ETSI TS 103 544-14 V1.3.0 (2017-10)

Publicly Available Specification (PAS); Intelligent Transport Systems (ITS); MirrorLink®; Part 14: Application Certificates

Reference

DTS/ITS-88-14

Keywords

interface, ITS, PAS, smartphone

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Intelligent Transport Systems (ITS).

The present document is part 14 of a multi-part deliverable. Full details of the entire series can be found in part 1 [i.1].

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document is part of the MirrorLink® specification which specifies an interface for enabling remote user interaction of a mobile device via another device. The present document 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.

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.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document.

[1] IETF RFC 3281: "An Internet Attribute Certificate Profile for Authorization", April 2002.

NOTE: Available at http://www.ietf.org/rfc/rfc3281.txt.

[2] IETF RFC 2459: "Internet X.509 Public Key Infrastructure Certificate and CRL Profile", January 1999.

NOTE: Available at http://www.ietf.org/rfc/rfc2459.txt.

[3] IETF RFC 6960: "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", June 2013.

NOTE: Available at http://tools.ietf.org/html/rfc6960.

[4] ETSI TS 103 544-16 (V1.3.0): "Publicly Available Specification (PAS); Intelligent Transport Systems (ITS); MirrorLink®; Part 16: Application Developer Certificates".

[5] ETSI TS 103 544-9 (V1.3.0): "Publicly Available Specification (PAS); Intelligent Transport Systems (ITS); MirrorLink®; Part 9: UPnP Application Server Service".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 103 544-1 (V1.3.0): "Publicly Available Specification (PAS); Intelligent Transport Systems (ITS); MirrorLink®; Part 1: Connectivity".

3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ACMS	Application Certification Management System
BT	Bluetooth
CCC	Car Connectivity Consortium
ML	MirrorLink
OCSP	Online Certificate Status Protocol
RFB	Remote Framebuffer
UPnP	Universal Plug and Play
USB	Universal Serial Bus
VNC	Virtual Network Computing

4 Application Certification Concept

MirrorLink distinguishes three main categories of applications:

1) A **MirrorLink-Aware Application** describes an application that implements software interfaces, which can be used via MirrorLink. A MirrorLink-Aware Application does not have MirrorLink or CCC Member certification, as described below.

2) A **MirrorLink-Certified Application** describes the certification status of a MirrorLink-Aware Application, which is additionally fulfilling CCC application certification criteria. This category comes in two sub categories:

   a) A **MirrorLink Base-Certified Application** is fulfilling CCC application certification criteria for basic MirrorLink Client and Server interoperability, usability and reliability.

   b) A **MirrorLink Drive-Certified Application** is a MirrorLink Base-Certified Application, which is additionally approved by the CCC for use in a MirrorLink Client and Server system by a driver, while the vehicle is in motion.

3) A **Member-certified Application** describes the certification status of a MirrorLink-Aware Application, which is additionally fulfilling CCC Member application certification criteria. This category comes in two sub-categories:

   a) A **Member Base-Certified Application** is fulfilling the CCC Member's certification criteria for basic MirrorLink Server and CCC Member's MirrorLink Client interoperability, usability and reliability.

   b) A **Member Drive-Certified Application** is a Member Base-Certified Application and is approved by the CCC Member for use in a MirrorLink Server and CCC Member's MirrorLink Client system by a driver, while the vehicle in in motion.

Certified applications will have an Application Certificate containing information about the application, relevant for allowing it in drive or non-drive mode (App Info), along with information (App ID) how the application can be securely identified on the MirrorLink Server device.

As shown in Figure 1, an application is downloaded from any application store and installed on the MirrorLink Server device. The application may come with a self-signed application certificate, which provides necessary information for the application advertisements as a MirrorLink-Aware Application.

In addition, the MirrorLink Server will retrieve the Application's associated MirrorLink or Member Certificate from the Application Certificate Management System (ACMS). The application identification information is used to securely link the application certificate to the downloaded and installed application. The MirrorLink Server device will be able to validate with the ACMS, whether the available application certificate is still valid.

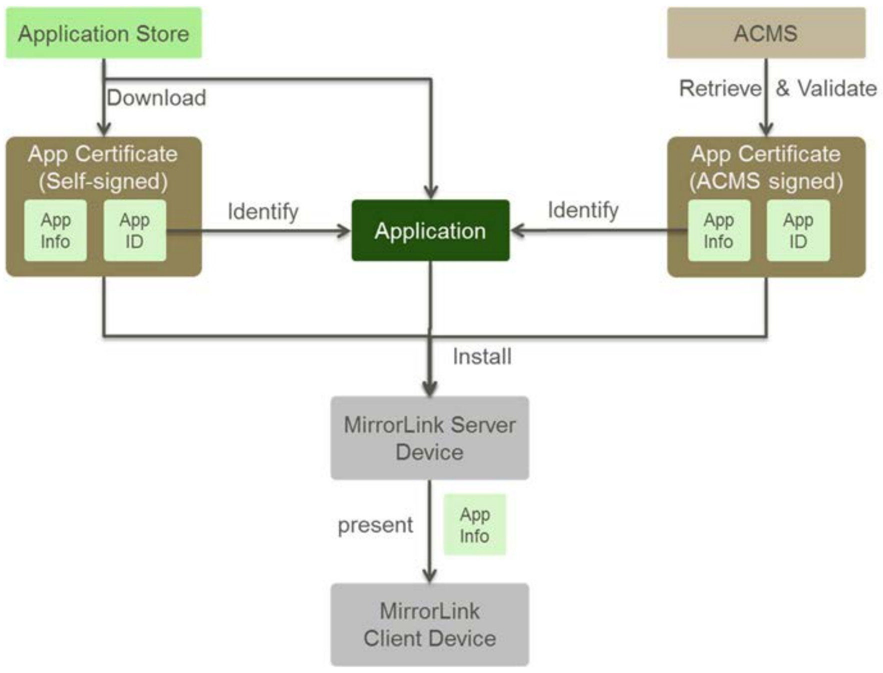

Figure 1: Application Certification Architecture (MirrorLink Server View)

The MirrorLink Server device will take the application information out of the validated application certificate and present the information to the MirrorLink Client devices.

Within the present document, we use the term **restricted** mode, to refer to the condition, when driver distraction rules have to be followed (e.g. while driving). The term **non- restricted** mode is used to refer to the condition, when driver distraction rules have not to be followed (e.g. while being parked).

5 Application Certificate Structure

5.1 X.509 Certificate

5.1.1 Application Certificate

Application Certificates shall be a public key X.509 version 3 certificate as specified in [1].

MirrorLink uses long-lived Application Certificates. The signing Certification Authority should set an expiration date of 10 years from the date of signing, but it shall not be longer than the expiration date of the signing root or intermediate certificate.

Application Certificates shall use 2048-bit RSA keys with SHA-256 or SHA-512 signature algorithms.

5.1.2 Intermediate Certificate

Hierarchy of certification authorities (CAs) may be used for application certification. In case intermediate CAs are used, the entire certificate chain up to the root CA shall be provided to the MirrorLink Server together with the application certificate.

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

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

An Intermediate Certificate should have an expiration date of 20 years from the date of signing, but it shall not be longer than the expiration date of the signing root certificate.

Any Intermediate Certificate shall use 4096-bit RSA keys with SHA-512 signature algorithms.

5.1.3 Root Certificate

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

Expiration date of the root certificate shall be 20 years from the date of signing.

Root Certificate shall use 4096-bit RSA keys with SHA-512 signature algorithms.

The Root Certificate shall be identical to the DAP Root Certificate.

5.2 MirrorLink Extension

5.2.1 Extension Header

The X.509 extension header shall have the following format. The CCC-MirrorLink Extension Id is provided from IANA. The identifier shall be provided without any "<>" delimiter. Its value is outside the scope of the present document:

```
X509v3 extensions:
    CCC-MirrorLink Extension:
        extnId:    1.3.6.1.4.1.41577.2.1
        critical:  no
        extnValue: DER:<DER encoded XML, as specified below>
```

The CCC-MirrorLink-OCSP extensions for *queryPeriod*, *driveGrace* and *baseGrace* are defined later on clause 6.4.3.

5.2.2 CCC-MirrorLink Extension Value

The DER encoded XML shall follow the format below. Detailed description of the elements can be found in Table 1.

