

Chapter 5. Closures

The Cluster Library is made of individual chapters such as this one. References between chapters are made using a X.Y notation where X is the chapter and Y is the sub-section within that chapter.

5.1. General Description

5.1.1. Introduction

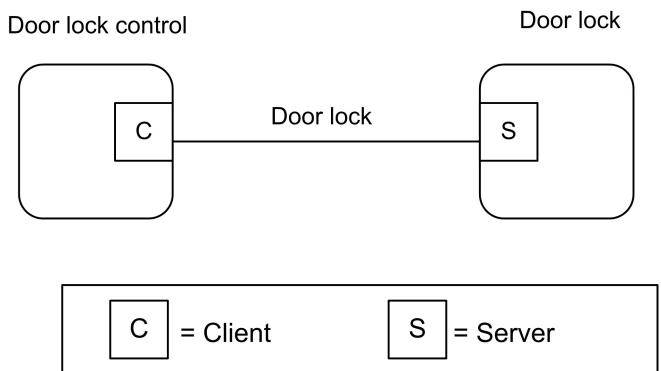
The clusters specified in this document are for use typically in applications involving closures (e.g., shades, windows, doors), but MAY be used in any application domain.

5.1.2. Cluster List

This section lists the closures specific clusters as specified in this chapter.

Table 37. Overview of the Closures Clusters

Cluster ID	Cluster Name	Description
0x0101	Door Lock	An interface to a generic way to secure a door
0x0102	Window Covering	Commands and attributes for controlling a window covering



Note: Device names are examples for illustration purposes only

Figure 14. Typical Usage of the Closures Clusters

5.2. Door Lock

5.2.1. Overview

The door lock cluster provides an interface to a generic way to secure a door. The physical object that provides the locking functionality is abstracted from the cluster. The cluster has a small list of mandatory attributes and functions and a list of optional features.

5.2.2. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	mandatory global ClusterRevision attribute added; CCB 1811 1812 1821
2	CCB 2430
3	CCB 2629 2630
4	All Hubs changes and added feature map
5	
6	New data model format and notation. Added User features. General cleanup of functionality
7	Added support for European door locks (unbolt feature)

5.2.3. Classification

Hierarchy	Role	Context	PICS Code
Base	Application	Endpoint	DRLK

5.2.4. Cluster ID

ID	Name
0x0101	Door Lock

5.2.5. Features

This cluster SHALL support the Feature Map global attribute with these bits defined.

Bit	Code	Feature	Conformance	Summary
0	PIN	PIN Credential	O	Lock supports PIN credentials (via keypad, or over-the-air)
1	RID	RFID Credential	O	Lock supports RFID credentials
2	FGP	Finger Credentials	P, O	Lock supports finger related credentials (fingerprint, finger vein)

Bit	Code	Feature	Conformance	Summary
3	LOG	Logging	O	Lock supports local/on-lock logging when Events are not supported
4	WDSCH	Week Day Access Schedules	O	Lock supports week day user access schedules
5	DPS	Door Position Sensor	O	Lock supports a door position sensor that indicates door's state
6	FACE	Face Credentials	P, O	Lock supports face related credentials (face, iris, retina)
7	COTA	Credential Over-the-Air Access	O	PIN codes over-the-air supported for lock/unlock operations
8	USR	User	[PIN RID FGP FACE]	Lock supports the user commands and database
9	NOT	Notification	O	Operation and Programming Notifications
10	YDSCH	Year Day Access Schedules	O	Lock supports year day user access schedules
11	HDSCH	Holiday Schedules	O	Lock supports holiday schedules
12	UBOLT	Unbolting	O	Lock supports unbolting

5.2.5.1. PIN Credential Feature

If the User Feature is also supported then any PIN Code stored in the lock SHALL be associated with a User.

A lock MAY support multiple credential types so if the User feature is supported the UserType, UserStatus and Schedules are all associated with a User index and not directly with a PIN index. A User index may have several credentials associated with it.

5.2.5.2. RFID Credential Feature

If the User Feature is also supported then any RFID credential stored in the lock SHALL be associated with a User.

A lock MAY support multiple credential types so if the User feature is supported the UserType, UserStatus and Schedules are all associated with a User index and not directly with a RFID index. A User Index may have several credentials associated with it.

5.2.5.3. Finger Credentials Feature

Currently the cluster only defines the metadata format for notifications when a fingerprint/ finger vein credential is used to access the lock and doesn't describe how to create fingerprint/finger vein credentials. If the Users feature is also supported then the User that a fingerprint/finger vein is associated with can also have its UserType, UserStatus and Schedule modified.

A lock MAY support multiple credential types so if the User feature is supported the UserType, UserStatus and Schedules are all associated with a User index and not directly with a Finger index. A User Index may have several credentials associated with it.

5.2.5.4. Logging Feature

If Events are not supported the logging feature SHALL replace the Event reporting structure. If Events are supported the logging feature SHALL NOT be supported.

5.2.5.5. Week Day Access Schedules Feature

If the User feature is supported then Week Day Schedules are applied to a User and not a credential.

Week Day Schedules are used to restrict access to a specified time window on certain days of the week. The schedule is repeated each week. When a schedule is cleared this clears the access restrictions and grants unrestricted access to the user. The lock MAY automatically adjust the UserType when a schedule is created or cleared.

5.2.5.6. Door Position Sensor Feature

If this feature is supported this indicates that the lock has the ability to determine the position of the door which is separate from the state of the lock.

5.2.5.7. Face Credentials Feature

Currently the cluster only defines the metadata format for notifications when a face recognition, iris, or retina credential is used to access the lock and doesn't describe how to create face recognition, iris, or retina credentials. If the Users feature is also supported then the User that a face recognition, iris, or retina credential is associated with can also have its UserType, UserStatus and Schedule modified.

A lock MAY support multiple credential types so if the User feature is supported the UserType, UserStatus and Schedules are all associated with a User and not directly with a credential.

5.2.5.8. Credential Over-the-Air Access Feature

If this feature is supported then the lock supports the ability to verify a credential provided in a lock/unlock command. Currently the cluster only supports providing the PIN credential to the lock/unlock commands. If this feature is supported then the PIN Credential feature SHALL also be supported.

5.2.5.9. User Feature

If the User Feature is supported then a lock employs a User database. A User within the User database is used to associate credentials and schedules to single user record within the lock. This also means the UserType and UserStatus fields are associated with a User and not a credential.

5.2.5.10. Notification Feature

This is a feature used before support of events. This feature supports notification commands and masks used to filter these notifications.

5.2.5.11. Year Day Access Schedules Feature

If the User feature is supported then Year Day Schedules are applied to a User and not a credential.

Year Day Schedules are used to restrict access to a specified date and time window. When a schedule is cleared this clears the access restrictions and grants unrestricted access to the user. The lock MAY automatically adjust the UserType when a schedule is created or cleared.

5.2.5.12. Holiday Schedules Feature

This feature is used to setup Holiday Schedule in the lock device. A Holiday Schedule sets a start and stop end date/time for the lock to use the specified operating mode set by the Holiday Schedule.

5.2.5.13. Unbolting Feature

Locks that support this feature differentiate between unbolting and unlocking. The Unbolt Door command retracts the bolt without pulling the latch. The Unlock Door command fully unlocks the door by retracting the bolt and briefly pulling the latch. While the latch is pulled, the lock state changes to Unlatched. Locks without unbolting support don't differentiate between unbolting and unlocking and perform the same operation for both commands.

5.2.5.13.1. Recommended steps for creating a new User

It is RECOMMENDED that the Administrator query the door lock for what users already exist in its database to find an available UserIndex for creating a new User (see [Get User Command](#)).

1. Use [Set User Command](#) with an available UserIndex to set the user record fields as applicable.
2. Use [Set Credential Command](#) with same UserIndex to add one or more credentials as applicable.
3. Use [Set Week Day Schedule Command](#) or [Set Year Day Schedule Command](#) with same UserIndex to add one or more schedule restrictions as applicable.

5.2.6. Attributes

Id	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	LockState	enum8	desc	X P S		R V	M
0x0001	LockType	enum8	desc			R V	M
0x0002	ActuatorEnabled	bool	all			R V	M
0x0003	DoorState	enum8	desc	X P		R V	DPS
0x0004	DoorOpenEvents	uint32	all			RW VM	[DPS]
0x0005	Door-ClosedEvents	uint32	all			RW VM	[DPS]
0x0006	OpenPeriod	uint16	all			RW VM	[DPS]
0x0010	NumberOfLoggingRecordsSupported	uint16	all	F	0	R V	LOG
0x0011	NumberOfTotalUsersSupported	uint16	all	F	0	R V	USR
0x0012	NumberOfPINUsersSupported	uint16	all	F	0	R V	PIN
0x0013	NumberOfRFIDUsersSupported	uint16	all	F	0	R V	RID
0x0014	NumberOfWeek-DaySchedulesSupportedPerUser	uint8	all	F	0	R V	WDSCH

Id	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0015	NumberOfYear-DaySchedulesSupportedPerUser	uint8	all	F	0	R V	YDSCH
0x0016	NumberOfHolidaySchedulesSupported	uint8	all	F	0	R V	HDSCH
0x0017	MaxPIN-Code-Length	uint8	all	F	MS	R V	PIN
0x0018	MinPIN-Code-Length	uint8	all	F	MS	R V	PIN
0x0019	MaxRFID-Code-Length	uint8	all	F	MS	R V	RID
0x001A	MinRFID-Code-Length	uint8	all	F	MS	R V	RID
0x001B	CredentialRulesSupport	map8	all	F	1	R V	USR
0x001C	NumberOfCredentialsSupportedPerUser	uint8	all	F	0	R V	USR
0x0020	EnableLogging	bool	all	P	0	R[W] VA	LOG
0x0021	Language	string	max 3	P	MS	R[W] VM	O
0x0022	LEDSettings	uint8	desc	P	0	R[W] VM	O
0x0023	AutoRe-lockTime	uint32	all	P	MS	R[W] VM	O

Id	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0024	SoundVolume	uint8	desc	P	0	R[W] VM	O
0x0025	Operating-Mode	enum8	desc	P	0	R[W] VM	M
0x0026	SupportedOperatingModes	map16	all	F	0xFFFF6	R V	M
0x0027	Default-ConfigurationRegister	map16	all	P	0	R V	O
0x0028	EnableLocalProgramming	bool	all	P	1	R[W] VA	O
0x0029	EnableOne-TouchLocking	bool	all	P	0	RW VM	O
0x002A	EnableInsideStatusLED	bool	all	P	0	RW VM	O
0x002B	EnablePrivacyMode-Button	bool	all	P	0	RW VM	O
0x002C	LocalProgrammingFeatures	map8	all	P	0	R[W] VA	O
0x0030	Wrong-CodeEntryLimit	uint8	1 to 255	P	MS	R[W] VA	PIN RID
0x0031	UserCode-TemporaryDisableTime	uint8	1 to 255	P	MS	R[W] VA	PIN RID
0x0032	Send-PINOverTheAir	bool	all	P	0	R[W] VA	[!USR & PIN]

Id	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0033	RequirePINforRemoteOperation	bool	all	P	0	R[W] VA	COTA & PIN
0x0034	SecurityLevel				0		D
0x0035	ExpiringUserTimeout	uint16	1 to 2880	P	MS	R[W] VA	[USR]
0x0040	AlarmMask	map16	all	P	0xFFFF	RW VA	O
0x0041	KeypadOperationEventMask	map16	all	P	0xFFFF	RW VA	[NOT & PIN]
0x0042	RemoteOperationEventMask	map16	all	P	0xFFFF	RW VA	[NOT]
0x0043	ManualOperationEventMask	map16	all	P	0xFFFF	RW VA	[NOT]
0x0044	RFIDOperationEventMask	map16	all	P	0xFFFF	RW VA	[NOT & RID]
0x0045	KeypadProgrammingEventMask	map16	all	P	0xFFFF	RW VA	[NOT & PIN]
0x0046	RemoteProgrammingEventMask	map16	all	P	0xFFFF	RW VA	[NOT]
0x0047	RFIDProgrammingEventMask	map16	all	P	0xFFFF	RW VA	[NOT & RID]

AutoRelockTime, OperatingMode and SupportedOperatingModes represent mandatory fields that

were previously not present or optional. Implementors should be aware that older devices may not implement them.

