or party, a "Member") is prohibited. The legal rights and obligations of each Member are governed by the Agreement and their applicable Membership Agreement, including without limitation those contained in Article 10 of the LLC Agreement.

CCC LLC hereby grants each Member a right to use and to make verbatim copies of the Specification for the purposes of implementing the technologies specified in the Specification to their products ("Implementing Products") under the terms of the Agreement (the "Purpose"). Members are not permitted to make available or distribute this Specification or any copies thereof to non-Members other than to their Affiliates (as defined in the Agreement) and subcontractors but only to the extent that such Affiliates and subcontractors have a need to know for carrying out the Purpose and provided that such Affiliates and subcontractors accept confidentiality obligations similar to those contained in the Agreement. Each Member shall be responsible for the observance and proper performance by such of its Affiliates and subcontractors of the terms and conditions of this Legal Notice and the Agreement. No other license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

Any use of the Specification not in compliance with the terms of this Legal Notice, the Agreement and Membership Agreement is prohibited and any such prohibited use may result in termination of the applicable Membership Agreement and other liability permitted by the applicable Agreement or by applicable law to CCC LLC or any of its members for patent, copyright and/or trademark infringement.

THE SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS, AND COMPLIANCE WITH APPLICABLE LAWS.

Each Member hereby acknowledges that its Implementing Products may be subject to various regulatory controls under the laws and regulations of various jurisdictions worldwide. Such laws and regulatory controls may govern, among other things, the combination, operation, use, implementation and distribution of Implementing Products. Examples of such laws and regulatory controls include, but are not limited to, road safety regulations, telecommunications regulations, technology transfer controls and health and safety regulations. Each Member is solely responsible for the compliance by their Implementing Products with any such laws and regulations and for obtaining any and all required authorizations, permits, or licenses for their Implementing Products related to such regulations within the applicable jurisdictions.

Each Member acknowledges that nothing in the Specification provides any information or assistance in connection with securing such compliance, authorizations or licenses**.**

NOTHING IN THE SPECIFICATION CREATES ANY WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING SUCH LAWS OR REGULATIONS. ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY INTELLECTUAL PROPERTYRIGHTS OR FOR NONCOMPLIANCE WITH LAWS, RELATING TO USE OF THE SPECIFICATION IS EXPRESSLY DISCLAIMED. BY USE OF THE SPECIFICATION, EACH MEMBER EXPRESSLY WAIVES ANY CLAIM AGAINST CCC LLC AND ITS MEMBERS RELATED TO USE OF THE SPECIFICATION.

CCC LLC reserve the right to adopt any changes or alterations to the Specification as it deems necessary or appropriate.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TERMS AND ABBREVIATIONS

A2DP	Bluetooth Advanced Audio Distribution Profile
ARP	Address Resolution Protocol
BT	Bluetooth
CDC	Communications Device Class; specified from USB Device Working Group
CE	Consumer Electronics; CE devices are referred to as mobile devices within this specification
DAP	Device Attestation Protocol
DHCP	Dynamic Host Configuration Protocol
HFP	Bluetooth Hands-free Profile
HSP	Bluetooth Headset Profile
HMI	Human Machine Interface
HU	Head-unit (this term is used interchangeably with the MirrorLink client)
HS	Head-set
IP	Internet Protocol
NCM	Network Control Model; part of the CDC device class
Pointer Event	Pointer events are used to describe touch screen action in which the user touches the screen with one (virtual) finger only at a single location.
RFB	Remote Framebuffer
RTP	Real-time Transport Protocol
TCP	Transmission Control Protocol
Touch Event	Touch events are used to describe touch screen action in which the user touches the screen with two or more separate fingers at different locations. Touch events are used to describe more complex touch action, like pinch-open or pinch-close.
UDP	User Datagram Protocol
UI	User Interface
UPnP	Universal Plug and Play
USB	Universal Serial Bus
VNC	Virtual Network Computing

MirrorLink is a registered trademark of Car Connectivity Consortium LLC

Bluetooth is a registered trademark of Bluetooth SIG Inc.

RFB and VNC are registered trademarks of RealVNC Ltd.

UPnP is a registered trademark of UPnP Forum.

Other names or abbreviations used in this document may be trademarks of their respective owners.

1 ABOUT

This document is part of the MirrorLink specification which specifies an interface for enabling remote user interaction of a mobile device via another device. This specification is written having a vehicle head-unit to interact with the mobile device in mind, but it will similarly apply for other devices, which provide a color display, audio input/output and user input mechanisms.

The document will focus on the interface functionality, its parameters and protocols only. It does not provide any guidelines for implementing the protocol. If there is a reference towards an implementation, this is of an informative nature only.

The specification lists a series of requirements, either explicitly or within the text, which are mandatory elements for compliant solutions. Recommendations are given, to ensure optimal usage and to provide suitable performance. All recommendations are optional.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are following the notation as described in RFC 2119 [2].

