ONVIF™
Access Control Service Specification

Version 1.0.3
June 2014

CONTENTS

Contributors

Version 1

Axis Communications AB	JohanAdolfsson
Axis Communications AB	MarcusJohansson
AxxonSoft	YuriTimenkov
Bosch	MohaneCaliaperoumal
Hirsch Electronics/ Identive Group	RobZivney
Honeywell	MarineDrive
Honeywell	NeelendraBhandari
PACOM	EugeneScully
PACOM	SteveBarton
Schneider Electric	MikeBerube
Siemens Building Technologies	Klaus Baumgartner
Siemens Building Technologies	SureshRaman

1 Scope

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 the ONVIF core specification for more information.

2 Normative references

ONVIF Core Specification
<http://www.onvif.org/specs/core/ONVIF-Core-Spec-v220.pdf>
.

3 Terms and Definitions

3.1 Definitions

Access Controller	From 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 controller receives information that it uses to determine through which doors and at what times cardholders are granted access to secure areas. Based on that information, the controller can lock/unlock doors, sound alarms, and communicate status to a host computer.
Access Point	A logical composition of a physical door and ID point(s) 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 ID Point may or may not be disabled or shut down. Clients may still be able to command the Door Controller to control associated door even though that door is also referenced by a disabled access point.
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 Number	A sequence of bytes uniquely identifying a credential at an Access Point.
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 Controller	From ONVIF perspective, it isa device or system implementing at least the Door Control Service, but not the Access Control Service. Often, it is a microprocessor 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 electromagnet, electric strike, etc.
Door Monitor	Also known as Door Contact Sensor
Duress	Forcing a person to provide access to a secure area against that person's wishes.
ID Point	A device that converts reader signals to protocols recognized by an authorization engine. It can be card reader, REX, biometric reader etc.

3.2 Abbreviations

ACMS	Access Control Management System

BMS	Building Management System
DCU	Door Control Unit
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

4.1 Introduction

This specification provides interfaces to enable integration of physical security equipment with other devices (e.g., video cameras) and systems (e.g., video monitoring system, PSIM, BMS etc.).

The standard specifies only the data and control flow between a client and the ONVIF services without reference to any physical device as the services required to implement an ONVIF compliant PACS can be not necessarily implemented on a single device (i.e., all services can be run on a control panel, event aggregator software on PC, etc.).

The standard does not define internal communication between an Access Controller and its components if they are implemented on a single device. However, future versions may provide interfaces to integrate these parts from different vendors.

4.2 Interoperability

The ONVIF specification provides new interoperability opportunities by separating configuration from control and monitoring. In traditional systems, the central management system pushes all configurations data to devices on startup and expects that this configuration data is not changed by other clients. Instead, each ONVIF client shall expect that all information is stored on end-devices and can be changed by others.

ONVIF PACS relies on Service-Oriented Architecture principles. This allows installations where different components can be replaced or updated independently.

4.3 Event handling

Event handling is a crucial part of Access Control operations. In addition to real-time event delivery ONVIF provides the means for accessing stored events on the edge to deliver them if connection is lost.

Events are divided into 3 groups depending on their origin and purpose:

1. Configuration change events. These events are provided to achieve interoperability between several clients that control a single device simultaneously.

2. Transaction events. The core functionality of PACS that provides daily monitoring of all access events, including access granted events designed to notify clients about all detailed information (who, when and probably where have passed) on every particular access granted event, access denial events (that may or may not contain reason information), etc.

3. Alarms and faults events. These events provide health status monitoring allowing operators take action in case of hardware failure, intrusion or other suspicious activity.

Please refer to ONVIF Core specification for details on event delivery mechanism and section 6 for the list of events defined by this document.

4.4 Architecture

The ONVIF specification does not mandate any specific physical device layout. The scheme provided below is not intended to be taken as a pattern but to serve as a reference for better understanding of the given specification. Based on the definitions below, different physical configurations of an access controlled door are possible.