5.2.6.1. Scene Table Extensions

If the Scenes server cluster is implemented on the same endpoint, the following attribute SHALL be part of the ExtensionFieldSetStruct of the Scene Table.

- LockState

When the LockState attribute is part of a Scene table, the attribute is treated as a writable command; that is:

- Setting the LockState to Locked SHALL command the lock to lock.
- Setting the LockState to Unlocked SHALL command the lock to unlock.
- If a lock supports the [Unbolting Feature](#), setting the LockState to Unlocked SHALL unlock the door without pulling the latch.
- Setting the LockState attribute to Unlatched SHALL command the lock to unlock and pull the latch.
- Trying to write the LockState attribute to Not Fully Locked SHALL be ignored and not change the state of the lock.

The Transition Time field in the Scene table will be treated as a delay before setting the LockState attribute; that is, it is possible to activate a scene with the lock actuation some seconds later.

Locks that do not have an actuation mechanism SHOULD not support the Scene table extension.

5.2.6.2. LockState Attribute

This attribute has the following possible values:

Value	Name	Conformance	Summary
0	Not Fully Locked	M	Lock state is not fully locked
1	Locked	M	Lock state is fully locked
2	Unlocked	M	Lock state is fully unlocked
3	Unlatched	O	Lock state is fully unlocked and the latch is pulled

The LockState Attribute may be NULL if the lock hardware does not currently know the status of the locking mechanism. For example, a lock may not know the LockState status after a power cycle until the first lock actuation is completed.

The Not Fully Locked value is used by a lock to indicate that the state of the lock is somewhere

Matter Application Cluster Specification R1.2 Connectivity Standards Alliance Document 23-27350 October 18, 2023 between Locked and Unlocked so it is only partially secured. For example, a deadbolt could be partially extended and not in a dead latched state.

5.2.6.3. LockType Attribute

The LockType attribute is indicated by an enumeration:

Value	Name	Summary
0	Dead bolt	Physical lock type is dead bolt
1	Magnetic	Physical lock type is magnetic
2	Other	Physical lock type is other
3	Mortise	Physical lock type is mortise
4	Rim	Physical lock type is rim
5	Latch Bolt	Physical lock type is latch bolt
6	Cylindrical Lock	Physical lock type is cylindrical lock
7	Tubular Lock	Physical lock type is tubular lock
8	Interconnected Lock	Physical lock type is interconnected lock
9	Dead Latch	Physical lock type is dead latch
10	Door Furniture	Physical lock type is door furniture
11	Eurocylinder	Physical lock type is eurocylinder

5.2.6.4. ActuatorEnabled Attribute

The ActuatorEnabled attribute indicates if the lock is currently able to (Enabled) or not able to (Disabled) process remote Lock, Unlock, or Unlock with Timeout commands.

This attribute has the following possible values:

Boolean Value	Summary
0	Disabled
1	Enabled

5.2.6.5. DoorState Attribute

The current door state as defined in [DoorStateEnum](#).

This attribute SHALL be null only if an internal error prevents the retrieval of the current door state.

5.2.6.6. DoorOpenEvents Attribute

This attribute holds the number of door open events that have occurred since it was last zeroed.

5.2.6.7. DoorClosedEvents Attribute

This attribute holds the number of door closed events that have occurred since it was last zeroed.

5.2.6.8. OpenPeriod Attribute

This attribute holds the number of minutes the door has been open since the last time it transitioned from closed to open.

5.2.6.9. NumberOfLogRecordsSupported Attribute

The number of available log records.

5.2.6.10. NumberOfTotalUsersSupported Attribute

Number of total users supported by the lock.

5.2.6.11. NumberOfPINUsersSupported Attribute

The number of PIN users supported.

5.2.6.12. NumberOfRFIDUsersSupported Attribute

The number of RFID users supported.

5.2.6.13. NumberOfWeekDaySchedulesSupportedPerUser Attribute

The number of configurable week day schedule supported per user.

5.2.6.14. NumberOfYearDaySchedulesSupportedPerUser Attribute

The number of configurable year day schedule supported per user.

5.2.6.15. NumberOfHolidaySchedulesSupported Attribute

The number of holiday schedules supported for the entire door lock device.

5.2.6.16. MaxPINCodeLength Attribute

An 8 bit value indicates the maximum length in bytes of a PIN Code on this device.

5.2.6.17. MinPINCodeLength Attribute

An 8 bit value indicates the minimum length in bytes of a PIN Code on this device.

5.2.6.18. MaxRFIDCodeLength Attribute

An 8 bit value indicates the maximum length in bytes of a RFID Code on this device. The value

Matter Application Cluster Specification R1.2 Connectivity Standards Alliance Document 23-27350 October 18, 2023 depends on the RFID code range specified by the manufacturer, if media anti-collision identifiers (UID) are used as RFID code, a value of 20 (equals 10 Byte ISO 14443A UID) is recommended.

5.2.6.19. MinRFIDCodeLength Attribute

An 8 bit value indicates the minimum length in bytes of a RFID Code on this device. The value depends on the RFID code range specified by the manufacturer, if media anti-collision identifiers (UID) are used as RFID code, a value of 8 (equals 4 Byte ISO 14443A UID) is recommended.

5.2.6.20. CredentialRulesSupport Attribute

This bitmap contains a bit for every value of [CredentialRuleEnum](#) supported on this device.

Bit	Name	Summary
0	Single	Only one credential is required for lock operation
1	Dual	Any two credentials are required for lock operation
2	Tri	Any three credentials are required for lock operation

5.2.6.21. NumberOfCredentialsSupportedPerUser Attribute

The number of credentials that could be assigned for each user.

Depending on the value of [NumberOfRFIDUsersSupported](#) and [NumberOfPINUsersSupported](#) it may not be possible to assign that number of credentials for a user.

For example, if the device supports only PIN and RFID credential types, [NumberOfCredentialsSupportedPerUser](#) is set to 10, [NumberOfPINUsersSupported](#) is set to 5 and [NumberOfRFIDUsersSupported](#) is set to 3, it will not be possible to actually assign 10 credentials for a user because maximum number of credentials in the database is 8.

5.2.6.22. EnableLogging Attribute

Enable/disable event logging. When event logging is enabled, all event messages are stored on the lock for retrieval. Logging events can be but not limited to Tamper Alarm, Lock, Unlock, AutoRelock, User Code Added, User Code Cleared, Schedule Added, and Schedule Cleared. For a full detail of all the possible alarms and events, please refer to the full list in the Alarm and Event Masks Attribute Set.

5.2.6.23. Language Attribute

Modifies the language for the on-screen or audible user interface using a 2-byte language code from ISO-639-1.

5.2.6.24. OperatingMode Attribute

The current operating mode of the lock (see [OperatingModeEnum](#)).

5.2.6.25. SupportedOperatingModes Attribute

This bitmap contains all operating bits of the Operating Mode Attribute supported by the lock. All operating modes NOT supported by a lock SHALL be set to one. The value of the OperatingMode enumeration defines the related bit to be set, as shown below:

Bit	Name	Conformance
0	Normal	M
1	Vacation	O
2	Privacy	O
3	NoRemoteLockUnlock	M
4	Passage	O

5.2.6.26. LEDSettings Attribute

The settings for the LED support three different modes, shown below:

Value	Name
0	Never use LED for signalization
1	Use LED signalization except for access allowed events
2	Use LED signalization for all events

5.2.6.27. AutoRelockTime Attribute

The number of seconds to wait after unlocking a lock before it automatically locks again. 0=disabled. If set, unlock operations from any source will be timed. For one time unlock with timeout use the specific command.

5.2.6.28. SoundVolume Attribute

The sound volume on a door lock has four possible settings: silent, low, high and medium volumes, shown below:

Value	Name
0	Silent Mode
1	Low Volume
2	High Volume
3	Medium Volume

5.2.6.29. DefaultConfigurationRegister Attribute

This attribute represents the default configurations as they are physically set on the device (example: hardware dip switch setting, etc...) and represents the default setting for some of the attributes

within this cluster (for example: LED, Auto Lock, Sound Volume, and Operating Mode attributes).

This is a read-only attribute and is intended to allow clients to determine what changes MAY need to be made without having to query all the included attributes. It MAY be beneficial for the clients to know what the device's original settings were in the event that the device needs to be restored to factory default settings.

If the Client device would like to query and modify the door lock server's operating settings, it SHOULD send read and write attribute requests to the specific attributes.

For example, the Sound Volume attribute default value is Silent Mode. However, it is possible that the current Sound Volume is High Volume. Therefore, if the client wants to query/modify the current Sound Volume setting on the server, the client SHOULD read/write to the Sound Volume attribute.

This attribute has the following possible values:

Bit	Name	Summary
0	Local Programming	0 - Enable Local Programming Attribute default value is 0 (disabled) 1 - Enable Local Programming Attribute default value is 1 (enabled)
1	Keypad Interface	0 - Keypad Interface default access is disabled 1 - Keypad Interface default access is enabled
2	Remote Interface	0 - Remote Interface default access is disabled 1 - Remote Interface default access is enabled
5	Sound Volume	0 – Sound Volume attribute default value is 0 (Silent Mode) 1 – Sound Volume attribute default value is equal to something other than 0
6	Auto Relock	0 – Auto Relock Time attribute default value = 0 1 – Auto Relock Time attribute default value is equal to something other than 0

Bit	Name	Summary
7	LED Settings	0 – LED Settings attribute default value = 0 1 – LED Settings attribute default value is equal to something other than 0

5.2.6.30. EnableLocalProgramming Attribute

Enable/disable local programming on the door lock of certain features (see LocalProgrammingFeatures attribute). If this value is set to TRUE then local programming is enabled on the door lock for all features. If it is set to FALSE then local programming is disabled on the door lock for those features whose bit is set to 0 in the LocalProgrammingFeatures attribute. Local programming SHALL be enabled by default.

5.2.6.31. EnableOneTouchLocking Attribute

Enable/disable the ability to lock the door lock with a single touch on the door lock.

5.2.6.32. EnableInsideStatusLED Attribute

Enable/disable an inside LED that allows the user to see at a glance if the door is locked.

5.2.6.33. EnablePrivacyModeButton Attribute

Enable/disable a button inside the door that is used to put the lock into privacy mode. When the lock is in privacy mode it cannot be manipulated from the outside.

5.2.6.34. LocalProgrammingFeatures Attribute

The local programming features that will be disabled when EnableLocalProgramming attribute is set to False. If a door lock doesn't support disabling one aspect of local programming it SHALL return CONSTRAINT_ERROR during a write operation of this attribute. If the EnableLocalProgramming attribute is set to True then all local programming features SHALL be enabled regardless of the bits set to 0 in this attribute.

The features that can be disabled from local programming are defined in the following bitmap.