1. MUST: This word, or the terms "REQUIRED" or "SHALL", means that the definition is an absolute requirement of the specification.
2. MUST NOT: This phrase, or the phrase "SHALL NOT", means that the definition is an absolute prohibition of the specification.
3. SHOULD: This word, or the adjective "RECOMMENDED", means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
4. SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED" means that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
5. MAY: This word, or the adjective "OPTIONAL", means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option MUST be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides.)

2 DEVICE ATTESTATION

The term “device attestation” in this context refers to the MirrorLink client verifying that the MirrorLink server is from a compliant manufacturer and running approved software. The attestation will be based on standard X.509 certificates [3] and attestation mechanisms standardized by Trusted Computing Group [5].

3 MANAGING A DAP SESSION

3.1 Bindings

3.1.1 TCP Binding

3.1.1.1 Identifying DAP Server

The identification of the DAP server is described in [10].

3.1.1.2 Device Attestation Launch

The DAP server start-up is facilitated via the UPnP TmApplicationServer:1 service LaunchApplication action, as defined in [10]. The LaunchApplication action MUST return with a URL to the DAP server.

If the returned URL is already used from any established DAP session, this session will continue without any change.

Otherwise a new DAP session MUST be established, given the following steps:

1 DAP server MUST listen for the DAP client to make TCP connection at the provided URL.

2 DAP client MUST make a TCP connection to the provided URL.

3 DAP server and client MUST start DAP according to the steps defined in Section 4.

3.1.2 Intentionally Terminating the DAP Session

The DAP server MUST NOT intentionally terminate a DAP session.

The DAP client MUST intentionally terminate a DAP session, using the following steps:

1. UPnP Control Point uses TmApplicationServer:1 service's TerminateApplication SOAP action to send termination request.
2. DAP client MUST disconnect the TCP connection
3. DAP server SHOULD disconnect the TCP connection on detection of the client TCP disconnect or 5 seconds after responding to the TerminateApplication SOAP action, whichever comes first.

The DAP client MUST wait for any outstanding Device Attestation Response messages, prior to terminating the DAP session.

3.1.3 Unintentionally Terminating the DAP Session

Unintentional termination of the DAP session MAY happen at any time due to error conditions. In the case of unintentional termination of the DAP session, the respective DAP server or client MUST disconnect the TCP connection. The respective counterpart SHOULD disconnect as well.

If the MirrorLink Client decides to re-establish the DAP session, it MUST follow the steps given in Section 3.1.1.2.

To avoid the DAP server or client being in a TCP TIME-WAIT time-out loop as a result of an unintentional active disconnect, the TCP socket SHOULD be established using the SO_REUSEADDR option (or similar platform specific variant), allowing the operating system to reuse a port address, even it is currently in the TIME-WAIT state or the DAP server SHOULD use a different, unaffected port number.

3.2 Other Bindings

Besides TCP/IP, it will be possible to run MirrorLink Device Attestation Protocol on top of other protocols like Bluetooth RFCOMM, but how to discover and establish connection for such configuration is outside the scope of this specification.

3.3 Testing Considerations

If the MirrorLink Client is in a dedicated testing state (as part of the MirrorLink Certification), it MUST launch a new DAP session (either initiated automatically or manually from the user), whenever the DAP server has unintentionally terminated the DAP session.

The MirrorLink Client MUST have a mechanism to allow a test engineer to launch a DAP session (either automatically or manually).

For DAP testing purposes, the MirrorLink Client MUST use a CTS root certificate to validate responses from the CTS Server. This CTS root certificate MUST be decoupled from the regular CCC root certificate used during production.

The CTS root certificate MUST be accepted from the MirrorLink Client in test setup only. The CTS public key (DER encoded) and the 32-byte SHA-256 hash of the CTS public key (Base64 encoded) are provided separately.

The DAP trust chain, for testing purposes is shown in Figure 1.

