ONVIF™
Door Control Service Specification

Version 18.12
December 2018

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 doors. This includes but is not limited to controlling them and monitoring their state.

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

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

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>

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.

Barrier	A barrier controls access for vehicles. Examples of barriers are booms, gates, rising curbs, bollards, etc. Barriers are used at parking places, gates of properties or military sites.

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 Number	A sequence of bytes uniquely identifying a credential at an access point.
Delay Time Before Relock	A configuration parameter for a door. The time from when the door is physically opened until the latch goes back to locked state.
Door	A physical door, barrier, turnstile, etc. which restricts access between two areas. A door is usually equipped with an electronic lock and a door monitor.
Door Alarm	An abnormal state of the door where door is forced open or held open beyond the permitted time duration
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 Mode	Logical state of the door indicating whether the door is locked, unlocked, blocked, locked down or locked open 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 electromagnetic lock or armature plate. Also known as door contact sensor.
Door Type	The type of door, e.g. mantrap, turnstile, revolving door, barrier or regular door.
Extended Open Time	A configuration parameter for a door. Time added to Open Time to allow for individuals in need of longer time to pass through the door.
Extended Release Time	A configuration parameter for a door. Time added to Release Time to allow for individuals in need of longer time to access the door.
Lock	An operation after which a door is locked and alarm is unmasked.
Mantrap	A mantrap is a combination of two interlocked doors and a small area between these doors. The mantrap has a restriction that the first door must be closed before the second door can be opened.
Momentary Access	An operation which invokes the same logic as upon normal access being granted to a credential.
Open Time	A configuration parameter for a door. The time from when the door is physically opened until the door starts warning about it being opened too long.
Pre-Alarm Time	A configuration parameter for a door. The time from when the door starts warning about it being opened too long until an alarm state is generated.
Release Time	A configuration parameter for a door. The time from when the latch is unlocked until it is relocked again (unless the door is physically opened).
Revolving Door	A revolving door is similar to a turnstile, but has a different form factor. Revolving doors may be used to prevent tailgating.
Tailgating	Person or entity, passing through an access point without using credentials by following a person or entity for whom access has been granted.
Tamper Detector	Mechanism commonly available for doors, access points and controllers to detect physical tamper

Turnstile	A turnstile is a special kind of door that allows only one person to pass at a time and prevents tailgating. Turnstiles can be setup to enforce one-way traffic of people.
Unlock	An operation to allow a door to be freely used for passage without any door alarms being triggered.

3.2 Abbreviated terms

PACS	Physical Access Control System

4 Overview

The door control service provides mechanisms for controlling physical door instances and monitoring their status.

The term Door in this specification can refer to such physical objects as an automatic barrier or a door equipped with electric lock. Turnstiles which can restrict access in either direction can be represented with a pair of doors.

Please refer to the [ONVIF PACS Architecture and Design Considerations] for generic operation guidelines and design principles behind ONVIF PACS services family.

The service includes the following operations:

- Getting list of doors including their capabilities (e.g., supported operations).
- Getting actual state (e.g., open or closed, locked or unlocked, health status).
- Locking and unlocking.
- Blocking door in locked state such that it can’t be accessed.
- Holding door in either unlocked (locked open) or locked (locked down) state and releasing the hold.
- Momentary access.
- Double lock (also known as secure lock) for preventing night-time access.

The service also defines a number of events for real-time monitoring:

- Door physical status change (e.g., open or closed).
- Lock physical state change (e.g., locked or unlocked).
- Operation mode change (e.g., blocked, locked down or locked open).
- Alarm (if door was forced open or was open for too long during momentary access).
- Tamper (an attempt to physically damage its components).
- Hardware malfunction.

5 Door control

5.1 General

This service offers commands to retrieve status information and to control door instances of a device.

5.2 Service capabilities

5.2.1 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.

5.2.2 Data structures

5.2.2.1 ServiceCapabilities

ServiceCapabilities structure reflects 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.

- **MaxDoors**

  Indicates the maximum number of doors supported by the device.

- **ClientSuppliedTokenSupported**

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

5.2.3 GetServiceCapabilities command

This operation returns the capabilities of the door control service. A device shall support this command.