Bit	Name	Summary
0	Add Locally	0 - The ability to add Users/credentials/Schedules locally is disabled 1 - The ability to add Users/Credentials/Schedules locally is enabled

Bit	Name	Summary
1	Modify Locally	0 - The ability to modify Users/credentials/Schedules locally is disabled
		1 - The ability to modify Users/Credentials/Schedules locally is enabled
2	Clear Locally	0 - The ability to clear Users/credentials/Schedules locally is disabled
		1 - The ability to clear Users/Credentials/Schedules locally is enabled
3	Adjust Locally	0 - The ability to adjust lock settings locally is disabled
		1 - The ability to adjust lock settings locally is enabled

5.2.6.35. WrongCodeEntryLimit Attribute

The number of incorrect Pin codes or RFID presentment attempts a user is allowed to enter before the lock will enter a lockout state. The value of this attribute is compared to all failing forms of credential presentation, including Pin codes used in an Unlock Command when RequirePINforRemote-Operation is set to true. Valid range is 1-255 incorrect attempts. The lockout state will be for the duration of UserCodeTemporaryDisableTime. If the attribute accepts writes and an attempt to write the value 0 is made, the device SHALL respond with CONSTRAINT_ERROR.

The lock MAY reset the counter used to track incorrect credential presentations as required by internal logic, environmental events, or other reasons. The lock SHALL reset the counter if a valid credential is presented.

5.2.6.36. UserCodeTemporaryDisableTime Attribute

The number of seconds that the lock shuts down following wrong code entry. Valid range is 1-255 seconds. Device can shut down to lock user out for specified amount of time. (Makes it difficult to try and guess a PIN for the device.) If the attribute accepts writes and an attempt to write the attribute to 0 is made, the device SHALL respond with CONSTRAINT_ERROR.

5.2.6.37. SendPINOverTheAir Attribute

Boolean set to True if it is ok for the door lock server to send PINs over the air. This attribute determines the behavior of the server's TX operation. If it is false, then it is not ok for the device to send PIN in any messages over the air.

The PIN field within any door lock cluster message SHALL keep the first octet unchanged and

masks the actual code by replacing with 0xFF. For example (PIN "1234"): If the attribute value is True, 0x04 0x31 0x32 0x33 0x34 SHALL be used in the PIN field in any door lock cluster message payload. If the attribute value is False, 0x04 0xFF 0xFF 0xFF 0xFF SHALL be used.

5.2.6.38. **RequirePINForRemoteOperation** Attribute

Boolean set to True if the door lock server requires that an optional PINs be included in the payload of remote lock operation events like Lock, Unlock, Unlock with Timeout and Toggle in order to function.

5.2.6.39. **ExpiringUserTimeout** Attribute

Number of minutes a PIN, RFID, Fingerprint, or other credential associated with a user of type ExpiringUser SHALL remain valid after its first use before expiring. When the credential expires the UserStatus for the corresponding user record SHALL be set to OccupiedDisabled.

5.2.6.40. **AlarmMask** Attribute

This attribute is only supported if the Alarms cluster is on the same endpoint. The alarm mask is used to turn on/off alarms for particular functions. Alarms for an alarm group are enabled if the associated alarm mask bit is set. Each bit represents a group of alarms. Entire alarm groups can be turned on or off by setting or clearing the associated bit in the alarm mask.

This attribute has the following possible values:

This mask DOES NOT apply to the Events mechanism of this cluster.

Name	Bit	Summary	Conformance
Alarm Code 0	0	Locking Mechanism Jammed	M
Alarm Code 1	1	Lock Reset to Factory Defaults	O
Alarm Code 2	2	Reserved	O
Alarm Code 3	3	RF Module Power Cycled	O
Alarm Code 4	4	Tamper Alarm - wrong code entry limit	PIN RID
Alarm Code 5	5	Tamper Alarm - front escutcheon removed from main	O
Alarm Code 6	6	Forced Door Open under Door Locked Condition	O

5.2.6.41. KeypadOperationEventMask Attribute

Event mask used to turn on and off the transmission of keypad operation events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Operation Event Notification Command.

This attribute has the following possible values:

Event Source	Name	Bit	Summary
0	Operation Event Code 0	0	Unknown or manufacturer-specific keypad operation event
0	Operation Event Code 1	1	Lock, source: keypad
0	Operation Event Code 2	2	Unlock, source: keypad
0	Operation Event Code 3	3	Lock, source: keypad, error: invalid PIN
0	Operation Event Code 4	4	Lock, source: keypad, error: invalid schedule
0	Operation Event Code 5	5	Unlock, source: keypad, error: invalid code
0	Operation Event Code 6	6	Unlock, source: keypad, error: invalid schedule
0	Operation Event Code 15	7	Non-Access User operation event, source keypad.

This mask DOES NOT apply to the Events mechanism of this cluster.

5.2.6.42. RemoteOperationEventMask Attribute

Event mask used to turn on and off the transmission of remote operation events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Operation Event Notification Command.

This attribute has the following possible values:

Event Source	Name	Bit	Summary
1	Operation Event Code 0	0	Unknown or manufacturer-specific remote operation event
1	Operation Event Code 1	1	Lock, source: remote
1	Operation Event Code 2	2	Unlock, source: remote
1	Operation Event Code 3	3	Lock, source: remote, error: invalid code

Event Source	Name	Bit	Summary
1	Operation Event Code 4	4	Lock, source: remote, error: invalid schedule
1	Operation Event Code 5	5	Unlock, source: remote, error: invalid code
1	Operation Event Code 6	6	Unlock, source: remote, error: invalid schedule

This mask DOES NOT apply to the Events mechanism of this cluster.

5.2.6.43. ManualOperationEventMask Attribute

Event mask used to turn on and off manual operation events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Operation Event Notification Command.

This attribute has the following possible values:

Event Source	Name	Bit	Summary
2	Operation Even Code 0	0	Unknown or manufacturer-specific manual operation event
2	Operation Even Code 1	1	Thumbturn Lock
2	Operation Even Code 2	2	Thumbturn Unlock
2	Operation Even Code 7	3	One touch lock
2	Operation Even Code 8	4	Key Lock
2	Operation Even Code 9	5	Key Unlock
2	Operation Even Code 10	6	Auto lock
2	Operation Even Code 11	7	Schedule Lock
2	Operation Even Code 12	8	Schedule Unlock
2	Operation Even Code 13	9	Manual Lock (Key or Thumbturn)
2	Operation Even Code 14	10	Manual Unlock (Key or Thumbturn)

This mask DOES NOT apply to the Events mechanism of this cluster.

5.2.6.44. RFIDOperationEventMask Attribute

Event mask used to turn on and off RFID operation events. This mask DOES NOT apply to the stor-

This attribute has the following possible values:

Event Source	Name	Bit	Summary
3	Operation Event Code 0	0	Unknown or manufacturer-specific keypad operation event
3	Operation Event Code 1	1	Lock, source: RFID
3	Operation Event Code 2	2	Unlock, source: RFID
3	Operation Event Code 3	3	Lock, source: RFID, error: invalid RFID ID
3	Operation Event Code 4	4	Lock, source: RFID, error: invalid schedule
3	Operation Event Code 5	5	Unlock, source: RFID, error: invalid RFID ID
3	Operation Event Code 6	6	Unlock, source: RFID, error: invalid schedule

This mask DOES NOT apply to the Events mechanism of this cluster.

5.2.6.45. KeypadProgrammingEventMask Attribute

Event mask used to turn on and off keypad programming events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Programming Event Notification Command.

This attribute has the following possible values:

Event Source	Name	Bit	Summary
0	Program Event Code 0	0	Unknown or manufacturer-specific keypad programming event

Event Source	Name	Bit	Summary
0	Program Event Code 1	1	<p>Programming PIN code changed, source: keypad</p> <p>User ID: programming user ID.</p> <p>PIN: default or programming PIN code if codes can be sent over the air per attribute.</p> <p>User type: default</p> <p>User Status: default</p>
0	Program Event Code 2	2	<p>PIN added, source: keypad</p> <p>User ID: user ID that was added.</p> <p>PIN: code that was added (if codes can be sent over the air per attribute.)</p> <p>User type: default or type added.</p> <p>User Status: default or status added.</p>
0	Program Event Code 3	3	<p>PIN cleared, source: keypad</p> <p>User ID: user ID that was cleared.</p> <p>PIN: code that was cleared (if codes can be sent over the air per attribute.)</p> <p>User type: default or type cleared.</p> <p>User Status: default or status cleared.</p>

Event Source	Name	Bit	Summary
0	Program Event Code 4	4	<p>PIN changed</p> <p>Source: keypad</p> <p>User ID: user ID that was changed</p> <p>PIN: code that was changed (if codes can be sent over the air per attribute.)</p> <p>User type: default or type changed.</p> <p>User Status: default or status changed.</p>

This mask DOES NOT apply to the Events mechanism of this cluster.

5.2.6.46. RemoteProgrammingEventMask Attribute

Event mask used to turn on and off remote programming events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Programming Event Notification Command.

This attribute has the following possible values:

Event Source	Name	Bit	Summary
1	Program Event Code 0	0	Unknown or manufacturer-specific remote programming event.
1	Program Event Code 2	2	<p>PIN added, source remote</p> <p>Same as keypad source above</p>
1	Program Event Code 3	3	<p>PIN cleared, source remote</p> <p>Same as keypad source above.</p>

Event Source	Name	Bit	Summary
1	Program Event Code 4	4	PIN changed Source remote Same as keypad source above
1	Program Event Code 5	5	RFID code added, Source remote
1	Program Event Code 6	6	RFID code cleared, Source remote

This mask DOES NOT apply to the Events mechanism of this cluster.

5.2.6.47. RFIDProgrammingEventMask Attribute

Event mask used to turn on and off RFID programming events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Programming Event Notification Command.

This attribute has the following possible values:

Event Source	Name	Bit	Summary
3	Program Event Code 0	0	Unknown or manufacturer-specific keypad programming event
3	Program Event Code 5	5	ID Added, Source: RFID User ID: user ID that was added. ID: ID that was added (if codes can be sent over the air per attribute.) User Type: default or type added. User Status: default or status added.

Event Source	Name	Bit	Summary
3	Program Event Code 6	6	<p>ID cleared, Source: RFID</p> <p>User ID: user ID that was cleared.</p> <p>ID: ID that was cleared (if codes can be sent over the air per attribute.)</p> <p>User Type: default or type cleared.</p> <p>User Status: default or status cleared.</p>

This mask DOES NOT apply to the Events mechanism of this cluster.

5.2.7. Commands

Id	Name	Direction	Response	Access	Conformance
0x00	Lock Door	Client ⇒ Server	Y	T O	M
0x01	Unlock Door	Client ⇒ Server	Y	T O	M
0x02	Toggle	Client ⇒ Server	Y	T O	X
0x03	Unlock with Timeout	Client ⇒ Server	Y	T O	O
0x04	Get Log Record	Client ⇒ Server	Get Log Record Response	M	LOG
0x04	Get Log Record Response	Server ⇒ Client	N		LOG
0x05	Set PIN Code	Client ⇒ Server	Y	T A	!USR & PIN
0x06	Get PIN Code	Client ⇒ Server	Get PIN Code Response	A	!USR & PIN
0x06	Get PIN Code Response	Server ⇒ Client	N		!USR & PIN
0x07	Clear PIN Code	Client ⇒ Server	Y	T A	!USR & PIN
0x08	Clear All PIN Codes	Client ⇒ Server	Y	T A	!USR & PIN

Id	Name	Direction	Response	Access	Conformance
0x09	Set User Status	Client ⇒ Server	Y	A	!USR & (PIN RID FGP)
0x0A	Get User Status	Client ⇒ Server	Get User Status Response	A	!USR & (PIN RID FGP)
0x0A	Get User Status Response	Server ⇒ Client	N		!USR
0x0B	Set Week Day Schedule	Client ⇒ Server	Y	A	WDSCH
0x0C	Get Week Day Schedule	Client ⇒ Server	Get Week Day Schedule Response	A	WDSCH
0x0C	Get Week Day Schedule Response	Server ⇒ Client	N		WDSCH
0x0D	Clear Week Day Schedule	Client ⇒ Server	Y	A	WDSCH
0x0E	Set Year Day Schedule	Client ⇒ Server	Y	A	YDSCH
0x0F	Get Year Day Schedule	Client ⇒ Server	Get Year Day Schedule Response	A	YDSCH
0x0F	Get Year Day Schedule Response	Server ⇒ Client	N		YDSCH
0x10	Clear Year Day Schedule	Client ⇒ Server	Y	A	YDSCH
0x11	Set Holiday Schedule	Client ⇒ Server	Y	A	HDSCH
0x12	Get Holiday Schedule	Client ⇒ Server	Get Holiday Schedule Response	A	HDSCH
0x12	Get Holiday Schedule Response	Server ⇒ Client	N		HDSCH
0x13	Clear Holiday Schedule	Client ⇒ Server	Y	A	HDSCH
0x14	Set User Type	Client ⇒ Server	Y	A	!USR & (PIN RID FGP)
0x15	Get User Type	Client ⇒ Server	Get User Type Response	A	!USR & (PIN RID FGP)

