ONVIF™
Credential Service Specification

Version 19.12
December 2019

Contents

Contributors

ASSA ABLOY	Patrik Björling Rygert
ASSA ABLOY	Mattias Rengstedt
Axis Communications AB	Marcus Johansson
Axis Communications AB	Robert Rosengren
Axis Communications AB	Derek Wang
Axis Communications AB	Emil Selinder
Bosch	Mohane Caliaperoumal
Bosch	Dirk Schreiber
Honeywell	Uvaraj Thangarajan
Honeywell	Neelendra Bhandari
Honeywell	Mayur Salgar
Honeywell	Vinay Ghule
PACOM	Eugene Scully
Siemens AG	Lokeshwar K
Siemens AG	Suresh Raman
Siemens AG	Suresh Krishnamurthy
Dahua Technology	Weiming Mao
Dahua Technology	Hui Xu
Hikvision Digital Technology	Shuanglong Liao
Hikvision Digital Technology	Yuanyuan Zheng

1 Scope

1.1 General

This specification defines the web service interface for integration with physical access control systems. This includes discovering components and support of the configuration of the credentials components.

Supplementary dedicated services such as access control and access rules services will be defined in separate document.

Web service usage and common ONVIF functionality are outside the scope of this document. Please refer to [ONVIF Core Specification] for more information.

1.2 Conventions

The key words “shall”, “shall not”, “should”, “should not”, “may”, “need not”, “can”, “cannot” in this specification are to be interpreted as described in Annex H of [ISO/IEC Directives].

1.3 Namespaces

This document references the following namespaces:

Table 1: Referenced namespaces (with prefix)

Prefix	Namespace URI
env	http://www.w3.org/2003/05/soap-envelope
ter	http://www.onvif.org/ver10/error
xs	http://www.w3.org/2001/XMLSchema
tt	http://www.onvif.org/ver10/schema
pt	http://www.onvif.org/ver10/pacs
tns1	http://www.onvif.org/ver10/topics
tcr	http://www.onvif.org/ver10/credential/wsdl

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ONVIF Core Specification
<http://www.onvif.org/specs/core/ONVIF-Core-Specification.pdf>

ONVIF PACS Architecture and Design Considerations
<https://www.onvif.org/specs/wp/ONVIF-PACS-Architecture-and-Design-Considerations.pdf>

ONVIF Access Rules Service Specification
<http://www.onvif.org/specs/srv/access/ONVIF-AccessRules-Service-Spec.pdf>

ONVIF Door Control Service Specification
<https://www.onvif.org/specs/srv/door/ONVIF-DoorControl-Service-Spec.pdf>

ISO 16484-5, *ISO 16484-5:2017, Building automation and control systems (BACS) – Part 5: Data communication protocol, Annex P – BACnet Encoding of Standard Authentication Factor Formats (Normative)*
<https://www.iso.org/obp/ui/#iso:std:iso:16484:-5:ed-6:v1:en>

ISO 8601, *ISO 8601:2004, Data elements and interchange formats – Information interchange – Representation of dates and times*
<https://www.iso.org/obp/ui/#iso:std:iso:8601:ed-3:v1:en>

ISO/IEC Directives, *ISO/IEC Directives Part 2, Principles and rules for the structure and drafting of ISO and IEC documents, Edition 7.0, May 2016*
<http://www.iec.ch/members_experts/refdocs/iec/isoiecdir-2%7Bed7.0%7Den.pdf>

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

Anti-Passback	Operating mode which requires user validation when leaving a security controlled area in order to be able to re-enter and vice versa.
Anti-Passback Violation State	A signal stating if the anti-passback rules have been violated for a credential.
Access Profile	A collection of access policies. Is used to define role based access.
Credential	A physical/tangible object, a piece of knowledge, or a facet of a person's physical being, that enables an individual access to a given physical facility or computer-based information system.
Credential Format	The credential data can be formatted in many different ways. ONVIF supports the BACnet format types in [ISO 16484-5].
Credential Holder	Associates a credential with a user. Typically it holds a reference to a credential and a reference to a user.
Credential Identifier	Card number, unique card information, PIN, fingerprint, or other biometric information, etc., that can be validated in an access point.
Credential Number	A sequence of bytes uniquely identifying a credential at an access point.
Credential State	The credential state indicates if a credential is enabled or disabled. The state also indicates if anti-passback has been violated or not. The state may also contain a reason why the credential was disabled.
Duress	Forcing a person to provide access to a secure area against that person's wishes.
Format Type	See Credential Format.
Validity Period	From a certain point in time, to a later point in time.
Reset Anti-Passback Violation	See Credential Forgive.
Credential Forgive	Command which re-enables a credential that has violated the anti-passback rules.

3.2 Abbreviated terms

PACS	Physical Access Control System

4 Overview

The credential service specification defines the commands to configure credentials.

A credential holds information that can be validated in an access point, such as unique card information, PIN, biometric information, etc. A credential also holds information on what the credential can access via credential access profiles, which ties the credential to access profiles as described in [ONVIF Access Rules Service Specification].

A credential is assigned to a person (called credential holder) and has a validity that specifies the period during which the credential can be used to get access.

Consider the following example:

A credential is assigned to a consultant to be temporarily used during one week (April 2nd-April 6th). The start date/time of the validity is set to the morning of April 2nd and the end date/time of the validity is set to the evening of April 6th.

This particular credential is a card with number 987654321 and the consultant is given the personal pin code 1234. Both pieces of information are stored in the credential. The card number is a credential identifier of type pt:Card and the pin code is a credential identifier of type pt:PIN (see section 5.2.2.1 for supported identifier types).