Table 1: MirrorLink Extension Header extnValue XML

Element	Description	Parent	Availability
certificate	MirrorLink Application Certificate	-	Required
version	Version of the certificate Note: This version corresponds to the Certificate Version, mainly the structure of this XML. It does not correspond to the MirrorLink specification version.	certificate	Optional
majorVersion	Major Version (it shall be 1) **Type**: Unsigned integer **Default**: 1	version	Optional

Element	Description	Parent	Availability
minorVersion	Minor Version **Type**: Unsigned integer **Default**: 0	version	Optional
appIdentifier	Platform specific application identifier (defined in Annex B)	certificate	Required
appListEntry	Application entry for the UPnP Application Server Service A_ARG_TYPE_AppList listing	certificate	Required
name	Application name	appListEntry	Required
provider Name	Name of the application provider	appListEntry	Optional
providerURL	URL of the application provider's website	appListEntry	Optional
description	Text description of application	appListEntry	Optional
iconList	List of available application icons	appListEntry	Platform specific
icon*	Describes an application icon The MirrorLink server shall include an icon for all applications with <protocolID> = VNC.	iconList	Platform specific
mimetype	Type of icon image (A_ARG_TYPE_String)	icon	Required
width	Width of icon (A_ARG_TYPE_INT)	icon	Required
height	Height of icon (A_ARG_TYPE_INT)	icon	Required
depth	Color depth of icon (A_ARG_TYPE_INT)	icon	Required
url	URL where icon is available within the install package (platform specific). (A_ARG_TYPE_URI)	icon	Required
appInfo	Information about the listed application	appListEntry	Optional
appCategory	Application category	appInfo	Optional
displayInfo	Information about display content	appListEntry	Optional
content Category	Visual content categories used	displayInfo	Optional
audioInfo	Information about audio content	appListEntry	Optional
audioType	Audio type	audioInfo	Optional
content Category	Audio content categories used	audioInfo	Optional
appCert InfoEntry	Application entry for the UPnP Application Server Srevice A_ARG_TYPE_AppCertificateInfo listing	certificate	Required
appUUID	UUID of the application The UUID shall be unique across all mobile device platform variants and application versions.	appCert InfoEntry	Optional
entity*	Certifying entity	appCert InfoEntry	Optional
name	Entity name Unique identifier of the entity certifying the application. Allowed values are specified in Table 2.	entity	Required
targetList	Target	entity	Optional
target*	Target name Entry is undefined in case of the CCC entity and shall be ignored from the MirrorLink Client. Otherwise the format is OEM specific. The OEM may use this entry to implement a white and/or black list of supported targets.	targetList	Required
restricted	List of locales for restricted use	entity	Required
non Restricted	List of locales for non-restricted use	entity	Required

Element	Description	Parent	Availability
serviceList	List of used data services	entity	Required
service*	Service name	serviceList	Required
properties	Application properties Contains an UTF-8 XML representation of certified application properties. The XML representation is out-of-scope of the present document.	appCert InfoEntry	Optional
server Properties	MirrorLink Server Properties	certificate	Required
platform	Platform supported from application A separate certificate is provided for each platform/runtime.	server Properties	Required
platformID	Platform identifier, as defined in Table 3.	platform	Required
blacklisted Platform Versions	Comma separated list of black-listed platform versions. Version information is platform specific. Version information shall be complete. An empty version tag indicates that the application certificate is not dependent on a specific version of the host OS in question.	platform	Required
runtimeID	Runtime identifier, as defined in Table 4	platform	Required
blacklisted RuntimeVersions	Comma separated list of black-listed runtime versions. Version information is runtime specific. Version information shall be complete. An empty version tag indicates that the application certificate is not dependent on a specific runtime version.	platform	Required

Elements marked with a (*) can have multiple instances.

In case the entity is missing from the Application Certificate, the application shall be treated as a MirrorLink-Aware application.

5.2.3 Certificate Signing Entities

The following entity names are currently registered with CCC. The MirrorLink Client shall ignore any unknown entries.

Table 2: Certificate Signing Entities

Entity Name	Description
CCC	Car Connectivity Consortium The application follows application guidelines, as specified from CCC, for the certified regions.
DEVELOPER	MirrorLink Developer Application The application is a developer self-signed MirrorLink aware application, as specified in [4]. The application need not follow any application guidelines, as specified from the CCC.
ACMS	MirrorLink Aware Application The application is a self-signed MirrorLink aware application. The MirrorLink Server shall check with the ACMS for a CCC or Member-signed certificate.
<Empty String> OR Tag missing OR Unknown entity name	MirrorLink Aware Application The application is a self-signed MirrorLink aware application. The MirrorLink Server shall not check with the ACMS for a CCC or Member-signed certificate.

Entity Name	Description
<CCC Member Name>	CCC Member It is the member's responsibility that the application is following all necessary application guidelines. The unique list of CCC member names is maintained separately from the present document. The CCC member name shall be identical to the *manufacturer* entry in *A_ARG_TYPE_ClientProfile* structure as provided in the UPnP Set Client Profile service.

The MirrorLink Server shall only consider a CCC Member certified application as a certified application:

- if the *manufacturer* entry set in the UPnP Client Profile is matching the certificate's signing entity in *the A_ARG_TYPE_CertificateInfo* entry; and
- if any of the `"name"` entries in the certificate' s signing entities is matching the certificate's signing entity in the *AppCertFilter*.

The application certificate may contain divergent certification related information, originating from a CCC signing entity and from a relevant member-signing entity (according to the above statement). In case the MirrorLink Server has to decide on the certification status of those applications, it shall follow the rules given below:

- Merge the *restricted/nonRestricted* entries from the Member- and CCC-Certifying entities (Note: it does not matter, whether one locale is missing from one of the two lists, as the application is certified according to either the CCC or the Member-certification statement).
- Merge the *serviceList* entries from the Member- and CCC-Certifying entities.
- Use the *targetList* from the Member-Certifying Entity section.

NOTE: This entry does not exist for CCC-Certifying entities.

5.2.4 MirrorLink Server Platform Identifier

The following platform identifiers and their respective versions are currently registered with CCC. The platform identifier shall be used case-sensitive.

Known platform versions are covered in a separate document.

Table 3: MirrorLink Server Platform Identifier

Platform Identifier	Description
Android	Android™
WP	Windows Phone
Symbian	Symbian™
MeeGo	MeeGo™
BlackBerry	BlackBerry®

5.2.5 MirrorLink Server Runtime Identifier

The following runtime identifiers and their respective versions are currently registered with CCC. The runtime identifier shall be used case-sensitive.

Table 4: MirrorLink Server Runtime Identifier

Runtime Identifier	Description	Known Runtime Versions
Native	Native environment	NOT USED

5.2.6 Application identifier

Each application shall have an application identifier, which identifies a particular version of an application executable within the given platform. The MirrorLink Server shall be able to detect any change to the application's executable after the application certificate has been signed. The application identifier shall be used case-sensitive.

Platform specific identifiers are specified in separate documents.

5.2.7 Mapping of Locales

CCC has defined a set of guidelines for CCC drive certification in the European Union, North America and Japan. Table 5 table lists the locales (for the appropriate regions), which shall be included into the `restricted` element within the CCC signing entity, in case the application has passed the respective certification.

Table 5: Mapping of Locales for CCC Drive Certification

Regions	List of Locales
Default	EPE,RUS,AMERICA,BRA,AUS,KOR,CHN,HKG,TPE,IND,APAC,AFRICA
European Union	EU,EPE,RUS,AMERICA,BRA,AUS,KOR,CHN,HKG,TPE,IND,APAC,AFRICA
North America	EU,EPE,RUS,AMERICA,BRA,AUS,KOR,CHN,HKG,TPE,IND,APAC,AFRICA,USA, CAN
Japan	EPE,RUS,AMERICA,BRA,AUS,KOR,CHN,HKG,TPE,IND,APAC,AFRICA,JPN
Global	EU,EPE,RUS,AMERICA,BRA,AUS,KOR,CHN,HKG,TPE,IND,APAC,AFRICA,USA, CAN,JPN,WORLD

CCC has defined a set of guidelines for CCC base certification, which are currently not region independent. Table 6 lists the locales, which shall be included into the *nonRestricted* element within the CCC signing entity, in case the application has passed the respective certification.

NOTE: CCC may define regional base-certification criteria in the future.

Table 6: Mapping of Locales for CCC Base Certification

Regions	List of Locales
Default	EU,EPE,RUS,AMERICA,BRA,AUS,KOR,CHN,HKG,TPE,IND,APAC,AFRICA,USA, CAN,JPN,WORLD
European Union	
North America	
Japan	
Global	

The entries are provided from the ACMS within the X.509 certificate. The MirrorLink Server shall not modify those. The CCC may decide to change the mappings, which would then be implemented by the ACMS; this is fully transparent to the MirrorLink Client and Server devices.

6 Application Certificate Life Cycle

6.1 General

The Application Certificate Life Cycle of applications, coming with a self-signed Application Certificate, containing an `"entity"` tag with a `"name"` tag value of `"DEVELOPER"`, are specified in [4].

The high-level Application Certificate life-cycle is given below; the detailed specification is described in the following clauses:

1) The MirrorLink Server checks for the application certificate with the ACMS on application download for self-signed certificates with `"ACMS"` as a signing entity name.

2) On failure to retrieve the application certificate, the MirrorLink Server checks for the certificate again within the query period.