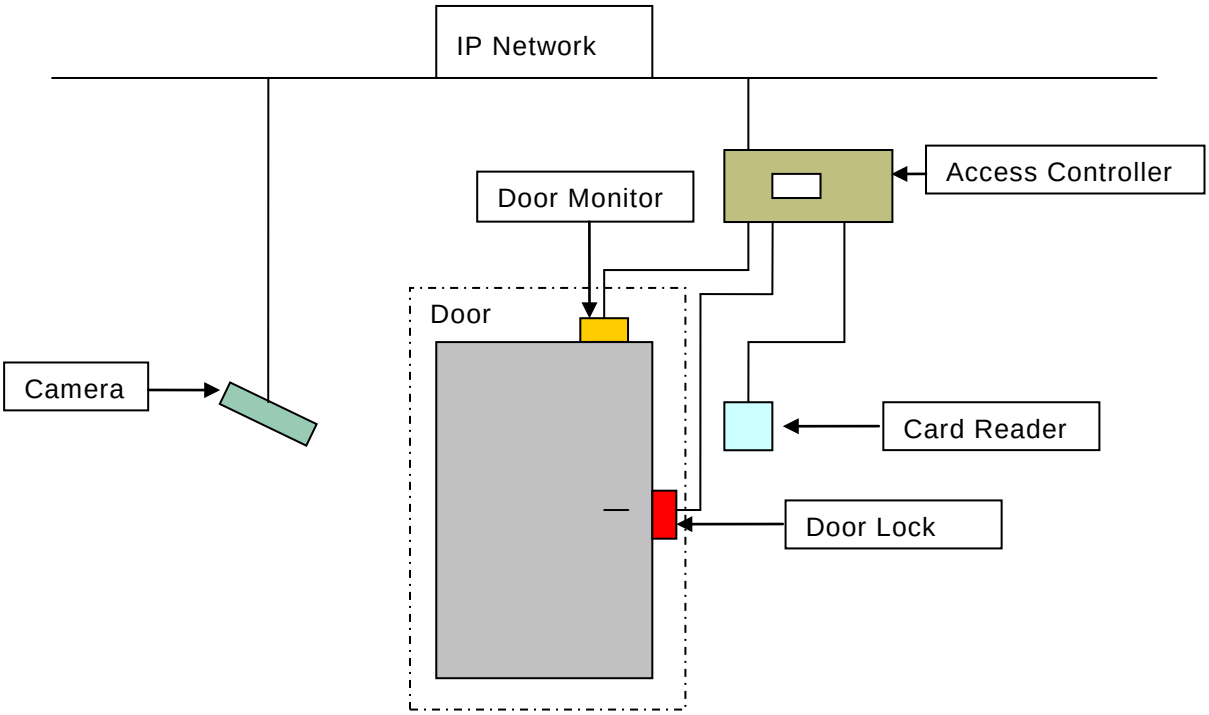

Figure 1. A typical access controller door

A door that is controlled by a physical access control system is equipped with the following devices:

- An access controller that provides connections for card reader, door sensor, door lock and additional digital inputs and outputs. This panel enables the software to interact with the physical devices. Sometimes these panels also contain storage and local intelligence to provide an offline functionality, so that the door will work as expected, even if there is no management system above available.
- A card reader that is able to read the identification data from a credential. In most cases a card reader is only mounted at the outer (unsecure) side of the door. If the system monitors when somebody is leaving an area, a card reader will be mounted on both sides of the door.
- A door monitor that signals the control panel, that the door is open or closed.
- A magnetic door lock that locks the door the most of the time and that can be engaged by the control panel to release the door, in case an authorized credential is recognized.
- (Optional) a camera that shows the person waiting for the door to be opened.

A logical composition of a physical door and an ID point (reader, REX etc.) which allows authentication and passage in one direction is referred to as an access point in this specification. A door can belong to multiple access points at the same time. A typical case is one access point for entry and another for exit, both referencing the same door.

4.5 External authorization

External authorization is a feature used to take access decisions for an access point outside the access controller. External authorization entails but is not limited to a policy within the access controller where the access controller delegates the access decisions to an outside entity such as a guard or ACMS.