Id	Name	Direction	Response	Access	Conformance
0x15	Get User Type Response	Server ⇒ Client	N		!USR
0x16	Set RFID Code	Client ⇒ Server	Y	T A	!USR & RID
0x17	Get RFID Code	Client ⇒ Server	Get RFID Code Response	A	!USR & RID
0x17	Get RFID Code Response	Server ⇒ Client	N		!USR & RID
0x18	Clear RFID Code	Client ⇒ Server	Y	T A	!USR & RID
0x19	Clear All RFID Codes	Client ⇒ Server	Y	T A	!USR & RID
0x1A	Set User	Client ⇒ Server	Y	T A	USR
0x1B	Get User	Client ⇒ Server	Get User Response	A	USR
0x1C	Get User Response	Server ⇒ Client	N		USR
0x1D	Clear User	Client ⇒ Server	Y	T A	USR
0x20	Operating Event Notification	Server ⇒ Client	N		[NOT]
0x21	Programming Event Notification	Server ⇒ Client	N		[NOT]
0x22	Set Credential	Client ⇒ Server	Set Credential Response	T A	USR
0x23	Set Credential Response	Server ⇒ Client	N		USR
0x24	Get Credential Status	Client ⇒ Server	Get Credential Status Response	A	USR
0x25	Get Credential Status Response	Server ⇒ Client	N		USR
0x26	Clear Credential	Client ⇒ Server	Y	T A	USR
0x27	Unbolt Door	Client ⇒ Server	Y	T O	UBOLT

5.2.7.1. Lock Door Command

This command causes the lock device to lock the door. This command includes an optional code for the lock. The door lock MAY require a PIN depending on the value of the [RequirePINForRemoteOperation attribute](#).

Id	Field	Type	Constraint	Quality	Default	Conformance
0	PINCode	octstr				[COTA & PIN]

† The PIN/RFID Code is an obsolete field name, use PINCode instead.

5.2.7.1.1. PINCode Field

If the [RequirePINforRemoteOperation attribute](#) is True then PINCode field SHALL be provided and the door lock SHALL NOT grant access if it is not provided.

If the PINCode field is provided, the door lock SHALL verify PINCode before granting access regardless of the value of [RequirePINForRemoteOperation attribute](#).

When the PINCode field is provided an invalid PIN will count towards the WrongCodeEntryLimit and the UserCodeTemporaryDisableTime will be triggered if the WrongCodeEntryLimit is exceeded. The lock SHALL ignore any attempts to lock/unlock the door until the UserCodeTemporaryDisableTime expires.

5.2.7.2. Unlock Door Command

This command causes the lock device to unlock the door. This command includes an optional code for the lock. The door lock MAY require a code depending on the value of the [RequirePINForRemoteOperation attribute](#).

Note: If the attribute AutoRelockTime is supported the lock will transition to the locked state when the auto relock time has expired.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	PINCode	octstr				[COTA & PIN]

† The PIN/RFID Code is an obsolete field name, use PINCode instead.

5.2.7.2.1. PINCode Field

See [PINCode Field](#).

5.2.7.3. Unlock with Timeout Command

This command causes the lock device to unlock the door with a timeout parameter. After the time in seconds specified in the timeout field, the lock device will relock itself automatically. This timeout parameter is only temporary for this message transition and overrides the default relock time as specified in the AutoRelockTime attribute. If the door lock device is not capable of or does not want to support temporary Relock Timeout, it SHOULD NOT support this optional command.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Timeout	uint16				M
1	PINCode	octstr				[COTA & PIN]

† The PIN/RFID Code is an obsolete field name, use PINCode instead.

5.2.7.3.1. Timeout Field

The timeout in seconds to wait before relocking the door lock. This value is independent of the AutoRelockTime attribute value.

5.2.7.3.2. PINCode Field

See [PINCode Field](#).

5.2.7.4. Get Log Record Command

Request a log record. Log number is between 1 – [Number of Log Records Supported attribute]. If log number 0 is requested then the most recent log entry is returned.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Log Index	uint16	desc			M

Log record format: The log record format is defined in the description of the Get Log Record Response command.

5.2.7.5. Get Log Record Response Command

Returns the specified log record. If an invalid log entry ID was requested, it is set to 0 and the most recent log entry will be returned.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Log Entry ID	uint16	desc			M

Id	Field	Type	Constraint	Quality	Default	Conformance
1	Timestamp	epoch-s	all			M
2	Event Type	enum8	desc			M
3	Source	uint8	desc			M
4	Event ID/Alarm Code	uint8	desc			M
5	User ID	uint16	desc			M
6	PIN	octstr				M

5.2.7.5.1. Log Entry ID Field

This is the index into the log table where this log entry is stored. If the log entry requested is 0, the most recent log is returned with the appropriate log entry ID.

5.2.7.5.2. Timestamp Field

This is a timestamp for all events and alarms on the door lock in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day of the event.

5.2.7.5.3. Event Type Field

This indicates the type of event that took place on the door lock.

Value	Name
0	Operation
1	Programming
2	Alarm

5.2.7.5.4. Source Field

This is a source value where available sources are (see Operation Event Sources).

Value	Name
0x00	Keypad
0x01	Remote
0x02	Manual
0x03	RFID
0xFF	Indeterminate

If the Event type is 0x02 (Alarm) then the source SHOULD be, but does not have to be 0xFF (Indeterminate).

5.2.7.5. Event ID Field

This indicates the type of event that took place on the door lock depending on the event code table provided for a given event type and source. See Operation Event Codes.

5.2.7.5.6. User ID Field

This indicates the ID of the user who generated the event on the door lock if one is available. Otherwise, the value is 0xFFFF.

5.2.7.5.7. PIN Field

This is a string indicating the PIN code or RFID code that was used to create the event on the door lock if one is available.

5.2.7.6. Set PIN Code Command

Set a PIN Code into the lock.

Id	Field	Type	Constraint	Quality	Default	Confor- mance
0	User ID	uint16	desc			M
1	User Status	uint8	UserStatusEnum	X	OccupiedEnabled	M
2	User Type	enum8	UserTypeEnum	X	UnrestrictedUser	M
3	PIN	octstr				M

User ID is between 0 - [# of PIN Users Supported attribute]. Only the values 1 (Occupied/Enabled) and 3 (Occupied/Disabled) are allowed for User Status.

Return status is a global status code or a cluster-specific status code from the [DoorLockStatus](#) table and SHALL be one of the following values:

Name	Summary
SUCCESS	Setting PIN code was successful.
FAILURE	Setting PIN code failed.
CONSTRAINT_ERROR	Setting PIN code failed because User ID requested was out of range.
DUPLICATE	Setting PIN code failed because it would create a duplicate PIN code.

5.2.7.7. Get PIN Code Command

Retrieve a PIN Code. User ID is between 0 - [# of PIN Users Supported attribute].

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M

5.2.7.8. Get PIN Code Response Command

Returns the PIN for the specified user ID.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M
1	User Status	uint8	UserStatusEnum	X	Available	M
2	User Type	enum8	UserTypeEnum	X		M
3	PIN Code	octstr		X	empty	M

If the requested User ID is valid and the Code doesn't exist, Get RFID Code Response SHALL have the following format:

User ID = requested User ID

UserStatus = 0 (available)

UserType = 0xFF (not supported)

PIN Code = 0 (zero length)

If the requested User ID is invalid, send Default Response with an error status. The error status SHALL be equal to CONSTRAINT_ERROR when User_ID is less than the max number of users supported, and NOT_FOUND if greater than or equal to the max number of users supported.

5.2.7.9. Clear PIN Code Command

Clear a PIN code or all PIN codes.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	PINSlotIndex	uint16	1 to NumberOfPINUsersSupported, 0xFFFF			M

† The User ID is an obsolete field name, use PINSlotIndex instead.

For each PIN Code cleared whose user doesn't have a RFID Code or other credential type, then cor-

5.2.7.9.1. PINSlotIndex

Specifies a valid PIN code slot index or 0xFFFF to indicate all PIN code slots SHALL be cleared.

5.2.7.10. Clear All PIN Codes Command

Clear out all PINs on the lock.

Note: On the server, the clear all PIN codes command SHOULD have the same effect as the Clear PIN Code command with respect to the setting of user status, user type and schedules.

5.2.7.11. Set User Status Command

Set the status of a user ID. User Status value of 0 is not allowed. In order to clear a user id, the Clear ID Command SHALL be used. For user status value please refer to User Status Value.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M
1	User Status	uint8	UserStatusEnum			M

5.2.7.12. Get User Status Command

Get the status of a user.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M

5.2.7.13. Get User Status Response Command

Returns the user status for the specified user ID. If the requested User ID is invalid, the Status field SHALL be set to FAILURE. If the command is successful, the Status field SHALL be set to SUCCESS.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M
1	User Status	enum8	UserStatusEnum			M

5.2.7.14. Set Week Day Schedule Command

Set a weekly repeating schedule for a specified user.

Id	Name	Type	Constraint	Quality	Default	Conformance
0	WeekDayIndex	uint8	1 to NumberOfWeekDaySchedulesSupportedPerUser			M
	Schedule ID†					
1	UserIndex	uint16	1 to NumberOfTotalUsersSupported			M
	User ID†					
2	DaysMask	DaysMaskMap				M
3	StartHour	uint8	0 to 23			M
4	StartMinute	uint8	0 to 59			M
5	EndHour	uint8	0 to 23			M
6	EndMinute	uint8	0 to 59			M

† The Schedule ID and User ID are obsolete field names, use WeekDayIndex and UserIndex instead, respectively.

The associated UserType MAY be changed to ScheduleRestrictedUser by the lock when a Week Day schedule is set.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Setting Week Day schedule was successful.
FAILURE	Some unexpected internal error occurred setting Week Day schedule.
INVALID_COMMAND	<p>One or more fields violates constraints or is invalid.</p> <p>Door lock is unable to set Week Day schedule for more than one day in DaysMask map (e.g. need to use separate schedules for each day).</p>

5.2.7.14.1. StartHour

This indicates the starting hour for the Week Day schedule.

5.2.7.14.2. StartMinute

This indicates the starting minute for the Week Day schedule.

5.2.7.14.3. EndHour

The ending hour for the Week Day schedule. EndHour SHALL be equal to or greater than StartHour.

5.2.7.14.4. EndMinute

The ending minute for the Week Day schedule. If EndHour is equal to StartHour then EndMinute SHALL be greater than StartMinute.

If the EndHour is equal to 23 and the EndMinute is equal to 59 the Lock SHALL grant access to the user up until 23:59:59.

5.2.7.15. Get Week Day Schedule Command

Retrieve the specific weekly schedule for the specific user.

ID	Field	Type	Constraint	Quality	Default	Conformance
0	WeekDayIndex Schedule ID†	uint8	1 to NumberOfWeek-DaySchedulesSupportedPerUser			M
1	UserIndex User ID†	uint16	1 to NumberOfTotalUsersSupported			M

† The Schedule ID and User ID are obsolete field names, use WeekDayIndex and UserIndex instead, respectively.

