ONVIF™
Access Control Service Specification

Version 21.06

June 2021

CONTENTS

Contributors

ASSA ABLOY	Patrik Björling Rygert
ASSA ABLOY	Mattias Rengstedt
Axis Communications AB	Johan Adolfsson
Axis Communications AB	Marcus Johansson
Axis Communications AB	Robert Rosengren
Axis Communications AB	Derek Wang
Axis Communications AB	Emil Selinder
AxxonSoft	Yuri Timenkov
Bosch	Mohane Caliaperoumal
Bosch	Dirk Schreiber
Hirsch Electronics/ Identive Group	Rob Zivney
Honeywell	Marine Drive
Honeywell	Neelendra Bhandari
Honeywell	Uvaraj Thangarajan
Honeywell	Vinay Ghule
PACOM	Eugene Scully
PACOM	Steve Barton
Schneider Electric	Mike Berube
Siemens AG	Klaus Baumgartner
Siemens AG	Suresh Raman
Siemens AG	Suresh Krishnamurthy

1 Scope

1.1 General

This specification defines the web service interface for interaction with physical access control systems. This includes discovering components and their logical composition and controlling them.

Supplementary dedicated services such as low-level door control, schedule management will be defined in separate documents.

Web service usage and common ONVIF functionality are outside of 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
tmd	http://www.onvif.org/ver10/deviceIO/wsdl
tdc	http://www.onvif.org/ver10/doorcontrol/wsdl
tac	http://www.onvif.org/ver10/accesscontrol/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 Door Control Service Specification

<http://www.onvif.org/specs/srv/door/ONVIF-DoorControl-Service-Spec.pdf>
ONVIF Authentication Behavior Service Specification

<http://www.onvif.org/specs/srv/auth/ONVIF-AuthenticationBehavior-Service-Spec.pdf>
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.

Access Control Service	A device implementing the ONVIF Access Control Service Specification (this specification).
Access Control Unit	Part of an access control system that interfaces with readers, locking devices and sensing devices, making a decision to grant or deny access through a portal. From an ONVIF perspective, it is a device or system implementing at least the access control service. Often, it is a microprocessor-based circuit board that manages access to a secure area. The access control unit receives information that it uses to determine through whichdoorsand at what times credential holders are granted access to secure areas. Based on that information, the access control unit can lock/unlock doors, sound alarms, and communicate status to a host computer.
Access Point	A logical composition of a physical door, reader(s) and/or a request-to-exit device controlling access in one direction.
Access Point Disable	If an access point is disabled, it will not be considered in the decision-making process and no commands will be issued from that access point to the door configured for that access point. When an access point is disabled, the associated reader(s) may or may not be disabled or shut down. Clients may still be able to command the door control unit to control associated door even though that door is also referenced by a disabled access point.
Area	A protected or controlled area defined by a physical boundary, through which passage is controlled by means of one or more doors.
Client	An ONVIF service requester. A typical ONVIF network system may have multiple clients that handle device configuration and device management operations for numerous devices. A device providing services may also act as a client to other devices.
Credential	A logical object holding related credential identifiers for a credential holder. E.g. if a PIN is associated with a specific credential number, then both of these identifiers are stored in one credential. Note that the PIN is normally not stored in the physical credential.
Credential Holder	A person holding a credential.
Credential Identifier	Information either memorized or held within a physical credential. Could be a credential number, PIN, biometric information, etc., that can be validated in an access point.
Credential Number	A sequence of bytes uniquely identifying a physical credential at an access point.
Device	An ONVIF service provider implementing one or more ONVIF services. E.g. an access control unit or a door control unit.
Door	A physical door, barrier, turnstile, etc. which can be controlled remotely and restricts access between two areas. A door is usually equipped with an electronic lock and a door monitor.
Door Control Service	A device implementing [ONVIF Door Control Service Specification].
Door Control Unit	From an ONVIF perspective, it is a device or system implementing at least the door control service, but not the access control service. Often, it is a microproces-

	sor-based circuit board that manages door locks and/or door monitors for one or more doors.
Door Lock	A device that secures a door to prevent access, except when explicitly allowed by the access control system. Lock types include electromagnetic, electric strike, etc.
Door Monitor	Electrical component used to monitor the open or closed status of a door, or locked/ unlocked status of a locking device, or the secure/unsecure status of an electro-magnetic lock or armature plate. Also known as door contact sensor.
Duress	Forcing a person to provide access to a secure area against that person's wishes.
Peripheral Device	An I/O device physically wired to the access control unit. Some peripheral devices are associated with an access point (e.g. reader or request-to-exit button), and some are associated with a door (e.g. door monitor).
Physical Credential	Portable device containing a readable unique credential number that can be associated with the credential holder’s data and access rules stored within the electronic access control system. Examples are card, key, fob, smart phone, etc.
Reader	Device for the input of credentials. Examples include card readers, biometric readers, etc.
Request-to-Exit Device	A peripheral device associated with an access point used to initiate free exit.

3.2 Abbreviations

ACMS	Access Control Management System
BMS	Building Management System
HTTP	Hypertext Transfer Protocol
PACS	Physical Access Control System
PSIM	Physical Security Information Management
REX	Request-to-Exit
TLS	Transport Layer Security
VMS	Video Management System

4 Overview

Physical access control is all about who (credential holders) can access what (areas), when (schedules) and how (security levels).

Access points represents the sides of a door, and are the points of access that you need to pass to get to a protected area. An access point is typically equipped with a reader or a request-to-exit device controlling access from one are to another area.

An access point may also be associated with an authentication profile that defines how (i.e. which security level) to get access to an area and when (defined by a schedule). A typical example is an access point that requires card during day time and card plus PIN code during night time.

Every access point offers different capabilities, such as support for duress or what security levels are supported.

The access control service offers commands to manage the access points and areas, to retrieve status information and to control access point instances.