Table 2: 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 DoorControl service capabilities using a hierarchical XML capability structure.* tdc:ServiceCapabilities **Capabilities [1][1]** *(extendable)*

5.3 Door information

5.3.1 General

The following figure shows an overview of the related objects of a door:

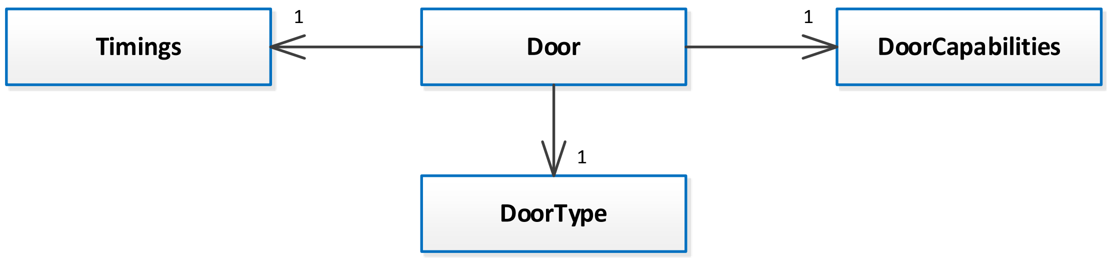

Figure 1: The related objects of a door

5.3.2 Data structures

5.3.2.1 DoorInfo

The DoorInfo type represents a door as a physical object. The structure contains information and capabilities of a specific door instance. An ONVIF compliant device shall provide the following fields for each Door instance:

- **token**

  A service-unique identifier of the door.

- **Name**

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

- **Capabilities**

  The capabilities of the door; is of type DoorCapabilities.

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

- **Description**

  A user readable description. It shall be up to 1024 characters.

5.3.2.2 Door

The door structure shall include all properties of the DoorInfo structure and also a timings structure.

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

- **DoorType**

  The type of door. The type is used by the device to understand the operational mode of the door. Door types starting with the prefix pt: are reserved to define ONVIF-specific types. For custom defined door types, free text can be used. The following types are defined by ONVIF:

pt:Door	A regular door
pt:Mantrap	See Terms and Definitions
pt:Turnstile	See Terms and Definitions
pt:RevolvingDoor	See Terms and Definitions

  - o pt:Barrier See Terms and Definitions

- **Timings**

  A structure defining times such as how long the door is unlocked when accessed, extended grant time, etc.

5.3.2.3 Timings

A structure defining times such as how long the door is unlocked when accessed, extended grant time, etc.

The following diagram shows how the timings are used:

