Car Connectivity Consortium

MirrorLink$^{®}$

Handling of Application Developer Certificates

Version 1.1.6
(CCC-TS-044)

VERSION HISTORY

Version	Date	Comment
1.1.0	22 March 2013	Approved Version
1.1.1	18 June 2013	Approved Errata Version
1.1.2	12 September 2013	Approved Errata Version
1.1.3	05 November 2013	Approved Errata Version
1.1.4	18 March 2014	Approved Errata Version
1.1.5	12 June 2014	Approved Errata Version
1.1.6	18 March 2015	Approved Errata Version

LIST OF CONTRIBUTORS

Brakensiek, Jörg (Editor)	Microsoft Corporation
Pichon, Ed	E-Qualus (for CCC)

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.

Copyright © 2011-2015. CCC LLC.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TERMS AND ABBREVIATIONS

ACMS	Application Certification Management System
BT	Bluetooth
ML	MirrorLink
OCSP	Online Certificate Status Protocol
RFB	Remote Framebuffer
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 specifies the handling of MirrorLink Application Certificates from the MirrorLink Server device.

The specification lists a series of requirements, either explicitly or within the text, which are mandatory elements for a 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 [1].

1. MUST: This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.
2. MUST NOT: This phrase, or the phrase "SHALL NOT", mean that the definition is an absolute prohibition of the specification.
3. SHOULD: This word, or the adjective "RECOMMENDED", mean 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" mean 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 Developer Application Concept

MirrorLink provides the ability to run certified applications on MirrorLink server devices that can be launched from the MirrorLink client device. In order to improve safety and ensure a quality user experience, an application certification program is implemented that will control which applications can be used with MirrorLink in drive on in non-drive situations. Application developers will be able to use specific application development certificates, which simplifies the development of applications on the one side, but which will be usable only on a small set of MirrorLink Server devices – as well as a potentially restricted set of MirrorLink Client devices.

Each application under development, which can be uniquely identified by a platform specific application identifier (App ID), will come with an Application Development Certificate (App Dev Certificate), which contains the App ID; necessary application information, provided to the MirrorLink Client (App Info); and the Developer ID (Dev ID). The Application Development Certificate is self-signed by either the application developer or the MirrorLink Server's software development kit.

The MirrorLink Server will use the information from the App Development Certificate to validate the MirrorLink Application, and to link it to the Developer Identifier Certificate (Dev ID Certificate). The Dev ID Certificate contains a unique Developer Identifier (Dev ID), and one or more Server Device Identifiers (Server Device IDs) for which the Dev ID Certificate is valid. An optional list of Client Device Identifiers (Client Device IDs) defines a black list of client devices, for which the Dev ID Certificate is not valid.

As shown in Figure 1, the App Dev and the Dev ID Certificates are stored on the MirrorLink Server Device. It is the responsibility of the MirrorLink Server to check, whether the Dev ID Certificate has not been revoked and whether it is valid for the MirrorLink Server and Client combination. In case the App Dev Certificate is valid, the corresponding MirrorLink application will be presented to the MirrorLink Client Device as an application coming with a certificate distributed by CCC.