The consultant will help out with the installation of a server so he needs access to the server room, and of course also access to the other common facilities at the office. The access profile “IT support access” gives access to the server room during office hours, and the access profile “Staff access” gives access to the rest of the office, The references to both access profiles are stored on the credential in the CredentialAccessProfile structure.

A credential service object model representation is shown in Figure 1

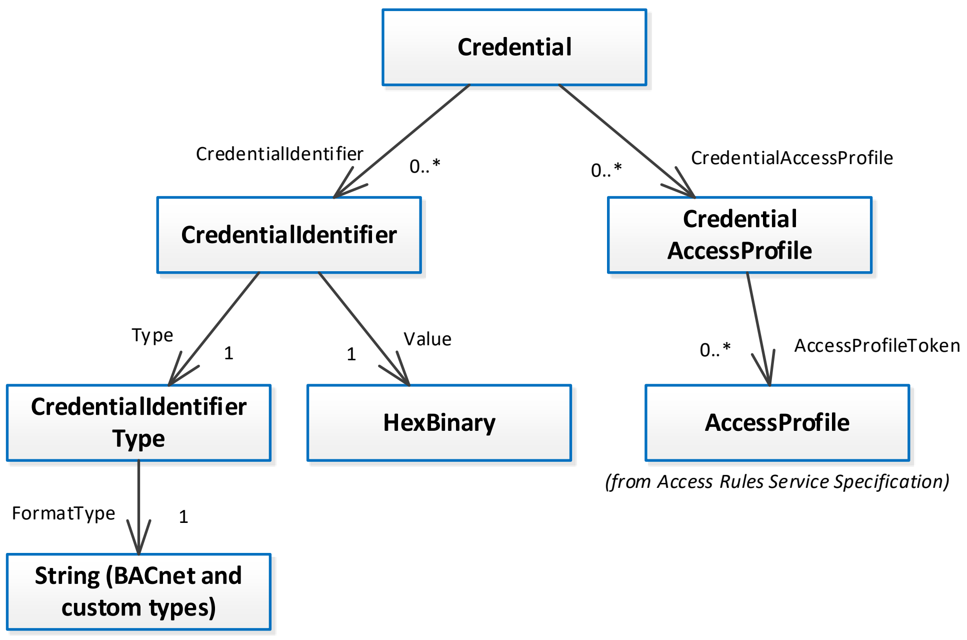

Figure 1: Main data structures in the credential service

5 Credentials

5.1 General

This service offers commands for configuring the credentials.

5.2 Service capabilities

5.2.1 General

The device shall provide service capabilities in two ways:

1. With the GetServices method of Device service when IncludeCapability is true. Please refer to [ONVIF Core Specification] for more details.

2. With the GetServiceCapabilities method.

5.2.2 Data structures

5.2.2.1 ServiceCapabilities

The service capabilities reflect optional functionality of a service. The information is static and does not change during device operation. The following capabilities are available:

- **MaxLimit**

  The maximum number of entries returned by a single Get<Entity>List or Get<Entity> request. The device shall never return more than this number of entities in a single response.

- **CredentialValiditySupported**

  Indicates that the device supports credential validity.

- **CredentialAccessProfileValiditySupported**

  Indicates that the device supports validity on the association between a credential and an access profile.

- **ValiditySupportsTimeValue**

  Indicates that the device supports both date and time value for validity. If set to false, then the time value is ignored.

- **MaxCredentials**

  The maximum number of credential supported by the device.

- **MaxAccessProfilesPerCredential**

  The maximum number of access profiles for a credential.

- **ResetAntipassbackSupported**

  Indicates the device supports resetting of anti-passback violations and notifying on anti-passback violations.

- **SupportedIdentifierType**

  A list of identifier types that the device supports. Is of type text.

  Identifier types starting with the prefix pt: are reserved to define ONVIF-specific types as defined in pt:RecognitionType. Please note that pt:REX is not an identifier type. For custom defined identifier types, free text can be used.

- **SupportedExemptionType**

  A list of exemptions that the device supports. Exemption types starting with the prefix pt: are reserved to define ONVIF-specific types. For custom defined exemption types, free text can be used. The following types are defined by ONVIF:

o pt:ExemptFromAuthentication	Supports ExemptedFromAuthentication in section 5.3.2.3.

- **ClientSuppliedTokenSupported**

  Indicates that the client is allowed to supply the token when creating credentials. To enable the use of the command SetCredential, the value must be set to true.

- **DefaultCredentialSuspensionDuration**

  The default time period that the credential will temporary be suspended (e.g. by using the wrong PIN a predetermined number of times).

  The time period is defined as an [ISO 8601] duration string (e.g. “PT5M”).

- **MaxWhitelistedItems**

  The maximum number of whitelisted credential identifiers supported by the device. Defaults to zero.

- **MaxBlacklistedItems**

  The maximum number of blacklisted credential identifiers supported by the device. Defaults to zero.

5.2.3 GetServiceCapabilities command

This operation returns the capabilities of the credential service.

REQUEST:

This is an empty message.

RESPONSE:

- **Capabilities – [tcr:ServiceCapabilities]**
  List of capabilities as defined above.

FAULTS:

None

ACCESS CLASS:

PRE_AUTH

5.3 Credential information

5.3.1 General

A credential holds information that can be validated in an access point (card information, PIN, etc.) and what the credential can access (access profiles).