The following picture shows the main data structures involved in the access control service:

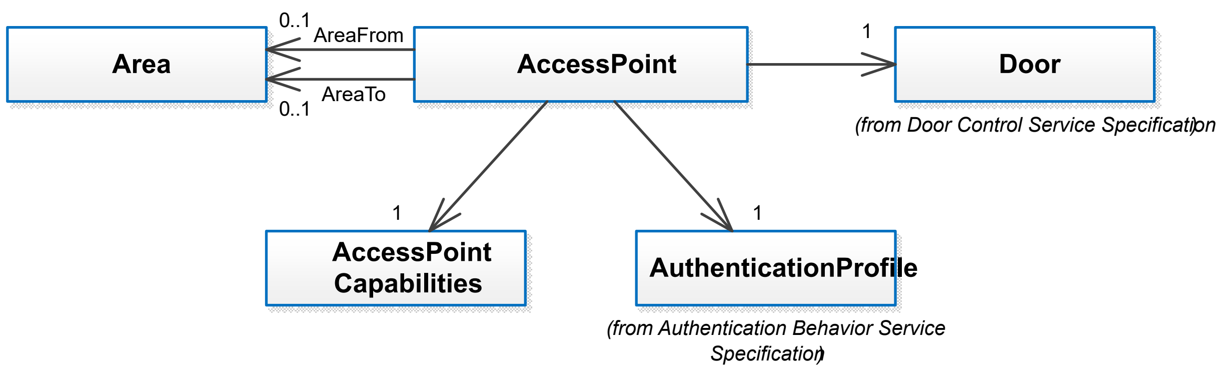
Area
0..1
AreaFrom
AreaTo
0..1
AccessPoint
1
Door
(from Door Control Service Specification)
1
AccessPoint Capabilities
1
AuthenticationProfile
(from Authentication Behavior Service Specification)

Figure 1: Main data structures in the access control service

5 Access control

5.1 Service capabilities

5.1.1 General

A device shall provide service capabilities in two ways:

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

5.1.2 Data structures

5.1.2.1 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.
MaxAccessPoints	Indicates the maximum number of access points supported by the device.
MaxAreas	Indicates the maximum number of areas supported by the device.
ClientSuppliedTokenSupport-ed	Indicates that the client is allowed to supply the token when creating access points and areas. To enable the use of the commands SetAccessPoint and SetArea, the value must be set to true.
AccessPointManagementSup-ported	Indicates that the client can perform CRUD operations (create, read, update and delete) on access points. To enable the use of the com-mands GetAccessPoints, GetAccessPointList, CreateAccessPoint, Mod-ifyAccessPoint, DeleteAccessPoint, SetAccessPointAuthenticationProfile and DeleteAccessPointAuthenticationProfile, the value must be set to true.
AreaManagementSupported	Indicates that the client can perform CRUD operations (create, read, up-date and delete) on areas. To enable the use of the commands GetAreas, GetAreaList, CreateArea, ModifyArea and DeleteArea, the value must be set to true.

5.1.3 GetServiceCapabilities

This operation returns the capabilities of the access control service.

A device which provides the access control service shall implement this method.

REQUEST:

This message is empty

RESPONSE:

- **Capabilities [tac:Serviceapabilities]**
  Set of indicators for function groups as described above.

FAULTS:

None

ACCESS CLASS:

PRE_AUTH

5.2 Access point information

5.2.1 Data structures

5.2.1.1 AccessPointInfo

The AccessPointInfo structure contains basic information about an access point instance. An access point defines an entity a credential can be granted or denied access to. The AccessPointInfo structure provides basic information on how access is controlled in one direction for a door (from which area to which area).

Multiple access points may cover the same door. A typical case is one access point for entry and another for exit, both referencing the same door.

The device shall provide the following fields for each access point instance:

- token

  A service-unique identifier of the access point.

- Name

  A user readable name. It shall be up to 64 characters.

- Entity

  Reference to the entity used to control access; the entity type may be specified by the optional Entity-Type field explained below but is typically a door.

- Capabilities

  The capabilities for the access point.

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

- Description

  Optional user readable description for the access point. It shall be up to 1024 characters.

- AreaFrom

  Optional reference to the area from which access is requested.

- AreaTo

  Optional reference to the area to which access is requested.

- EntityType

  Optional entity type; if missing, a Door type as defined by [ONVIF Door Control Service Specification] should be assumed. This can also be represented by the QName value “tdc:Door” – where tdc is the namespace of the door control service: “http://www.onvif.org/ver10/doorcontrol/wsdl”. This field is provided for future extensions; it will allow an access point being extended to cover entity types other than doors as well.

5.2.1.2 AccessPoint

The AccessPoint structure shall include all properties of the AccessPointInfo structure, and optionally a reference to an authentication profile instance.

The device may provide the following optional fields for each access point instance:

- **AuthenticationProfileToken**

  A reference to an authentication profile which defines the authentication behavior of the access point.

  During the installation phase of an access control system, the authentication behavior is typically not defined, but the list of supported security levels are (see section 5.2.1.3). At a later stage, someone (e.g. a security officer) will define the authentication behavior for the access point by calling the SetAccessPointAuthenticationProfile command (see section 5.2.10).

5.2.1.3 AccessPointCapabilities

The access point capabilities reflect optional functionality of a particular physical entity. Different access point instances may have different set of capabilities. This information may change during device operation, e.g. if hardware settings are changed. The following capabilities are available:

- **DisableAccessPoint**

  Indicates whether or not this access point instance supports the EnableAccessPoint and DisableAccessPoint commands.

- **Duress**

  Indicates whether or not this access point instance supports generation of duress events.

- **AnonymousAccess**

  Indicates whether or not this access point has a REX switch or other input that allows anonymous access.

- **AccessTaken**

  Indicates whether or not this access point instance supports generation of AccessTaken and AccessNotTaken events. If AnonymousAccess and AccessTaken are both true, it indicates that the Anonymous versions of AccessTaken and AccessNotTaken are supported.

- **ExternalAuthorization**

  Indicates whether or not this access point instance supports the ExternalAuthorization operation and the generation of Request events. If AnonymousAccess and ExternalAuthorization are both true, it indicates that the Anonymous version is supported as well.

- **SupportedSecurityLevels**

  A list of security level tokens that this access point supports. See [ONVIF Authentication Behavior Service Specification].

This field is optional, and if omitted, the device cannot support multi-factor authentication for this access point.

Please note that when an access point is updated, then any previously supported security levels are replaced with the new list.

- **SupportedRecognitionTypes**

  A list of recognition types that the device supports. This field is only relevant for devices that are not aware of security levels (see [ONVIF Authentication Behavior Service Specification]).