5.2.7.16. Get Week Day Schedule Response Command

Returns the weekly repeating schedule data for the specified schedule index.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	WeekDayIndex Schedule ID†	uint8	1 to NumberOfWeek-DaySchedulesSupportedPerUser			M
1	UserIndex User ID†	uint16	1 to NumberOfTotalUsersSupported			M
2	Status	enum8	desc		SUCCESS	M

ID	Name	Type	Constraint	Quality	Default	Conformance
3	DaysMask	DaysMaskMap				0
4	StartHour	uint8	0 to 23			0
5	StartMinute	uint8	0 to 59			0
6	EndHour	uint8	0 to 23			0
7	EndMinute	uint8	0 to 59			0

† The Schedule ID and User ID are obsolete field names, use WeekDayIndex and UserID instead, respectively.

5.2.7.16.1. Status

Status SHALL be one of the following values:

- SUCCESS if both WeekDayIndex and UserID are valid and there is a corresponding schedule entry.
- INVALID_COMMAND if either WeekDayIndex and/or UserID values are not within valid range
- NOT_FOUND if no corresponding schedule entry found for WeekDayIndex.
- NOT_FOUND if no corresponding user entry found for UserID.

If this field is SUCCESS, the optional fields for this command SHALL be present. For other (error) status values, only the fields up to the status field SHALL be present.

5.2.7.16.2. StartHour

This indicates the starting hour for the Week Day schedule.

5.2.7.16.3. StartMinute

This indicates the starting minute for the Week Day schedule.

5.2.7.16.4. EndHour

This indicates the ending hour for the Week Day schedule. EndHour SHALL be equal to or greater than StartHour.

5.2.7.16.5. EndMinute

The ending minute for the Week Day schedule. If EndHour is equal to StartHour then EndMinute SHALL be greater than StartMinute.

5.2.7.17. Clear Week Day Schedule Command

Clear the specific weekly schedule or all weekly schedules for the specific user.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	WeekDayIndex Schedule ID†	uint8	1 to NumberOfWeek-DaySchedulesSupportedPerUser, 0xFE			M
1	UserIndex User ID†	uint16	1 to NumberOfTotalUsersSupported			M

† The Schedule ID and User ID are obsolete field names, use WeekDayIndex and UserIndex instead, respectively.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Clearing Week Day schedule(s) was successful.
FAILURE	Some unexpected internal error occurred clearing Week Day schedule.
INVALID_COMMAND	One or more fields violates constraints or is invalid.

5.2.7.17.1. WeekDayIndex

The Week Day schedule index to clear or 0xFE to clear all Week Day schedules for the specified user.

5.2.7.18. Set Year Day Schedule Command

Set a time-specific schedule ID for a specified user.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	YearDayIndex Schedule ID†	uint8	1 to NumberOfYear-DaySchedulesSupportedPerUser			M
1	UserIndex User ID†	uint16	1 to NumberOfTotalUsersSupported			M

Id	Field	Type	Constraint	Quality	Default	Conformance
2	LocalStartTime	epoch-s	all			M
3	LocalEndTime	epoch-s	all			M

† The Schedule ID and User ID are obsolete field names, use YearDayIndex and UserID instead, respectively.

The associated UserType MAY be changed to ScheduleRestrictedUser by the lock when a Year Day schedule is set.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Setting Year Day schedule was successful.
FAILURE	Some unexpected internal error occurred setting Year Day schedule.
INVALID_COMMAND	One or more fields violates constraints or is invalid.

5.2.7.18.1. LocalStartTime

The starting time for the Year Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value.

5.2.7.18.2. LocalEndTime

The ending time for the Year Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. LocalEndTime SHALL be greater than LocalStartTime.

5.2.7.19. Get Year Day Schedule Command

Retrieve the specific year day schedule for the specific schedule and user indexes.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	YearDayIndex	uint8	1 to NumberOfYearDaySchedulesSupportedPerUser			M

Id	Field	Type	Constraint	Quality	Default	Conformance
1	UserIndex User ID†	uint16	1 to NumberOfTotalUsersSupported			M

† The Schedule ID and User ID are obsolete field names, use YearDayIndex and UserIndex instead, respectively.

5.2.7.20. Get Year Day Schedule Response Command

Returns the year day schedule data for the specified schedule and user indexes.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	YearDayIndex Schedule ID†	uint8	1 to NumberOfYearDaySchedulesSupportedPerUser			M
1	UserIndex User ID†	uint16	1 to NumberOfTotalUsersSupported			M
2	Status	enum8	desc		SUCCESS	M
2	LocalStartTime	epoch-s	all			O
3	LocalEndTime	epoch-s	all			O

† The Schedule ID and User ID are obsolete field names, use YearDayIndex and UserIndex instead, respectively.

5.2.7.20.1. Status

Status SHALL be one of the following values:

- SUCCESS if both YearDayIndex and UserIndex are valid and there is a corresponding schedule entry.
- INVALID_COMMAND if either YearDayIndex and/or UserIndex values are not within valid range
- NOT_FOUND if no corresponding schedule entry found for YearDayIndex.
- NOT_FOUND if no corresponding user entry found for UserIndex.

If this field is SUCCESS, the optional fields for this command SHALL be present. For other (error)

status values, only the fields up to the status field SHALL be present.

5.2.7.20.2. LocalStartTime

The starting time for the Year Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. This SHALL be null if the schedule is not set for the YearDayIndex and UserIndex provided.

5.2.7.20.3. LocalEndTime

The ending time for the Year Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. LocalEndTime SHALL be greater than LocalStartTime. This SHALL be null if the schedule is not set for the YearDayIndex and UserIndex provided.

5.2.7.21. Clear Year Day Schedule Command

Clears the specific year day schedule or all year day schedules for the specific user.

Id	Field	Type	Constraint	Quality	Default	Confor- mance
0	YearDayIndex Schedule ID†	uint8	1 to NumberOfYearDaySchedulesSupportedPerUser, 0xFE			M
1	UserIndex User ID†	uint16	1 to NumberOfTotalUsersSupported			M

† The Schedule ID and User ID are obsolete field names, use YearDayIndex and UserIndex instead, respectively.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Clearing Year Day schedule(s) was successful.
FAILURE	Some unexpected internal error occurred clearing Year Day schedule.
INVALID_COMMAND	One or more fields violates constraints or is invalid.

5.2.7.21.1. YearDayIndex

The Year Day schedule index to clear or 0xFE to clear all Year Day schedules for the specified user.

5.2.7.22. Set Holiday Schedule Command

Set the holiday Schedule by specifying local start time and local end time with respect to any Lock Operating Mode.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	HolidayIndex Holiday Schedule ID†	uint8	1 to NumberOfHolidaysSupported			M
1	LocalStartTime	epoch-s	all			M
2	LocalEndTime	epoch-s	all			M
3	OperatingMode	OperatingModeEnum	all			M

† The Holiday Schedule ID is an obsolete field name, use HolidayIndex instead.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Setting Holiday schedule was successful.
FAILURE	Some unexpected internal error occurred setting Holiday schedule.
INVALID_COMMAND	One or more fields violates constraints or is invalid.

5.2.7.22.1. LocalStartTime

The starting time for the Holiday Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value.

5.2.7.22.2. LocalEndTime

The ending time for the Holiday Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. LocalEndTime SHALL be greater than LocalStartTime.

5.2.7.22.3. OperatingMode

The operating mode to use during this Holiday schedule start/end time.

5.2.7.23. Get Holiday Schedule Command

Get the holiday schedule for the specified index.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	HolidayIndex Holiday Schedule ID†	uint8	1 to NumberOfHolidaySchedulesSupported			M

† The Holiday Schedule ID is an obsolete field name, use HolidayIndex instead.

5.2.7.24. Get Holiday Schedule Response Command

Returns the Holiday Schedule Entry for the specified Holiday ID.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	HolidayIndex Holiday Schedule ID†	uint8	1 to NumberOfHolidaySchedulesSupported			M
1	Status	enum8	desc		SUCCESS	M
2	LocalStartTime	epoch-s	all	X		O
3	Local End Time	epoch-s	all	X		O
4	Operating-Mode	Operating-ModeEnum	all	X		O

† The Holiday Schedule ID is an obsolete field name, use HolidayIndex instead.

5.2.7.24.1. Status

Status SHALL be one of the following values:

- FAILURE if the attribute NumberOfHolidaySchedulesSupported is zero.
- SUCCESS if the HolidayIndex is valid and there is a corresponding schedule entry.
- INVALID_COMMAND if the HolidayIndex is not within valid range
- NOT_FOUND if the HolidayIndex is within the valid range, however, there is not corresponding schedule entry found.

If this field is SUCCESS, the optional fields for this command SHALL be present. For other (error) status values, only the fields up to the status field SHALL be present.

5.2.7.24.2. LocalStartTime

The starting time for the Holiday schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. This SHALL be null if the schedule is not set for the HolidayIndex provided.

5.2.7.24.3. LocalEndTime

The ending time for the Holiday schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. LocalEndTime SHALL be greater than LocalStartTime. This SHALL be null if the schedule is not set for the HolidayIndex provided.

5.2.7.24.4. OperatingMode

The operating mode to use during this Holiday schedule start/end time. This SHALL be null if the schedule is not set for the HolidayIndex provided.

5.2.7.25. Clear Holiday Schedule Command

Clears the holiday schedule or all holiday schedules.

Id	Field	Type	Constraint	Quality	Default	Confor- mance
0	HolidayIndex Holiday Schedule ID†	uint8	1 to NumberOfHolidaySchedulesSupported, 0xFE			M

† The Holiday Schedule ID is an obsolete field name, use HolidayIndex instead.

5.2.7.25.1. HolidayIndex

The Holiday schedule index to clear or 0xFE to clear all Holiday schedules.

5.2.7.26. Set User Type Command

Set the user type for a specified user.

For user type value please refer to User Type Value.

Id	Field	Type	Constraint	Quality	Default	Confor- mance
0	User ID	uint16	desc			M

Id	Field	Type	Constraint	Quality	Default	Conformance
1	User Type	enum8	UserType-Enum			M

If UserType is currently YearDayScheduleUser, WeekDayScheduleUser, or ScheduleRestrictedUser and the new UserType is UnrestrictedUser then all existing Year Day and/or Week Day schedules SHALL be ignored or disabled (if this transition is supported by the door lock). If UserType is ScheduleRestrictedUser and the new UserType is ScheduleRestrictedUser then all existing Year Day and/or Week Day schedules SHALL be applied or enabled.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Setting User Type was successful.
FAILURE	Some unexpected internal error occurred setting User Type.
INVALID_COMMAND	<p>One or more fields violates constraints or is invalid.</p> <p>Door lock is unable to switch from restricted to unrestricted user (e.g. need to clear schedules to switch).</p>

5.2.7.27. Get User Type Command

Retrieve the user type for a specific user.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M

5.2.7.28. Get User Type Response Command

Returns the user type for the specified user ID. If the requested User ID is invalid, send Default Response with an error status equal to FAILURE.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M
1	User Type	enum8	UserType-Enum			M

5.2.7.29. Set RFID Code Command

Set an ID for RFID access into the lock.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M
1	User Status	uint8	UserStatusEnum	X	OccupiedEnabled	M
2	User Type	enum8	UserTypeEnum	X	UnrestrictedUser	M
3	RFID Code	octstr				M

User ID: Between 0 - [# of RFID Users Supported attribute]. Only the values 1 (Occupied/Enabled) and 3 (Occupied/Disabled) are allowed for User Status.

User Status: Used to indicate what the status is for a specific user ID. The values are according to “Set PIN” while not all are supported.

Value	User Status Byte	Conformance
1	Occupied / Enabled (Access Given)	M
3	Occupied / Disabled	M

User Type: The values are the same as used for “Set PIN Code.”