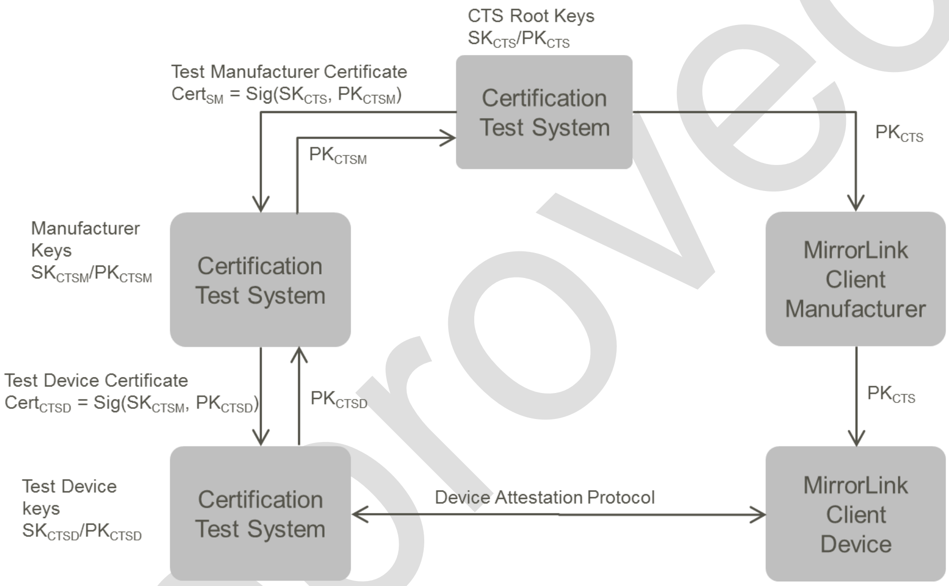
CTS Root Keys
SK_{CTS}/PK_{CTS}
Test Manufacturer Certificate
$Cert_{SM}$ = Sig(SK_{CTS}, PK_{CTSM})
Certification Test System
PK_{CTSM}
PK_{CTS}
Manufacturer Keys
SK_{CTSM}/PK_{CTSM}
Certification Test System
MirrorLink Client Manufacturer
Test Device Certificate
$Cert_{CTSD}$ = Sig(SK_{CTSM}, PK_{CTSD})
PK_{CTSD}
PK_{CTS}
Test Device keys
SK_{CTSD}/PK_{CTSD}
Certification Test System
Device Attestation Protocol
MirrorLink Client Device

Figure 1: Device Attestation Certification Infrastructure – Testing Only

The MirrorLink Client MUST NOT accept and request the CTS root certificate outside DAP testing and certification.

For testing MirrorLink Servers, the CTS MUST only accept legitimate certificates signed by the CCC’s Root Certificate. For the purposes of testing, MirrorLink Server manufacturers are allowed to issue a small number (<100) of device certificates with an expiration of 3 months from the date of signing before successful completion of their Server Device Audit as described in [16]. MirrorLink Server manufacturers must still have a Passed/Conditional result for their Server Manufacturer Audit in order to receive a Manufacturer Certificate.

4 DEVICE ATTESTATION PROTOCOL

The prerequisite of successful Device Attestation Protocol run is that the MirrorLink server has a X.509 device certificate (with Extended Key Usage tcg-kp-AIKCertificate OID 2.23.133.8.3 as specified in [14]) for its device key pair from the server device manufacturer. Additionally, the server MUST have one X.509 manufacturer certificate signed from the CCC DAP management system. The server device's private key(s) MUST be stored securely. The secure storage is manufacturer specific and MAY use (1) Hardware-based Mobile Trusted Module (MTM) [7] implementation or equivalent, (2) Storage on OS, which integrity has been verified with hardware-assisted secure boot process, (3) Or storage on OS alone.

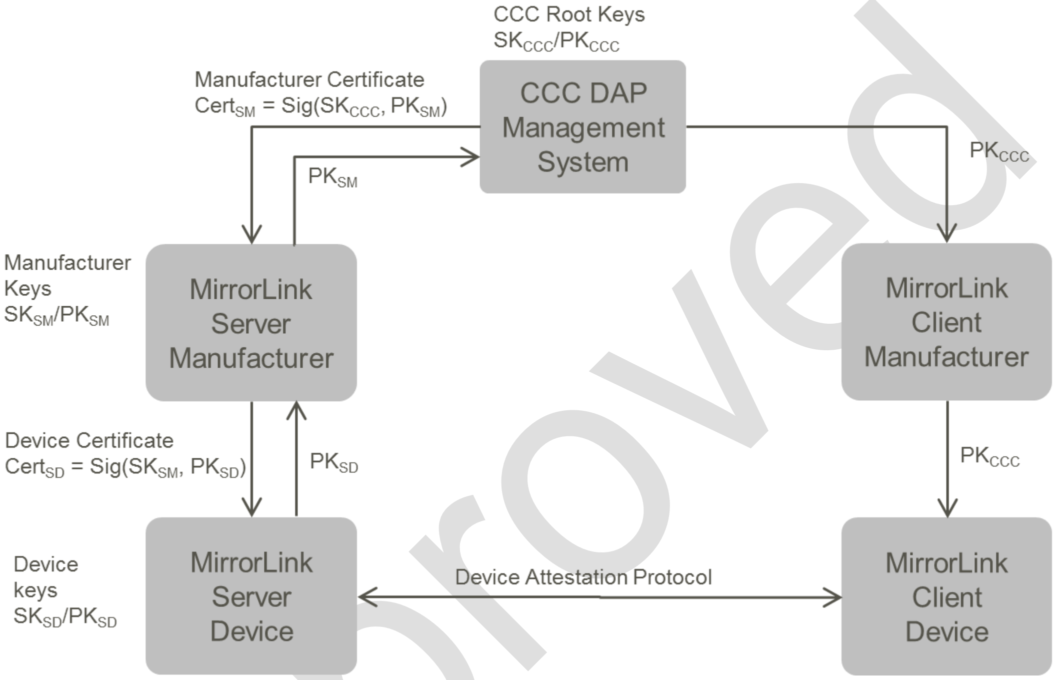

Figure 2: Device Attestation Certification Infrastructure

MirrorLink assumes pre-established trust relationships and security associations between the MirrorLink server device manufacturers and MirrorLink client device manufacturers via a central CCC controlled DAP Management System which extends to both client and server devices. This is achieved using a standard X.509 certificate chain.