  Please note that when an access point is updated, then any previously supported recognition types are replaced with the new list.

  Recognition types starting with the prefix pt: are reserved to define ONVIF-specific types as defined in pt:RecognitionType. For custom defined identifier types, free text can be used.

- **IdentifierAccess**

  Indicates whether or not this access point supports the AccessControl/Request/Identifier event to request external authorization.

  Identifier access requires that ExternalAuthorization is set to true.

  The IdentifierAccess capability is typically enabled for devices that do not have any knowledge of credential tokens. When IdentifierAccess is set to true then the device shall support the identifier events.

- **SupportedFeedbackTypes**

  List of supported feedback types. Feedback types starting with the prefix pt: are reserved to define ONVIF-specific types as defined in tac:FeedbackType. For custom defined feedback types, free text can be used.

5.2.2 GetAccessPointInfo

This operation requests a list of AccessPointInfo 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 AccessPointInfo items to get.

RESPONSE:

- **AccessPointInfo - optional, unbounded [tac:AccessPointInfo]**
  List of AccessPointInfo items.

FAULTS:

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

ACCESS CLASS:

READ_SYSTEM

5.2.3 GetAccessPointInfoList

This operation requests a list of all AccessPointInfo 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 Limit is omitted or if the value of Limit is higher than what the device supports, then the device shall return its maximum amount of entries.
- **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.
- **AccessPointInfo - optional, unbdounded [tac:AccessPointInfo]**
  List of AccessPointInfo 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

5.2.4 GetAccessPoints

This operation requests a list of AccessPoint 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.

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

REQUEST:

- **Token - unbounded [pt:ReferenceToken]**
  Tokens of AccessPoint items to get

RESPONSE:

- **AccessPoint - optional, unbounded [tac:AccessPoint]**
  List of AccessPoint items.

FAULTS:

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

ACCESS CLASS:

READ_SYSTEM

5.2.5 GetAccessPointList

This operation requests a list of all AccessPoint 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.

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

REQUEST:

- **Limit - optional [xs:int]**
  Maximum number of entries to return. If Limit is omitted or if the value of Limit is higher than what the device supports, then the device shall return its maximum amount of entries.
- **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.
- **AccessPoint optional, unbounded [tac:AccessPoint]**
  List of access point 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

5.2.6 CreateAccessPoint

This operation creates the specified access point in the device.

The token field of the AccessPoint structure shall be empty and the device shall allocate a token for the access point. 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.

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

REQUEST:

- **AccessPoint [tac:AccessPoint]**
  AccessPoint item to create

RESPONSE:

- **Token [pt:ReferenceToken]**
  Token of created AccessPoint item

FAULTS:

- **env:Receiver - ter:CapabilityViolated - ter:MaxAccessPoints**
  There is not enough space to add new AccessPoint, see the MaxAccessPoints capability
- **env:Sender - ter:InvalidArgVal - ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).

ACCESS CLASS:

WRITE_SYSTEM

5.2.7 SetAccessPoint

This method is used to synchronize an access point in a client with the device.

If an access point with the specified token does not exist in the device, the access point is created. If an access point with the specified token exists, then the access point is modified.

A call to this method takes an AccessPoint structure as input parameter. The token field of the AccessPoint structure shall not be empty.

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:

- **AccessPoint [tac:AccessPoint]**
  AccessPoint item 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:Receiver - ter:CapabilityViolated - ter:MaxAccessPoints**
  There is not enough space to add new AccessPoint, see the MaxAccessPoints capability
- **env:Sender - ter:InvalidArgVal - ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).

ACCESS CLASS:

WRITE_SYSTEM

5.2.8 ModifyAccessPoint

This operation modifies the specified access point.

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

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

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

REQUEST:

- **AccessPoint [tac:AccessPoint]**
  AccessPoint item to modify

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender - ter:InvalidArgVal ter:NotFound**
  Access point token is not found.
- **env:Sender - ter:InvalidArgVal - ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).

ACCESS CLASS:

WRITE_SYSTEM

5.2.9 DeleteAccessPoint

This operation deletes the specified access point.

If it is associated with one or more entities some devices may not be able to delete the access point, 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.

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

REQUEST:

- **Token [pt:ReferenceToken]**
  Token of AccessPoint item to delete.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:NotFound**
  Access point token is not found.
- **env:Sender - ter:InvalidArgVal - ter:ReferenceInUse**
  Failed to delete, Access point token is in use

ACCESS CLASS:

WRITE_SYSTEM

5.2.10 SetAccessPointAuthenticationProfile

This operation defines the authentication behavior for an access point.

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

REQUEST:

- **Token [pt:ReferenceToken]**
  Token of the AccessPoint
- **AuthenticationProfileToken [pt:ReferenceToken]**
  Token of the AuthenticationProfile

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:NotFound**
  The specified token is not found.
- **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:SupportedSecurityLevels**
  The referred authentication profile contains security levels that is not supported by this access point.

ACCESS CLASS:

ACTUATE

5.2.11 DeleteAccessPointAuthenticationProfile

This operation reverts the authentication behavior for an access point to its default behavior.

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

REQUEST:

- **Token [pt:ReferenceToken]**
  Token of the AccessPoint

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:NotFound**
  The specified token is not found.

ACCESS CLASS:

ACTUATE

5.3 Area Information

5.3.1 Data structures

5.3.1.1 AreaInfo

The AreaInfo structure contains basic information about an area instance.

The device shall provide the following fields for each area instance:

- **token**

A service-unique identifier of the area.

- **Name**

  User readable name. It shall be up to 64 characters.

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

- **Description**

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

5.3.1.2 Area

The Area structure shall include all properties of the AreaInfo structure.

Please note that this structure is a placeholder for future attributes.

5.3.2 GetAreaInfo

This operation requests a list of AreaInfo 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 AreaInfo items to get.

RESPONSE:

- **AreaInfo - optional, unbounded [tac:AreaInfo]**
  List of AreaInfo items.

FAULTS:

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

ACCESS CLASS:

  READ_SYSTEM

5.3.3 GetAreaInfoList

This operation requests a list of all AreaInfo 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 Limit is omitted or if the value of Limit is higher than what the device supports, then the device shall return its maximum amount of entries.