3) The MirrorLink Server will continue to perform step 2, until it can retrieve a certificate or after 6 months (whatever comes first); after 6 months, the MirrorLink Server will stop performing step 2 and the application is considered uncertified.

4) If a certificate is successfully retrieved by the MirrorLink Server, it performs OCSP checks within a query period

5) On failure to perform the OCSP check the MirrorLink Server performs the OCSP check again within the query period.

6) The MirrorLink Server will continue to perform step 5, until it can perform the OCSP check; after a grace period, the application will be considered unchecked; after a 6 month, the MirrorLink Server will stop performing step 5 and the application is considered uncertified.

6.2 Certificate Retrieval and Validation

6.2.1 Certificate Retrieval

An application, which is downloaded and installed from a market place or via other mechanism onto a MirrorLink certified server device shall come with a MirrorLink developer self-signed Application Certificate.

In case that developer self-signed Application Certificate, does not contain an *entity* tag or a *name* tag with an empty string, the MirrorLink Server shall treat the application as a MirrorLink-Aware Application and shall not attempt to retrieve CCC distributed certificates from the ACMS.

In case that developer self-signed Application Certificate, does contain an *entity* tag with a *name* tag value of `"ACMS"`, the MirrorLink Server shall attempt to get a CCC distributed Application Certificate from the Application Certification Management System (ACMS), as defined in clause 6.3.1. As long as the MirrorLink Application Certificate is not available, the MirrorLink application shall be considered a MirrorLink-Aware Application.

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

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

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

```
GET /obtainCertificate.html?
certificateVersion=1.0&
platformID=<Platform Identifier>&
runtimeID=<Runtime Identifier>&
```

```
appID=<applicationIdentifier>
HTTP/1.1<CR><LF>
Host: acms.carconnectivity.org:80<CR><LF>
<CR><LF>
```

NOTE: Application identifier is platform specific.

In case a sufficient data connection is available, the MirrorLink Server shall initiate the application certificate retrieval according to the following time, whatever comes first:

- Immediately (not later than 10 min) when the MirrorLink Server is already in a MirrorLink session; or
- As soon as the MirrorLink Server is entering a MirrorLink session (not later than 10 min).

In case a sufficient data connection is not available, the MirrorLink Server shall initiate the application certificate retrieval not later than 1h after a sufficient data connection becomes available.

The ACMS's HTTP Server shall return the application 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 shall provide one of the following error codes.

Table 7: Certificate Retrieval Error Codes

HTTP Error Code	CCC Error Code	Description
1xx	N/A	MirrorLink Server shall handle the HTTP response in compliance with the HTTP protocol (implementation specific).
200	N/A	MirrorLink Server shall validate the receive application certificate, in accordance with clause 6.2.2.
2xx	N/A	MirrorLink Server shall handle the HTTP response in compliance with the HTTP protocol (implementation specific).
3xx	N/A	MirrorLink Server shall 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 shall consider the application as a MirrorLink-Aware Application. The MirrorLink Server shall not retry the request.
4xx	N/A	MirrorLink Server shall not retry the request
500	800	No certificate available for the given parameter The MirrorLink Server shall consider the application as a MirrorLink-Aware Application. The MirrorLink Server shall retry between 50 % and 100 % of the query period after the last HTTP-Get attempt for at least 6 months.
500	801	Certification Database currently offline The MirrorLink Server shall consider the application as a MirrorLink-Aware Application, if a certificate is not already on the server. The MirrorLink Server shall retry between 1 h and 24 h after the last HTTP-Get attempt
500	8xx	Reserved for future use The MirrorLink Server shall retry between 50 % and 100 % of the query period after the last HTTP-Get attempt for at least 6 months.
500	900	Certificate has been revoked. The MirrorLink Server shall consider the application as a MirrorLink-Aware Application. The MirrorLink Server shall not retry the request.
500	9xx	Reserved for future use The MirrorLink Server shall not retry the request.

HTTP Error Code	CCC Error Code	Description
500	xxx	Reserved for future use The MirrorLink Server shall retry between 50 % and 100 % of the query period after the last HTTP-Get attempt for at least 6 months.
5xx	N/A	The MirrorLink Server shall retry between 50 % and 100 % of the query period after the last HTTP-Get attempt for at least 6 months.

The CCC Error Code is provided in the HTTP response header, as described in the following example for CCC Error Code 800:

```
HTTP/1.0 500 Internal Server Error
Date: Thu, 22 Aug 2013 09:25:10 GMT
Pragma: no-cache
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Cache-Control: no-cache
Cache-Control: no-store
Content-Type: text/plain
Content-Length: 3
X-Cache: MISS from firewall.HQ
X-Cache-Lookup: MISS from firewall.HQ:8080
Via: 1.0 firewall.HQ:8080 (squid/2.6.STABLE21)
Connection: close

800
```

6.2.2 Certificate Validation

If the MirrorLink Server detects a **newly** installed or reinstalled application, it shall follow the steps given below:

- Validate the application certificate (i.e. check expiration dates, format, and signature).
- Validate the certificate's trust chain.
- Validate the application identifier (i.e. check the application identifier with the installed package). This is platform specific. The application identifier for self-signed certificates shall not be validated. The application identifier from other certificates shall be validated.
- Validate that the application is certified for the MirrorLink Server's platform and runtime and that their versions are not blacklisted.
- Verify from the Application Certification Management System (ACMS) that the application certificate of the installed application has not been revoked. This step does require Internet connectivity to the ACMS.

The MirrorLink Server shall execute these steps immediately (not later than 10 min) when the MirrorLink Server is already in a MirrorLink session, or as soon as the MirrorLink Server is entering a MirrorLink session (not later than 10 min).

If any of the above steps fail, the application is considered to be non-certified and the MirrorLink Server shall not add the application to the certified application list (*A_ARG_TYPE_CertifiedAppList*). The MirrorLink Server should store the application identifier for later validation needs.

The following steps shall be executed not later than the end of the query period since the last check:

- Validate the application certificate (i.e. check expiration dates, format, and signature).
- Validate the application identifier (i.e. check the application identifier with the installed package). This is platform specific.
- Validate that the application is certified for the MirrorLink Server's platform and runtime and that their versions are not blacklisted.

If any of the steps fail, the application is non-certified and the MirrorLink Server shall not add the application to the certified application list (*A_ARG_TYPE_CertifiedAppList*).

The MirrorLink Server shall not retry to download a new application certificate in case any of the following validation steps failed:

- Validation of the trust chain failed.
- Validation of the certificate's signature failed.
- Validation of the application identifier failed.

The MirrorLink Server shall retry to download a new application certificate between 50 % and 100 % of the query period after the last HTTP-Get attempt in case the following validation steps failed:

- Application certificate is expired.
- Application is not certified for the MirrorLink Server's platform or the platform version is blacklisted.
- Application is not certified for the MirrorLink Server's runtime or the runtime version is blacklisted.

Applications, which failed validation, may be included in the regular application list (*A_ARG_TYPE_AppList*). In the case, the application shall not have a trust level of `"Application Certificate"`.

Figure 2 shows the state machine diagram of the application certificate retrieval and validation process.