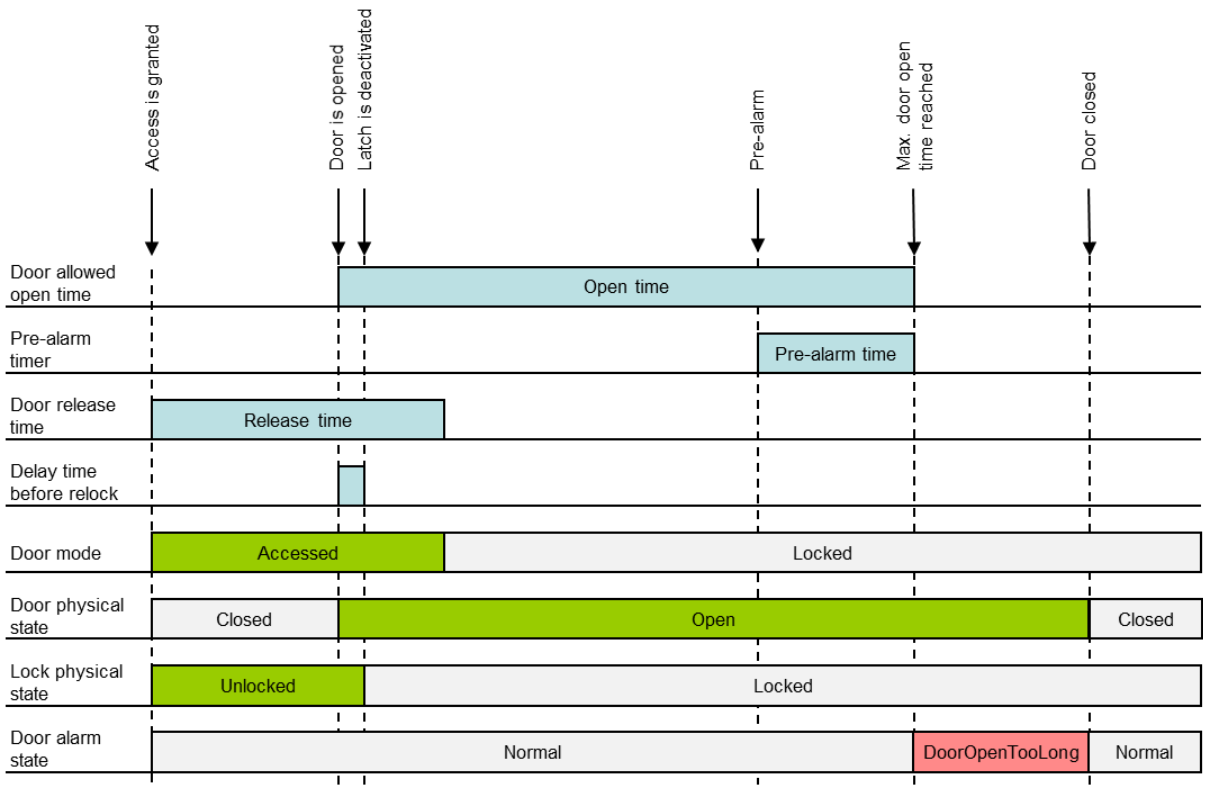

Figure 2: Door timings

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

- **ReleaseTime**

  When access is granted (door mode becomes Accessed, see section 5.4.2.9), the latch is unlocked. ReleaseTime is the time from when the latch is unlocked until it is relocked again (unless the door is physically opened).

- **OpenTime**

  The time from when the door is physically opened until the door is set in the DoorOpenTooLong alarm state (see section 5.4.2.4).

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

- **ExtendedReleaseTime**

  Some individuals need extra time to open the door before the latch relocks. If supported, ExtendedReleaseTime shall be added to ReleaseTime if UseExtendedTime is set to true in the AccessDoor command (see section 5.5.2).

- **DelayTimeBeforeRelock**

  If the door is physically opened after access is granted, then DelayTimeBeforeRelock is the time from when the door is physically opened until the latch goes back to locked state.

- **ExtendedOpenTime**

  Some individuals need extra time to pass through the door. If supported, ExtendedOpenTime shall be added to OpenTime if UseExtendedTime is set to true in the AccessDoor command (see section 5.5.2).

- **PreAlarmTime**

  Before a DoorOpenTooLong alarm state is generated, a signal will sound to indicate that the door must be closed. PreAlarmTime defines how long before DoorOpenTooLong the warning signal shall sound.

5.3.2.4 DoorCapabilities

DoorCapabilities reflect optional functionality of a particular physical entity. Different door 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:

- **Access**

  Indicates whether or not this Door instance supports AccessDoor command to perform momentary access.

- **AccessTimingOverride**

  Indicates that this Door instance supports overriding configured timing in the AccessDoor command.

- **Lock**

  Indicates that this Door instance supports LockDoor command to lock the door.

- **Unlock**

  Indicates that this Door instance supports UnlockDoor command to unlock the door.

- **Block**

  Indicates that this Door instance supports BlockDoor command to block the door.

- **DoubleLock**

  Indicates that this Door instance supports DoubleLockDoor command to lock multiple locks on the door.

- **LockDown**

  Indicates that this Door instance supports LockDown (and LockDownRelease) commands to lock the door and put it in LockedDown mode.

- **LockOpen**

  Indicates that this Door instance supports LockOpen (and LockOpenRelease) commands to unlock the door and put it in LockedOpen mode.

- **DoorMonitor**

  Indicates that this Door instance has a DoorMonitor and supports the DoorPhysicalState event.

- **LockMonitor**

  Indicates that this Door instance has a LockMonitor and supports the LockPhysicalState event.

- **DoubleLockMonitor**

  Indicates that this Door instance has a DoubleLockMonitor and supports the DoubleLockPhysicalState event.

- **Alarm**

  Indicates that this Door instance supports door alarm and the DoorAlarm event.

- **Tamper**

  Indicates that this Door instance has a Tamper detector and supports the DoorTamper event.