- **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.
- **AreaInfo optional, unbounded [tac:AreaInfo]**
  List of area info 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

5.3.4 GetAreas

This operation requests a list of Area 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.

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

REQUEST:

- **Token - unbounded [pt:ReferenceToken]**
  Tokens of Area items to get

RESPONSE:

- **Area - optional, unbounded [tac:Area]**
  List of Area items.

FAULTS:

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

ACCESS CLASS:

READ_SYSTEM

5.3.5 GetAreaList

This operation requests a list of all Area 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.

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

REQUEST:

- **Limit - optional [xs:int]**
  Maximum number of entries to return. If Limit is omitted or if the value of Limit is higher than what the device supports, then the device shall return its maximum amount of entries.
- **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.
- **Area optional, unbounded [tac:Area]**
  List of areas.

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

5.3.6 CreateArea

This operation creates the specified area in the device.

The token field of the Area structure shall be empty and the device shall allocate a token for the area. 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.

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

REQUEST:

- **Area [tac:Area]**
  Area item to create

RESPONSE:

- **Token [pt:ReferenceToken]**
  Token of created Area item

FAULTS:

- **env:Receiver - ter:CapabilityViolated - ter:MaxAreas**
  There is not enough space to add the new area, see the MaxAreas capability
- **env:Sender - ter:InvalidArgVal - ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).

ACCESS CLASS:

WRITE_SYSTEM

5.3.7 SetArea

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

If an area with the specified token does not exist in the device, the area is created. If an area with the specified token exists, then the area is modified.

A call to this method takes an Area structure as input parameter. The token field of the Area structure shall not be empty.

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:

- **Area [tac:Area]**
  Area item 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:Receiver - ter:CapabilityViolated - ter:MaxAreas**
  There is not enough space to add new area, see the MaxAreas capability
- **env:Sender - ter:InvalidArgVal - ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).

ACCESS CLASS:

WRITE_SYSTEM

5.3.8 ModifyArea

This operation modifies the specified area.

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

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

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

REQUEST:

- **Area [tac:Area]**
  Area item to modify

RESPONSE:

This is an empty message.

FAULTS:

- **env:Receiver - ter:CapabilityViolated - ter:OccupancyControlSupported**
  An OccupancyControl structure was passed in the request although occupancy control is not supported by this device.
- **env:Receiver - ter:CapabilityViolated - ter:AntipassbackSupported**
  An Antipassback structure was passed in the request although anti-passback is not supported by this device.

- **env:Sender - ter:InvalidArgVal - ter:NotFound**
  Area token is not found.
- **env:Sender - ter:InvalidArgVal - ter:ReferenceNotFound**
  A referred entity token is not found (some devices may not validate referred entities).

ACCESS CLASS:

WRITE_SYSTEM

5.3.9 DeleteArea

This operation deletes the specified area.

If it is associated with one or more entities some devices may not be able to delete the area, 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.

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

REQUEST:

- **Token [pt:ReferenceToken]**
  Token of Area item to delete.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender - ter:InvalidArgVal - ter:NotFound**
  Area token is not found.
- **env:Sender - ter:InvalidArgVal - ter:ReferenceInUse**
  Failed to delete, the area token is in use

ACCESS CLASS:

WRITE_SYSTEM

5.4 Access point status

5.4.1 General

The state of the access point is determined by a number of operations that can be performed on it depending on its capabilities (please refer to access point capabilities in section 5.2).

5.4.2 Data structures

5.4.2.1 AccessPointState

The AccessPointState contains state information for an access point. A device shall provide the following fields for each access point instance:

- **Enabled**

  Indicates that the access point is enabled. By default this field value shall be True, if the DisableAccessPoint capabilities is not supported.

5.4.3 GetAccessPointState

This operation requests the AccessPointState for the access point instance specified by the token.

REQUEST:

- **Token [pt:ReferenceToken]**
  Token of AccessPoint instance to get AccessPointState for.

RESPONSE:

- **AccessPointState [tac:AccessPointState]**
  AccessPointState item.

FAULTS:

- **env:Sender ter:InvalidArgVal ter:NotFound**
  AccessPoint is not found

ACCESS CLASS:

READ_SYSTEM_SENSITIVE

5.5 Access control commands

5.5.1 General

The service control commands contain operations that allow modifying access point states and controlling access points.

5.5.2 Data structures

5.5.2.1 Enumeration: Decision

The Decision enumeration represents a choice of two available options for an access request:

- **Granted**

  The decision is to grant access.

- **Denied**

  The decision is to deny access.

5.5.2.2 Enumeration: FeedbackType

Non-normative enumeration that describes the ONVIF defined feedback types. These types are used in string fields where extendibility is desired. Strings starting with the prefix pt: are reserved to define ONVIF-specific types. For custom defined feedback types, free text can be used.

The following types are defined by ONVIF:

- **pt:Disabled**

  Indication that the access point is disabled.

- **pt:Idle**

  Indication of normal idle state.

- **pt:DoorLocked**

  Indication that the door is locked.

- **pt:DoorUnlocked**

Indication that the door is unlocked.

- **pt:DoorOpenTooLong**

  Indication that the door has been held open too long.

- **pt:DoorPreAlarmWarning**

  Indication that the door must be closed to avoid an alarm.

- **pt:RequireIdentifier**

  Indication that at least one more identifier is required to grant access.

- **pt:TextMessage**

  Display the provided text message.

- **pt:Processing**

  Indication that processing is in progress.

- **pt:RetryIdentifier**

  Indication that the credential holder needs to retry providing the identifier. This could be due to card read failure, face detection problem etc.

- **pt:AccessGranted**

  Indication that access is granted.

- **pt:AccessDenied**

  Indication that access is denied.

- **pt:Ok**

  Generic ok indication.

- **pt:Fault**

  Generic fault indication.

- **pt:Warning**

  Generic warning indication.

- **pt:Alarm**

  Generic alarm indication.

5.5.3 EnableAccessPoint

This operation allows enabling an access point.

A device that signals support for DisableAccessPoint capability for a particular access point instance shall implement this command.

REQUEST:

- **Token [pt:ReferenceToken]**
  Token of the AccessPoint instance to enable.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender ter:InvalidArgVal ter:NotFound**
  The specified token is not found.
- **env:Receiver ter:ActionNotSupported ter:NotSupported**
  The operation is not supported.