4.6 Security considerations

Thespecification assumes possibility of building PACS systems interacting on device level. This implies more security consideration than regular client-server interaction. ONVIF Core specification defines several mechanisms to achieve this. They include, but are not limited to

- TLS for transport encryption
- HTTP and WS-Security for client authentication
- User management and Access Policies for client authorization
- IEEE 802.1X certificate management for server authentication and spoofing protection.

Please refer to the respective whitepapers and specifications for more information.

4.7 Physical and logical security

This specification distinguishes two types of security:

- Physical security prevents unauthorized personnel, attackers or accidental intruders from physically accessing a building, room or etc.
- Logical security protects information and restricts access to managing equipment.

4.8 Design considerations

4.8.1 Instance-level capabilities

A single PACS device may have diverse components of the same type. For example, a controller may operate two doors: one at the entrance to the building which has secure locking, monitoring and alarm abilities, and the other one is internal which can be only locked and unlocked.

Therefore, capabilities can be divided into 2 groups:

- Overall service capabilities;
- Capabilities for a particular entity in the service. It can also work in conjunction with GetEventProperties function to provide finer control over system.

Please refer to section 5.1 for more information.

4.8.2 Retrieving status

The PACS family of ONVIF services defines 2 parallel mechanisms for retrieving status information for most entities:

- Get<*Entity*>State functions return a cumulative snapshot of the current state, operating mode and other run-time information.
- The Event Service returns up-to-date and consistent states of entities. Each entity provides a set of events (usually one per each field in the State type) to notify a client about status changes. As far as these events are property events, a client receives the currentstate whenever a new subscription is initialized.

4.8.3 Retrieving system configuration

The PACS family of ONVIF services defines several Get-functions that can return data incrementally. These functions allow the processing ofa large number of entities even though resources are highly constrained.

To return data incrementally, these functions make use of a parameter called StartReference. StartReference is a device internal identifier used to continue fetching data from the last position, and allows a client to iterate over a large dataset in smaller chunks. The device

handles a reasonable number of different StartReferences at the same time and they live for a reasonable time so that clients are able to fetch complete datasets.

An ONVIF compliant client always passes the value returned from a previous request to continue fetching data. Client do not use the same reference more than once.

For example, the StartReference can be incrementing start position number or underlying database transaction identifier.

The returned NextStartReferenceis used as the StartReference parameter in successive calls, and may be changed by device in each call.

The following pseudo-code demonstrates how information about all Access Points can be obtained from a device:

```
StartRef = null
do {
Response = GetAccessPointInfoList(StartReference = StartRef)
if (Response.AccessPointInfo != null) {
AllAccessPoints.Append(Response.AccessPointInfo)
}
StartRef = Response.NextStartReference
} while (StartRef != null)
```

5 Access Control

This service offers commands to retrieve status information and to control AccessPoint instances.

5.1 Service capabilities

5.1.1 General

An ONVIF compliant device shall provide service capabilities in two ways:

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

2. 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.

5.1.3 GetServiceCapabilities command

This operation returns the capabilities of the Access Control service. A device shall support this command.

Table 1: GetServiceCapabilities command

GetServiceCapabilities	Access Class: PRE_AUTH
Message name	**Description**
GetServiceCapabilitiesRequest	*This message shall be empty*
GetServiceCapabilitiesResponse	*This message contains:* • *"Capabilities": The capability response message contains the requested Access Control service capabilities using a hierarchical XML capability structure.* Tac:ServiceCapabilities **Capabilities [1][1]**

5.2 Access Point Information

5.2.1 Data Structures

5.2.1.1 AccessPointInfo

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

Multiple AccessPoints may cover the same Door. A typical case is one AccessPoint for entry and another for exit, both referencing the same Door.

A device shall provide the following fields for each AccessPoint instance:

- **token**
  A service-unique identifier of the AccessPoint.

- **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 EntityType field explained below but is typically a Door.

- **Capabilities**
  The capabilities for the AccessPoint.

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

- **Description**
  Optional user readable description for the AccessPoint. 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 the ONVIF DoorControl service 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 AccessPoint being extended to cover entity types other than Doors as well.

5.2.1.2 AccessPointCapabilities

The AccessPoint capabilities reflect optional functionality of a particular physical entity. Different AccessPoint 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 AccessPoint instance supports EnableAccessPoint and DisableAccessPointcommands.