- **Fault**

  Indicates that this Door instance supports door fault and the DoorFault event.

5.3.3 GetDoorInfo command

This operation requests a list of DoorInfo 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.

Table 3: GetDoorInfo command

GetDoorInfo	Access Class: READ_SYSTEM
Message name	**Description**
GetDoorInfoRequest	*This message contains:* • *"Token": Tokens of DoorInfo items to get.* pt:ReferenceToken **Token [1][unbounded]**
GetDoorInfoResponse	*This message contains:* • *"DoorInfo": List of DoorInfo items.* tdc:DoorInfo **DoorInfo [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgs ter:TooManyItems	*Too many items were requested, see MaxLimit capability.*

5.3.4 GetDoorInfoList command

This operation requests a list of all DoorInfo 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 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.

Table 4: GetDoorInfoList command

GetDoorInfoList	Access Class: READ_SYSTEM
Message name	**Description**
GetDoorInfoListRequest	*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]**
GetDoorInfoListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"DoorInfo": List of DoorInfo items.* xs:string **NextStartReference [0][1]** tdc:DoorInfo **DoorInfo [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*

5.3.5 GetDoors command

This operation requests a list of Door 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 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 5 GetDoors command

GetDoors	Access Class: READ_SYSTEM
Message name	**Description**
GetDoorsRequest	*This message contains:* • *"Token": Tokens of Door items to get* pt:ReferenceToken **Token [1][unbounded]**
GetDoorsResponse	*This message contains:* • *"Door": List of Door items.* tdc:Door **Door [0][unbounded]**
Fault codes	**Description**
env:Sender ter:InvalidArgs ter:TooManyItems	*Too many items were requested, see MaxLimit capability.*

5.3.6 GetDoorList command

This operation requests a list of all Door 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.

Table 6 GetDoorList command

GetDoorList	Access Class: READ_SYSTEM
Message name	**Description**
GetDoorListRequest	*This message contains:* • *"Limit": Maximum number of entries to return. If not specified, less than one or higher than what the device supports, the number of items is determined by the device.* • *"StartReference": 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]**
GetDoorListResponse	*This message contains:* • *"NextStartReference": StartReference to use in next call to get the following items. If absent, no more items to get.* • *"Door": List of Door items.* xs:string **NextStartReference [0][1]** tdc:Door **Door [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.*

5.3.7 CreateDoor command

This operation creates the specified door in the device.

The token field of the Door structure shall be empty and the device shall allocate a token for the door. 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.

Table 7 CreateDoor command

CreateDoor	Access Class: WRITE_SYSTEM
Message name	**Description**
CreateDoorRequest	*This message contains:* • *"Door": The Door item to create* tdc:Door **Door [1][1]**
CreateDoorResponse	*This message contains:* • *"Token": The token of the created Door item* pt:ReferenceToken **Token [1][1]**
Fault codes	**Description**
env:Receiver ter:CapabilityViolated ter:MaxDoors	*There is not enough space to add the new door, see the MaxDoors capability*
env:Sender ter:InvalidArgVal ter:ReferenceNotFound	*A referred entity token is not found (some devices may not validate referred entities).*

5.3.8 SetDoor command

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

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

A call to this method takes a door structure as input parameter. The token field of the Door 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.

Table 8 SetDoor command

SetDoor	Access Class: WRITE_SYSTEM
Message name	**Description**
SetDoorRequest	*This message contains:* • *"Door": The Door item to create or modify* tdc:Door **Door [1][1]**
SetDoorResponse	*This message shall be empty*
Fault codes	**Description**
env:Receiver ter:CapabilityViolated ter:ClientSuppliedTokenSupported	*The device does not support that the client supplies the token*
env:Receiver ter:CapabilityViolated ter:MaxDoors	*There is not enough space to add new door, see the MaxDoors capability*
env:Sender ter:InvalidArgVal ter:ReferenceNotFound	*A referred entity token is not found (some devices may not validate referred entities).*

5.3.9 ModifyDoor command

This operation modifies the specified door.

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

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

Table 9 ModifyDoor command

ModifyDoor	Access Class: WRITE_SYSTEM
Message name	**Description**
ModifyDoorRequest	*This message contains:* • *“Door”: The details of the door* tdc:Door **Door [1][1]**
ModifyDoorResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*Door token is not found.*
env:Sender ter:InvalidArgVal ter:ReferenceNotFound	*A referred entity token is not found (some devices may not validate referred entities).*

5.3.10 DeleteDoor command

This operation deletes the specified door.