ACCESS CLASS:

ACTUATE

5.5.4 DisableAccessPoint

This operation allows disabling an access point.

A device that signals support for the DisableAccessPoint capability for a particular access point instance shall implement this command.

REQUEST:

- **Token [pt:ReferenceToken]**
  Token of the AccessPoint instance to disable.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender ter:InvalidArgVal ter:NotFound**
  The specified token is not found.
- **env:Receiver ter:ActionNotSupported ter:NotSupported**
  The operation is not supported.

ACCESS CLASS:

ACTUATE

5.5.5 ExternalAuthorization

This operation allows to deny or grant decision at an access point instance.

A device that signals support for ExternalAuthorization capability for a particular access point instance shall implement this method.

REQUEST:

- **AccessPointToken [pt:ReferenceToken]**
  Token of the AccessPoint instance.
- **CredentialToken - optional [pt:ReferenceToken]**
  Optional token of the Credential involved.
- **Reason - optional [xs:string]**
  Optional reason for decision.
- **Decision [tac:Decision]**
  Decision - Granted or Denied (extendable).

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender ter:InvalidArgVal ter:NotFound**
  AccessPoint is not found.
- **env:Receiver ter:ActionNotSupported ter:NotSupported**
  The operation is not supported.

ACCESS CLASS:

ACTUATE

5.5.6 Feedback

This operation controls how the specified access point should indicate feedback.

A client can instruct the access point about the door status or that one or more identifiers are needed to grant access, etc. It is typically used in conjunction with the AccessControl/Request/Identifier event and the ExternalAuthorization operation to indicate progress and required recognition methods.

The supported feedback types are indicated in the SupportedFeedbackTypes field in the access point capabilities.

In cases where multiple combinations of recognition methods are possible, the RecognitionType field can contain multiple values. E.g. for the security level "Card+PIN or Card+Fingerprint", the feedback command for requesting the second recognition method would contain both pt:PIN and pt:Fingerprint in the RecognitionType field.

If the device supports at least one feedback type the device shall implement this command.

REQUEST:

- **AccessPointToken [pt:ReferenceToken]**
  Token of the access point to control.
- **FeedbackType [pt:string]**
  The feedback type to use. Feedback types starting with the prefix pt: are reserved to define ONVIF-specific types as defined in tac:FeedbackType. For custom defined feedback types, free text can be used. If feedbacktype is set to pt:RequireIdentifier, the RecognitionType field shall provide the required type(s). If the feedback type is not supported, it shall be ignored.
- **RecognitionType - optional, unbounded [xs:string]**
  Optional list of recognition types requested by a client to get closer to making a decision. Used if FeedbackType is set to pt:RequireIdentifier. If a recognition type is not supported, it shall be ignored.
- **TextMessage - optional [xs:string]**
  Optional textual feedback message. If not supported by the access point it shall be ignored.

RESPONSE:

This is an empty message.

FAULTS:

- **env:Sender ter:InvalidArgVal ter:NotFound**
  AccessPoint is not found.

ACCESS CLASS:

ACTUATE

6 Notification topics

6.1 General

This section defines notification topics specific to the access control service.

6.2 Event overview

6.2.1 General

The access control service specifies events to be used for access transactions, for example an access request is made and an access is granted or denied, when duress is detected, and when an important configuration has been changed.

The main topics for access transaction events are:

- tns1:AccessControl/AccessGranted/ - when access is granted.
- tns1:AccessControl/AccessTaken/ - when access is taken after being granted.
- tns1:AccessControl/AccessNotTaken/ - when access is not taken after being granted.
- tns1:AccessControl/Denied/ - when access is denied.
- tns1:AccessControl/Duress - when duress is detected.
- tns1:AccessControl/Request/ - when external authorization is requested or has timed out.

The main topic for status updates is:

- tns1:AccessPoint/State/ - for status updates.

The main topics for configuration change notifications are:

- tns1:Configuration/AccessPoint - when access point configuration has been changed.
- tns1:Configuration/Area - when area configuration has been changed.

Note that the term “main topic” here means that a client may subscribe to e.g. tns1:AccessControl/Access-Granted, but the actual event sent (and thus received) may be the main topic itself or any subtopic of the main topic (such as tns1:AccessControl/AccessGranted/Credential). New subtopics may be defined in the future.

6.3 Access granted

6.3.1 General

Whenever a positive access decision is made, i.e., access is granted, the device shall provide a corresponding event message as per the following sub-sections.

The event shall be sent immediately once the decision is made regardless of whether access was taken or not.

6.3.2 Anonymous

Whenever access is granted for an anonymous user at an access point, a device that signals AnonymousAc-cess capability for a particular access point instance shall provide the following event:

Topic: tns1:AccessControl/AccessGranted/Anonymous

```
<tt:MessageDescription IsProperty="false">
```

```
  <tt:Source>
 <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
 <tt:SimpleItemDescription Name="External" Type="xs:boolean"/>
  </tt:Data>
</tt:MessageDescription>
```

If the command was triggered as a result of the ExternalAuthorization command, the data element External shall be set to true, otherwise the element is optional.

6.3.3 Credential

Whenever a valid credential has passed all the necessary checks and a credential holder is granted access to the access point, but not yet accessed it (entered or exited), the device shall provide the following event data:

Topic: tns1:AccessControl/AccessGranted/Credential

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="External" Type="xs:boolean"/>
    <tt:SimpleItemDescription Name="CredentialToken" Type="pt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="CredentialHolderName" Type="xs:string"/>
    <tt:SimpleItemDescription Name="SecurityLevelToken" Type="xs:string"/>
    <tt:SimpleItemDescription Name="ExemptedAccess" Type="xs:boolean"/>
  </tt:Data>
</tt:MessageDescription>
```

If the command was triggered as a result of the ExternalAuthorization command, the data element External shall be set to true, otherwise the element is optional.

The data elements CredentialHolderName, SecurityLevelToken and ExemptedAccess are optional.

6.3.4 Identifier

Whenever a credential identifier is granted access, but the credential token is not known (e.g. when whitelisted), the device shall provide the following event:

Topic: tns1:AccessControl/AccessGranted/Identifier

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="IdentifierType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="FormatType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="IdentifierValue" Type="xs:hexBinary"/>
  </tt:Data>
</tt:MessageDescription>
```

The content of the identifier type string must be one of the ONVIF-specific types defined in pt:RecognitionType (except pt:REX) or a vendor-specific type.