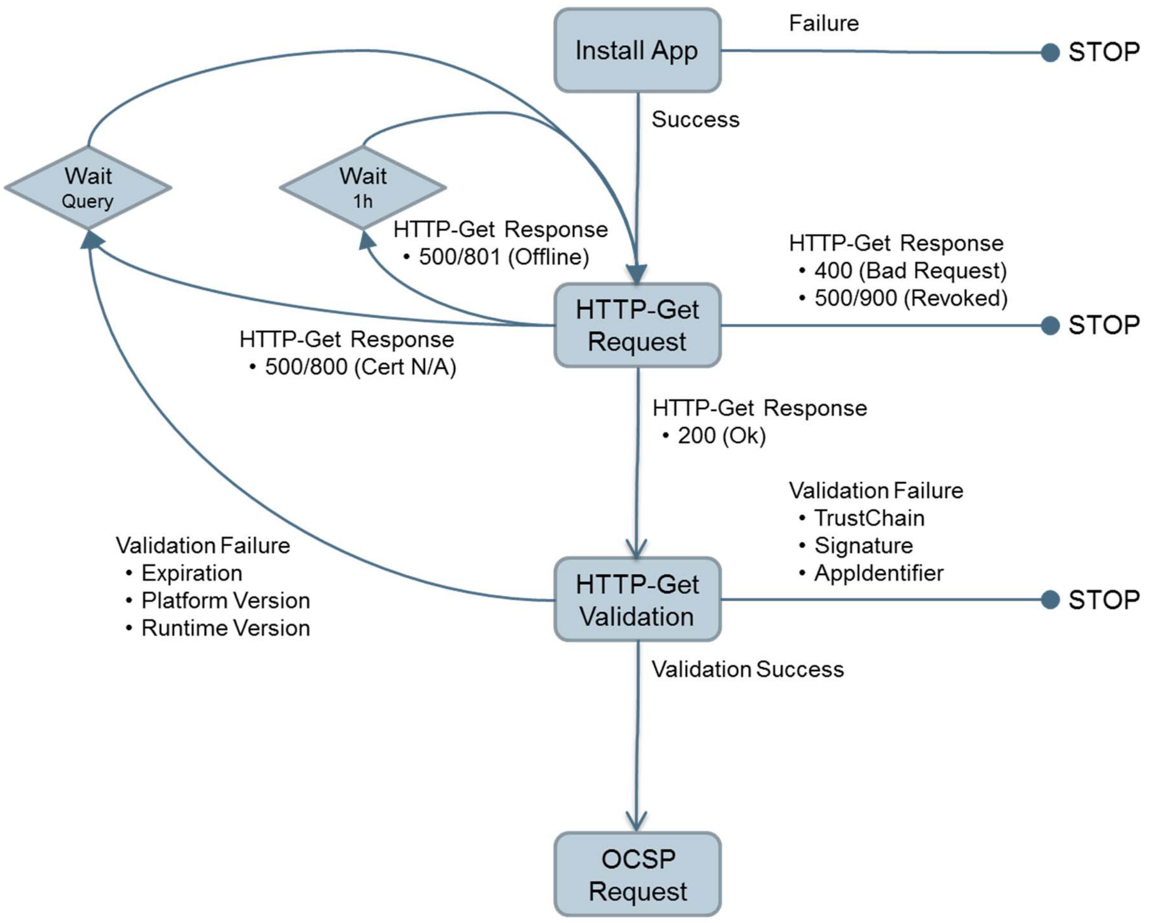
Install App
Failure
STOP
Success
Wait Query
Wait 1h
HTTP-Get Response • 500/801 (Offline)
HTTP-Get Response • 400 (Bad Request) • 500/900 (Revoked)
HTTP-Get Request
STOP
HTTP-Get Response • 500/800 (Cert N/A)
HTTP-Get Response • 200 (Ok)
Validation Failure • TrustChain • Signature • AppIdentifier
Validation Failure • Expiration • Platform Version • Runtime Version
HTTP-Get Validation
STOP
Validation Success
OCSP Request

Figure 2: State Machine Diagram - Certificate Retrieval and Validation

6.2.3 Testing Considerations

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

6.3 Certificate Revocation Checks

6.3.1 Revocation Protocol

Rather than downloading Certificate Revocation Lists (CRL), the Online Certificate Status Protocol (OCSP) [3] is used to verify the status of certificates. The URI, where the MirrorLink Server shall ask for the certificate status, shall be available from the *AuthorityInfoAccess* (AIA) field, as defined in [2], in the certificate. OCSP responses are signed.

The structure of the OCSP request is given in:

```
OCSPRequest := SEQUENCE {
  tbsRequest                  TBSRequest,
  optionalSignature       [0] EXPLICIT Signature OPTIONAL }

TBSRequest := SEQUENCE {
  version                 [0] EXPLICIT Version DEFAULT v1,
  requestorName           [1] EXPLICIT GeneralName OPTIONAL,
  requestList                 SEQUENCE OF Request,
  requestExtensions       [2] EXPLICIT Extensions OPTIONAL }

Request ::= SEQUENCE {
  reqCert                     CertID,
  singleRequestExtensions [0] EXPLICIT Extensions OPTIONAL }

CertID ::= SEQUENCE {
  hashAlgorithm               AlgorithmIdentifier,
  issuerNameHash              OCTET STRING,
  issuerKeyHash               OCTET STRING,
  serialNumber                CertificateSerialNumber }
```

The optional elements *optionalSignature* is not required.

The OCSP request is using the following elements, as specified in [3]:

- *hashAlgorithm* identifies the hash algorithm used - it shall use at least SHA256.
- *issuerNameHash* is the hash calculated over the DER encoding of the issuer's name field in the certificate being checked.
- *issuerKeyHash* is the hash calculated over the value (excluding tag and length) of the subject public key field in the issuer's certificate.
- *serialNumber* is the serial number of the certificate for which status is being requested.

The MirrorLink Server shall include a Nonce extension, with a random nonce value, into the OCSP request to prevent any replay attack. The MirrorLink Server shall ignore any OCSP response, if the Signature is either missing or wrong, or if the key of the issuing certificate is not the CCC root certificate.

OCSP responses shall be signed, with the signature algorithm and key of the issuing certificate.

The structure of the OCSP response is given in:

```
    OCSPResponse ::= SEQUENCE {
        responseStatus         OCSPResponseStatus,
        responseBytes          [0] EXPLICIT ResponseBytes OPTIONAL
    }
    OCSPResponseStatus ::= ENUMERATED {
        successful            (0),  --Response has valid confirmations
        malformedRequest      (1),  --Illegal confirmation request
        internalError         (2),  --Internal error in issuer
        tryLater              (3),  --Try again later
                                    --(4) is not used
        sigRequired           (5),  --Signature required
        unauthorized          (6)   --Request unauthorized
    }
    ResponseBytes ::= SEQUENCE {
        responseType         OBJECT IDENTIFIER,
        response             OCTET STRING
    }
```

OCSP responses shall be of responseType "`id-pkix-ocsp-basic`".

```
BasicOCSPResponse ::= SEQUENCE {
    tbsResponseData      ResponseData,
    signatureAlgorithm   AlgorithmIdentifier,
    signature            BIT STRING,
    certs                [0] EXPLICIT SEQUENCE OF Certificate
                             OPTIONAL
}
ResponseData ::= SEQUENCE {
    version              [0] EXPLICIT Version DEFAULT v1,
    responderID              ResponderID,
    producedAt               GeneralizedTime,
    responses                SEQUENCE OF SingleResponse,
    responseExtensions   [1] EXPLICIT Extensions OPTIONAL
}
ResponderID ::= CHOICE {
    byName               [1] Name,
    byKey                [2] KeyHash
}
KeyHash ::= OCTET STRING -- SHA-1 hash of responder's public key
(excluding the tag and length fields)

SingleResponse ::= SEQUENCE {
    certID                   CertID,
    certStatus               CertStatus,
    thisUpdate               GeneralizedTime,
    nextUpdate           [0] EXPLICIT GeneralizedTime,
    singleExtensions     [1] EXPLICIT Extensions OPTIONAL
}
CertStatus ::= CHOICE {
    good        [0]     IMPLICIT NULL,
    revoked     [1]     IMPLICIT RevokedInfo,
    unknown     [2]     IMPLICIT UnknownInfo
}
RevokedInfo ::= SEQUENCE {
    revocationTime           GeneralizedTime,
    revocationReason     [0] EXPLICIT CRLReason OPTIONAL
}
UnknownInfo ::= NULL -- this can be replaced with an enumeration
```

The MirrorLink Server shall use OCSP over HTTP to send and receive OCSP requests and responses. Their formatting is specified in Annex A of [3]. The signature algorithm shall be RSA with at least 2048 bits with at least SHA-256. The ACMS shall include a Nonce extension, with the nonce value provided from the MirrorLink Server, into the OCSP response.

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

• tryLater:	Shall retry within 50 % to 100 % of the query period, since last OCSP request.
• internalError:	Shall retry within 50 % to 100 % of the restricted-grace period, since last OCSP request.
• malformedRequest:	Shall not send any further OCSP requests.
• sigRequired:	Shall not send any further OCSP requests.
• unauthorized:	Shall not send any further OCSP requests.

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

• unknown:	Shall not send any further OCSP requests.
• good:	See clause 6.3.2.
• revoked:	See clauses 6.3.3 and 6.3.4.

The MirrorLink Server shall retry the OCSP request within 50 % to 100 % of the query period, since last 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 shall continue to retry until receiving a response, which does not require a further retry, or until receiving retry responses for at least 6 months.

The OCSP response may include multiple entries. The MirrorLink Server shall evaluate each individual entry according to the above.

Figure 3 shows the state machine diagram of the application certificate revocation process.