Return status is a global status code or a cluster-specific status code from the [DoorLockStatus](#) table and SHALL be one of the following values:

Name	Summary
SUCCESS	Setting RFID code was successful.
FAILURE	Setting RFID code failed.
CONSTRAINT_ERROR	Setting RFID code failed because User ID requested was out of range.
DUPLICATE	Setting RFID code failed because it would create a duplicate RFID code.

5.2.7.30. Get RFID Code Command

Retrieve an RFID code. User ID is between 0 - [# of RFID Users Supported attribute].

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M

5.2.7.31. Get RFID Code Response Command

Returns the RFID code for the specified user ID.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	User ID	uint16	desc			M
1	User Status	enum8	UserStatusEnum	X	Available	M
2	User Type	enum8	UserTypeEnum	X		M
3	RFID Code	octstr		X	empty	M

If the requested User ID is valid and the Code doesn't exist, Get RFID Code Response SHALL have the following format:

User ID = requested User ID

UserStatus = 0 (available)

UserType = 0xFF (not supported)

RFID Code = 0 (zero length)

If requested User ID is invalid, send Default Response with an error status. The error status SHALL be equal to CONSTRAINT_ERROR when User_ID is less than the max number of users supported, and NOT_FOUND if greater than or equal to the max number of users supported.

5.2.7.32. Clear RFID Code Command

Clear an RFID code or all RFID codes.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	RFIDSlotIndex	uint16	1 to NumberOfRFIDUsersSupported, 0xFFFFE			M

† The User ID is an obsolete field name, use RFIDSslotIndex instead.

For each RFID Code cleared whose user doesn't have a PIN Code or other credential type, then the corresponding user record's UserStatus value SHALL be set to Available, and UserType value SHALL be set to UnrestrictedUser and all schedules SHALL be cleared.

5.2.7.32.1. RFIDSslotIndex

Specifies a valid RFID code slot index or 0xFFFFE to indicate all RFID code slots SHALL be cleared.

5.2.7.33. Clear All RFID Codes Command

Clear out all RFIDs on the lock. If you clear all RFID codes and this user didn't have a PIN code, the user status has to be set to "0 Available", the user type has to be set to the default value, and all schedules which are supported have to be set to the default values.

5.2.7.34. Set User Command

Set User into the lock.

Id	Field	Type	Constraint	Quality	Default	Confor- mance
0	Opera- tionType	DataOpera- tionType- Enum	Add, Modify			M
1	UserIndex	uint16	1 to Num- berOfTo- talUsersSup- ported			M
2	UserName	string	max 10	X	empty	M
3	UserUnique ID	uint32	all	X	0xFFFFFFFF	M
4	UserStatus	UserSta- tusEnum	OccupiedEn- abled, Occu- piedDisabled	X	OccupiedEn- abled	M
5	UserType	UserType- Enum	Unrestrict- edUser, NonAcces- sUser, Force- dUser, Dis- posableUser, ExpiringUser , ScheduleRe- strictedUser, RemoteOn- lyUser	X	Unrestrict- edUser	M
6	Credential- Rule	Credential- RuleEnum		X	Single	M

Fields used for different use cases:

Use Case	Description
Create a new user record	<ul style="list-style-type: none"> • OperationType SHALL be set to Add. • UserIndex value SHALL be set to a user record with UserType set to Available. • UserName MAY be null causing new user record to use empty string for UserName otherwise UserName SHALL be set to the value provided in the new user record. • UserUniqueID MAY be null causing new user record to use 0xFFFFFFFF for UserUniqueID otherwise UserUniqueID SHALL be set to the value provided in the new user record. • UserStatus MAY be null causing new user record to use OccupiedEnabled for UserStatus otherwise UserStatus SHALL be set to the value provided in the new user record. • UserType MAY be null causing new user record to use UnrestrictedUser for UserType otherwise UserType SHALL be set to the value provided in the new user record. • CredentialRule MAY be null causing new user record to use Single for CredentialRule otherwise CredentialRule SHALL be set to the value provided in the new user record. <p>CreatorFabricIndex and LastModifiedFabricIndex in the new user record SHALL be set to the accessing fabric index.</p> <p>A LockUserChange event SHALL be generated after successfully creating a new user.</p>

Use Case	Description
Modify an existing user record	<ul style="list-style-type: none"> • OperationType SHALL be set to Modify. • UserIndex value SHALL be set for a user record with UserType NOT set to Available. • UserName SHALL be null if modifying a user record that was not created by the accessing fabric. • INVALID_COMMAND SHALL be returned if UserName is not null and the accessing fabric index doesn't match the CreatorFabricIndex in the user record otherwise UserName SHALL be set to the value provided in the user record. • UserUniqueID SHALL be null if modifying the user record that was not created by the accessing fabric. • INVALID_COMMAND SHALL be returned if UserUniqueID is not null and the accessing fabric index doesn't match the CreatorFabricIndex in the user record otherwise UserUniqueID SHALL be set to the value provided in the user record. • UserStatus MAY be null causing no change to UserStatus in user record otherwise UserStatus SHALL be set to the value provided in the user record. • UserType MAY be null causing no change to UserType in user record otherwise UserType SHALL be set to the value provided in the user record. • CredentialRule MAY be null causing no change to CredentialRule in user record otherwise CredentialRule SHALL be set to the value provided in the user record. <p>CreatorFabricIndex SHALL NOT be changed in the user record. LastModifiedFabricIndex in the new user record SHALL be set to the accessing fabric index.</p> <p>A LockUserChange event SHALL be generated after successfully modifying a user.</p>

Return status is a global status code or a cluster-specific status code from the [DoorLockStatus](#) table

and SHALL be one of the following values:

- SUCCESS, if setting User was successful.
- FAILURE, if some unexpected internal error occurred setting User.
- OCCUPIED, if OperationType is Add and UserIndex points to an occupied slot.
- INVALID_COMMAND, if one or more fields violate constraints or are invalid or if OperationType is Modify and UserIndex points to an available slot.

5.2.7.34.1. UserName

A string to use as a human readable identifier for the user.

If UserName is null then:

- If the OperationType is Add, the UserName in the resulting user record SHALL be set to an empty string.
- If the OperationType is Modify, the UserName in the user record SHALL NOT be changed from the current value.

If UserName is not null, the UserName in the user record SHALL be set to the provided value.

5.2.7.34.2. UserUniqueID

A fabric assigned number to use for connecting this user to other users on other devices from the fabric's perspective.

If UserUniqueID is null then:

- If the OperationType is Add, the UserUniqueID in the resulting user record SHALL be set to default value specified above.
- If the OperationType is Modify, the UserUniqueID in the user record SHALL NOT be changed from the current value.

If UserUniqueID is not null, the UserUniqueID in the user record SHALL be set to the provided value.

5.2.7.34.3. UserStatus

A UserStatus to assign to this user when created or modified.

If UserStatus is null then:

- If the OperationType is Add, the UserStatus in the resulting user record SHALL be set to default value specified above.
- If the OperationType is Modify, the UserStatus in the user record SHALL NOT be changed from the current value.

If UserStatus is not null, the UserStatus in the user record SHALL be set to the provided value.

5.2.7.34.4. UserType

A UserType to assign to this user when created or modified.

If UserType is null then:

- If the OperationType is Add, the UserType in the resulting user record SHALL be set to default value specified above.
- If the OperationType is Modify, the UserType in the user record SHALL NOT be changed from the current value.

If UserType is not null, the UserType in the user record SHALL be set to the provided value.

5.2.7.34.5. CredentialRule

The CredentialRule to use for this user.

The valid CredentialRule enumeration values depends on the bits in the [CredentialRulesSupport](#) map. Each bit in the map identifies a valid CredentialRule that can be used.

If CredentialRule is null then:

- If the OperationType is Add, the CredentialRule in the resulting user record SHALL be set to default value specified above.
- If the OperationType is Modify, the CredentialRule in the user record SHALL NOT be changed from the current value.

If CredentialRule is not null, the CredentialRule in the user record SHALL be set to the provided value.

5.2.7.35. Get User Command

Retrieve User.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	UserIndex	uint16	1 to NumberOfTotalUsersSupported			M

An InvokeResponse command SHALL be sent with an appropriate error (e.g. FAILURE, INVALID_COMMAND, etc.) as needed otherwise the [Get User Response Command](#) SHALL be sent implying a status of SUCCESS.

5.2.7.36. Get User Response Command

Returns the User for the specified UserIndex.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	UserIndex	uint16	1 to NumberOfTotalUsersSupported			M
1	UserName	string	max 10	X	empty	M
2	UserUniqueID	uint32	all	X	0	M
3	UserStatus	UserStatusEnum	all	X	Available	M
4	UserType	UserTypeEnum	all	X	UnrestrictedUser	M
5	Credential-Rule	Credential-RuleEnum	desc	X	Single	M
6	Credentials	list[CredentialStruct]	0 to NumberOfCredentialsSupportedPerUser	X		M
7	Creator-FabricIndex	fabric-idx	all	X		M
8	LastModified-FabricIndex	fabric-idx	all	X		M
9	NextUserIndex	uint16	1 to NumberOfTotalUsersSupported	X		M

If the requested UserIndex is valid and the UserStatus is Available for the requested UserIndex then UserName, UserUniqueID, UserStatus, UserType, CredentialRule, Credentials, CreatorFabricIndex, and LastModifiedFabricIndex SHALL all be null in the response.

5.2.7.36.1. CreatorFabricIndex

The user's creator fabric index. CreatorFabricIndex SHALL be null if UserStatus is set to Available or when the creator fabric cannot be determined (for example, when user was created outside the Interaction Model) and SHALL NOT be null otherwise. This value SHALL be set to **0** if the original creator fabric was deleted.

5.2.7.36.2. LastModifiedFabricIndex

The user's last modifier fabric index. LastModifiedFabricIndex SHALL be null if UserStatus is set to Available or when the modifier fabric cannot be determined (for example, when user was modified

5.2.7.36.3. NextUserIndex

The next occupied UserIndex in the database which is useful for quickly identifying occupied user slots in the database. This SHALL NOT be null if there is at least one occupied entry after the requested UserIndex in the User database and SHALL be null if there are no more occupied entries.

5.2.7.37. Clear User Command

Clears a User or all Users.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	UserIndex	uint16	1 to NumberOfTotalUsersSupported, 0xFFFF			M

For each User to clear, all associated credentials (e.g. PIN, RFID, fingerprint, etc.) SHALL be cleared and the User entry values SHALL be reset to their default values (e.g. UserStatus SHALL be Available, UserType SHALL be UnrestrictedUser) and all associated schedules SHALL be cleared.

A LockUserChange event with the provided UserIndex SHALL be generated after successfully clearing users.

5.2.7.37.1. UserIndex

Specifies a valid User index or 0xFFFF to indicate all User slots SHALL be cleared.

5.2.7.38. Operation Event Notification Command

The door lock server sends out operation event notification when the event is triggered by the various event sources. The specific operation event will only be sent out if the associated bitmask is enabled in the various attributes in the Event Masks Attribute Set.

All events are optional.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Operation Event Source	uint8	desc			M
1	Operation Event Code	uint8	desc			M
2	User ID	uint16	desc			M

Id	Field	Type	Constraint	Quality	Default	Conformance
3	PIN	octstr				M
4	LocalTime	epoch-s	all			M
5	Data	string				O

5.2.7.38.1. Operation Event Sources

This field indicates where the event was triggered from.

Value	Source
0	Keypad
1	Remote
2	Manual
3	RFID
0xFF	Indeterminate

5.2.7.38.2. Operation Event Codes

The door lock optionally sends out notifications (if they are enabled) whenever there is a significant operational event on the lock. When combined with a source from the Event Source table above, the following operational event codes constitute an event on the door lock that can be both logged and sent to a bound device using the Operation Event Notification command.