- **Duress**
  Indicates whether or not this AccessPoint instance supports generation of duress events.

- **AnonymousAccess**
  Indicates whether or not this AccessPoint has a REX switch or other input that allows anonymous access.

- **AccessTaken**
  Indicates whether or not this AccessPoint instance supports generation of AccessTaken and AccessNotTaken events.
  If AnonymousAccessandAccessTaken are both true, it indicates that the Anonymous versions of AccessTaken and AccessNotTaken are supported.

- **ExternalAuthorization**
  Indicates whether or not this AccessPoint 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.

5.2.2 GetAccessPointInfoList command

This operation requests a list of all of AccessPointInfo items provided by the device. A device shall support this command.

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 section 4.8.3 for more details.

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

Table 2: GetAccessPointInfoList command

GetAccessPointInfoList	Access Class: READ_SYSTEM
Message name	**Description**
GetAccessPointInfoListRequest	*This message contains:* • *"Limit": 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": Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.* Xs:int **Limit [0][1]** xs:string**StartReference [0][1]**
GetAccessPointInfoListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"AccessPointInfo": List of AccessPointInfo items.* Xs:string**NextStartReference [0][1]** tac:AccessPointInfo**AccessPointInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:InvalidStartReference	*StartReference is invalid or has timed out. Client needs to start fetching from the beginning.*

5.2.3 GetAccessPointInfo command

This operation requests a list of AccessPointInfo items matching the given tokens.A device shall support this command.

The device shall ignore tokens it cannot resolve and shall return an empty list if there are no items matching 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.

Table 3: GetAccessPointInfo command

GetAccessPointInfo	Access Class: READ_SYSTEM
Message name	**Description**
GetAccessPointInfoRequest	*This message contains:* • *"Token": Tokens of AccessPointInfo items to get.* Pt:ReferenceToken**Token [1][unbounded]**
GetAccessPointInfoResponse	*This message contains:* • *"AccessPointInfo": List of AccessPointInfo items.* Tac:AccessPointInfo**AccessPointInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgs ter:TooManyItems	*Too many items were requested, see MaxLimit capability.*

5.3 Area Information

5.3.1 Data Structures

5.3.1.1 AreaInfo

The AreaInfo structure contains basic information about an Area. A device shall provide the following fields for each Area:

- **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.

5.3.2 GetAreaInfoList command

This operation requests a list of all AreaInfo items provided by the device. A device shall support this command.

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 section 4.8.3 for more details.

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

Table 4: GetAreaInfoList command

GetAreaInfoList	Access Class: READ_SYSTEM
Message name	**Description**
GetAreaInfoListRequest	*This message contains:*

	• *"Limit": 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.by the device.* • *"StartReference": Start returning entries from this start reference. If not specified, entries shall start from the beginning of the dataset.* Xs:int **Limit [0][1]** xs:string**StartReference [0][1]**
GetAreaInfoListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"AreaInfo": List of AreaInfo items.* Xs:string**NextStartReference [0][1]** tac:AreaInfo**AreaInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:InvalidStartReference	*StartReference is invalid or has timed out.Client needs to start fetching from the beginning.*

5.3.3 GetAreaInfo command

This operation requests a list of AreaInfo items matching the given tokens. A device shall support this command.

The device shall ignore tokens it cannot resolve and may return an empty list if there are no items matching specified tokens.

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

Table 5: GetAreaInfo command

GetAreaInfo	Access Class: READ_SYSTEM
Message name	**Description**
GetAreaInfoRequest	*This message contains:* • *"Token": Tokens of AreaInfo items to get.* Pt:ReferenceToken**Token [1][unbounded]**
GetAreaInfoResponse	*This message contains:* • *"AreaInfo": List of AreaInfo items.* Tac:AreaInfo**AreaInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgs ter:TooManyItems	*Too many items were requested, see MaxLimit capability.*

5.4 Access Point Status

5.4.1 General

The state of the AccessPoint 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 AccessPoint. A device shall provide the following fields for each AccessPoint instance:

- **Enabled**
  Indicates that the AccessPoint is enabled. By default this field value shall be True, if the DisableAccessPoint capabilities is not supported.