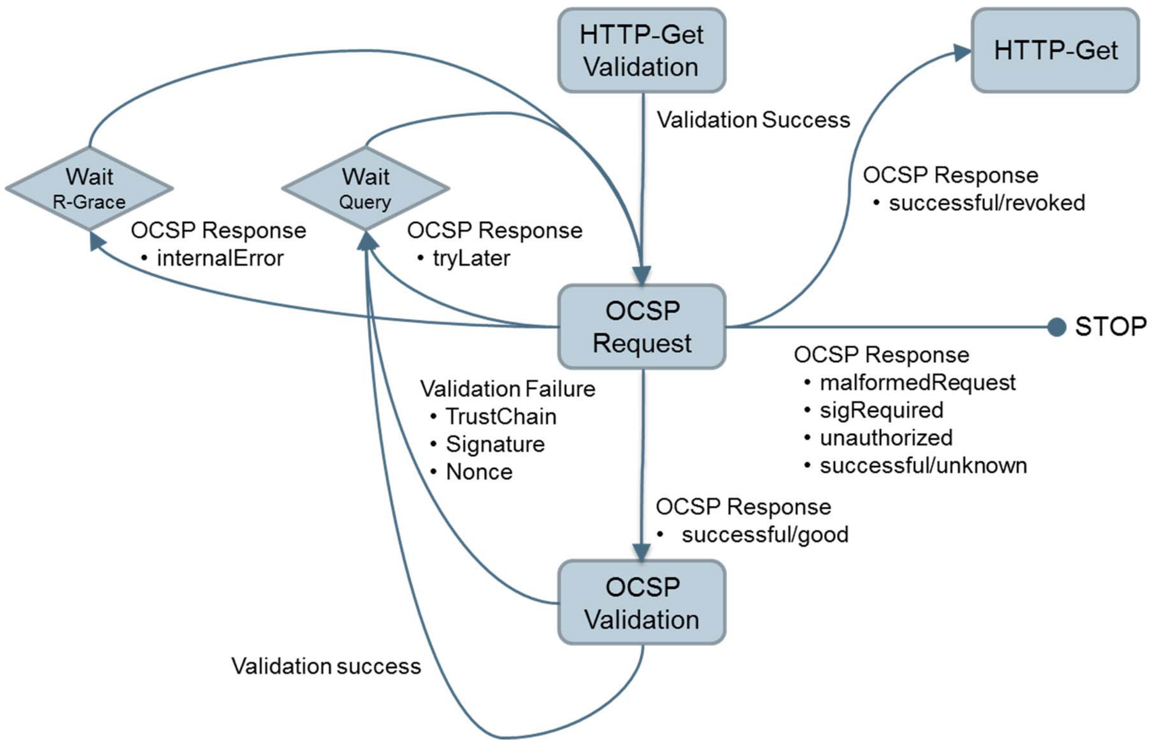

Figure 3: State Machine Diagram - Certificate Revocation Checks

In certain conditions, as given above, the MirrorLink Server shall retry the initial OCSP request for at least 6 months within Query period times. This behavior is shown in Figure 4, for an example query and restricted (R) and non-restricted (NR) grace periods.

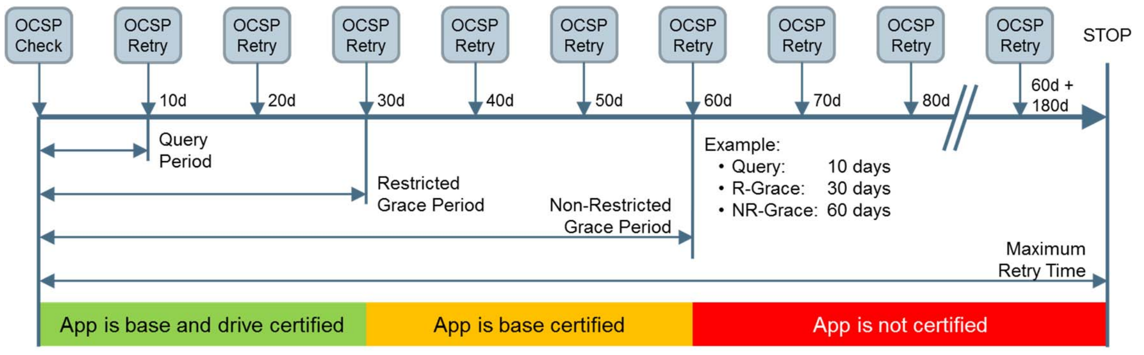

Figure 4: Example Retries of OCSP Requests

6.3.2 Certificate Valid

The MirrorLink Server shall consider an application certificate to be valid, if:

- The OCSP *certStatus* is `"good"`.

The MirrorLink Server shall update the relevant entries (e.g. *AppList* and *CertifiedApplicationList*) through the UPnP Application Server Service, whenever an application's certificate is getting validated, after it had been previously considered unchecked. The MirrorLink Server shall immediately inform the MirrorLink Client about the updated entries through an UPnP *AppListUpdate* event. The application's *appID* value in the *AppList* shall not change due to a validated application certificate.

6.3.3 Certificate Revoked

The ACMS will need to revoke an application certificate in one of the following cases:

1) On request by the application developer; or

2) On request by the Application Certification Body, when a critical driver distraction issue has been detected within the application.

The MirrorLink Server shall consider an application 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 for the same application. Possible HTTP-Get retry behavior shall be followed, as specified in Table 7.

Until the ACMS returns the HTTP-Get response with Error Code 500/900, the MirrorLink Server shall not change the certification status of the application, or extend or reset the query and grace periods. But the MirrorLink Server shall treat the certificate as being unchecked, specified in clause 6.3.5, if it fails to retrieve the HTTP-Get response within the remaining grace periods.

The MirrorLink Server shall not send any further HTTP-Get and OCSP request for a revoked application certificate. The MirrorLink Server shall consider any application, which application certificate got revoked, as a non-certified application.

The MirrorLink Server shall update the relevant entries (e.g. *AppList* and *CertifiedApplicationList*) through the UPnP Application Server Service, whenever an application's certificate is revoked. The MirrorLink Server shall immediately inform the MirrorLink Client about the updated entries through an UPnP *AppListUpdate* event. The application's *appID* value in the *AppList* shall not change due to a revoked application certificate.

6.3.4 Certificate Updated

The ACMS will need to update an application certificate in one of the following cases:

1) The current application certificate expired; or

2) Entries within the current application certificate have been added; or

3) Entries within the current application certificate have been removed.

NOTE: An updated application certificate does not change the application.

The MirrorLink Server shall consider an application 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 for the same application. Possible HTTP-Get retry behavior shall be followed, as specified in Table 7.

Until the new application certificate has been received, the MirrorLink Server shall not change the certification status of the application, or extend or reset the query and grace periods. But the MirrorLink Server shall treat the certificate as being unchecked, specified in clause 6.3.5, if it fails to retrieve and validate the new application certificate within the remaining grace periods.

The retrieved updated application certificate shall be validated (in accordance with clause 6.2.2), including an OCPS request for the updated application certificate.

The MirrorLink Server shall update the certification status of the updated application through the relevant entries (e.g. *AppList* and *CertifiedApplicationList*) through the UPnP Application Server Service, whenever an application's certificate is updated. The MirrorLink Server shall immediately inform the MirrorLink Client about the updated entries through an UPnP *AppListUpdate* event. The application's *appID* value in the *AppList* shall not change due to an updated application certificate.

6.3.5 Unchecked Certificates

An unchecked certificate is a certificate, whose revocation status has not been determined during either the restricted or non-restricted grace period. The MirrorLink Server shall continue checking for the revocation status for at least 6 months after the application certificate became unchecked. After 6 months, the MirrorLink Server may stop checking.

The MirrorLink Server shall update the certification status of the updated application through the relevant entries (e.g. *AppList* and *CertifiedApplicationList*) through the UPnP Application Server Service, whenever an application's certificate is becoming unchecked. The MirrorLink Server shall immediately inform the MirrorLink Client about the updated entries through an UPnP *AppListUpdate* event. The application's *appID* value in the *AppList* shall not change due to an unchecked application certificate.

In case an application's certificate becomes unchecked for restricted drive mode, the MirrorLink Server:

- Shall set the restricted entry in the *A_ARG_TYPE_CertificateInfo* to an Empty String; and
- Shall disable the car mode bit in the *displayInfo/contentCategory* value of the application's *A_ARG_TYPE_AppList* entry and in the respective VNC Context Information.

In case an application's certificate becomes unchecked for both drive and non-drive mode, the MirrorLink Server:

- Shall not provide A_ARG_TYPE_CertificateInfo; and
- Shall disable the car mode bit in the displayInfo/contentCategory value of the application's A_ARG_TYPE_AppList entry and in the respective VNC Context Information; and
- Shall not set both trustLevel entries in the application's A_ARG_TYPE_AppList entry and in the respective VNC Context Information to `"0x00A0"` (Application certificate).

The MirrorLink Server shall consider any application, with an unchecked application certificate for both drive and park mode, as a non-certified MirrorLink-Aware Application.

The MirrorLink Server shall not remove the application certificate.

The MirrorLink Server shall provide access to the application certificate from the MirrorLink Client via the *appCertificateURL* value of the application's *A_ARG_TYPE_AppList* entry, if the application with an unchecked certificate is included in the *A_ARG_TYPE_AppList*.

6.3.6 Testing Consideration

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

For OCSP testing purposes during MirrorLink device certification, the MirrorLink Server shall provide a mechanism for the test engineer to manually trigger an OCSP certificate revocation check any time.

6.4 Query and Grace Periods

6.4.1 Query Period

The MirrorLink Server shall verify from the ACMS, whether any application certificate has been revoked. Validation shall happen within 50 % to 100 % of the query period, if sufficient connectivity is available. It shall not be necessary for the MirrorLink Server device to be connected to a MirrorLink Client device in order to validate the MirrorLink application certificates. Validation REQUIRES a data connection to the Internet. The MirrorLink Server may offer different methods for Internet access, like 2G/3G/4G cellular connectivity, access to Wi-Fi networks, via a PC or via other means.