Not all operation event codes are applicable to each of the event source. The following table marks each event code with “A” if the event code is applicable to the event source.

Value	Name	Confor- mance	Applicable			
			Keypad	Remote	Manual	RFID
0	UnknownOrMfgSpecific	O	A	A	A	A
1	Lock	O	A	A	A	A
2	Unlock	O	A	A	A	A
3	LockFailureInvalidPINorRFID	O	A	A		A
4	LockFailureInvalidSchedule	O	A	A		A

Value	Name	Confor-mance	Applicable			
5	UnlockFail-ureInvalid-PINorRFID	O	A	A		A
6	UnlockFail-ureInvalid-Schedule	O	A	A		A
7	One-TouchLock	O			A	
8	KeyLock	O			A	
9	KeyUnlock	O			A	
10	AutoLock	O			A	
11	Schedule-Lock	WDSCH YDS CH			A	
12	ScheduleUn-lock	WDSCH YDS CH			A	
13	Manual Lock (Key or Thumb-turn)	O			A	
14	Manual Unlock (Key or Thumb-turn)	O			A	
15	Non-access User Operation Event	O	A			

5.2.7.38.3. User ID

The User ID who performed the event.

5.2.7.38.4. PIN

The PIN that is associated with the User ID who performed the event.

5.2.7.38.5. LocalTime

The time when the event was triggered in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. If time is not supported, the field SHALL be populated with default not used value 0xFFFFFFFF.

5.2.7.38.6. Data

The operation event notification command contains a variable string, which can be used to pass data associated with a particular event. Generally this field will be left empty. However, manufacturer can choose to use this field to store/display manufacturer-specific information.

5.2.7.38.7. Keypad Operation Event Notification

Keypad Operation Event Notification feature is enabled by setting the associated bitmasks in the [Keypad Operation Event Mask attribute].

5.2.7.38.8. Remote Operation Event Notification

Remote Operation Event Notification feature is enabled by setting the associated bitmasks in the [Remote Operation Event Mask attribute].

5.2.7.38.9. Manual Operation Event Notification

Manual Operation Event Notification feature is enabled by setting the associated bitmasks in the [Manual Operation Event Mask attribute] attribute.

5.2.7.38.10. RFID Operation Event Notification

RFID Operation Event Notification feature is enabled by setting the associated bitmasks in the [RFID Operation Event Mask attribute].

5.2.7.39. Programming Event Notification Command

The door lock server sends out a programming event notification whenever a programming event takes place on the door lock.

As with operational events, all programming events can be turned on and off by flipping bits in the associated event mask.

The programming event notification command includes an optional string of data that can be used by the manufacturer to pass some manufacturer-specific information if that is required.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Program Event Source	uint8	desc			M
1	Program Event Code	uint8	desc			M
2	User ID	uint16	desc			M
3	PIN	octstr				M
4	User Type	enum8	desc			M
5	User Status	enum8	desc			M

Id	Field	Type	Constraint	Quality	Default	Conformance
6	LocalTime	epoch-s	all			M
7	Data	string				O

5.2.7.39.1. Operation Event Sources

This field indicates where the event was triggered from.

Value	Source
0	Keypad
1	Remote
2	Reserved (Manual in Operation Event)
3	RFID
0xFF	Indeterminate

5.2.7.39.2. Programming Event Codes

The door lock optionally sends out notifications (if they are enabled) whenever there is a significant programming event on the lock. When combined with a source from the Event Source table above, the following programming event codes constitute an event on the door lock that can be both logged and sent to a bound device using the Programming Event Notification command.

Not all event codes are applicable to each of the event source. The following table marks each event code with “A” if the event code is applicable to the event source.

Value	Programming Event Code	Keypad	Remote	RFID
0	UnknownOrMfgSpecific	A	A	A
1	ProgrammingCodeChanged	A		
2	PINCodeAdded	A	A	
3	PINCodeCleared	A	A	
4	PINCodeChanged	A	A	
5	RFIDCodeAdded			A
6	RFIDCodeCleared			A

5.2.7.39.3. User ID

The User ID who performed the event

5.2.7.39.4. PIN

The PIN that is associated with the User ID who performed the event

5.2.7.39.5. User Type

The User Type that is associated with the User ID who performed the event

5.2.7.39.6. User Status

The User Status that is associated with the User ID who performed the event

5.2.7.39.7. LocalTime

The time when the event was triggered in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. If time is not supported, the field SHALL be populated with default not used value 0xFFFFFFFF.

5.2.7.39.8. Data

The programming event notification command contains a variable string, which can be used to pass data associated with a particular event. Generally this field will be left empty. However, manufacturer can choose to use this field to store/display manufacturer-specific information.

5.2.7.39.9. Keypad Programming Event Notification

Keypad Programming Event Notification feature is enabled by setting the associated bitmasks in the [Keypad Programming Event Mask attribute].

5.2.7.39.10. Remote Programming Event Notification

Remote Programming Event Notification feature is enabled by setting the associated bitmasks in the [Remote Programming Event Mask attribute].

5.2.7.39.11. RFID Programming Event Notification

RFID Programming Event Notification feature is enabled by setting the associated bitmasks in the [RFID Programming Event Mask attribute].

5.2.7.40. Set Credential Command

Set a credential (e.g. PIN, RFID, Fingerprint, etc.) into the lock for a new user, existing user, or ProgrammingUser.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	OperationType	DataOperationTypeEnum	Add, Modify			M
1	Credential	Credential-Struct				M

Id	Field	Type	Constraint	Quality	Default	Conformance
2	Credential-Data	octstr	desc			M
3	UserIndex	uint16	1 to NumberOfTotalUsersSupported	X		M
4	UserStatus	UserStatusEnum	OccupiedEnabled, OccupiedDisabled	X	OccupiedEnabled	M
5	UserType	UserTypeEnum	UnrestrictedUser, ProgrammingUser, NonAccessUser, ForcedUser, DisposableUser, ExpiringUser, RemoteOnlyUser	X	UnrestrictedUser	M

Fields used for different use cases:

Use Case	Description
Create a new credential and a new user record	<ul style="list-style-type: none"> • OperationType SHALL be set to Add. • UserIndex SHALL be set to null and the lock will find a user record with a UserStatus value of Available and associate its UserIndex with the CredentialIndex in CredentialStruct provided. • CredentialIndex in CredentialStruct SHALL be for an unoccupied credential slot. • UserStatus MAY be null. If it is null, the new user record SHALL have UserStatus set to OccupiedEnabled. Otherwise the new user record SHALL have UserStatus set to the provided value. • UserType MAY be null. If it is null, the new user record SHALL have UserType set to UnrestrictedUser. Otherwise the new user record SHALL have UserType set to the provided value. • UserType SHALL NOT be set to ProgrammingUser for this use case. <p>CreatorFabricIndex and LastModifiedFabricIndex in new user and credential records SHALL be set to the accessing fabric index.</p> <p>A LockUserChange event SHALL be generated after successfully creating a new credential and a new user. The UserIndex of this LockUserChange event SHALL be the UserIndex that was used to create the user. The DataIndex of this LockUserChange event SHALL be the CredentialIndex that was used to create the credential.</p>

Use Case	Description
Add a new credential to existing user record	<ul style="list-style-type: none"> • OperationType SHALL be set to Add. • UserIndex SHALL NOT be null and SHALL NOT already be associated with the CredentialIndex in CredentialStruct provided otherwise INVALID_COMMAND status response SHALL be returned. • INVALID_COMMAND SHALL be returned if the accessing fabric index doesn't match the CreatorFabricIndex in the user record pointed to by UserIndex. • CredentialIndex in CredentialStruct provided SHALL be for an available credential slot. • UserStatus SHALL be null. • UserType SHALL be null. <p>CreatorFabricIndex SHALL NOT be changed in the user record. LastModifiedFabricIndex in the user record SHALL be set to the accessing fabric index.</p> <p>CreatorFabricIndex and LastModifiedFabricIndex in the new credential record SHALL be set to the accessing fabric index.</p> <p>A LockUserChange event SHALL be generated after successfully adding a new credential.</p>

Use Case	Description
Modify credential for an existing user record	<ul style="list-style-type: none"> • OperationType SHALL be set to Modify. • UserIndex value SHALL already be associated with the CredentialIndex in CredentialStruct provided otherwise INVALID_COMMAND status response SHALL be returned. • INVALID_COMMAND SHALL be returned if the accessing fabric index doesn't match the CreatorFabricIndex in the user record pointed to by UserIndex. • INVALID_COMMAND SHALL be returned if the accessing fabric index doesn't match the CreatorFabricIndex in the credential record pointed to by the CredentialIndex field value of the Credential parameter. • CredentialIndex in CredentialStruct provided SHALL be for an occupied credential slot • UserStatus SHALL be null. • UserType SHALL be null. <p>CreatorFabricIndex SHALL NOT be changed in user and credential records. LastModifiedFabricIndex in user and credential records SHALL be set to the accessing fabric index.</p> <p>A LockUserChange event SHALL be generated after successfully modifying a credential.</p>

Use Case	Description
Modify credential for a Programming User	<ul style="list-style-type: none"> • OperationType SHALL be set to Modify. • UserIndex SHALL be null. • INVALID_COMMAND SHALL be returned if the accessing fabric index doesn't match the CreatorFabricIndex in the credential record pointed to by the CredentialIndex field value of the Credential parameter. • CredentialType in CredentialStruct SHALL be set to ProgrammingPIN. • CredentialIndex in CredentialStruct SHALL be 0. • UserStatus SHALL be null. • UserType SHALL be set to ProgrammingUser. <p>CreatorFabricIndex SHALL NOT be changed in the credential record. LastModifiedFabricIndex in the credential record SHALL be set to the accessing fabric index.</p> <p>A LockUserChange event SHALL be generated after successfully modifying a ProgrammingUser PIN code.</p>

5.2.7.40.1. OperationType

The set credential operation type requested.

5.2.7.40.2. Credential

A credential structure that contains the [CredentialTypeEnum](#) and the credential index (if applicable or 0 if not) to set.

5.2.7.40.3. CredentialData

The credential data to set for the credential being added or modified. The length of the credential data SHALL conform to the limits of the CredentialType specified in the Credential structure otherwise an INVALID_COMMAND status SHALL be returned in the [Set Credential Response Command](#).

5.2.7.40.4. UserIndex

The user index to the user record that corresponds to the credential being added or modified. This SHALL be null if OperationType is add and a new credential and user is being added at the same time.

5.2.7.40.5. UserStatus

The user status to use in the new user record if a new user is being created. This SHALL be null if OperationType is Modify. This MAY be null when adding a new credential and user.

5.2.7.40.6. UserType

The user type to use in the new user record if a new user is being created. This SHALL be null if OperationType is Modify. This MAY be null when adding a new credential and user.

5.2.7.41. Set Credential Response Command

Returns the status for setting the specified credential.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Status	status	desc			M
1	UserIndex	uint16	1 to NumberOfTotalUsersSupported	X	0	M
2	NextCredentialIndex	uint16	desc	X		O

5.2.7.41.1. Status

Status comes from the [DoorLockStatus](#) table and SHALL be one of the following values:

- SUCCESS, if setting user credential was successful.
- FAILURE, if some unexpected internal error occurred setting user credential.
- OCCUPIED, if OperationType is Add and CredentialIndex in Credential structure points to an occupied slot.
- OCCUPIED, if OperationType is Modify and CredentialIndex in Credential structure does not match the CredentialIndex that is already associated with the provided UserIndex.
- DUPLICATE, if CredentialData provided is a duplicate of another credential (e.g. duplicate PIN code).
- RESOURCE_EXHAUSTED, if OperationType is Add and the user referred to by UserIndex already has NumberOfCredentialsSupportedPerUser credentials associated.
- INVALID_COMMAND, if one or more fields violate constraints or are invalid or if OperationType is Modify and UserIndex points to an available slot.