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

Table 10 DeleteDoor command

DeleteDoor	Access Class: WRITE_SYSTEM
Message name	**Description**
DeleteDoorRequest	*This message contains:* • *"Token": The token of the door to delete.* pt:ReferenceToken **Token [1][1]**
DeleteDoorResponse	*This message shall be empty*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*Door token is not found.*
env:Sender ter:InvalidArgVal ter:ReferenceInUse	*Failed to delete, the door token is in use*

5.4 Door status

5.4.1 General

The state of the door may be affected by a number of operations that can be performed on it depending on its capabilities: LockDoor, UnlockDoor, AccessDoor, BlockDoor, DoubleLockDoor, LockDownDoor, LockDownReleaseDoor, LockOpenDoor and LockOpenReleaseDoor.

5.4.2 Data Structures

5.4.2.1 DoorState

The DoorState structure contains current aggregate runtime status of a door.

The device shall provide the following fields:

- **DoorMode**

  The logical operating mode of the door; it is of type DoorMode. An ONVIF compatible device shall report current operating mode in this field.

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

- **DoorPhysicalState**

  Physical state of the Door; it is of type DoorPhysicalState. A device that signals support for DoorMonitor capability for a particular door instance shall provide this field.

- **LockPhysicalState**

  Physical state of the Lock; it is of type LockPhysicalState. A device that signals support for LockMonitor capability for a particular door instance shall provide this field.

- **DoubleLockPhysicalState**

  Physical state of the DoubleLock; it is of type LockPhysicalState. A device that signals support for DoubleLockMonitor capability for a particular door instance shall provide this field.

- **Alarm**

  Alarm state of the door; it is of type DoorAlarmState. A device that signals support for Alarm capability for a particular door instance shall provide this field.

- **Tamper**

  Tampering state of the door; it is of type DoorTamper. A device that signals support for Tamper capability for a particular door instance shall provide this field.

- **Fault**

  Fault information for door; it is of type DoorFault. A device that signals support for Fault capability for a particular door instance shall provide this field.

The following data types define states of DoorState elements.

5.4.2.2 Enumeration: DoorPhysicalState

The physical state of a door. The following values are available:

- **Unknown**

  Value is currently unknown (possibly due to initialization or monitors not giving a conclusive result).

- **Open**

  Door is open.

- **Closed**

  Door is closed.

- **Fault**

  Door monitor fault is detected.

5.4.2.3 Enumeration: LockPhysicalState

The physical state of a Lock (including Double Lock). The following values are available:

- **Unknown**

  Value is currently not known.

- **Locked**

  Lock is activated.

- **Unlocked**

  Lock is not activated.

- **Fault**

  Lock fault is detected.

5.4.2.4 Enumeration: DoorAlarmState

Describes the state of a door with regard to alarms. The following values are available:

- **Normal**

  No alarm.

- **DoorForcedOpen**

  Door is forced open.

- **DoorOpenTooLong**

  Door is held open too long.

5.4.2.5 DoorTamper

Tampering information for a door. The following fields are available:

- **Reason**

  Optional field; Details describing tampering state change (e.g., reason, place and time).

NOTE: All fields (including this one) which are designed to give end-user prompts can be localized to the customer’s native language.

▪ **State**

State of the tamper detector; it is of type DoorTamperState.

5.4.2.6 Enumeration: DoorTamperState

Describes the state of a Tamper detector. The following values are available:

▪ **Unknown**

Value is currently not known.

▪ **NotInTamper**

No tampering is detected.

▪ **TamperDetected**

Tampering is detected.

5.4.2.7 DoorFault

Fault information for a door. This can be extended with optional attributes in the future. The following fields are available:

- **Reason**

  Optional reason for fault.

- **State**

  Overall fault state for the door; it is of type DoorFaultState. If there are any faults, the value shall be: FaultDetected. Details of the detected fault shall be found in the Reason field, and/or the various DoorState fields and/or in extensions to this structure.

5.4.2.8 Enumeration: DoorFaultState

Describes the state of a door fault. The following values are available:

- **Unknown**

  Fault state is unknown.