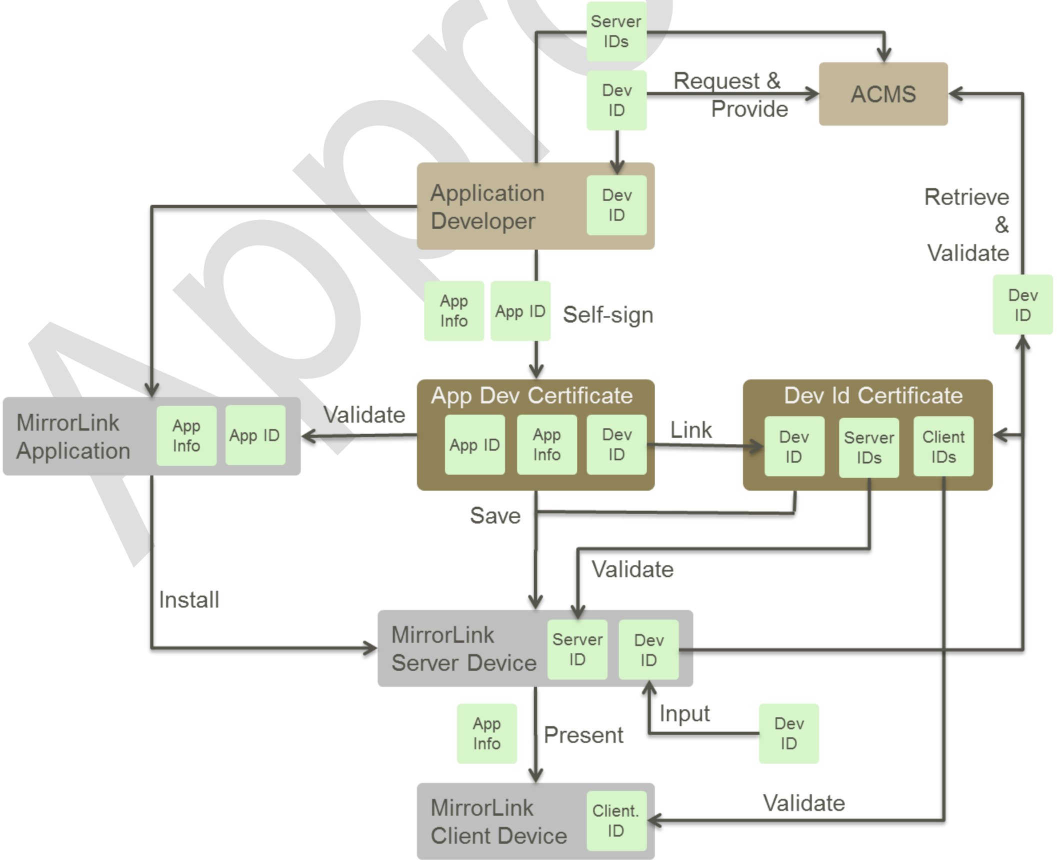

Figure 1: Application Developer Certification Architecture (MirrorLink Server View)

A MirrorLink Client will not see the difference from any regular non-development version, besides a different signing entity name and an additional X.509 v3 extension.

Support for development applications as described above MAY be restricted to specific MirrorLink Server Developer devices; those MUST be made available to application developers. Therefore, a regular MirrorLink Server device MAY NOT be able to run development applications as certified applications.

3 APPLICATION DEVELOPER CERTIFICATE STRUCTURE

3.1 Application Development Certificate

MirrorLink Application Development Certificates MUST be a public key X.509 version 3 certificate as specified in [2].

The certificate is a self-signed certificate. The signing authority MUST NOT set an expiration date of longer than 1 month from the date of signing.

Application Development Certificate MUST use 2048-bit RSA keys with SHA-256 or SHA-512 signature algorithms.

3.1.1 Extension Header

The X.509 extension header MUST have the following format:

```
X509v3 extensions:

   CCC-MirrorLink-Developer-Id Extension:
        extnId:    1.3.6.1.4.1.41577.3.1
        critical:  no
        extnValue: DER:OCTET STRING

   CCC-MirrorLink Extension:
        extnId:    1.3.6.1.4.1.41577.2.1
        critical:  no
        extnValue: DER:<DER encoded XML, as specified below>
```

3.1.2 Extension Values

3.1.2.1 CCC-MirrorLink-Developer-Id

Developer Id, as provided from the Application Certification Management System (ACMS), MUST be formatted as a character string of up to 40 alphanumeric characters (`‘a’-‘z’, ‘0’-‘9’`).

3.1.2.2 CCC-MirrorLink Extension Value

The DER encoded XML of the application information, as specified in [8].

The Signing Entity Name of application development certificates MUST be “`DEVELOPER`”.

3.2 Developer Identification Certificate

The MirrorLink Dev ID Certificate MUST be a public key X.509 version 3 certificate as specified in [2].

The certificate MUST be signed by the CCC’s Root Certificate. A hierarchy of certification authorities (CAs) MAY be used for Dev ID certificates. In case intermediate CAs are used, the entire certificate chain up to the root CA MUST be provided to the MirrorLink Server together with the Dev ID certificate. Any intermediate certificate MUST NOT have an expiration date of more than 1 year from the date of signing.