5.3.2 Data structures

5.3.2.1 CredentialInfo

The CredentialInfo type represents the credential as a logical object. The structure contains the basic information of a specific credential instance. The device shall provide the following fields for each credential.

- **token**

  A service unique identifier of the credential.

- **CredentialHolderReference**

  An external reference to a person holding this credential. The reference is a username or used ID in an external system, such as a directory service.

To provide more information, the device may include the following optional fields:

- **Description**

  User readable description for the credential. It shall be up to 1024 characters.

- **ValidFrom**

  The start date/time validity of the credential. If the ValiditySupportsTimeValue capability is set to false, then only date is supported (time is ignored).

- **ValidTo**

  The expiration date/time validity of the credential. If the ValiditySupportsTimeValue capability is set to false, then only date is supported (time is ignored).

  The end time is exclusive, meaning that the exact moment in time is not part of the period.

If ValiditySupportsTimeValue is set to false, then a ValidFrom date implicitly means 00:00 that day, and a ValidTo date implicitly means that the whole day is included in the range. E.g. the validity June 6 to June 7 is the same as June 6 00:00 to June 8 00:00 (note that the end time is exclusive).

5.3.2.2 Credential

A Credential is a physical/tangible object, a piece of knowledge, or a facet of a person’s physical being, that enables an individual access to a given physical facility or computer-based information system. A credential holds one or more credential identifiers. To gain access one or more identifiers may be required.

The device shall include all properties of the CredentialInfo structure and also a list of credential identifiers and a list of credential access profiles.

- **CredentialIdentifier**

  A list of credential identifier structures. At least one credential identifier is required. Maximum one credential identifier structure per type is allowed.

To provide more information, the device may include the following optional fields:

- **CredentialAccessProfile**

  A list of credential access profile structures.

- **Attribute**

  A list of credential attributes as name value pairs. Key name types starting with the prefix pt: are reserved to define ONVIF-specific types. For custom defined key name types, free text can be used.

- **ExtendedGrantTime**

  A boolean indicating that the credential holder needs extra time to get through the door. ExtendedReleaseTime will be added to ReleaseTime, and ExtendedOpenTime will be added to OpenTime (see section 5.3.2.4 in [ONVIF Door Control Service Specification]).

5.3.2.3 CredentialIdentifier

A credential identifier is a card number, unique card information, PIN or biometric information such as fingerprint, iris, vein, face recognition, that can be validated in an access point.

- **Type**

  Contains the details of the credential identifier type. Is of type CredentialIdentifierType.

- **ExemptedFromAuthentication**

  If set to true, this credential identifier is not considered for authentication. For example if the access point requests Card plus PIN, and the credential identifier of type PIN is exempted from authentication, then the access point will not prompt for the PIN.

- **Value**

  The value of the identifier in hexadecimal representation.

5.3.2.4 CredentialIdentifierType

Specifies the name of credential identifier type and its format for the credential value.

- **Name**

  The name of the credential identifier type, such as pt:Card, pt:PIN, etc. (see section 5.2.2.1 for supported credential identifier types).

- **FormatType**

  Specifies the format of the credential value for the specified identifier type name. See section 5.3.2.5 below.

5.3.2.5 CredentialIdentifierFormatTypeInfo

Contains information about a format type.

- **FormatType**

  A format type supported by the device. A list of supported format types is provided in [ISO 16484-5]. The BACnet type “CUSTOM” is not used in this specification. Instead device manufacturers can define their own format types.

- **Description**

  User readable description of the credential identifier format type. It shall be up to 1024 characters. For custom types, it is recommended to describe how the octet string is encoded (following the structure in column *Authentication Factor Value Encoding* of [ISO 16484-5]).

5.3.2.6 CredentialAccessProfile

The association between a credential and an access profile.

- **AccessProfileToken**

  The reference token of the associated access profile. Access profiles are defined in [ONVIF Access Rules Service Specification].

The device may include the following optional fields:

- **ValidFrom**

  The start date/time of the validity for the association between the credential and the access profile. If the ValiditySupportsTimeValue capability is set to false, then only date is supported (time is ignored).

- **ValidTo**

  The end date/time of the validity for the association between the credential and the access profile. The end time is exclusive, meaning that the exact moment in time is not part of the period.

Devices without ValiditySupportsTimeValue support shall ignore the time value and interpret the provided date as inclusive.

For example if ValiditySupportsTimeValue is set to false, then a ValidFrom date implicitly means 00:00 that day, and a ValidTo date implicitly means that the whole day is included in the range. E.g. the validity June 6 to June 7 is the same as June 6 00:00 to June 8 00:00 (note that the end time is exclusive).

5.3.2.7 CredentialData

A combination of a credential and its state.

- **Credential**

  The credential.

- **CredentialState**

  The state of the credential.

5.3.2.8 CredentialState

The CredentialState structure contains information about the state of the credential and optionally the reason of why the credential was disabled.

- **Enabled**

  True if the credential is enabled or false if the credential is disabled.

The device may include the following optional fields:

- **Reason**

  The reason of disabling the credential. Reason types starting with the prefix pt: are reserved to define ONVIF-specific types. For custom defined reason types, free text can be used. The following types are defined by ONVIF:

o pt:CredentialLockedOut	Access is denied due to credential locked out
o pt:CredentialBlocked	Access is denied because the credential has deliberately been blocked by the operator
o pt:CredentialLost	Access is denied due to the credential being reported as lost
o pt:CredentialStolen	Access is denied due to the credential being reported as stolen
o pt:CredentialDamaged	Access is denied due to the credential being reported as damaged
o pt:CredentialDestroyed	Access is denied due to the credential being reported as destroyed
o pt:CredentialInactivity	Access is denied due to credential inactivity
o pt:CredentialExpired	Access is denied because the credential has expired
o pt:CredentialRenewalNeeded	Access is denied because the credential requires a renewal (e.g. new PIN or fingerprint enrollment)

- **AntipassbackState**

  A structure indicating the anti-passback state. This field shall be supported if the ResetAntipassbackSupported capability is set to true.