5.4.3 GetAccessPointState command

This operation requests the AccessPointState for the AccessPoint instance specified by Token.A device shall support this command.

Table 6: GetAccessPointState command

GetAccessPointState	Access Class: READ_SYSTEM_SENSITIVE
Message name	**Description**
GetAccessPointStateRequest	*This message contains:* • *"Token": Token of AccessPoint instance to get AccessPointState for.* Pt:ReferenceToken **Token [1][1]**
GetAccessPointStateResponse	*This message contains:* • *"AccessPointState": AccessPointState item.* Tac:AccessPointState**AccessPointState [1][1]**
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*AccessPoint is not found*

5.5 Access control commands

5.5.1 General

The service control commands contain operations that allow modifying AccessPoint states and controlling AccessPoints.

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.

5.5.3 EnableAccessPoint command

This operation allows enabling an access point.

A device that signals support for EnableAccessPoint capability for a particular AccessPoint instance shall support this command.

Table 7: EnableAccessPoint command

EnableAccessPoint	Access Class: ACTUATE
Message name	**Description**
EnableAccessPointRequest	*This message contains:* • *"Token": Token of the AccessPoint instance to enable.* Pt:ReferenceToken **Token [1][1]**
EnableAccessPointResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
Env:Receiver ter:ActionNotSupported ter:NotSupported	*The operation is not supported.*

5.5.4 DisableAccessPoint command

This operation allows disabling an access point.

A device that signals support for DisableAccessPoint capability for a particular AccessPoint instance shall support this command.

Table 8: DisableAccessPoint command

DisableAccessPoint	Access Class: ACTUATE
Message name	**Description**
DisableAccessPointRequest	*This message contains:* • *"Token": Token of the AccessPoint instance to disable.* Pt:ReferenceToken **Token [1][1]**
DisableAccessPointResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
Env:Receiver ter:ActionNotSupported ter:NotSupported	*The operation is not supported.*

5.5.5 ExternalAuthorization command

This operation allows to Deny or Grant decision at an AccessPoint instance.

A device that signals support for ExternalAuthorization capability for a particular AccessPoint instance shall support this method.

Table 9: ExternalAuthorization command

ExternalAuthorization	Access Class: ACTUATE
Message name	**Description**
ExternalAuthorizationRequest	*This message contains:* • *"AccessPointToken": Token of the Access Point instance.* • *"CredentialToken": Optional token of the Credential involved.* • *"Reason": Optional reason for decision.* • *"Decision": Decision - Granted or Denied.* Pt:ReferenceToken**AccessPointToken [1][1]** pt:ReferenceToken**CredentialToken [0][1]** xs:string **Reason [0][1]** tac:Decision **Decision [1][1]** *(extendable)*
ExternalAuthorizationResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*AccessPoint is not found.*
Env:Receiver ter:ActionNotSupported ter:NotSupported	*The operation is not supported.*

6 Notification topics

6.1 Event overview

The AccessControl Service specifies events to be used for access transactions, e.g 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 AccessPointconfiguration has been changed.
- tns1:Configuration/Area - when Area configuration has been changed.

The term “main topic” here means that a client may subscribe to e.g.Tns1:AccessControl/AccessGranted, 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.

6.2 General transaction event layout

All transaction events use the following message scheme:

```
<tt:MessageDescriptionIsProperty="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="Reason"
                              Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

where AccessPointToken identifies AccessPoint where the decision was made.

The Data parameters are optional. See the definitions in the following sections for their requirements. The parameter CredentialTokenshall be present and CredentialHolderNamemay be present for credential based transactions. The optional

parameter Reason holds error reason information and the optional parameter External signals whether the transaction was triggered as a result of the ExternalAuthorization command.

6.3 Access granted

6.3.1 General

Whenever a positive access decision is made (i.e., access is granted), a 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 AnonymousAccess capability for particular AccessPoint instance shall provide the following event:

```
Topic: tns1:AccessControl/AccessGranted/Anonymous

<tt:MessageDescriptionIsProperty="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 user or card holder is granted access to the AccessPoint, but not yet accessed it (entered or exited), the device shall provide the following event data:

```
Topic: tns1:AccessControl/AccessGranted/Credential

<tt:MessageDescriptionIsProperty="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: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 is optional.

6.4 Access taken

6.4.1 General

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

6.4.2 Anonymous

The device that signals support for AnonymousAccesscapability for a particular AccessPoint instance shall provide the following event:

```
Topic: tns1:AccessControl/AccessTaken/Anonymous

<tt:MessageDescriptionIsProperty="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, it shall provide the following event:

```
Topic: tns1:AccessControl/AccessTaken/Credential

<tt:MessageDescriptionIsProperty="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 Access not taken

6.5.1 General

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

6.5.2 Anonymous

The device that signals support for AnonymousAccesscapability for a particular AccessPoint instance shall provide the following event:

```
Topic: tns1:AccessControl/AccessNotTaken/Anonymous

<tt:MessageDescriptionIsProperty="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, it shall provide the following event:

```
Topic: tns1:AccessControl/AccessNotTaken/Credential

<tt:MessageDescriptionIsProperty="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.

6.6 Access denied

6.6.1 General

An ONVIF compliant device shallprovide one of the eventsas per the following sub-sections whenever a person is denied to access. The following applies to all subsections:

The denial reason shall be present in the parameter Reason. If there are multiple reasons for denial, the device shall send only one notification. The following strings shall be used for the reason field:

- **CredentialNotEnabled**
  The device shall provide the following event, 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 the following event, 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 the following event, whenever a valid credential was presented after its expiry date.

- **InvalidPIN**
  The device shall provide the following event, whenever an entered PIN code does not match the credential.

- **NotPermittedAtThisTime**
  The device shall provide the following eventwhenever a valid credential is denied access to the requested AccessPoint because the credential is not permitted at the moment.

- **Unauthorized**
  The device shall provide the following event, whenever the presented credential is not authorized.

- **Other**
  The device shall provide the following event, whenever the request is denied and no other specific event matches it or is supported by the service.

6.6.2 Anonymous

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

```
Topic: tns1:AccessControl/Denied/Anonymous

<tt:MessageDescriptionIsProperty="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, it shall provide the following event:

```
Topic: tns1:AccessControl/Denied/Credential

<tt:MessageDescriptionIsProperty="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="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 is optional.

6.6.4 CredentialNotFound

Under some circumstances a device may be not able to resolve authentication data to a credential token. Whenever this happens the device should provide a corresponding event message as per the following sub-sections.

Depending on which authentication factors were used message payload may differ. Currently only a single subset is defined.

6.6.4.1 Card

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:MessageDescriptionIsProperty="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

The device that signals support for Duress capability for a particular AccessPoint instance shall provide the following event whenever a condition of Duress is detected.

```
Topic: tns1:AccessControl/Duress

<tt:MessageDescriptionIsProperty="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

The device that signals support for ExternalAuthorization capability for a particular AccessPoint instance shall provide 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 device.

6.8.2 Anonymous

Whenever a device that signals AnonymousAccess capability for particular AccessPoint 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:MessageDescriptionIsProperty="false">
  <tt:Source>
    <tt:SimpleItemDescription Name="AccessPointToken"
                              Type="pt:ReferenceToken"/>
  </tt:Source>
</tt:MessageDescription>
```

6.8.3 Credential

Whenever the device requests external agent to authorize a person, the device shall send the following event:

```
Topic: tns1:AccessControl/Request/Credential

<tt:MessageDescriptionIsProperty="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 can not be resolved.

6.8.4 Timeout

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

```
Topic: tns1:AccessControl/Request/Timeout

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

6.8.5 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 the PACS entity status is changed. A 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 DisableAccessPoint capability for 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:MessageDescriptionIsProperty="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>
```

6.10 Configuration changes

6.10.1 General

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

6.10.2 Access Point

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

```
Topic: tns1:Configuration/AccessPoint/Changed

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

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

```
Topic: tns1:Configuration/AccessPoint/Removed

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

Annex A. Revision History

Rev.	Date	Editor	Changes
1.0	Apr-2013	Yuri Timenkov	Initial version
1.01	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