The Intermediate certificate, which is signed by the CCC root CA, MUST have a Common Name (CN) in the issuer information, identical to “`ACMS CA`”; otherwise the certificate MUST NOT be accepted. A valid example issuer information is given below:

```
Issuer: O=Car Connectivity Consortium, CN=ACMS CA
```

Any intermediate certificate MUST use 4096-bit RSA keys with SHA-512 signature algorithms.

3.2.1 Extension Header

The X.509 extension header MUST have the following format:

```
X509v3 extensions:

    CCC-MirrorLink-Developer-Id Extension:
        extnId:    1.3.6.1.4.1.41577.3.1
        critical:  no
        extnValue: DER:OCTET STRING

    CCC-MirrorLink-Developer-Server-Ids Extension:
        extnId:    1.3.6.1.4.1.41577.3.2
        critical:  no
        extnValue: DER:OCTET STRING

    CCC-MirrorLink-Client-Manufacturer-Ids Extension:
        extnId:    1.3.6.1.4.1.41577.3.3
        critical:  no
        extnValue: DER:OCTET STRING

```

3.2.2 Extension Values

3.2.2.1 CCC-MirrorLink-Developer-Id

Developer Id, as provided from the Application Certification Management System (ACMS), MUST be formatted as a character string of up to 40 alphanumeric characters (`‘a’-‘z’, ‘0’-‘9’`).

3.2.2.2 CCC-MirrorLink-Developer-Server-Ids

A comma-delimited list of Server Ids, for which the Dev ID certificate is valid; each entry MUST be formatted as a string (UTF-8).

Server IDs are the IMEI/IMEISV number (or equivalent unique identifier) of the MirrorLink Server devices on which development applications can be used.

3.2.2.3 CCC-MirrorLink-Client-Manufacturer-Ids

Comma-separated list of MirrorLink Client manufacturer Ids, for which the Dev ID certificate is not valid (black list). Each entry MUST be formatted as a string (UTF-8).

Each list entry represents a manufacturer name, and MUST match the manufacturer name (as provided from the UPnP Client Profile service [9]) or the `AppCertFilter`’s entity name (as used in the UPnP Application Server service [6]).

3.3 Root Certificate

The signing certification authority’s Root Certificate, a hash of it or a hash of its public key MUST be stored in the MirrorLink Server. Access to the certificate’s public key MUST be read-only.

Expiration date of the root certificate MUST be 20 year from the date of signing.

Root certificate MUST use 4096-bit RSA keys with SHA-512 signature algorithms. The root certificate MUST be identical to the DAP root certificate.

4 DEVELOPER IDENTIFICATION CERTIFICATE LIFE CYCLE

4.1 Certificate Retrieval and Validation

4.1.1 Certificate Retrieval

The MirrorLink Server MUST use HTTP-GET to obtain the MirrorLink Dev ID certificate from the Application Certification Management System using the following URL:

```
http://acms.carconnectivity.org:80
```

The following GET command MUST be used to obtain the application certificate:

```
GET /obtainDeveloperCertificate.html?
      certificateVersion=1.0&
      developerID=<Developer Identifier>&
      serverID=<Server Identifier>
HTTP/1.1<CR><LF>
Host: acms.carconnectivity.org:80<CR><LF>
<CR><LF>
```

The provided "serverID" MUST uniquely identify a particular MirrorLink Server device. The MirrorLink Server MUST use the IMEI/IMEISV number (or equivalent unique identifier) of the MirrorLink Server device for "serverID". Devices without an IMEI/IMEISV number MUST NOT be used for Application development at this time.

The MirrorLink Server MUST retrieve the Dev ID Certificate before it can use any self-signed application development certificates. The MirrorLink Server MUST NOT retrieve the Dev ID Certificate, unless the device is going to be used for MirrorLink application development.

The MirrorLink Server MUST NOT install any Dev ID Certificate, if it has not passed the DAP Audit with respect to the Application Developer section.

The ACMS's HTTP Server MUST return the Dev ID certificate and the entire chain of intermediate certificates, Base 64 encoded. Blank lines separate the certificates, starting from the certificate signed directly by the CCC root CA.