5.3.2.9 Anti-passback State

A structure containing anti-passback related state information.

- **AntipassbackViolated**

  Indicates if anti-passback is violated for the credential.

5.3.2.10 CredentialIdentifierItem

A credential identifier is a card number, unique card information, PIN or biometric information such as fingerprint, iris, vein, face recognition, that can be validated in an access point.

- **Type**

  Contains the details of the credential identifier type. Is of type CredentialIdentifierType.

- **Value**

  The value of the identifier in hexadecimal representation.

5.3.3 GetCredentialInfo command

This operation requests a list of CredentialInfo items matching the given tokens.

The device shall ignore tokens it cannot resolve and shall return an empty list if there are no items matching the specified tokens. The device shall not return a fault in this case.

If the number of requested items is greater than MaxLimit, a TooManyItems fault shall be returned.

REQUEST:

- **Token – unbounded [pt:ReferenceToken]**
  Tokens of CredentialInfo items to get.

RESPONSE:

- **CredentialInfo – optional, unbounded [tcr:CredentialInfo]**
  List of CredentialInfo items.

FAULTS:

- **env:Sender – ter:InvalidArgs – ter:TooManyItems**
  Too many items were requested, see MaxLimit capability.

ACCESS CLASS:

READ_SYSTEM

5.3.4 GetCredentialInfoList command

This operation requests a list of all CredentialInfo items provided by the device.

A call to this method shall return a StartReference when not all data is returned and more data is available. The reference shall be valid for retrieving the next set of data. Please refer to section 4.8.3 in [ONVIF PACS Architecture and Design Considerations] for more details.

The number of items returned shall not be greater than the Limit parameter.

REQUEST:

- **Limit – optional [xs:int]**
  Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.
- **StartReference – optional [xs:string]**
  Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.

RESPONSE:

- **NextStartReference – optional [xs:string]**
  StartReference to use in next call to get the following items. If absent, no more items to get.
- **CredentialInfo – optional, unbounded [tcr: CredentialInfo]**
  List of CredentialInfo items.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:InvalidStartReference**
  StartReference is invalid or has timed out. Client needs to start fetching from the beginning.

ACCESS CLASS:

5.3.5 GetCredentials command

This operation requests a list of Credential items matching the given tokens.

The device shall ignore tokens it cannot resolve and shall return an empty list if there are no items matching the specified tokens. The device shall not return a fault in this case.

If the number of requested items is greater than MaxLimit, a TooManyItems fault shall be returned.

REQUEST:

- **Token – unbounded [pt:ReferenceToken]**
  Tokens of Credential items to get.

RESPONSE:

- **Credential – optional, unbounded [tcr:Credential]**
  List of Credential items.

FAULTS:

- **env:Sender – ter:InvalidArgs – ter:TooManyItems**
  Too many items were requested, see MaxLimit capability.

ACCESS CLASS:

READ_SYSTEM_SECRET

5.3.6 GetCredentialList command

This operation requests a list of all Credential items provided by the device.

A call to this method shall return a StartReference when not all data is returned and more data is available. The reference shall be valid for retrieving the next set of data. Please refer to section 4.8.3 in [ONVIF PACS Architecture and Design Considerations] for more details.

The number of items returned shall not be greater than the Limit parameter.

REQUEST:

- **Limit – optional [xs:int]**
  Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.
- **StartReference – optional [xs:string]**
  Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.

RESPONSE:

- **NextStartReference – optional [xs:string]**
  StartReference to use in next call to get the following items. If absent, no more items to get.
- **Credential – optional, unbounded [tcr:Credential]**
  List of Credential items.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:InvalidStartReference**
  StartReference is invalid or has timed out. Client needs to start fetching from the beginning.

ACCESS CLASS:

READ_SYSTEM_SECRET

5.3.7 CreateCredential command

This operation creates the specified credential in the device.

A call to this method takes a credential structure and a credential state structure as input parameters. The credential state can be created in disabled or enabled state.

The token field of the Credential structure shall be empty and the device shall allocate a token for the credential. The allocated token shall be returned in the response.

If the client sends any value in the token field, the device shall return InvalidArgVal as a generic fault code.

REQUEST:

- **Credential – [tcr:Credential]**
  The credential to create.
- **State – [tcr:CredentialState]**
  The state of the credential.

RESPONSE:

- **Token – [pt:ReferenceToken]**
  The token of the created credential.