The key pair SK_A/PK_A, as shown in Figure 2 consists of the private key SK_A and the public key PK_A. The certificate $Cert_A = Sig(SK_B, PK_A)$ is an X.509 public key certificate with subject public key PK_A and signed with private key SK_B (i.e., the certificate issuer is B).

After the MirrorLink Server device manufacturers have been certified from the central CCC DAP management system ($Cert_{SM} = Sig(SK_{CCC}, PK_{SM})$), they can certify individual devices they produce ($Cert_{SD} = Sig(SK_{SM}, PK_{SD})$). Again, MirrorLink specifies using standard X.509 certificates. This certification will typically take place during device manufacturing time, but some device manufacturers MAY have proprietary mechanisms and the device cert MAY be bootstrapped during first boot. This operation is done only once per each device.

Using the device certificate the MirrorLink Server device can authenticate/attest itself to the MirrorLink Client device. For this MirrorLink specifies using Device Attestation Protocol (DAP). This process (DAP) will be run for each connection from a MirrorLink Server device to the MirrorLink Client device. Both the device and the manufacturer certificates are included in the DAP message exchange. Using this chain of certificates the MirrorLink Client device can verify at runtime that the MirrorLink Server device is a genuine/compliant device from an authorized MirrorLink Server device manufacturer. The MirrorLink Client MUST have the public key PK_{CCC} available, to be able to validate the MirrorLink Server device manufacturer certificate.

An overview of device and software attestation protocol is shown in Figure 3 below. The protocol is two-flow protocol: MirrorLink client sends `attestationRequest` message and MirrorLink server replies with `attestationResponse` message. Both of these messages are XML formatted.

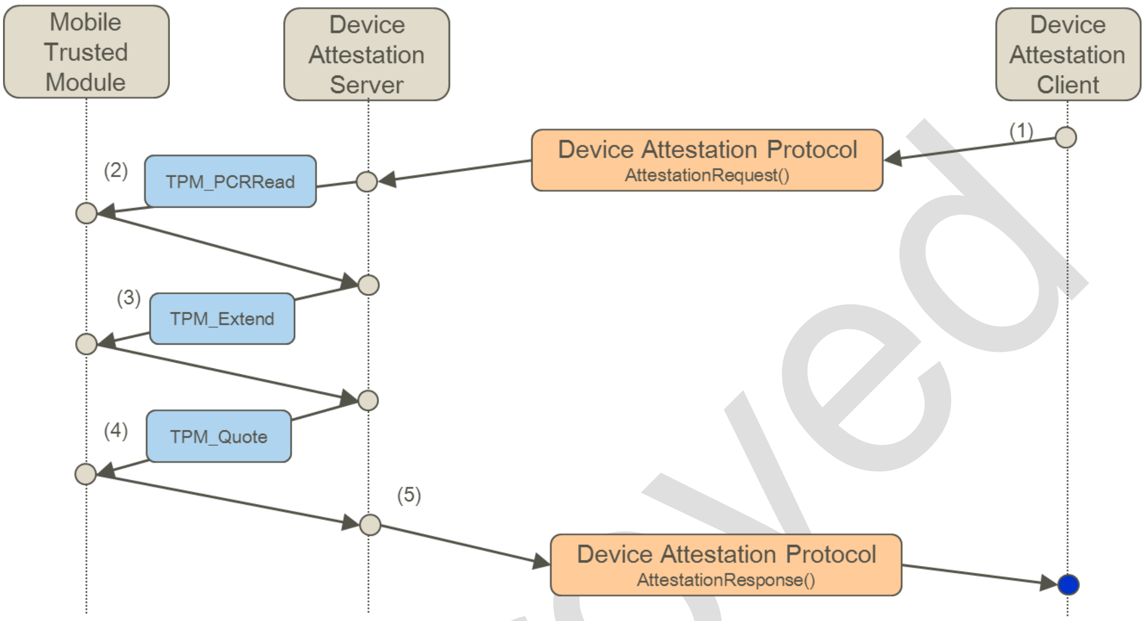

Figure 3: Device and software attestation protocol overview

In more detail, the Device Attestation Protocol consists of following steps:

1. Device Attestation Client first picks random nonce and sends that together with identifier of the requested attested component(s) to Device Attestation Server as part of `AttestationRequest` message.
2. Device Attestation Server is a trusted software component on the MirrorLink server device. It first measures[1] the requested component(s). If the measurement matches expected value, the attestation protocol proceeds. The Device Attestation Server first reads current value of Platform Configuration Register (PCR) 10 from Mobile Trusted Module using `TPM_PCRRead` command [6]. Device Attestation Server saves the current PCR value for later use. (Device Attestation Server protection and software component measurement mechanisms are platform and implementation specific and intentionally left out of this specification).
3. Device Attestation Server extends attestationEvidence to PCR 10 using `TPM_Extend` command [6]. attestationEvidence = componentID || URL || SHA1(applicationPublicKey), where || denotes to concatenation (for more details on these data elements, see descriptions of `attestationRequest` and `attestationResponse` below).
4. Device Attestation Server performs `TPM_Quote` command [6] on PCR 10 using the nonce as an input. This operation signs the PCR structure using a certified device key. The TPM_Quote operation returns TPM_PCR_COMPOSITE structure using which the server constructs matching TPM_QUOTE_INFO structure.
5. The saved PCR value, the resulting TPM_Quote signature, URL, the device certificate, and the device manufacturer certificate are sent to Device Attestation Client as part of `AttestationResponse`. Device Attestation Client verifies the device manufacturer certificate (using pre-installed

[1] Measurement of a software component is a platform specific mechanism of identifying a software component on the device. In practice, this could involve verification of a unique software component identifier provided by the underlying operating system security framework, or calculation of a hash over the software component binary executable, or other equivalent mechanisms.

trust root), verifies device certificate using verified device manufacturer certificate, and finally verifies the attestation signature with respect to the TPM_QUOTE_INFO using the verified device certificate.

Implementation considerations:

It is the responsibility of the MirrorLink server that the usage of certified device key is restricted to authorized components only, such as hardware-based MTM implementation or trusted software component. The exact protection mechanisms are platform and implementation specific.

The elements of `attestationRequest` XML message are defined in Table 1 (the XSD schema of the XML is given in Appendix A):

Element	Description	Parent	Availability
attestationRequest	Main container of attestation request	None	Optional
version	MirrorLink Version information	attestation Request	Mandatory
majorVersion	Major version number	version	Mandatory
minorVersion	Minor version number	version	Mandatory
trustRoot	Identifier of the trust root used by MirrorLink client for attestation verification. 32-byte SHA-256 hash of CCC root public key Base64 encoded from SubjectPublicKeyInfo DER format [1]. Based on this identifier the server can send correct manufacturer certificate(s) to client.	attestation Request	Mandatory
nonce	20-byte random number Base64-encoded [1]	attestation Request	Mandatory
componentID	Identifier of the software component to be attested. Possible values are described in Table 2.	attestation Request	Mandatory

Table 1: Device Attestation – attestationRequest elements

Device Attestation Protocol can be used to enable the following attestations:

- Attest that the connected MirrorLink Server device is manufactured from a trusted manufacturer.
- Attest that software components on the connected MirrorLink Server device are original (i.e. certified).

The components, which can be attested, are listed in Table 2 with their component IDs. The Protocol ID is used as part of the attestation's response URL binding

componentID	Description of what functionality is attested	Protocol ID
TerminalMode:VNC-Server	VNC server functionality as specified in [12]	VNC
TerminalMode:UPnP-Server	UpnP TmServerDevice server services as specified in [9], [10] and [11] The attestation includes the service advertisements, using UDP broadcasts and the SOAP and eventing mechanisms based on HTTP.	HTTP
TerminalMode:RTP-Server	RTP-Server as specified in [13]	RTP
TerminalMode:RTP-Client	RTP-Client as specified in [13]	RTP
MirrorLink:CDB-Endpoint	CDB Endpoint, as specified in [15]	CDB

componentID	Description of what functionality is attested	Protocol ID
MirrorLink:Device	Device is certified and manufactured from a CCC member.	-
*	Wildcard. All components, which can be attested from the MirrorLink Server, MUST be attested. In this case the MirrorLink server MUST reply with a single `attestationResponse` message, which includes the attestation elements of all attested components.	-

Table 2: Device Attestation – Component List

The MirrorLink Server MUST support attesting of the following components (DAP Minimum Set):

- "`MirrorLink:Device`"
- "`TerminalMode:UPnP-Server`", including the `URL` and the `applicationPublicKey`

The MirrorLink Client MUST accept a MirrorLink Server, implementing the above listed DAP Minimum Set, in Park and Drive Mode.

The MirrorLink Server SHOULD provide support for attesting other components. Components, the MirrorLink Server is not able to attest, MUST NOT be included in the `attestationResponse`.

Proprietary components MUST be identified with a *`domainName`*`:`*`componentID`*, where *`domainName`* follows the Java namespace convention (e.g. `com.daimler`). The MirrorLink server MUST ignore the attestation of proprietary componentIDs, which are not supported by it.