Definition: *Query Period* - Number of hours between status checks by the MirrorLink Server. The query period can be changed by the Application Certification Management System. The initial notional period is 168 hours (7 days).

The MirrorLink Server may provide a mechanism to manually trigger an OCSP certificate revocation check any time.

The MirrorLink Server shall not start the query period timer, prior the first connection establishment to a MirrorLink Client. The MirrorLink Server shall reset the query period if it has received a valid OCSP response, whose *certStatus* is `"good"` and which has been successfully validated.

Failure to receive an OCSP response shall not invalidate any application certificate, or extend or reset the query period.

The sequence of OCSP checks within the query period is shown in Figure 5.

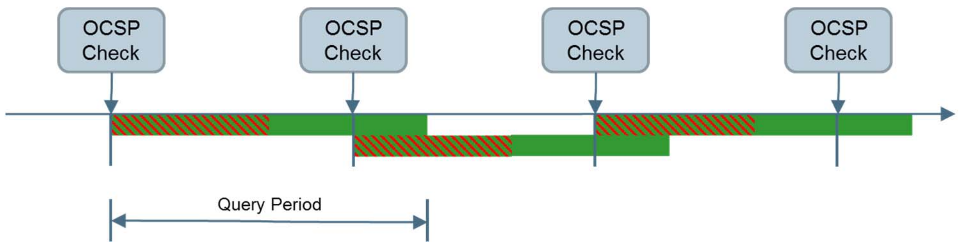

Figure 5: OCSP Checks - Within Query Period

If a successful OCSP response is not received within the query period, the MirrorLink shall enter the grace period. OCSP certificate revocation checks during the query period shall not require an ongoing MirrorLink connection. If the query period is set to 0, all application shall be checked on MirrorLink connection setup.

The Query Period applies to all applications installed on the MirrorLink Server. Different query periods SHALL NOT be tracked for individual applications. The MirrorLink Server shall use the Query Period provided in the most recent valid OCSP response. Updates to the Query period shall not be applied retrospectively to applications, which are already in their Query period. The MirrorLink Server may synchronize all OCSP requests locally on the device, but shall not aim to synchronize requests across devices.

NOTE: The MirrorLink Server may send the first OCSP request earlier than 50 % of the query period, for a newly installed application certificate, in order to synchronize all requests on the device.

6.4.2 Grace Period

In case the MirrorLink Server cannot access the OCSP server within the query period, the query period can be extended.

Definition: *Restricted Grace Period* - The number of hours without a successful query before a MirrorLink Server no longer allows an application, with an application certificate distributed by CCC, to be used in restricted mode. The restricted grace period can be changed by the Application Certification Management System. The initial notional restricted grace period is 720 hours (30 days). This is the maximum time that a MirrorLink application, certified for restricted use, that has had its certification revoked, can be used in restricted mode in the field.

Definition: *Non-restricted Grace Period* - The number of hours without a successful query before a MirrorLink Server no longer allows an application, with an application certificate distributed by CCC, to be used in non- restricted mode. The non-restricted grace period can be changed by the Application Certification Management System. The initial notional non-restricted grace period is 2 160 hours (90 days). This is the maximum time that an app that has had its certification revoked can be used with a MirrorLink head unit in the field.

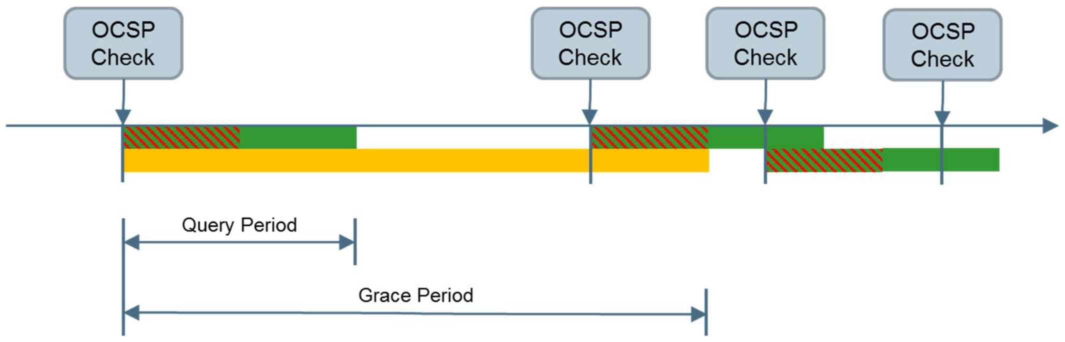

Figure 6: OCSP Checks - Within Grace Period

The sequence of OCSP checks within the grace period is shown in Figure 6.

In case the application certificate is within the grace period, the MirrorLink Server shall send an OCSP request within 1h after sufficient connectivity becomes available.

Applications, which fail to receive a OCSP certificate revocation check within the respective grace period shall be considered unchecked, as specified in clause 6.3.5. If the respective grace period is equal to the query period, all affected application certificates shall be immediately considered unchecked, as specified in clause 6.3.5, when there is no successful OCSP response received within the query period. The grace periods shall not be smaller than the query period.

The sequence of successful OCSP checks after the grace period is shown in Figure 7, leaving the application not certified for a brief period of time, until the next successful OCSP check.

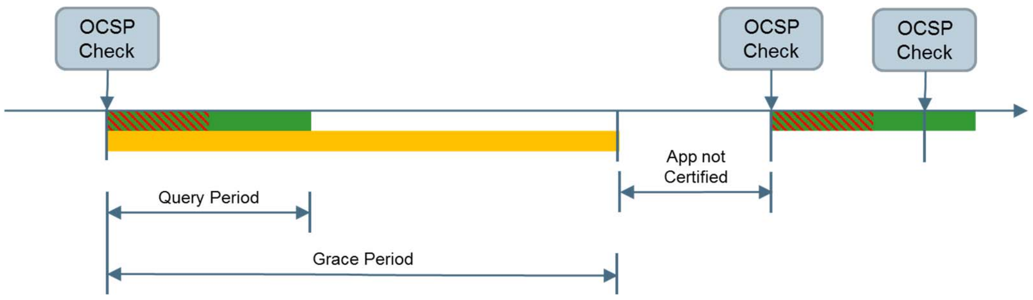

Figure 7: OCSP Checks - After Grace Period

The interaction between the query, restricted (R) grace and non-restricted (NR) grace periods and their dependencies on the application certification status of a drive-certified application is shown in Figure 8.

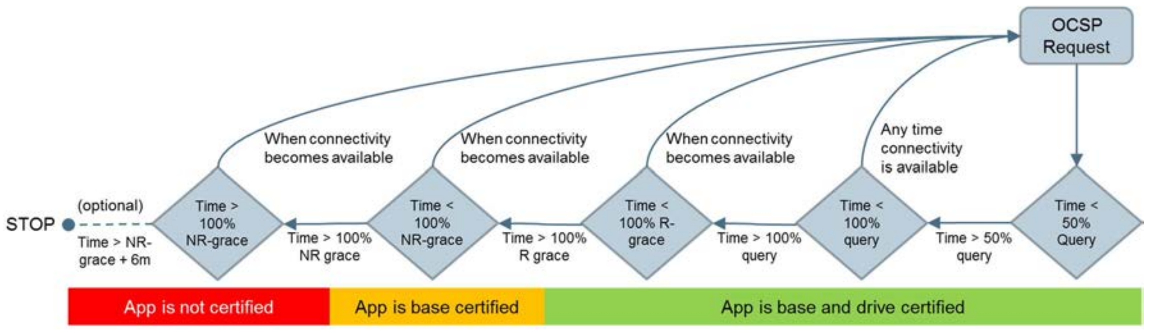

Figure 8: Dependencies of Query and Grace Periods on App Certification Status

Failure to receive an OCSP response shall not extend or reset the respective grace period. Failure to receive a successful OCSP response within the respective grace period shall be notified to the user. An OCSP request during the grace period shall not require an ongoing MirrorLink connection.

The Grace Periods applies to all applications installed on the MirrorLink Server. Different grace periods SHALL NOT be tracked for individual applications. The MirrorLink Server shall use the Grace Periods provided in the most recent valid OCSP response. Updates to the Grace period shall not be applied retrospectively to applications, which are already in their Grace period.

6.4.3 Period Update

The ACMS shall provide an update to the revocation and grace period on every certificate revocation check.

The ACMS shall use the *responseExtension* field of the following structure:

```
Extensions ::= SEQUENCE {
    queryPeriod   [0] EXPLICIT queryPeriod DEFAULT 168,
    driveGrace    [1] EXPLICIT restrictedGrace DEFAULT 720,
    baseGrace     [2] EXPLICIT nonRestrictedGrace DEFAULT 2160
}
```

All values are given in hours.