FAULTS:

- **env:Sender – ter:CapabilityViolated – ter:MaxCredentials**
  There is not enough space to create a new credential, see the MaxCredentials capability.
- **env:Sender – ter:CapabilityViolated – ter:MaxAccessProfilesPerCredential**
  There are too many access profiles per credential, see the MaxAccessProfilesPer-Credential capability.
- **env:Sender – ter:CapabilityViolated – ter:CredentialValiditySupported**
  Credential validity is not supported by device, see the CredentialValiditySupported capability.
- **env:Sender – ter:CapabilityViolated – ter:CredentialAccessProfileValidity-Supported**
  Credential access profile validity is not supported by the device, see the Credential-AccessProfileValiditySupported capability.
- **env:Sender – ter:CapabilityViolated – ter:SupportedIdentifierType**
  Specified identifier type is not supported by device, see the SupportedIdentifierType capability.
- **env:Sender – ter:InvalidArgVal – ter:DuplicatedIdentifierType**
  The same identifier type was used more than once.
- **env:Sender – ter:InvalidArgVal – ter:InvalidFormatType**
  Specified identifier format type is not supported by the device.
- **env:Sender – ter:InvalidArgVal – ter:InvalidIdentifierValue**
  Specified identifier value is not as per FormatType definition.
- **env:Sender – ter:InvalidArgVal – ter:DuplicatedIdentifierValue**
  The same combination of identifier type, format and value was used more than once (some devices may not support duplicate identifier values).
- **env:Sender – ter:InvalidArgVal – ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).

- **env:Sender – ter:CapabilityViolated – ter:SupportedExemptionType**
  Specified exemption type is not supported by the device, see the Supported-ExemptionType capability.

ACCESS CLASS:

WRITE_SYSTEM

5.3.8 SetCredential command

This method is used to synchronize a credential in a client with the device.

A call to this method takes a credential structure and a credential state structure as input parameters. The token field of the credential must not be empty.

If a credential with the specified token does not exist in the device, the credential is created. The credential state can be created in disabled or enabled state.

If a credential with the specified token exists in the device, then the credential is modified. The credential state will be set according to the specified state (this behavior differs from the ModifyCredential command). All existing credential identifiers and credential access profiles are removed and replaced with the specified entities.

A device that signals support for the ClientSuppliedTokenSupported capability shall implement this command.

If no token was specified in the request, the device shall return InvalidArgs as a generic fault code.

REQUEST:

- **CredentialData – [tcr:CredentialData]**
  The credential and state to create or modify.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Receiver – ter:CapabilityViolated – ter:ClientSuppliedTokenSupported**
  The device does not support that the client supplies the token.
- **env:Sender – ter:CapabilityViolated – ter:MaxCredentials**
  There is not enough space to create a new credential, see the MaxCredentials capability.
- **env:Sender – ter:CapabilityViolated – ter:MaxAccessProfilesPerCredential**
  There are too many access profiles per credential, see the MaxAccessProfilesPer-Credential capability.
- **env:Sender – ter:CapabilityViolated – ter:CredentialValiditySupported**
  Credential validity is not supported by device, see the CredentialValiditySupported capability.
- **env:Sender – ter:CapabilityViolated – ter:CredentialAccessProfileValidity-Supported**
  Credential access profile validity is not supported by the device, see the Credential-AccessProfileValiditySupported capability.
- **env:Sender – ter:CapabilityViolated – ter:SupportedIdentifierType**
  Specified identifier type is not supported by device, see the SupportedIdentifierType capability.
- **env:Sender – ter:InvalidArgVal – ter:DuplicatedIdentifierType**
  The same identifier type was used more than once.

- **env:Sender – ter:InvalidArgVal – ter:InvalidFormatType**
  Specified identifier format type is not supported by the device.
- **env:Sender – ter:InvalidArgVal – ter:InvalidIdentifierValue**
  Specified identifier value is not as per FormatType definition.
- **env:Sender – ter:InvalidArgVal – ter:DuplicatedIdentifierValue**
  The same combination of identifier type, format and value was used more than once (some devices may not support duplicate identifier values).
- **env:Sender – ter:InvalidArgVal – ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).
- **env:Sender – ter:CapabilityViolated – ter:SupportedExemptionType**
  Specified exemption type is not supported by the device, see the Supported-ExemptionType capability.

ACCESS CLASS:

WRITE_SYSTEM

5.3.9 ModifyCredential command

This operation modifies the specified credential.

The token of the credential to modify is specified in the token field of the Credential structure and shall not be empty. All other fields in the structure shall overwrite the fields in the specified credential.

When an existing credential is modified, the state is not modified explicitly. The only way for a client to change the state of a credential is to explicitly call the EnableCredential, DisableCredential or ResetAntipassback command.

All existing credential identifiers and credential access profiles are removed and replaced with the specified entities.

If no token was specified in the request, the device shall return InvalidArgs as a generic fault code.

REQUEST:

- **Credential – [tcr:Credential]**
  The credential to modify.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.
- **env:Sender – ter:CapabilityViolated – ter:MaxAccessProfilesPerCredential**
  There are too many access profiles per credential, see the MaxAccessProfilesPerCredential capability
- **env:Sender – ter:CapabilityViolated – ter:CredentialValiditySupported**
  Credential validity is not supported by device, see the CredentialValiditySupported capability
- **env:Sender – ter:CapabilityViolated – ter:CredentialAccessProfileValidity-Supported**
  Credential access profile validity is not supported by device, see the Credential-AccessProfileValiditySupported capability

- **env:Sender – ter:CapabilityViolated – ter:SupportedIdentifierType**
  Specified identifier type is not supported by device, see the SupportedIdentifierType capability
- **env:Sender – ter:InvalidArgVal – ter:DuplicatedIdentifierType**
  The same identifier type was used more than once.
- **env:Sender – ter:InvalidArgVal – ter:InvalidFormatType**
  Specified identifier format type is not supported by the device.
- **env:Sender – ter:InvalidArgVal – ter:InvalidIdentifierValue**
  Specified identifier value is not as per FormatType definition.
- **env:Sender – ter:InvalidArgVal – ter:DuplicatedIdentifierValue**
  The same combination of identifier type, format and value was used more than once (some devices may not support duplicate identifier values).
- **env:Sender – ter:InvalidArgVal – ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).
- **env:Sender – ter:CapabilityViolated – ter:SupportedExemptionType**
  Specified exemption type is not supported by the device, see the Supported-ExemptionType capability.