Below is an example of `attestationRequest` message:

```
<attestationRequest>
   <version>
      <majorVersion>1</majorVersion>
      <minorVersion>1</minorVersion>
   </version>
   <trustRoot>dbR5…dT5S3=</trustRoot>
   <nonce>7Thg34saHd5...4hkjd=</nonce>
   <componentID>TerminalMode:VNC-Server</componentID>
</attestationRequest>
```

The elements of `attestationResponse` XML message are defined in Table 3 (the XSD schema of the XML is given in Appendix A):

Element	Description	Parent	Availability
attestation Response	Main container of attestation response	None	Optional
version	MirrorLink Version information	attestation Response	Mandatory
major Version	Major version number	version	Mandatory
minor Version	Minor version number	version	Mandatory
result	Indicates success/failure of attestation operation. Possible values are defined in Table 4.	attestation Response	Optional
sizeOfSelect	UINT 16 value of the sizeOfSelect as used by the ML Server in the TPM_PCR_SELECTION structure when calling TPM_Quote. (**Default**: 3)	attestation Response	Optional

Element	Description	Parent	Availability
attestation*	Contains attestation of one component.	attestation Response	Mandatory[2]
componentID	Identifier of the attested component. Possible values are listed in Table 2.	attestation	Mandatory
oldValue	The old value of the PCR 10 reserved for device and software attestation use. 20-byte binary value Base64-encoded [1].	attestation	Mandatory
quoteInfo	TPM_QUOTE_INFO structure (as specified in [6]) over which the quoteSignature is calculated. Contains TPM_COMPOSITE_HASH value derived from the content of PCR 10. Base64-encoded [1]	attestation	Mandatory
quote Signature	RSA PKCS#1 v1.5 formatted signature produced by TPM_Quote command as specified in [6]. 256-byte binary value Base64-encoded [1].	attestation	Mandatory
URL	URL that defines the Protocol ID (according Table 2), IP address and port number that the attested software component is currently holding, according the following format: `[ProtocolID]://[IP]:[Port]` MUST be left empty for MirrorLink:Device component identifier. Multiple URLs of the same component MUST be added as separate attestation elements.	attestation	Mandatory
application PublicKey	Public part of key pair that the attested application MAY use to authenticate (e.g. sign) transferred data. (The private part of this key pair SHOULD be accessible only by the attested application. The mechanism used to protect the private key depends on the platform of server device.) The key pair MUST be 2048 bit RSA key and the public part MUST be Base64 encoded [1] from X.509 SubjectPublicKeyInfo DER format [3].	attestation	Optional
device Certificate	X.509v3 [3] certificate issued by the MirrorLink server device manufacturers. The certificate contains the public part of the 2048-bit RSA device key with SHA-256 or SHA-512. The certificate MAY have variable length. The certificate is Base64 encoded from ASN.1 DER format [8].	attestation Response	Mandatory[3]
manufacturer Certificate*	A (chain of) X.509v3 [3] certificate(s) issued for the MirrorLink server manufacturer by the CCC DAP management system. The certificate contains the public part of the 2048-bit or 4096-bit RSA manufacturer key with SHA-512. The certificate(s) MAY have variable length. The certificate(s) are Base64 encoded from DER format.	attestation Response	Mandatory[4]

1 Table 3: Device Attestation – attestationResponse Elements

[2] Mandatory only in case of successful attestation (result == 0).

[3] Mandatory only in case of successful attestation (result == 0).

[4] Mandatory only in case of successful attestation (result == 0).

The MirrorLink Client MUST be able to verify a X.509 trust chain, using only RSA with SHA-2. Algorithms are defined in [4].

The elements marked with an (*) can have multiple instances.

The MirrorLink Server MUST NOT use a `componentID` value equal to the wildcard ‟*”within the `attestationResponse` message.

The MirrorLink Server MUST NOT include more than 3 entries into the manufacturer certificate chain. To simplify the validation for the MirrorLink Client, the MirrorLink Server MUST include the manufacturer certificates in trust chain order, i.e. the certificate from the CA, which signed the device certificate, MUST be first and the certificate, which was signed by the CCC root CA MUST be last.

An example with 3 entries is shown below.

```
<deviceCertificate>
   Device Certificate, signed by manufacturer Sub-Sub-CA
</deviceCertificate>
<manufacturerCertificate>
   Manufacturer Sub-Sub-CA Certificate, signed by manufacturer Sub-CA
</manufacturerCertificate>
<manufacturerCertificate>
   Manufacturer Sub-CA Certificate, signed by manufacturer CA
</manufacturerCertificate>
<manufacturerCertificate>
   Manufacturer CA certificate, signed by CCC root CA
</manufacturerCertificate>
```

An example with 1 entry is shown below.

```
<deviceCertificate>
   Device Certificate, signed by manufacturer CA
</deviceCertificate>
<manufacturerCertificate>
   Manufacturer CA certificate, signed by CCC root CA
</manufacturerCertificate>
```

Possible result values indicating success/failure of the attestation operation are given in Table 4. In case the MirrorLink Client request attestation of an individual component, which is not available for attestation, the MirrorLink Server MUST respond with the ‟`Component not existing`” response.