The response extension values are identified via the following identifiers:

```
CCC-MirrorLink-OCSP-queryPeriod Extension:
    extnId:    1.3.6.1.4.1.41577.1.1
    critical:  no
    extnValue: DER:INTEGER

CCC-MirrorLink-OCSP-driveGrace Extension:
    extnId:    1.3.6.1.4.1.41577.1.2
    critical:  no
    extnValue: DER:INTEGER

CCC-MirrorLink-OCSP-baseGrace Extension:
    extnId:    1.3.6.1.4.1.41577.1.3
    critical:  no
    extnValue: DER:INTEGER
```

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

Query and Grace Period updates are governed by the CCC, as described within a separate Process Management Document (PMD). It has to be understood that in case the query and grace period are both set to zero and the MirrorLink Server does not have network coverage, no certified applications are available.

7 Handling of Applications with a Certificate distributed by CCC

7.1 Application Installation

If the installation of an application fails, the MirrorLink Server device shall not install the Application's Certificate.

The installation of an application can happen, while the MirrorLink Server device is not connected to a MirrorLink Client device.

If the installation of an application succeeds, the MirrorLink Server device shall follow the steps defined in clause 6.2 to retrieve the Application's Certificate distributed by the CCC. If the application comes with an Application Certificate, distribute by the CCC, the MirrorLink Server shall validate it, as defined in clause 6.2.2.

The MirrorLink Server shall treat a re-installation or update of any MirrorLink-Certified or Member-Certified Application as a fresh installation as defined above. The MirrorLink Server shall then trigger the retrieval and installation of the new application certificate. The MirrorLink Server shall update the certification status of the re-installed application through the relevant entries (e.g. *AppList* and *CertifiedApplicationList*) through the UPnP Application Server Service immediately.

The MirrorLink Server shall remove the Application Certificate of any MirrorLink-Certified or Member-Certified Application, which has been updated with a new version, which does not have such an Application Certificate.

7.2 Application Filtering

The basic underlying assumption is that any MirrorLink-Certified or Member-Certified Application, successfully installed on a MirrorLink certified Server device is going to show up on a MirrorLink certified Client device. Some level of filter may happen at the Server and/or Client side. Filtering shall be transparent to the end-user. In addition to such applications, any MirrorLink Server device may have other application as well:

Definition: *Certified Application List* - List of applications, which have been successfully installed on a MirrorLink Server device, and which are MirrorLink-Certified or Member-Certified.

Definition: *Non-Certified Application List* - List of applications, which have been successfully installed on a MirrorLink Server device, and which are MirrorLink-Aware, but not already included in the Certified Application List.

A Member-Certified Application shall be treated as a MirrorLink-Aware Application, if the list of signing entities does not include `"CCC",` the manufacturer's name (as provided from the UPnP Client Profile service), or the *AppCertFilter*'s entity name (as used in the UPnP Application Server service).

From the available, installed applications, the MirrorLink Server may report (i.e. advertise) only a subset to the MirrorLink Client. This Server-side filtering can be described as follows:

Definition: *Reported Certified Application List* - List of applications, which have an application certificate distributed by CCC, and which are included into UPnP Application advertisements via *GetCertifiedApplicationsList* action. List is subject to the use of the *AppCertFilter*.

Definition: *Reported Non-Certified Application List* - List of applications, which are included into UPnP Application advertisements via *GetApplicationList* action. List is subject to the use of the *AppListingFilter*.

The MirrorLink Server shall include all application from the Certified Application List in the Reported Certified Application List. Exceptions shall be approved from the Certification Body.

The MirrorLink Server shall include any application, listed in the Reported Certified Application List, into the Reported Non-Certified Application List.

The MirrorLink Server need not include every application from the Non-Certified Application List into the Reported Non-Certified Application List. The MirrorLink Server shall not include any application from the Non-Certified Application List in the Reported Certified Application List.

MirrorLink Clients shall filter non-certified applications independent of the MirrorLink Server's version; i.e. if a MirrorLink Client provides access to non-certified applications in park mode from a MirrorLink 1.0 Server, it shall provide access to the same applications from a MirrorLink 1.1 Server as well.

From the reported application lists, the MirrorLink Client may show only a subset to the end-user on its MirrorLink Client display. This Client-side filtering can be described as follows:

Definition: *Presented Certified Application List* - List of MirrorLink Reported Certified applications, from the Reported Certified Application List, which are available from the MirrorLink Client.

Definition: *Presented Non-Certified Application List* - List of MirrorLink Reported Non-Certified applications, from the Reported Non-Certified Application List, which are available from the MirrorLink Client.

The MirrorLink Client shall include all applications from the Reported Certified Application List in the Presented Certified Application List, while in Non-Drive Mode. The MirrorLink Client shall include all Drive-Certified applications from the Reported Certified Application List into the Presented Certified Application List, while in Drive Mode. Exceptions shall be approved from the Certification Body.

The MirrorLink Client shall not show the graphical user interface of any non-drive-certified application on the MirrorLink Client display, within Drive Mode at any time. The MirrorLink Client should not list applications, which are not drive-certified within Drive Mode.

Exception: A MirrorLink 1.1 Client may show non-certified Car Mode applications from select MirrorLink 1.0 Servers, while in drive mode, depending on application context information.

NOTE: Selected because of the known qualifications of the MirrorLink Server manufacturer with respect to the shown Car Mode applications.

Those Car Mode applications shall be limited to the following application categories:

- `0x0001 0001`: Home Screen.
- `0x0002 0000`: General Phone Call applications.
- `0x0003 0001`: Music.
- `0x0005 0000`: General Navigation.

7.3 Updating UPnP Application Server Services

7.3.1 Eventing

The MirrorLink Server shall notify the MirrorLink Client, using the UPnP eventing mechanism for the *AppListUpdate* status variable defined in [5] for any of the following reasons:

1) An application has been added to the reported certified or non-certified application list; or
2) An application has been removed from the reported certified or non-certified application list; or
3) Certificate, distributed by CCC, has been updated.

7.3.2 A_ARG_TYPE_AppList

The MirrorLink Server shall include all applications on the Reported Certified and Reported Non-Certified Application List, as defined in clause 7.2, into the response to the UPnP Application Server Service's *GetApplicationList* action, defined in [5], if the application matches the *AppListingFilter* filter criteria. MirrorLink-aware applications should be included as non-certified applications in the UPnP application listing, until the MirrorLink Server has retrieved and validated their regular application certificates.

The MirrorLink Server may provide a mechanism allowing the end-user to prevent the inclusion of an application into the *A_ARG_TYPE_AppList* defined in [5]. In that case the MirrorLink Server shall provide a mechanism to allow the end-user to add the application back to the application listing. This mechanism allows the end-user to restrict the available list of applications to his preferred ones.

The MirrorLink Server shall copy all entries from the Application Certificate, matching the *A_ARG_TYPE_AppList* entries. The MirrorLink Server shall not set any entry, which is missing from the application certificate, to anything other but to the default value in the *A_ARG_TYPE_AppList*. The MirrorLink Server may change the following entries in the *A_ARG_TYPE_AppList*, but this shall be done only for localization purposes:

- *name*
- *providerName*
- *providerURL*

- *description*

The MirrorLink Server shall replace the *icon/url* entry, according to the platform specific rules (in case those exist).

If the validated Application Certificate has been distributed by the AMCS, the MirrorLink Server shall set both Trust Level entries to `"0x00A0"` (Application Certificate) in the UPnP *A_ARG_TYPE_AppList* as well as in the VNC Context Information, otherwise the MirrorLink Server shall set the Trust Level entries to `"0x0080"` (Registered application) or `"0x0060"` (Self-registered application).

The MirrorLink Server shall provide a link to the application certificate in the *appCertificateURL* entry for any application on the Reported Certified Application List, if the application certificate has been distributed by CCC, as being distributed from the ACMS.

7.3.3 A_ARG_TYPE_CertifiedAppList

The MirrorLink Server shall include all applications on the Reported Certified Application List, as defined in clause 7.2, into the response to the UPnP Application Server service's *GetCertifiedApplicationsList* action, defined in [5], if the application matches the *AppCertFilter* filtering criteria.

The MirrorLink Server may provide a mechanism allowing the end-user to prevent the inclusion of an application into the *A_ARG_TYPE_CertifiedAppList*, defined in [5]. In that case the MirrorLink Server shall provide a mechanism to allow the end-user to add the application back to the certified application listing.

7.3.4 A_ARG_TYPE_AppCertificateInfo

The MirrorLink Server shall use all entries, unmodified, from the Application Certificate, matching the *A_ARG_TYPE_AppCertificateInfo* entries, defined in [5], for the given application.