ACCESS CLASS:

WRITE_SYSTEM

5.3.10 DeleteCredential command

This method deletes the specified credential.

If it is associated with one or more entities some devices may not be able to delete the credential, and consequently a ReferenceInUse fault shall be generated.

If no token was specified in the request, the device shall return InvalidArgs as a generic fault code.

REQUEST:

- **Token – [pt:ReferenceToken]**
  The token of the credential to delete.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.

ACCESS CLASS:

WRITE_SYSTEM

5.3.11 GetCredentialState command

This method returns the state for the specified credential.

If the capability ResetAntipassbackSupported is set to true, then the device shall supply the anti-passback state in the returned credential state structure.

REQUEST:

- **Token – [pt:ReferenceToken]**
  The token of the credential.

RESPONSE:

- **State – [tcr:CredentialState]**
  The state of the credential.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.

ACCESS CLASS:

READ_SYSTEM_SENSITIVE

5.3.12 EnableCredential command

This method is used to enable a credential.

REQUEST:

- **Token – [pt:ReferenceToken]**
  The token of the credential.
- **Reason – optional [pt:Name]**
  Reason for enabling the credential.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.

ACCESS CLASS:

ACTUATE

5.3.13 DisableCredential command

This method is used to disable a credential.

REQUEST:

- **Token – [pt:ReferenceToken]**
  The token of the credential.
- **Reason – optional [pt:Name]**
  Reason for disabling the credential.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.

ACCESS CLASS:

ACTUATE

5.3.14 ResetAntipassbackViolation command

This method is used to reset anti-passback violations for a specified credential.

REQUEST:

- **CredentialToken – [pt:ReferenceToken]**
  The token of the credential.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.

ACCESS CLASS:

ACTUATE

5.3.15 GetSupportedFormatTypes command

This method returns all the supported format types of a specified identifier type that is supported by the device.

REQUEST:

- **CredentialIdentifierTypeName – [xs:string]**
  Name of the credential identifier type.

RESPONSE:

- **FormatTypeInfo – unbounded [tcr:CredentialIdentifierFormatTypeInfo]**
  Identifier format types.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Identifier type is not found.

ACCESS CLASS:

READ_SYSTEM

5.3.16 GetCredentialIdentifiers command

This method returns all the credential identifiers for a credential.

REQUEST:

- **CredentialToken – [pt:ReferenceToken]**
  The token of the credential.

RESPONSE:

- **CredentialIdentifier – optional, unbounded [tcr:CredentialIdentifier]**
  Identifiers of the credential.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.

ACCESS CLASS:

READ_SYSTEM_SECRET

5.3.17 SetCredentialIdentifier command

This operation creates or updates a credential identifier for a credential.

If the type of specified credential identifier already exists, the current credential identifier of that type is replaced. Otherwise the credential identifier is added.

REQUEST:

- **CredentialToken – [pt:ReferenceToken]**
  The token of the credential.
- **CredentialIdentifier – [tcr:CredentialIdentifier]**
  Identifier of the credential.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.
- **env:Sender – ter:CapabilityViolated – ter:SupportedIdentifierType**
  Specified identifier type is not supported by device, see the SupportedIdentifierType capability.
- **env:Sender – ter:InvalidArgVal – ter:InvalidFormatType**
  Specified identifier format type is not supported by the device.
- **env:Sender – ter:InvalidArgVal – ter:InvalidIdentifierValue**
  Specified identifier value is not as per FormatType definition.

ACCESS CLASS:

WRITE_SYSTEM

5.3.18 DeleteCredentialIdentifier command

This method deletes all the identifier values for the specified type. However, if the identifier type name doesn’t exist in the device, it will be silently ignored without any response.

Note that each credential needs at least one identifier and an attempt to delete the last identifier will result in a fault.

REQUEST:

- **CredentialToken – [pt:ReferenceToken]**
  The token of the credential.
- **CredentialIdentifierTypeName – [pt:Name]**
  Name of the credential identifier type.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.
- **env:Sender – ter:ConstraintViolated – ter:MinIdentifiersPerCredential**
  The last (or unique) identifier cannot be removed.

ACCESS CLASS:

WRITE_SYSTEM

5.3.19 GetCredentialAccessProfiles command

This method returns all the credential access profiles for a credential.

REQUEST:

- **CredentialToken – [pt:ReferenceToken]**
  The token of the credential.

RESPONSE:

- **CredentialAccessProfile – optional, unbounded [tcr:CredentialAccessProfile]**
  Access profiles of the credential.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.

ACCESS CLASS:

READ_SYSTEM

5.3.20 SetCredentialAccessProfiles command

This operation adds or updates the credential access profiles for a credential.

The device shall update the credential access profile if the access profile token in the specified credential access profile matches. Otherwise the credential access profile is added.

REQUEST:

- **CredentialToken – [pt:ReferenceToken]**
  The token of the credential.
- **CredentialAccessProfile – unbounded [tcr:CredentialAccessProfile]**
  Access profiles of the credential.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.
- **env:Sender – ter:CapabilityViolated – ter:MaxAccessProfilesPerCredential**
  There are too many access profiles per credential, see the MaxAccessProfilesPer-Credential capability.
- **env:Sender – ter:CapabilityViolated – ter:CredentialAccessProfileValidity-Supported**
  Credential access profile validity is not supported by the device, see the Credential-AccessProfileValiditySupported capability.
- **env:Sender – ter:InvalidArgVal – ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).