- **NotInFault**

  No fault is detected.

- **FaultDetected**

  Fault is detected.

5.4.2.9 Enumeration: DoorMode

The DoorMode describe the mode of operation from a logical perspective. Setting a door mode reflects the intent to set a door in a physical state.

The following values are available:

- **Unknown**

  The mode of operation is unknown.

- **Locked**

  The intention is to set the door to a physical locked state. In this mode the device shall provide momentary access using the AccessDoor method if supported by the door instance.

- **Unlocked**

  The intention is to set the door to a physical unlocked state. Alarms related to door timing operations such as open too long or forced open are masked in this mode.

- **Accessed**

  The intention is to momentary set the door to a physical unlocked state. After a predefined time the device shall revert the door to its previous mode. Alarms related to timing operations such as door forced open are masked in this mode.

- **Blocked**

  The intention is to set the door to a physical locked state and the device shall not allow AccessDoor requests, i.e. it is not possible for the door to go to the accessed mode. All other requests to change the door mode are allowed.

- **LockedDown**

  The intention is to set the door to a physical locked state and the device shall only allow the LockDownReleaseDoor request. All other requests to change the door mode are not allowed.

- **LockedOpen**

  The intention is to set the door to a physical unlocked state and the device shall only allow the LockOpenReleaseDoor request. All other requests to change the door mode are not allowed.

- **DoubleLocked**

  The intention is to set the door with multiple locks to a physical double locked state. If the door does not support double locking the devices shall treat this as a normal locked mode. When changing to an unlocked mode from the double locked mode, the physical state of the door may first go to locked state before unlocking.

5.4.3 GetDoorState command

This operation requests the state of a door specified by the Token.

A device implementing the door control service shall be capable of reporting the status of a door using a DoorState structure available from the GetDoorState command.

Table 11 GetDoorState command

GetDoorState	Access Class: READ_SYSTEM_SENSITIVE
Message name	**Description**
GetDoorStateRequest	*This message contains:* • *"Token": Token of the Door instance to get the state for.* pt:ReferenceToken **Token [1][1]** *(extendable)*
GetDoorStateResponse	*This message contains:* • *"DoorState": The state of the door.* tdc:DoorState **DoorState [1][1]** *(extendable)*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*

5.5 Door control commands

5.5.1 General

The service control commands contain operations that allow modifying Door instances states and controlling Door instances of a device.

5.5.2 AccessDoor command

This operation allows momentarily accessing a door. It invokes the functionality typically used when a card holder presents a card to a card reader at the door and is granted access.

The DoorMode shall change to Accessed state. Please refer to Accessed mode in section 5.4.2.9 for more details.

The door shall remain accessible for the defined time. When the time span elapses, the DoorMode shall change back to its previous state.

If the request cannot be fulfilled, a Failure fault shall be returned.

Please refer to section 5.4.2.9 for details about door mode restrictions.

A device that signals support for Access capability for a particular Door instance shall support this method. A device that signals support for AccessTimingOverride capability for a particular Door instance shall also provide the following optional timing parameters when performing AccessDoor command:

• AccessTime	Overrides ReleaseTime in section 5.3.2.3
• OpenTooLongTime	Overrides OpenTime in section 5.3.2.3
• PreAlarmTime	Overrides PreAlarmTime in section 5.3.2.3

The device shall take the best effort approach for parameters not supported, it must fallback to preconfigured time or limit the time to the closest supported time if the specified time is out of range.

Table 12 AccessDoor command

AccessDoor	Access Class: ACTUATE
Message name	**Description**
AccessDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* • *"UseExtendedTime": Optional - Indicates that the configured extended time should be used.* • *"AccessTime": Optional - overrides ReleaseTime if specified.* • *"OpenTooLongTime": Optional - overrides OpenTime if specified.* • *"PreAlarmTime": Optional - overrides PreAlarmTime if specified.* • *"Extension": Future extension.* pt:ReferenceToken **Token [1][1]** xs:boolean **UseExtendedTime [0][1]** xs:duration **AccessTime [0][1]** xs:duration **OpenTooLongTime [0][1]** xs:duration **PreAlarmTime [0][1]** tdc:AccessDoorExtension **Extension [0][1]**
AccessDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to Accessed state and unlock the door.*

5.5.3 LockDoor command

This operation allows locking a door. The door mode shall change to Locked state. Please refer to Locked mode in section 5.4.2.9 for more details.