Annex A (normative): XSD MirrorLink Extension Value

The XSD scheme allows the XML structure to be extended (elements and attributes), allowing future extensions of the XML. Note - Members shall not extend the XML on their own.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="certificate">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="version" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="majorVersion" type="xs:nonNegativeInteger"
       minOccurs="0"/>
      <xs:element name="minorVersion" type="xs:nonNegativeInteger"
       minOccurs="0"/>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
       processContents="lax"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="appIdentifier" type="xs:string"/>
   <xs:element name="appListEntry">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="name" type="xs:string" minOccurs="1"/>
      <xs:element name="providerName" type="xs:string"
      minOccurs="0"/>
      <xs:element name="providerURL" type="xs:string" minOccurs="0"/>
      <xs:element name="description" type="xs:string" minOccurs="0"/>
      <xs:element name="iconList" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="icon" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="mimetype" type="xs:string"/>
            <xs:element name="width"    type="xs:positiveInteger"/>
            <xs:element name="height"   type="xs:positiveInteger"/>
            <xs:element name="depth"    type="xs:positiveInteger"/>
            <xs:element name="url"      type="xs:string"/>
            <xs:any namespace="##any" minOccurs="0"
             maxOccurs="unbounded" processContents="lax"/>
           </xs:sequence>
           <xs:anyAttribute namespace="##any" processContents="lax"/>
          </xs:complexType>
         </xs:element>
         <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
         processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="appInfo" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="appCategory" minOccurs="0">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:any namespace="##any" minOccurs="0"
          maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="displayInfo" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="contentCategory" minOccurs="0">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element name="orientation" minOccurs="0">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="landscape"/>
            <xs:enumeration value="portrait"/>
            <xs:enumeration value="both"/>
            <xs:enumeration value="mixed"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:any namespace="##any" minOccurs="0"
          maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="audioInfo" minOccurs="0">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="audioType" minOccurs="0">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="phone"/>
            <xs:enumeration value="application"/>
            <xs:enumeration value="all"/>
            <xs:enumeration value="none"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element name="contentCategory" minOccurs="0">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:pattern value="0[Xx][A-Fa-f0-9]{1,8}"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:any namespace="##any" minOccurs="0"
          maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:any namespace="##any" minOccurs="0"
       maxOccurs="unbounded" processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##any" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:element name="appCertInfoEntry">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="appUUID" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:pattern value="uuid:[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}"/>
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="entity" minOccurs="0" maxOccurs="unbounded">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="name" type="xs:string"/>
         <xs:element name="targetList" minOccurs="0">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="target" type="xs:string"
             maxOccurs="unbounded"/>
            <xs:any namespace="##other" minOccurs="0"
             maxOccurs="unbounded" processContents="lax"/>
           </xs:sequence>
           <xs:anyAttribute namespace="##any" processContents="lax"/>
          </xs:complexType>
         </xs:element>
         <xs:element name="restricted"    type="xs:string"/>
         <xs:element name="nonRestricted" type="xs:string"/>
         <xs:element name="serviceList">
          <xs:complexType>
           <xs:sequence>
            <xs:element name="service" type="xs:string"
             maxOccurs="unbounded"/>
            <xs:any namespace="##other" minOccurs="0"
             maxOccurs="unbounded" processContents="lax"/>
           </xs:sequence>
          </xs:complexType>
         </xs:element>
         <xs:any namespace="##any" minOccurs="0"
          maxOccurs="unbounded" processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:element name="properties" minOccurs="0" type="xs:string"/>
      <xs:any namespace="##any" minOccurs="0"
       maxOccurs="unbounded" processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##any" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:element name="serverProperties">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="platform">
       <xs:complexType>
        <xs:sequence>
         <xs:element name="platformID">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="Android"/>
            <xs:enumeration value="WP"/>
            <xs:enumeration value="Symbian"/>
            <xs:enumeration value="MeeGo"/>
            <xs:enumeration value="BlackBerry"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element name="blacklistedPlatformVersions"
          type="xs:string"/>
         <xs:element name="runtimeID" minOccurs="1">
          <xs:simpleType>
           <xs:restriction base="xs:string">
            <xs:enumeration value="Native"/>
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element name="blacklistedRuntimeVersions"
          type="xs:string"/>
         <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
          processContents="lax"/>
        </xs:sequence>
        <xs:anyAttribute namespace="##any" processContents="lax"/>
       </xs:complexType>
      </xs:element>
      <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
       processContents="lax"/>
     </xs:sequence>
     <xs:anyAttribute namespace="##any" processContents="lax"/>
    </xs:complexType>
   </xs:element>
   <xs:any namespace="##any" minOccurs="0" maxOccurs="unbounded"
    processContents="lax"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
 </xs:complexType>
</xs:element>
</xs:schema>
```

Annex B (informative): 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 12:57:44 2013 GMT
    Responses:
    Certificate ID:
      Hash Algorithm: sha1
      Issuer Name Hash: 8809099A55F562E1EA13273360FFB7F50B76F508
      Issuer Key Hash: BF37183EB53B43AD1D7237E59CE2FC4DF96C7BFC
      Serial Number: 6D
    Cert Status: good
    This Update: May 16 12:57:44 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:
            041035DA009D2912E3CEC403D34B319228D9
        1.3.6.1.4.1.41577.1.2:
            12
    Signature Algorithm: sha512WithRSAEncryption
         17:56:33:9e:9e:80:93:4b:34:db:3f:e2:c3:59:5d:d5:87:96:
         < ... >
         9e:42:6a:2c:45:b6:cc:0d:2f:71:e4:a9:a9:21:70:f0:31:b9:
         61:9f:43:fc:9f:74:47:c3
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
appCer.crt: good
    This Update: May 16 12:57:44 2013 GMT
```

Annex C (informative): Application Certificate Example

An example Application Certificate is given below (in a human readable format).

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

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 109 (0x6d)
    Signature Algorithm: sha512WithRSAEncryption
        Issuer: O=Car Connectivity Consortium, CN=ACMS CA
        Validity
            Not Before: May 16 12:05:23 2013 GMT
            Not After : Jul 23 12:05:23 2023 GMT
        Subject: CN=APP_ID:fake-app-id-for-testing-only-92794929abbfav
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:a9:54:01:07:0b:27:38:8e:91:18:32:58:da:39:
                    < ... >
                    0e:e3:4c:10:d9:38:fc:fa:43:b1:10:89:31:79:bf:
                    7d:1b
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature
            X509v3 Extended Key Usage:
                TLS Web Client Authentication
            1.3.6.1.4.1.41577.2.1:
                <certificate>
                 <version>
                  <majorVersion>1</majorVersion>
                  <minorVersion>0</minorVersion>
                 </version>
                 <appIdentifier>fake-app-id-92794929</appIdentifier>
                 <appListEntry>
                  <name>AppCert</name>
                  <providerName/>
                  <providerURL/>
                  <description/>
                  <iconList>
                   <icon>
                    <mimetype>PNG</mimetype>
                     <width>33</width>
                     <height>43</height>
                     <depth>43</depth>
                     <url>/icons/icon1.png</url>
                    </icon>
                   </iconList>
                   <appInfo>
                    <appCategory/>
                  </appInfo>
                  <displayInfo>
                   <contentCategory/>
                   <orientation/>
                  </displayInfo>
                  <audioInfo>
                   <audioType/>
                   <contentCategory/>
                   </audioInfo>
                 </appListEntry>
                 <appCertInfoEntry>
                  <appUUID/>
                  <entity>
                   <name/>
                   <targetList>
                    <target/>
                   </targetList>
                   <restricted/>
                   <nonRestricted/>
                   <serviceList>
                    <service/>
                   </serviceList>
                  </entity>
                  <properties/>
                 </appCertInfoEntry>
                 <serverProperties>
                  <platform>
                   <platformID>MeeGo</platformID>
                   <blacklistedPlatformVersions/>
                   <runtimeID>Native</runtimeID>
                   <blacklistedRuntimeVersions/>
                  </platform>
                 </serverProperties>
                </certificate>
            Authority Information Access:
                OCSP - URI:http://acms.carconnectivity.org/OCSP
            X509v3 Subject Key Identifier:
                2E:E2:41:6E:58:10:26:19:AA:DE:9C:6E:2D:4E:28:32:
                21:77:51:16
    Signature Algorithm: sha512WithRSAEncryption
         70:51:c6:81:3a:c0:47:b2:15:ec:5b:e6:1b:b0:c1:18:e3:d4:
         < ... >
         38:63:2f:6d:c9:99:c0:30:07:03:af:d7:32:86:4c:0a:10:fb:
         6c:2c:9f:f4:9b:59:7b:bc
```

Annex D (informative):
Authors and Contributors

The following people have contributed to the present document:

Rapporteur:	Dr. Jörg Brakensiek, E-Qualus (for Car Connectivity Consortium LLC)
Other contributors:	Risto Helin, Nokia Corporation
	Ed Pichon, E-Qualus (for Car Connectivity Consortium LLC)
	Kari Kostiainen, Nokia Corporation
	Murali Soundararajan, Samsung

History

Document history		
V1.3.0	October 2017	Publication