Result	Description of result value
0	Successful attestation (**default**)
1	Component not existing or attestation not available
2	Error in attestation: Version not supported
3	Error in attestation: unknown trust root
4	Error in attestation: unknown component ID
5	Error in attestation: Attestation failed

Table 4: Device Attestation – Possible Attestation Result Values

Below is an example of `attestationResponse` message:

```
<attestationResponse>
   <version>
      <majorVersion>1</majorVersion>
      <minorVersion>0</minorVersion>
   </version>
   <result>0</result>
```

```
      <attestation>
         <componentID>TerminalMode:VNC-Server</componentID>
         <oldvalue>j1FGh...kj=</oldValue>
         <quoteInfo>kDal2d33...26sE56jJsa</quoteInfo>
         <quoteSignature>IL7h9j9...4J3234Kg=</quoteSignature>
         <URL>VNC://192.168.64.1:5500</URL>
      </attestation>
      <deviceCertificate>gTsd7d3...763rJKh=</deviceCertificate>
      <manufacturerCertificate>6sbk5..7d4dkH= </manufacturerCertificate>
   </attestationResponse>
```

The MirrorLink Server MUST return a version number, which is equal or smaller than the MirrorLink Client’s version number. That means that the MirrorLink Server MUST NOT include components into a DAP response to a wildcard “*” DAP request, which have not been specified for the respective MirrorLink Client version, as given below:

- A MirrorLink 1.1 Server connected to a MirrorLink 1.0 Client MUST NOT include any MirrorLink:Device component
- A MirrorLink 1.2 Server connected to a MirrorLink 1.0 Client MUST NOT include any MirrorLink:Device, Mir-rorLink:HSML, MirrorLink:WFD:RTSP components
- A MirrorLink 1.2 Server connected to a MirrorLink 1.1 Client MUST NOT include any MirrorLink:HSML, Mir-rorLink:WFD:RTSP components

It MUST be noted though, that a MirrorLink Server MUST correctly respond to a DAP request for any individual component (i.e. excluding the wildcard), even if that component is not specified for the respective MirrorLink Client version (but is specified within the respective MirrorLink Server version).

To verify quoteSignature a MirrorLink client SHOULD construct a TPM_PCR_COMPOSITE structure and validate the TPM_Quote signature with respect to that. The TPM_PCR_COMPOSITE structure contains a TPM_PCR_SELECTION structure in which only PCR 10 is set (bit field starting with 0x00 0x04 and padded with 0x00 up to a length in bytes of sizeOfSelect). For more details on these structures see TPM Structures specification from [6]. The MirrorLink client SHOULD perform following steps:

1. Calculate hash H1 as SHA1(applicationPublicKey) if attestationResponse message included applicationPublicKey element
2. Calculate hash H2 as SHA1(oldValue || SHA1(componentID || URL || H1)). Include H1 if attestationResponse message included applicationPublicKey element.
3. Create TPM_PCR_COMPOSITE structure C
   a. Set C->select->sizeOfSelect to sizeOfSelect (UINT16 value)
   b. Set C->select->pcrSelect[0] to 0x00
   c. Set C->select->pcrSelect[1] to 0x04
   d. Pad C->select->pcrSelect with 0x00 (until size of C->select->pcrSelect is sizeOfSelect)
   e. Set C->valueSize to 20 (UINT32 value)
   f. Set C->pcrValue to H2
4. Calculate hash H3 as SHA1(C)
5. Verify that digestValue in quoteInfo equals to H3
6. Verify that externalData in quoteInfo equals to nonce in AttestationRequest
7. Verify that received quoteSignature is valid RSA-SHA1 PKCS#1 v1.5 signature over received quoteInfo using public part of device key extracted from deviceCertificate. (This means that quoteSignature SHOULD valid signature over SHA1(quoteInfo)).

The client MUST verify that the server device certificate has tcg-kp-AIKCertificate Extended Key Usage as specified in [14]. Otherwise the attestation MUST NOT be accepted.

If device and software attestation is mandated by the MirrorLink client, it SHOULD attest all software components on the MirrorLink server before presenting content received from these components to the user.

When a software component on the MirrorLink server is attested, the client SHOULD check that it has active (TCP) connection with the attested software component that matches the attested IP address and port number. Once the active (TCP) connection breaks the client SHOULD run the attestation protocol again for the same component (if mandated by the client).

Note: When device and software attestation protocol is run over networked connection, such as protected WiFi, an attacker *within* the same network MAY be able to masquerade as the attested device. Performing such an attack requires that the attacker is able to (1) spoof its IP address and (2) manipulate TCP connection parameters (such as sequence numbers). Both of these are possible with right kind of equipment, but the user MUST have intentionally added the malicious device into the protected WiFi network. In such a case it is RECOMMENDED to use the application specific key (that was bound to attestation of each component) to authenticate (e.g. sign on server and verify on client) all or a subset of the communication.