Otherwise it MUST provide one of the following error codes:

HTTP Error Code	CCC Error Code	Description
1xx	N/A	MirrorLink Server MUST handle the HTTP response in compliance with the HTTP protocol (implementation specific).
200	N/A	MirrorLink Server MUST validate the receive application certificate, in accordance with section **Error! Reference source not found.**.
2xx	N/A	MirrorLink Server MUST handle the HTTP response in compliance with the HTTP protocol (implementation specific).
3xx	N/A	MirrorLink Server MUST handle the HTTP response in compliance with the HTTP protocol (implementation specific).
400	N/A	Bad request – The request cannot be fulfilled due to bad syntax (e.g. missing, empty or wrongly formatted parameter). The MirrorLink Server MUST NOT retry the request.
4xx	N/A	MirrorLink Server MUST NOT retry the request
500	800	No certificate available for the given parameter

HTTP Error Code	CCC Error Code	Description
		The MirrorLink Server SHOULD retry the request.
500	801	Certification Database currently offline The MirrorLink Server MUST retry between 1h and 24h after the last HTTP-Get attempt.
500	8xx	Reserved for future use The MirrorLink Server SHOULD retry the request.
500	900	Certificate has been revoked. The MirrorLink Server MUST NOT retry the request.
500	9xx	Reserved for future use The MirrorLink Server MUST NOT retry the request.
500	xxx	Reserved for future use The MirrorLink Server SHOULD retry the request.
5xx	N/A	The MirrorLink Server SHOULD retry the request.

Table 1: Certificate Retrieval Error Codes

If the ACMS HTTP Server returns with an error response other than 200 (Ok response), the MirrorLink Server MUST consider the Dev ID certificate as not being available and any development application linked to the developer ID MUST be considered a MirrorLink aware-application only. The MirrorLink Server SHOULD retry the HTTP-Get request, unless otherwise stated above. If the MirrorLink Server retries the request, it MUST retry between 50% and 100% of the query period since the last HTTP-Get request. If no automatic retry is provided, the MirrorLink Server MUST provide a manual retry option (Note: there are no time constraints in case of manual retry).

4.1.2 Certificate Validation

The validation of Dev ID certificates is following the steps below:

1. Validate the Dev ID certificate and trust chain
2. Validate the MirrorLink Server identifier is in the list of Server IDs, as given in the CCC-MirrorLink-Developer-Server-Id X.509 extension.
3. Validate the MirrorLink Client's manufacturer identifier is not within the black list of certified manufacturer identifiers, as given in the CCC-MirrorLink-Developer-Manufacturer -Ids X.509 extension, as specified in chapter 3.2.2.3.

The MirrorLink Server MUST execute all certificate validation steps at MirrorLink connection setup, prior to including any development application into any certified application listing.

If any of the steps fail, all development applications, linked to the developer ID, MUST be considered to be non-certified and the MirrorLink Server MUST NOT add them to the certified application list (`A_ARG_TYPE_CertifiedAppList`).

Applications, which failed validation, MAY be included in the regular application list (`A_ARG_TYPE_AppList`). In that case, the applications MUST NOT have a trust level of "`Application Certificate`".

4.1.3 Testing Considerations

For Certification Validation testing purposes during MirrorLink device certification, the MirrorLink Server MUST accept the CTS root certificate to validate application certificates distributed by the ACMS. This Test Mode MUST NOT be accessible in production devices.

4.2 Certificate Revocation Checks

4.2.1 Revocation Protocol

The MirrorLink Server MUST use the Online Certificate Status Protocol (OCSP) [5] to verify the status of Dev ID certificate. The URI, where the MirrorLink Server MUST ask for the certificate status, MUST be available from the AuthorityInfoAccess (AIA) field, as defined in[3], in the certificate.

The MirrorLink Server MUST include a Nonce extension, with a random nonce value, into the OCSP request to prevent any replay attack. OCSP responses MUST be signed, with the signature algorithm and key of the issuing certificate. The signature algorithm MUST be RSA with at least 2048 bits with at least SHA-256.

The MirrorLink Server MUST use OCSP over HTTP to send and receive OCSP requests and responses. Their formatting is specified in Appendix A of [5].