ACCESS CLASS:

ACTUATE

5.3.21 DeleteCredentialAccessProfiles command

This method deletes credential access profiles for the specified credential token.

However, if no matching credential access profiles are found, the corresponding access profile tokens are silently ignored without any response.

REQUEST:

- **CredentialToken – [pt:ReferenceToken]**
  The token of the credential.
- **AccessProfileToken – unbounded [pt:ReferenceToken]**
  Tokens of access profiles.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:NotFound**
  Credential token is not found.

ACCESS CLASS:

ACTUATE

5.3.22 GetWhitelist command

This command requests a list of all whitelisted credential identifiers in the device.

A device with capability MaxWhitelistedItems greater than zero, shall implement this command.

REQUEST:

- **Limit – optional [xs:int]**
  Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.
- **StartReference – optional [xs:string]**
  Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.
- **IdentifierType – optional [xs:string]**
  Get only whitelisted credential identifiers with the specified identifier type.
- **FormatType – optional [xs:string]**
  Get only whitelisted credential identifiers with the specified identifier format type.
- **Value – optional [xs:hexBinary]**
  Get only whitelisted credential identifiers with the specified identifier value.

RESPONSE:

- **NextStartReference – optional [xs:string]**
  StartReference to use in next call to get the following items. If absent, no more items to get.
- **Identifier – optional, unbounded [tcr:CredentialIdentifierItem]**
  The whitelisted credential identifiers matching the request criteria.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:InvalidStartReference**
  StartReference is invalid or has timed out. Client needs to start fetching from the beginning.

- **env:Sender – ter:CapabilityViolated – ter:SupportedIdentifierType**
  Specified identifier type is not supported by the device, see the SupportedIdentifier-Type capability.
- **env:Sender – ter:InvalidArgVal – ter:InvalidFormatType**
  Specified identifier format type is not supported by the device.

ACCESS CLASS:

READ_SYSTEM_SECRET

5.3.23 AddToWhitelist command

This command adds the specified credential identifiers to the whitelist.

A device with capability MaxWhitelistedItems greater than zero, shall implement this command.

If a specified whitelist item also is blacklisted, the item shall be removed from the blacklist.

REQUEST:

- **Identifier – unbounded [tcr:CredentialIdentifierItem]**
  The credential identifiers to be added to the whitelist.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:CapabilityViolated – ter:MaxWhitelistedItems**
  There is not enough space to create a new whitelist item, see the MaxWhitelistedItems capability.
- **env:Sender – ter:InvalidArgVal – ter:TooManyItems**
  Too many items were specified, see MaxLimit capability.
- **env:Sender – ter:CapabilityViolated – ter:SupportedIdentifierType**
  Specified identifier type is not supported by the device, see the SupportedIdentifier-Type capability.
- **env:Sender – ter:InvalidArgVal – ter:InvalidFormatType**
  Specified identifier format type is not supported by the device.

ACCESS CLASS:

WRITE_SYSTEM

5.3.24 RemoveFromWhitelist command

This command removes the specified credential identifiers from the whitelist.

A device with capability MaxWhitelistedItems greater than zero, shall implement this command.

This command is idempotent and is safe to repeat even if the specified whitelist items do not exist.

REQUEST:

- **Identifier – unbounded [tcr:CredentialIdentifierItem]**
  The credential identifiers to be removed from the whitelist.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:TooManyItems**
  Too many items were specified, see MaxLimit capability.

ACCESS CLASS:

WRITE_SYSTEM

5.3.25 DeleteWhitelist command

This command deletes all credential identifiers from the whitelist.

A device with capability MaxWhitelistedItems greater than zero, shall implement this command.

This command is idempotent and is safe to repeat even if the whitelist already is empty.

REQUEST:

This is an empty message.

RESPONSE:

This is an empty message.

FAULTS:

None

ACCESS CLASS:

WRITE_SYSTEM

5.3.26 GetBlacklist command

This command requests a list of all blacklisted credential identifiers in the device.

A device with capability MaxBlacklistedItems greater than zero, shall implement this command.

REQUEST:

- **Limit – optional [xs:int]**
  Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.
- **StartReference – optional [xs:string]**
  Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.
- **IdentifierType – optional [xs:string]**
  Get only blacklisted credential identifiers with the specified identifier type.
- **FormatType – optional [xs:string]**
  Get only blacklisted credential identifiers with the specified identifier format type.
- **Value – optional [xs:hexBinary]**
  Get only blacklisted credential identifiers with the specified identifier value.

RESPONSE:

- **NextStartReference – optional [xs:string]**
  StartReference to use in next call to get the following items. If absent, no more items to get.
- **Identifier – optional, unbounded [tcr:CredentialIdentifierItem]**
  The blacklisted credential identifiers matching the request criteria.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:InvalidStartReference**
  StartReference is invalid or has timed out. Client needs to start fetching from the beginning.
- **env:Sender – ter:CapabilityViolated – ter:SupportedIdentifierType**
  Specified identifier type is not supported by the device, see the SupportedIdentifier-Type capability.
- **env:Sender – ter:InvalidArgVal – ter:InvalidFormatType**
  Specified identifier format type is not supported by the device.

ACCESS CLASS:

READ_SYSTEM_SECRET

5.3.27 AddToBlacklist command

This command adds the specified credential identifiers to the blacklist.

A device with capability MaxBlacklistedItems greater than zero, shall implement this command.

If a specified blacklist item also is whitelisted, the item shall be removed from the whitelist.