It is RECOMMENDED to terminate the Device Attestation Protocol, using the appropriate TerminateApplicaton() SOAP action after all components have been successfully attested. The Device Application Protocol can be launched again at any point in time.

5 REFERENCES

[1] XML Signature Syntax and Processing, http://www.w3.org/TR/xmldsig-core/, W3C Recommendation 10 June 2008

[2] IETF, RFC 2119, Keys words for use in RFCs to Indicate Requirement Levels, March 1997. http://www.ietf.org/rfc/rfc2119.txt

[3] IETF, RFC 5280, Internet X.509 Public Key Infrastructure Certificate, May 2008. http://tools.ietf.org/html/rfc5280

[4] IETF, RFC 3279, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", April 2002, http://tools.ietf.org/html/rfc3279

[5] Trusted Computing Group. http://www.trustedcomputinggroup.org/

[6] Trusted Platform Module (TPM) specifications. http://www.trustedcomputinggroup.org/resources/tpm_main_specification

[7] Trusted Computing Group. Mobile Trusted Module Specification. Version 1.0. April 2010. Available at: http://www.trustedcomputinggroup.org/

[8] ITU-T ASN.1 Encoding Rules. http://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf

[9] Car Connectivity Consortium, "MirrorLink – Server Device", Version 1.1; CCC-TS-030

[10] Car Connectivity Consortium, "MirrorLink – Application Server Service", Version 1.1; CCC-TS-024

[11] Car Connectivity Consortium, "MirrorLink – Client Profile Service", Version 1.1; CCC-TS-026

[12] Car Connectivity Consortium, "MirrorLink – VNC based Display and Control"; CCC-TS-010

[13] Car Connectivity Consortium, "MirrorLink – Audio", Version 1.1; CCC-TS-012

[14] Trusted Computing Group, "Credentials Profiles Specification 1.1", May 2007. http://www.trustedcomputinggroup.org/resources/infrastructure_work_group_tcg_credential_profiles_specification

[15] Car Connectivity Consortium, "MirrorLink – Common Data Bus", Version 1.1; CCC-TS-016

[16] Car Connectivity Consortium, "MirrorLink – DAP Audit Requirements and Certificate Management", Version 1.1, CCC-TS-035

APPENDIX A – XSD SCHEMA

DAP attestationRequest XSD Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="urn:schemas-upnp-org:dapserver"
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="quali-
fied" attributeFormDefault="unqualified" id="attestationRequest">
<xs:element name="attestationRequest">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="version" minOccurs="1" maxOccurs="1">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="majorVersion" type="xs:nonNegativeInteger"
      minOccurs="1" maxOccurs="1"/>
      <xs:element name="minorVersion" type="xs:nonNegativeInteger"
      minOccurs="1" maxOccurs="1"/>
      <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
      processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:element name="trustRoot" type="xs:string" minOccurs="1"
   maxOccurs="1"/>
   <xs:element name="nonce" type="xs:string" minOccurs="1"
   maxOccurs="1"/>
   <xs:element name="componentID" type="xs:string" minOccurs="1"
   maxOccurs="1"/>
   <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
   processContents="lax"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
 </xs:complexType>
</xs:element>
</xs:schema>
```

DAP attestationResponse XSD Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="urn:schemas-upnp-org:dapserver"
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="quali-
fied" attributeFormDefault="unqualified" id="attestationResponse">
<xs:element name="attestationResponse">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="version" minOccurs="1" maxOccurs="1">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="majorVersion" type="xs:nonNegativeInteger"
      minOccurs="1" maxOccurs="1"/>
      <xs:element name="minorVersion" type="xs:nonNegativeInteger"
      minOccurs="1" maxOccurs="1"/>
      <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
      processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:element name="result" minOccurs="0" maxOccurs="1" default="0"/>
   <xs:element name="sizeOfSelect" minOccurs="0"
   maxOccurs="1" default="3">
    <xs:simpleType>
     <xs:restriction base="xs:unsignedShort">
      <xs:minInclusive value="2"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="attestation " minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="componentID" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
      <xs:element name="oldValue" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
      <xs:element name="quoteInfo" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
      <xs:element name="quoteSignature" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
      <xs:element name="URL" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
      <xs:element name="applicationPublicKey" type="xs:string"
      minOccurs="1" maxOccurs="1"/>
      <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
      processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:element name="deviceCertificate" type="xs:string"
   minOccurs="0" maxOccurs="1"/>
   <xs:element name="manufacturerCertificate" type="xs:string"
   minOccurs="0" maxOccurs="3"/>
   <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
   processContents="lax"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
 </xs:complexType>
</xs:element>
</xs:schema>
```