The content of the format type string must be one of the BACnet formats referred to in the Credential Specification (e.g. WIEGAND26) or a vendor-specific format.

The content of the identifier value must contain the credential number, PIN or any other value identifying the credential that has been granted access.

6.4 Access taken

6.4.1 General

A device that signals support for AccessTaken capability for a particular access point instance shall provide a corresponding event to notify client whenever an authorized person takes access.

6.4.2 Anonymous

A device that signals support for AnonymousAccesscapability for a particular access point instance shall provide the following event:

Topic: tns1:AccessControl/AccessTaken/Anonymous

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

6.4.3 Credential

When the device detects that access is taken and the credential can be identified (credential token is known), it shall provide the following event:

Topic: tns1:AccessControl/AccessTaken/Credential

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="CredentialToken" Type="pt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="CredentialHolderName" Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

The data element CredentialHolderName is optional.

6.4.4 Identifier

When the device detects that access is taken, and the credential identifier is known but the credential token is not known, it shall provide the following event:

Topic: tns1:AccessControl/AccessTaken/Identifier

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="IdentifierType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="FormatType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="IdentifierValue" Type="xs:hexBinary"/>
  </tt:Data>
</tt:MessageDescription>
```

The content of the identifier type string must be one of the ONVIF-specific types defined in pt:RecognitionType (except pt:REX) or a vendor-specific type.

The content of the format type string must be one of the BACnet formats referred to in the Credential Specification (e.g. WIEGAND26) or a vendor-specific format.

The content of the identifier value must contain the credential number, PIN or any other value identifying the credential that has taken access.

6.5 Access not taken

6.5.1 General

A device that signals support for AccessTaken capability for a particular access point instance shall provide a corresponding event to notify client whenever a person was authorized but did not take access in time.

6.5.2 Anonymous

A device that signals support for AnonymousAccesscapability for a particular access point instance shall provide the following event:

Topic: tns1:AccessControl/AccessNotTaken/Anonymous

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

6.5.3 Credential

When the device detects that access is not taken and the credential can be identified (credential token is known), it shall provide the following event:

Topic: tns1:AccessControl/AccessNotTaken/Credential

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="CredentialToken" Type="pt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="CredentialHolderName" Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

The data element CredentialHolderName is optional.

6.5.4 Identifier

When the device detects that access is not taken, and the credential identifier is known but the credential token is not known, it shall provide the following event:

Topic: tns1:AccessControl/AccessNotTaken/Identifier

<tt:MessageDescription IsProperty="false">

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="IdentifierType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="FormatType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="IdentifierValue" Type="xs:hexBinary"/>
  </tt:Data>
```

The content of the identifier type string must be one of the ONVIF-specific types defined in pt:RecognitionType (except pt:REX) or a vendor-specific type.

The content of the format type string must be one of the BACnet formats referred to in the Credential Specification (e.g. WIEGAND26) or a vendor-specific format.

The content of the identifier value must contain the credential number, PIN or any other value identifying the credential that has not taken access.

6.6 Access denied

6.6.1 General

The device shall provide one of the events as per the following sub-sections whenever a person is denied to access. The following applies to all subsections:

Even if there are multiple reasons for denial, a device shall only send one reason and how the device chooses the reason in this case is outside the scope of this standard. The denial reason shall be present in the parameter Reason.

If the denial reason is due to the credential being disabled (CredentialNotEnabled, CredentialNotActive or CredentialExpired) more detailed information about the reason of disabling the credential can be found in the credential itself.

The following strings shall be used for the reason field:

- **CredentialNotEnabled**

  The device shall provide this reason whenever a valid credential is not enabled or has been disabled, e.g. due to credential being lost etc., to prevent unauthorized entry.

- **CredentialNotActive**

  The device shall provide this reason whenever a valid credential is presented though it is not active yet, e.g. the credential was presented before the start date.

- **CredentialExpired**

  The device shall provide this reason whenever a valid credential was presented after its expiry date.

- **InvalidPIN**

  The device shall provide this reason whenever an entered PIN code does not match the credential.

- **NotPermittedAtThisTime**

  The device shall provide this reason whenever a valid credential is denied access to the requested access point because the credential is not permitted at the moment.

- **Unauthorized**

  The device shall provide this reason whenever the presented credential is not authorized.

- **Other**

  The device shall provide this reason whenever the request is denied and no other specific event matches it or is supported by the service.

More values may be defined by either future revisions of this specification or as vendor specific extensions. To allow for this, a client shall treat unknown strings as “Other”.

While the Reason strings originally defined by this standard do not use a QName style syntax, extensions to the Reason values will always use a QName style syntax. The prefix “pt” is reserved for use by ONVIF, i.e. "pt:<reason>". Vendor specific extensions should choose a suitable prefix.

Current extensions are:

- **pt:InvalidIdentifierValue**

  The device shall provide this reason whenever an entered second recognition method (such as a fingerprint or a second credential identifier), does not match the initial credential identifier.

6.6.2 Anonymous

The device that signals support for the AnonymousAccesscapability for a particular access point instance shall provide the following event when access is denied and credential information is not provided:

Topic: tns1:AccessControl/Denied/Anonymous

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="External" Type="xs:boolean"/>
    <tt:SimpleItemDescription Name="Reason" Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

If the command was triggered as a result of the ExternalAuthorization command, the data element External shall be set to true, otherwise the element is optional.

6.6.3 Credential

When the device denies access and the credential can be identified (credential token is known), it shall provide the following event:

Topic: tns1:AccessControl/Denied/Credential

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="External" Type="xs:boolean"/>
    <tt:SimpleItemDescription Name="CredentialToken" Type="pt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="CredentialHolderName" Type="xs:string"/>
    <tt:SimpleItemDescription Name="SecurityLevelToken" Type="xs:string"/>
    <tt:SimpleItemDescription Name="Reason" Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

If the command was triggered as a result of the ExternalAuthorization command, the data element External shall be set to true, otherwise the element is optional.

The data element CredentialHolderName and SecurityLevelToken are optional.