The MirrorLink Server MUST take the following actions for the respective application certificate, in case the `ocspResponseStatus` has a value, indicated below:

• tryLater:	SHOULD retry
• internalError:	SHOULD retry
• malformedRequest:	MUST NOT send any further OCSP requests
• sigRequired:	MUST NOT send any further OCSP requests
• unauthorized:	MUST NOT send any further OCSP requests

The MirrorLink Server MUST take the following actions for the respective application certificate, in case the `ocspResponseStatus` is `successful` and the `certStatus` has a value, indicated below:

• unknown:	MUST NOT send any further OCSP requests
• good:	See section 4.2.2
• revoked:	See section 4.2.3 and 4.2.4

The MirrorLink Server SHOULD retry the OCSP request, in case OCSP response fails validation at least one of the following checks:

- Validation of the certificate trust chain
- Validation of the response signature
- Validation that the nonce value matched the one from the OCSP request

The MirrorLink Server SHOULD retry the OCSP request, unless otherwise stated above. If the MirrorLink Server retries the request, it MUST retry between 50% and 100% of the query period since the last OCSP request. If no automatic retry is provided, the MirrorLink Server MUST provide a manual retry option (Note: there are no time constraints in case of manual retry).

4.2.2 Certificate Valid

The MirrorLink Server MUST consider an developer ID certificate to be valid, if

- The OCSP “`certStatus`” is “`good`”.

4.2.3 Certificate Revoked

The MirrorLink Server MUST consider the developer ID certificate to be revoked if

- The OCSP “`certStatus`” is “`revoked`” and
- The ACMS returns the HTTP-Get response with Error Code 500/900, when requesting a new certificate.

The MirrorLink Server MUST NOT send any further HTTP-Get and OCSP request for a revoked developer ID certificate.

4.2.4 Certificate Updated

A developer ID certificate MUST be updated in the following cases

1. Current Dev ID certificate expired
2. Server or Client identifier fields in the certificate changed within the ACMS

The MirrorLink Server MUST consider the developer ID certificate as to be updated if

- The OCSP "`certStatus`" is "`revoked`" and
- The ACMS returns the HTTP-Get response with Error Code 200, when requesting a new certificate.

The retrieved updated application certificate MUST be validated (in accordance with section **Error! Reference source not found.**), including an OCPS request for the updated developer ID certificate.

4.2.5 Testing Consideration

For OCSP testing purposes during MirrorLink device certification, the MirrorLink Server MUST accept the CTS root certificate to validate responses from the ACMS. This Test Mode MUST NOT be accessible in production devices.

4.3 Query and Grace Periods

4.3.1 Query Period

The MirrorLink Server MUST verify from the ACMS, whether the developer ID certificate has been revoked. Validation MUST happen within 50% and 100% of the query period, as defined in [8].

Developer ID certificates MUST use the same query period as regular application certificates. If the query period is set to 0, the Developer ID MUST be checked on MirrorLink connection setup and at least every 24 h, while the MirrorLink connection lasts.

Failure to receive a revocation list update, after the query period, MUST invalidate the Dev ID certificate. The MirrorLink Server MUST remove any development application immediately from the certified application listing, if the query period expires during a MirrorLink connection. The MirrorLink Server MAY still provide access to the application certificate via the `appCertificateURL` entry in the UPnP application listing.

4.3.2 Grace Period

The MirrorLink Server MUST NOT allow for any Grace Period for Developer ID certificates.

5 APPLICATION DEVELOPMENT CERTIFICATE LIFE CYCLE

5.1 Certificate Retrieval and Validation

5.1.1 Certificate Retrieval

The MirrorLink Server manufacturer MUST provide a mechanism to create and self-sign an application development certificate, including the necessary application identifier. The application development certificate MUST include the developer identifier from the Dev ID Certificate.

The MirrorLink Server MUST NOT access the ACMS, using an HTTP-Get request, to retrieve any application development certificate.

5.1.2 Certificate Validation

The validation of Application Development certificates is following the steps below:

1. Validate the application development certificate
2. Validate the application identifier is identical with the given app id value.
3. Validate the developer identifier is identical to the validated one in the Developer ID certificate

The MirrorLink Server MUST execute all certificate validation steps at MirrorLink connection setup, prior including any development application into any certified application listing.