A device that signals support for Lock capability for a particular Door instance shall support this command.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.4.2.9 for more details about door mode restrictions.

Table 13 LockDoor command

LockDoor	Access Class: ACTUATE
Message name	**Description**
LockDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]**
LockDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to Locked state.*

5.5.4 UnlockDoor command

This operation allows unlocking a door. The door mode shall change to Unlocked state. Please refer to Unlocked mode in section 5.4.2.9 for more details.

A device that signals support for Unlock capability for a particular Door instance support this command.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.4.2.9 for more details about door mode restrictions.

Table 14 UnlockDoor command

UnlockDoor	Access Class: ACTUATE
Message name	**Description**
UnlockDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]**
UnlockDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to Unlocked state.*

5.5.5 BlockDoor command

This operation allows blocking a door and preventing momentary access (AccessDoor command). The door mode shall change to Blocked state. Please refer to Blocked mode in section 5.4.2.9 for more details.

A device that signals support for Block capability for a particular Door instance shall support this command.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.4.2.9 for more details about Door Modes restrictions.

Table 15 BlockDoor command

BlockDoor	Access Class: ACTUATE
Message name	**Description**
BlockDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]**
BlockDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to Blocked state.*

5.5.6 LockDownDoor command

This operation allows locking and preventing other actions until a LockDownRelease command is invoked. The DoorMode shall change to LockedDown state. Please refer to LockedDown mode in section 5.4.2.9 for more details.

The device shall ignore other door control commands until a LockDownRelease command is performed.

A device that signals support for LockDown capability for a particular Door instance shall support this command.

If a device supports DoubleLock capability for a particular Door instance, that operation may be engaged as well.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.4.2.9 for more details about door mode restrictions.

Table 16 LockDownDoor command

LockDownDoor	Access Class: ACTUATE
Message name	**Description**
LockDownDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]**
LockDownDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to a LockedDown state.*

5.5.7 LockDownReleaseDoor command

This operation allows releasing the LockedDown state of a door. The door mode shall change back to its previous/next state. It is not defined what the previous/next state shall be, but typically the Locked state. A device that signals support for LockDown capability for a particular Door instance shall support this command.

This method shall only succeed if the current door mode is LockedDown.

Table 17 LockDownReleaseDoor command

LockDownReleaseDoor	Access Class: ACTUATE
Message name	**Description**
LockDownReleaseDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]**
LockDownReleaseDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to leave LockedDown state.*

5.5.8 LockOpenDoor command

This operation allows unlocking a door and preventing other actions until LockOpenRelease method is invoked. The door mode shall change to LockedOpen state. Please refer to LockedOpen mode in section 5.4.2.9 for more details.

The device shall ignore other door control commands until a LockOpenRelease command is performed.

A device that signals support for LockOpen capability for a particular Door instance shall support this command.

If the request cannot be fulfilled, a Failure fault shall be returned. Please refer to section 5.4.2.9 for more details about Door Modes restrictions.

Table 18 LockOpenDoor command

LockOpenDoor	Access Class: ACTUATE
Message name	**Description**
LockOpenDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]**
LockOpenDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to LockedOpen state.*

5.5.9 LockOpenReleaseDoor command

This operation allows releasing the LockedOpen state of a door. The door mode shall change state from the LockedOpen state back to its previous/next state. It is not defined what the previous/next state shall be, but typically the Unlocked state. A device that signals support for LockOpen capability for a particular Door instance shall support this command.

This method shall only succeed if the current DoorMode is LockedOpen.

Table 19 LockOpenReleaseDoor command

LockOpenReleaseDoor	Access Class: ACTUATE
Message name	**Description**
LockOpenReleaseDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]**
LockOpenReleaseDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to leave LockedOpen state.*

5.5.10 DoubleLockDoor command

This operation is used for securely locking a door. A call to this method shall change door mode state to DoubleLocked. Please refer to DoubleLocked mode in section 5.4.2.9 for more details.

A device that signals support for DoubleLock capability for a particular Door instance shall support this command. Otherwise this method can be performed as a standard Lock operation (see section 5.5.3 LockDoor command).