6.6.4 Identifier

Whenever a credential identifier is denied access, but the credential token is not known (e.g. when blacklisted), the device shall provide the following event:

Topic: tns1:AccessControl/Denied/Identifier

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="IdentifierType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="FormatType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="IdentifierValue" Type="xs:hexBinary"/>
    <tt:SimpleItemDescription Name="Reason" Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

The content of the identifier type string must be one of the ONVIF-specific types defined in pt:RecognitionType (except pt:REX) or a vendor-specific type.

The content of the format type string must be one of the BACnet formats referred to in the Credential Specification (e.g. WIEGAND26) or a vendor-specific format.

The content of the identifier value must contain the credential number, PIN or any other value identifying the credential that was denied access.

6.6.5 CredentialNotFound

Under some circumstances a device may be not able to resolve authentication data to a credential token.

Whenever there is no credential matching the request stored in the device, the device shall provide the following event:

Topic: tns1:AccessControl/Denied/CredentialNotFound

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="IdentifierType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="FormatType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="IdentifierValue" Type="xs:hexBinary"/>
  </tt:Data>
</tt:MessageDescription>
```

The content of the identifier type string must be one of the ONVIF-specific types defined in pt:RecognitionType (except pt:REX) or a vendor-specific type.

The content of the format type string must be one of the BACnet formats referred to in the Credential Specification (e.g. WIEGAND26) or a vendor-specific format. This field is optional.

The content of the identifier value must contain the credential number, PIN or any other value identifying the credential that could not be found.

6.6.5.1 Card

Please note that the use of this event is provided for backward compatibility reasons. See also CredentialNotFound.

Whenever there is no credential matching the request stored in the device, the device shall provide the following event:

Topic: tns1:AccessControl/Denied/CredentialNotFound/Card

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="Card" Type="xs:string"/>
  </tt:Data>
```

```
</tt:MessageDescription>
```

The content of the Card string is vendor specific. It may contain the complete identification string of the card, part of this information or remain empty.

6.7 Duress

A device that signals support for the Duress capability for a particular access point instance shall provide the following event whenever a condition of duress is detected.

Topic: tns1:AccessControl/Duress

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="CredentialToken" Type="pt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="CredentialHolderName" Type="xs:string"/>
    <tt:SimpleItemDescription Name="Reason" Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

The data parameters CredentialToken and CredentialHolderName are optional and may be omitted for anonymous access.

6.8 External authorization

6.8.1 General

A device that signals support for ExternalAuthorization capability for a particular access point instance shall provide applicable events defined in this section whenever it requests for external authorization. These notification messages shall be used in conjunction with corresponding access control service operations which provide feedback to the device.

6.8.2 Anonymous

Whenever a device that signals AnonymousAccess capability for particular access point instance requests external agent to authorize a person when credential information is not available, e.g. when a REX button has been pressed and operator’s confirmation is needed, it shall provide the following event:

Topic: tns1:AccessControl/Request/Anonymous

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

6.8.3 Credential

Whenever the device requests an external authorization, and the credential token is known, the device shall send the following event:

Topic: tns1:AccessControl/Request/Credential

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="CredentialToken" Type="pt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="CredentialHolderName" Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

CredentialHolderName is optional and may be omitted or an empty string if it cannot be resolved.

6.8.4 Identifier

If the IdentifierAccess capability is set to true for the access point, then whenever the device requests an external authorization, and the credential identifier is known but the credential token is not known, the device shall send the following event:

Topic: tns1:AccessControl/Request/Identifier

```
<tt:MessageDescription IsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="IdentifierType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="FormatType" Type="xs:string"/>
    <tt:SimpleItemDescription Name="IdentifierValue" Type="xs:hexBinary"/>
  </tt:Data>
</tt:MessageDescription>
```

The content of the identifier type string must be one of the ONVIF-specific types defined in pt:RecognitionType (except pt:REX) or a vendor-specific type.

The content of the format type string must be one of the BACnet formats referred to in the Credential Specification (e.g. WIEGAND26) or a vendor-specific format.

The content of the identifier value must contain the credential number, PIN or any other value identifying the credential that is requesting access.

6.8.5 Timeout

A device shall provide the following event, whenever an external authorization request times out:

Topic: tns1:AccessControl/Request/Timeout

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

6.8.6 Example

A client that implements support for external authorization typically listens to the AccessControl/Request/<subtopic> events. When any of these events arrive, they are evaluated. If access is granted or denied, the ExternalAuthorization command is called on the device.

6.9 Status changes

6.9.1 General

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

6.9.2 Access point

The device that signals support for the DisableAccessPoint capability for a particular access point instance shall provide the following event whenever the state, enabled or disabled, of this access point is changed:

Topic: tns1:AccessPoint/State/Enabled

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State" Type="xs:boolean"/>
  </tt:Data>
</tt:MessageDescription>
```

6.9.3 Security level

If more than one security level is supported by the device (see SupportedSecurityLevels capability), then the device shall be capable of generating the following event whenever the active security level have changed for an access point.

Topic: tns1:AccessPoint/State/SecurityLevel