In case the certificate has been validated, the MirrorLink Server MUST treat the application as if the certificate has been signed by the CCC, i.e.

- The MirrorLink Server MUST apply all rules specified for CCC certified applications in [8]
- The MirrorLink Server MUST replace the Entity Name "`DEVELOPER`" by "`CCC`" within any provided A_ARG_TYPE_AppCertificateInfo structure
- The MirrorLink Server MUST include them into any response, when the MirrorLink Client is looking for "`CCC`" certified applications.

If any of the steps fail, the development application MUST be considered to be non-certified and the MirrorLink Server MUST NOT add it to the certified application list (`A_ARG_TYPE_CertifiedAppList`).

Applications, which failed validation, MAY be included in the regular application list (`A_ARG_TYPE_AppList`). In that case, the application MUST NOT have a trust level of "`Application Certificate`".

5.1.3 Certificate Update

An application development certificates MUST be updated in the following cases

1. New version of the application has been installed (the original certificate becomes invalid)
2. Current application development certificate expired
3. New application developer identifier

The MirrorLink Server manufacturer MUST provide a mechanism to update a self-sign an application development certificate, including the necessary application identifier.

On update of an application development certificate, the MirrorLink Server MUST immediately validate the application development certificate.

5.2 Certificate Revocation Checks

Application development certificates are not subject to any Revocation Protocol.

The MirrorLink Server MUST NOT use OCSP requests to check with the ACMS the revocation status of any application development certificate.

6 References

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

[2] IETF, RFC 3281, “An Internet Attribute Certificate Profile for Authorization”, April 2002, http://www.ietf.org/rfc/rfc3281.txt

[3] IETF, RFB 2459, “Internet X.509 Public Key Infrastructure Certificate and CRL Profile”, January 1999, http://www.ietf.org/rfc/rfc2459.txt

[4] IETF, RFC 5280, “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile”, May 2008, http://tools.ietf.org/html/rfc5280

[5] IETF, RFC 2560, “X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP”, June 1999, http://tools.ietf.org/html/rfc2560

[6] Car Connectivity Consortium, “MirrorLink - Application Server Service”, Version 1.1; CCC-TS-024

[7] Car Connectivity Consortium, “MirrorLink – VNC based Display and Control”, Version 1.1, CCC-TS-010

[8] Car Connectivity Consortium, “MirrorLink – Handling of Application Certificates”, Version 1.1, CCC-TS-036

[9] Car Connectivity Consortium, “MirrorLink – UPnP Client Profile Service”, Versions 1.1, CCC – TS-026

APPENDIX A – OCSP REQUEST & RESPONSE EXAMPLE

An example OCSP request is given below.

```
OCSP Request Data:
    Version: 1 (0x0)
    Requestor List:
        Certificate ID:
          Hash Algorithm: sha1
          Issuer Name Hash: 8809099A55F562E1EA13273360FFB7F50B76F508
          Issuer Key Hash: BF37183EB53B43AD1D7237E59CE2FC4DF96C7BFC
          Serial Number: 6D
    Request Extensions:
        OCSP Nonce:
            041035DA009D2912E3CEC403D34B319228D9
```

An example OCSP response is given below, which includes the query and grace periods update.