REQUEST:

- **Identifier – unbounded [tcr:CredentialIdentifierItem]**
  The credential identifiers to be added to the blacklist.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:CapabilityViolated – ter:MaxBlacklistedItems**
  There is not enough space to create a new blacklist item, see the MaxBlacklistedItems capability.
- **env:Sender – ter:InvalidArgVal – ter:TooManyItems**
  Too many items were specified, see MaxLimit capability.
- **env:Sender – ter:CapabilityViolated – ter:SupportedIdentifierType**
  Specified identifier type is not supported by the device, see the SupportedIdentifier-Type capability.
- **env:Sender – ter:InvalidArgVal – ter:InvalidFormatType**
  Specified identifier format type is not supported by the device.

ACCESS CLASS:

WRITE_SYSTEM

5.3.28 RemoveFromBlacklist command

This command removes the specified credential identifiers from the blacklist.

A device with capability MaxBlacklistedItems greater than zero, shall implement this command.

This command is idempotent and is safe to repeat even if the specified blacklist items do not exist.

REQUEST:

- **Identifier – unbounded [tcr:CredentialIdentifierItem]**
  The credential identifiers to be removed from the blacklist.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender – ter:InvalidArgVal – ter:TooManyItems**
  Too many items were specified, see MaxLimit capability.

ACCESS CLASS:

WRITE_SYSTEM

5.3.29 DeleteBlacklist command

This command deletes all credential identifiers from the blacklist.

A device with capability MaxBlacklistedItems greater than zero, shall implement this command.

This command is idempotent and is safe to repeat even if the blacklist already is empty.

REQUEST:

This is an empty message.

RESPONSE:

This is an empty message.

FAULTS:

None

ACCESS CLASS:

WRITE_SYSTEM

6 Notification topics

6.1 General

This section defines notification topics specific to the credential service.

6.2 Event overview (informative)

The credential service specifies events when credential state changes and when credentials are changed.

The main topics for status changes are:

- tns1:Credential/State/Enabled
- tns1:Credential/State/ApbViolation

The main topics for configuration change notifications are:

- tns1:Configuration/Credential/Changed
- tns1:Configuration/Credential/Removed

6.3 Status changes

6.3.1 General

The device shall provide the status change events to inform subscribed clients when credential entity status is changed. The device shall use the topics defined in this section associated with the respective message description.

6.3.2 Credential

Whenever the credential state (enabled or disabled) is changed, the device shall provide the following event (the Reason field is one of the listed in section 5.3.2.7, e.g. "pt:CredentialLost"):

```
Topic: tns1:Credential/State/Enabled

<tt:MessageDescription IsProperty=”false”>
  <tt:Source>
    <tt:SimpleItemDescription Name=”CredentialToken”
                              Type=”pt:ReferenceToken”/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name=”State”
                              Type=”xs:boolean”/>
    <tt:SimpleItemDescription Name=”Reason”
                              Type=”xs:string”/>
    <tt:SimpleItemDescription Name=”ClientUpdated”
                              Type=”xs:boolean”/>
  </tt:Data>
</tt:MessageDescription>
```

ClientUpdated is set to true if the state change was initiated by the client. ClientUpdated is set to false if the device initiated the state change (e.g. because the wrong PIN was entered three times in a row).

The device shall provide the following event whenever there is any anti-passback violation:

```
Topic: tns1:Credential/State/ApbViolation

<tt:MessageDescription IsProperty=”false”>
  <tt:Source>
    <tt:SimpleItemDescription Name=”CredentialToken”
                              Type=”pt:ReferenceToken”/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name=”ApbViolation”
                              Type=”xs:boolean”/>
    <tt:SimpleItemDescription Name=”ClientUpdated”
                              Type=”xs:boolean”/>
  </tt:Data>
</tt:MessageDescription>
```

6.4 Configuration changes

6.4.1 General

Whenever configuration data has been changed, added or been removed, the device shall provide these events to inform subscribed clients.

6.4.2 Credential

Whenever configuration data for a credential (including credential identifiers and credential access profiles) is changed, or if a credential is added, the device shall provide the following event:

```
Topic: tns1:Configuration/Credential/Changed

<tt:MessageDescription IsProperty=”false”>
  <tt:Source>
    <tt:SimpleItemDescription Name=”CredentialToken”
                              Type=”pt:ReferenceToken”/>
  </tt:Source>
</tt:MessageDescription>
```

Whenever a credential is removed, the device shall provide the following event:

```
Topic: tns1:Configuration/Credential/Removed

<tt:MessageDescription IsProperty=”false”>
  <tt:Source>
    <tt:SimpleItemDescription Name=”CredentialToken”
                              Type=”pt:ReferenceToken”/>
  </tt:Source>
</tt:MessageDescription>
```

Annex A. Revision History

Rev.	Date	Editor	Changes
1.0	Jun 2015	PACS WG	Initial version
17.06	Jun 2017	Hiroyuki Sano	Change request 2068, 2069, 2071, 2079, 2080, 2081, 2098, 2112, 2111, 2113
17.12	Dec 2017	Hiroyuki Sano	Change request 2147, 2149
18.06	Jun 2018	Patrik Björling Rygert	Added support for client-supplied tokens Added default duration for temporarily suspended credentials Added extended grant time for credentials
19.06	Jun 2019	Hiroyuki Sano	Change request 2450, 2455
19.12	Dec 2019	Hiroyuki Sano Patrik Björling Rygert	Change request 2470, 2637 Added support for white- and blacklisting of credential identifiers Updated command table format