```
<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken" Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="AuthenticationProfileToken" Type="pt:ReferenceToken"/>
    <tt:SimpleItemDescription Name="ActiveSecurityLevelToken" Type="pt:ReferenceToken"/>
  </tt:Data>
</tt:MessageDescription>
```

6.10 Configuration changes

6.10.1 General

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

6.10.2 Access Point

Whenever important configuration data for an access point is changed or an access point is added, the device shall provide the following event:

Topic: tns1:Configuration/AccessPoint/Changed

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

Whenever an access point is removed, the device shall provide the following event:

Topic: tns1: Configuration/AccessPoint/Removed

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

6.10.3 Area

Whenever configuration data for an area is changed or an area is added, the device shall provide the following event:

Topic: tns1:Configuration/Area/Changed

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

Whenever an area is removed, the device shall provide the following event:

Topic: tns1: Configuration/Area/Removed

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

Annex A. Use case examples

The following picture shows a scenario with multiple devices with different roles/profiles:

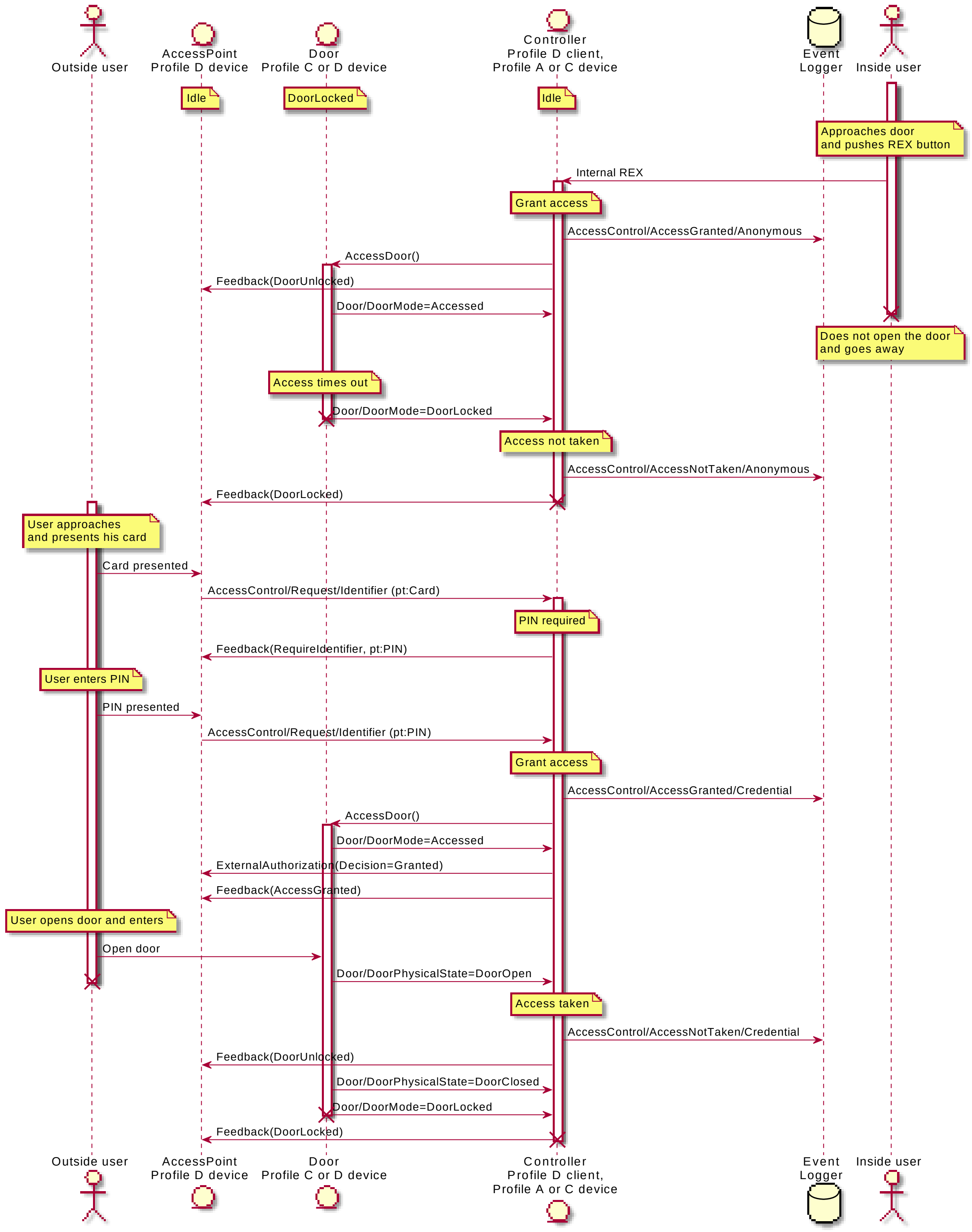
Outside user
AccessPoint Profile D device
Door Profile C or D device
Controller Profile D client, Profile A or C device
Event Logger
Inside user
Idle
DoorLocked
Idle
Approaches door and pushes REX button
Internal REX
Grant access
AccessControl/AccessGranted/Anonymous
AccessDoor()
Feedback(DoorUnlocked)
Door/DoorMode=Accessed
Does not open the door and goes away
Access times out
Door/DoorMode=DoorLocked
Access not taken
AccessControl/AccessNotTaken/Anonymous
Feedback(DoorLocked)
User approaches and presents his card
Card presented
AccessControl/Request/Identifier (pt:Card)
PIN required
Feedback(RequireIdentifier, pt:PIN)
User enters PIN
PIN presented
AccessControl/Request/Identifier (pt:PIN)
Grant access
AccessControl/AccessGranted/Credential
AccessDoor()
Door/DoorMode=Accessed
ExternalAuthorization(Decision=Granted)
Feedback(AccessGranted)
User opens door and enters
Open door
Door/DoorPhysicalState=DoorOpen
Access taken
AccessControl/AccessNotTaken/Credential
Feedback(DoorUnlocked)
Door/DoorPhysicalState=DoorClosed
Door/DoorMode=DoorLocked
Feedback(DoorLocked)
Outside user
AccessPoint Profile D device
Door Profile C or D device
Controller Profile D client, Profile A or C device
Event Logger
Inside user

Figure A.1: Use of AccessDoor, Feedback and ExternalAuthorization

Annex B. Revision History

Rev.	Date	Editor	Changes
1.0	Apr-2013	Yuri Timenkov	Initial version
1.0.1	Aug-2013	Hans Busch	Change Request 1053
1.0.2	Dec-2013	Michio Hirai	Change Request 1234
1.0.3	Jun-2014	Michio Hirai	Change Request 1363, 1367, 1355, 1357, 1364, 1366, 1347, 1348, 1365
1.1	Jun-2017	Patrik Björling Rygert	Change Request 1803
17.12	Dec-2017	Hiroyuki Sano	Change Request 2170, 2171
18.06	Jun-2018	Patrik Björling Rygert	Change Request 2300 Added support for client-supplied tokens Added full management support for access points Added full management support for areas Added support for authentication behavior
19.12	Dec-2019	Hiroyuki Sano	Change Request 2474, 2481, 2535, 2536, 2537, 2538, 2539, 2540, 2541, 2542, 2632, 2633, 2634
20.06	Jun-2020	Patrik Björling Rygert	Added support for feedback indication at access point
21.06	Jun-2021	Johan Adolfsson	Added Use case examples appendix