Note: The phrase < ... > indicates that the content has been shortened for readability purpose.

```
OCSP Response Data:
    OCSP Response Status: successful (0x0)
    Response Type: Basic OCSP Response
    Version: 1 (0x0)
    Responder Id: BF37183EB53B43AD1D7237E59CE2FC4DF96C7BFC
    Produced At: May 16 13:28:35 2013 GMT
    Responses:
    Certificate ID:
      Hash Algorithm: sha1
      Issuer Name Hash: 8809099A55F562E1EA13273360FFB7F50B76F508
      Issuer Key Hash: BF37183EB53B43AD1D7237E59CE2FC4DF96C7BFC
      Serial Number: 69
    Cert Status: good
    This Update: May 16 13:28:35 2013 GMT
    Response Extensions:
        1.3.6.1.4.1.41577.1.1:
            24
        1.3.6.1.4.1.41577.1.3:
            22
        OCSP Nonce:
            04101F0696B93BB03B5E84955AA32E16535F
        1.3.6.1.4.1.41577.1.2:
            12
    Signature Algorithm: sha512WithRSAEncryption
         5b:75:04:e2:40:12:fa:ea:85:67:c0:75:29:2b:b0:04:9a:8a:
         < ... >
         aa:de:96:58:4a:14:e3:6e:cc:28:92:f3:a9:cc:13:8e:f5:a7:
         62:00:51:b5:8d:53:1f:1e
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 12034049345340335056 (0xa701881ed68863d0)
    Signature Algorithm: sha512WithRSAEncryption
        Issuer: CN=CCC Root CA, O=Car Connectivity Consortium
        Validity
            Not Before: Apr 25 09:34:45 2013 GMT
            Not After : Oct 17 09:34:45 2032 GMT
        Subject: O=Car Connectivity Consortium, CN=ACMS CA
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:a3:8e:31:a8:dc:43:51:78:f8:c6:c8:a9:12:22:
                    < ... >
                    7e:e4:36:a8:01:51:ed:c7:4d:a3:9d:e8:62:9f:36:
                    03:10:25
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                BF:37:18:3E:B5:3B:43:AD:1D:72:37:E5:9C:E2:FC:4D:
                F9:6C:7B:FC
            X509v3 Authority Key Identifier:
                keyid:52:7C:16:40:94:8A:E4:D7:BA:01:24:72:AB:1E:95:E3:
                1A:12:0C:C3
                DirName:/CN=CCC Root CA/O=Car Connectivity Consortium
                serial:E3:EE:B1:5C:85:7B:63:B6
            X509v3 Basic Constraints:
                CA:TRUE
            X509v3 Key Usage:
                Certificate Sign, CRL Sign
    Signature Algorithm: sha512WithRSAEncryption
         1c:a1:c6:a2:ed:89:5d:19:ee:f1:07:1c:eb:c0:92:7e:d1:25:
         < ... >
         86:5b:a3:cc:45:1d:0a:4e:6f:ae:50:9e:80:a2:32:8f:7c:8d:
         cc:ed:75:81:63:be:83:31
-----BEGIN CERTIFICATE-----
MIIFozCCA4ugAwIBAgIJAKcBiB7WiGPQMA0GCSqGSIb3DQEBDQUAMDwxFDASBgNV
< ... >
EJaXdG/6JqHvY0sYyorzqjiPk/ww7sL+f0Nowu6GW6PMRR0KTm+uUJ6AojKPfI3M
7XWBY76DMQ==
-----END CERTIFICATE-----
Response verify OK
devCert.crt: good
      This Update: May 16 13:28:35 2013 GMT
```

APPENDIX B – APPLICATION DEVELOPER CERTIFICATE EXAMPLE

An example Application Certificate is given below.

Note: The phrase < ... > indicates that the content has been shortened for readability purpose.

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 105 (0x69)
    Signature Algorithm: sha512WithRSAEncryption
        Issuer: O=Car Connectivity Consortium, CN=ACMS CA
        Validity
            Not Before: May 16 00:29:28 2013 GMT
            Not After : Jul 23 00:29:28 2023 GMT
        Subject: CN=16542e60939048fba856ccd034b07f8b0506e249
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:d3:72:b9:cf:61:78:91:a5:b2:69:84:f0:77:34:
                    < ... >
                    4d:2e:49:ab:4b:50:c2:83:06:41:4f:6c:72:24:87:
                    97:b5
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature
            X509v3 Extended Key Usage:
                TLS Web Client Authentication
            1.3.6.1.4.1.41577.3.1:
                16542e60939048fba856ccd034b07f8b0506e249
            1.3.6.1.4.1.41577.3.2:
                1234,12345,123456,1,2,3,4,5,6,7,8,9,10,1234,
                < ... >
                34,12345,123456,1,2,3,4,5,6,7,8,9,10
            1.3.6.1.4.1.41577.3.3:
                EMPTY
    Signature Algorithm: sha512WithRSAEncryption
         01:da:0b:01:b9:1d:79:60:17:c1:e5:9e:97:00:29:d8:09:c4:
         < ... >
         ce:a7:b6:02:c4:c2:11:8c:16:3a:b5:ed:33:13:0f:bd:c4:bb:
         79:c4:b2:90:f0:e9:88:db
```