If the door has an extra lock that shall be locked as well.

If the request cannot be fulfilled, a Failure fault shall be returned.

Table 20 DoubleLockDoor command

DoubleLockDoor	Access Class: ACTUATE
Message name	**Description**
DoubleLockDoorRequest	*This message contains:* • *"Token": Token of the Door instance to control.* pt:ReferenceToken **Token [1][1]**
DoubleLockDoorResponse	*This message is typically empty, but is extendable*
Fault codes	**Description**
env:Sender ter:InvalidArgVal ter:NotFound	*The specified token is not found.*
env:Receiver ter:Action ter:Failure	*Failed to go to DoubleLocked state.*

6 Notification Topics

6.1 General

This section defines notification topics specific to door control service.

Please refer to [ONVIF PACS Architecture and Design Considerations] for generic operation guidelines and design principles behind ONVIF PACS services family.

6.2 Status changes

Whenever a door mode is changed, the device shall provide the following event:

```
Topic: tns1:Door/State/DoorMode

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="DoorToken"
                              Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State"
                              Type="tdc:DoorMode"/>
  </tt:Data>
</tt:MessageDescription>
```

A device that signals support for DoorMonitor capability for a particular door instance shall provide the following event whenever the physical state of this door is changed:

```
Topic: tns1:Door/State/DoorPhysicalState

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="DoorToken"
                              Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State"
                              Type="tdc:DoorPhysicalState"/>
  </tt:Data>
</tt:MessageDescription>
```

A device that signals support for LockMonitor capability for a particular door instance shall provide the following event whenever the physical state of this door's lock is changed:

```
Topic: tns1:Door/State/LockPhysicalState

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="DoorToken"
                              Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State"
                              Type="tdc:LockPhysicalState"/>
  </tt:Data>
</tt:MessageDescription>
```

A device that signals support for DoubleLockMonitor capability for a particular door instance shall provide the following event whenever the physical state of this door's secure lock is changed:

```
Topic: tns1:Door/State/DoubleLockPhysicalState

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="DoorToken"
                              Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State"
                              Type="tdc:LockPhysicalState"/>
  </tt:Data>
</tt:MessageDescription>
```

A device that signals support for Alarm capability for a particular door instance shall provide the following event whenever the alarm state of this door is changed:

```
Topic: tns1:Door/State/DoorAlarm

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="DoorToken"
                              Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State"
                              Type="tdc:DoorAlarmState"/>
  </tt:Data>
</tt:MessageDescription>
```

A device that signals support for Tamper capability for a particular door instance shall provide the following event whenever the tamper state of this door is changed:

```
Topic: tns1:Door/State/DoorTamper

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="DoorToken"
                              Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State"
                              Type="tdc:DoorTamperState"/>
  </tt:Data>
</tt:MessageDescription>
```

A device that signals support for Fault capability for a particular door instance shall provide the following event whenever the fault state of this door is changed:

```
Topic: tns1:Door/State/DoorFault

<tt:MessageDescription IsProperty="true">
  <tt:Source>
    <tt:SimpleItemDescription Name="DoorToken"
                              Type="pt:ReferenceToken"/>
  </tt:Source>
  <tt:Data>
    <tt:SimpleItemDescription Name="State"
                              Type="tdc:DoorFaultState"/>
    <tt:SimpleItemDescription Name="Reason"
                              Type="xs:string"/>
  </tt:Data>
</tt:MessageDescription>
```

The Reason element may be empty or absent. The device may also skip it unless the fault state is FaultDetected.

6.3 Configuration changes

6.3.1 Door

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

```
Topic: tns1:Configuration/Door/Changed

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

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

```
Topic: tns1:Configuration/Door/Removed

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

Annex A. Revision History

Rev.	Date	Editor	Changes
1.0	Apr-2013	Yuri Timenkov	First version
1.0.1	Aug-2013	Hans Busch	Change Request 1053
1.0.2	Jun-2014	Michio Hirai	Change Request 1368, 1369
17.12	Dec-2017	Hiroyuki Sano	Change Request 2172, 2173
18.06	Jun-2018	Patrik Björling Rygert	Added support for client-supplied tokens Added full management support for doors
18.12	Dec-2018	Hiroyuki Sano	2397