5.2.7.41.2. UserIndex

The user index that was created with the new credential. If the status being returned is not success then this SHALL be null. This SHALL be null if OperationType was Modify; if the OperationType was Add and a new User was created this SHALL NOT be null and SHALL provide the UserIndex

5.2.7.41.3. NextCredentialIndex

The next available index in the database for the credential type set, which is useful for quickly identifying available credential slots in the database. This SHALL NOT be null if there is at least one available entry after the requested credential index in the corresponding database and SHALL be null if there are no more available entries. The NextCredentialIndex reported SHALL NOT exceed the maximum number of credentials for a particular credential type.

5.2.7.42. Get Credential Status Command

Retrieve the status of a particular credential (e.g. PIN, RFID, Fingerprint, etc.) by index.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Credential	Credential-Struct				M

An InvokeResponse command SHALL be sent with an appropriate error (e.g. FAILURE, INVALID_COMMAND, etc.) as needed otherwise the [Get Credential Status Response Command](#) SHALL be sent implying a status of SUCCESS.

5.2.7.42.1. Credential

A credential structure that contains the [CredentialTypeEnum](#) and the credential index (if applicable or 0 if not) to retrieve the status for.

5.2.7.43. Get Credential Status Response Command

Returns the status for the specified credential.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	CredentialExists	bool	all			M
1	UserIndex	uint16	1 to NumberOfTotalUsersSupported	X		M
2	Creator-FabricIndex	fabric-idx	all	X		M
3	LastModified-FabricIndex	fabric-idx	all	X		M

Id	Field	Type	Constraint	Quality	Default	Conformance
4	NextCredentialIndex	uint16	desc	X		O

5.2.7.43.1. CredentialExists

A boolean value indicating the requested credential type and index exists and is populated for a given user index.

5.2.7.43.2. UserIDex

The credential's corresponding user index value if the credential exists. If CredentialType requested was ProgrammingPIN then UserIDex SHALL be null; otherwise, UserIDex SHALL be null if CredentialExists is set to False and SHALL NOT be null if CredentialExists is set to True.

5.2.7.43.3. CreatorFabricIndex

The credential's creator fabric index. CreatorFabricIndex SHALL be null if CredentialExists is set to False or when the creator fabric cannot be determined (for example, when credential was created outside the Interaction Model) and SHALL NOT be null otherwise. This value SHALL be set to **0** if the original creator fabric was deleted.

5.2.7.43.4. LastModifiedFabricIndex

The credential's last modifier fabric index. LastModifiedFabricIndex SHALL be null if CredentialExists is set to False or when the modifier fabric cannot be determined (for example, when credential was modified outside the Interaction Model) and SHALL NOT be null otherwise. This value SHALL be set to **0** if the last modifier fabric was deleted.

5.2.7.43.5. NextCredentialIndex

The next occupied index in the database for the credential type requested, which is useful for quickly identifying occupied credential slots in the database. This SHALL NOT be null if there is at least one occupied entry after the requested credential index in the corresponding database and SHALL be null if there are no more occupied entries. The NextCredentialIndex reported SHALL NOT exceed the maximum number of credentials for a particular credential type.

5.2.7.44. Clear Credential Command

Clear one, one type, or all credentials except ProgrammingPIN credential.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Credential	Credential-Struct	desc	X		M

Fields used for different use cases:

Use Case	Description
Clear a single credential	<ul style="list-style-type: none"> CredentialType in Credential structure SHALL be set to the credential type to be cleared. CredentialType in Credential structure SHALL NOT be set to ProgrammingPIN. CredentialIndex in Credential structure SHALL be set to the credential index to be cleared. <p>A LockUserChange event SHALL be generated after successfully clearing a credential.</p>
Clear all credentials of one type	<ul style="list-style-type: none"> CredentialType in Credential structure SHALL be set to the credential type to be cleared. CredentialType in Credential structure SHALL NOT be set to ProgrammingPIN. CredentialIndex in Credential structure SHALL be set to 0xFFFF to indicate all credentials of that type SHALL be cleared. <p>A single LockUserChange event SHALL be generated after successfully clearing credentials. This event SHALL have DataIndex set to the CredentialIndex in the Credential structure.</p>
Clear all credentials of all types	<ul style="list-style-type: none"> Credential field SHALL be null. <p>The ProgrammingPIN credential SHALL NOT be cleared.</p> <p>For each credential type cleared, a LockUserChange event with the corresponding Lock-DataType SHALL be generated. This event SHALL have DataIndex set to 0xFFFF.</p>

For each credential cleared whose user doesn't have another valid credential, the corresponding user record SHALL be reset back to default values and its UserStatus value SHALL be set to Available and UserType value SHALL be set to UnrestrictedUser and all schedules SHALL be cleared. In this case a LockUserChange event SHALL be generated for the user being cleared.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Clearing credential(s) was successful.

Name	Summary
FAILURE	Some unexpected internal error occurred clearing credential(s).
INVALID_COMMAND	One or more fields violate constraints or are invalid.

5.2.7.44.1. Credential

A credential structure that contains the [CredentialTypeEnum](#) and the credential index (0xFFFF for all credentials or 0 if not applicable) to clear. This SHALL be null if clearing all credential types otherwise it SHALL NOT be null.

5.2.7.45. Unbolt Door Command

This command causes the lock device to unlock the door without pulling the latch. This command includes an optional code for the lock. The door lock MAY require a code depending on the value of the [RequirePINForRemoteOperation](#) attribute.

Note: If the attribute AutoRelockTime is supported, the lock will transition to the locked state when the auto relock time has expired.

Id	Field	Type	Constraint	Quality	Default	Conformance
0	PINCode	octstr				[COTA & PIN]

5.2.7.45.1. PINCode Field

See [PINCode Field](#).

5.2.8. Events

This cluster SHALL support these events:

Id	Name	Priority	Access	Conformance
0	DoorLockAlarm	CRITICAL	V	M
1	DoorStateChange	desc	V	DPS
2	LockOperation	desc	V	M
3	LockOperationError	desc	V	M
4	LockUserChange	INFO	V	USR

The Events specified in this cluster are not intended to define the user experience. The events are only intended to define the metadata format used to notify any nodes that have subscribed for updates.

If the DoorState reported in the DoorStateChange event is not DoorClosed then the priority SHALL

be CRITICAL; otherwise it MAY be INFO.

If the LockOperationType reported in the LockOperation event is Unlock or ForcedUserEvent then the priority SHALL be CRITICAL; otherwise it MAY be INFO.

If the OperationError reported in the LockOperationError event is DisabledUserDenied or the LockOperationType is Lock the priority SHALL be CRITICAL; otherwise it MAY be INFO.

5.2.8.1. DoorLockAlarm Event

The door lock cluster provides several alarms which can be sent when there is a critical state on the door lock. The alarms available for the door lock cluster are listed in the [AlarmCodeEnum](#) section below.

The data of this event SHALL contain the following information:

Id	Field	Type	Constraint	Quality	Default	Conformance
0	AlarmCode	Alarm-CodeEnum	all			M

5.2.8.1.1. AlarmCode

The alarm code of the event that has happened.

5.2.8.2. DoorStateChange Event

The door lock server sends out a DoorStateChange event when the door lock door state changes.

The data of this event SHALL contain the following information:

Id	Field	Type	Constraint	Quality	Default	Conformance
0	DoorState	DoorState-Enum	all			M

5.2.8.2.1. DoorState Field

The new door state for this door event.

5.2.8.3. LockOperation Event

The door lock server sends out a LockOperation event when the event is triggered by the various lock operation sources.

The data of this event SHALL contain the following information:

Id	Field	Type	Constraint	Quality	Default	Conformance
0	LockOperationType	LockOperationType-Enum	all			M
1	OperationSource	OperationSourceEnum	all			M
2	UserIndex	uint16	all	X		M
3	FabricIndex	fabric-idx	all	X		M
4	SourceNode	node-id	all	X		M
5	Credentials	list[CredentialStruct]	1 to NumberOfCredentialsSupportedPerUser	X		[USR]

- If the door lock server supports the [Unbolt Door](#) command, it SHALL generate a LockOperation event with LockOperationType set to Unlock after an Unbolt Door command succeeds.
- If the door lock server supports the [Unbolting Feature](#) and an Unlock Door command is performed, it SHALL generate a LockOperation event with LockOperationType set to Unlatch when the unlatched state is reached and a LockOperation event with LockOperationType set to Unlock when the lock successfully completes the unlock → hold latch → release latch and return to unlock state operation.
- If the command fails during holding or releasing the latch but after passing the unlocked state, the door lock server SHALL generate a LockOperationError event with LockOperationType set to Unlatch and a LockOperation event with LockOperationType set to Unlock.
 - If it fails before reaching the unlocked state, the door lock server SHALL generate only a LockOperationError event with LockOperationType set to Unlock.
- Upon manual actuation, a door lock server that supports the [Unbolting Feature](#):
 - SHALL generate a LockOperation event of LockOperationType Unlatch when it is actuated from the outside.
 - MAY generate a LockOperation event of LockOperationType Unlatch when it is actuated from the inside.

5.2.8.3.1. LockOperationType

The type of the lock operation that was performed.

5.2.8.3.2. OperationSource

The source of the lock operation that was performed.

5.2.8.3.3. UserIndex

The lock UserIndex who performed the lock operation. This SHALL be null if there is no user index that can be determined for the given operation source. This SHALL NOT be null if a user index can be determined. In particular, this SHALL NOT be null if the operation was associated with a valid credential.

5.2.8.3.4. FabricIndex

The fabric index of the fabric that performed the lock operation. This SHALL be null if there is no fabric that can be determined for the given operation source. This SHALL NOT be null if the operation source is "Remote".

5.2.8.3.5. SourceNode

The Node ID of the node that performed the lock operation. This SHALL be null if there is no Node associated with the given operation source. This SHALL NOT be null if the operation source is "Remote".

5.2.8.3.6. Credentials

The list of credentials used in performing the lock operation. This SHALL be null if no credentials were involved.

5.2.8.4. LockOperationError Event

The door lock server sends out a LockOperationError event when a lock operation fails for various reasons.

The data of this event SHALL contain the following information:

Id	Field	Type	Constraint	Quality	Default	Confor-mance
0	LockOpera-tionType	LockOpera-tionType-Enum	all			M
1	Opera-tionSource	Opera-tionSourceE-num	all			M
2	Opera-tionError	Opera-tionErrorEn-um	all			M
3	UserIndex	uint16	all	X		M
4	FabricIndex	fabric-idx	all	X		M
5	SourceNode	node-id	all	X		M

Id	Field	Type	Constraint	Quality	Default	Conformance
6	Credentials	list[CredentialStruct]	1 to NumberOfCredentialsSupportedPerUser	X		[USR]

5.2.8.4.1. LockOperationType

The type of the lock operation that was performed.

5.2.8.4.2. OperationSource

The source of the lock operation that was performed.

5.2.8.4.3. OperationError

The lock operation error triggered when the operation was performed.

5.2.8.4.4. UserIndex

The lock UserIndex who performed the lock operation. This SHALL be null if there is no user id that can be determined for the given operation source.

5.2.8.4.5. FabricIndex

The fabric index of the fabric that performed the lock operation. This SHALL be null if there is no fabric that can be determined for the given operation source. This SHALL NOT be null if the operation source is "Remote".

5.2.8.4.6. SourceNode

The Node ID of the node that performed the lock operation. This SHALL be null if there is no Node associated with the given operation source. This SHALL NOT be null if the operation source is "Remote".

5.2.8.4.7. Credentials

The list of credentials used in performing the lock operation. This SHALL be null if no credentials were involved.

5.2.8.5. LockUserChange Event

The door lock server sends out a LockUserChange event when a lock user, schedule, or credential change has occurred.

The data of this event SHALL contain the following information:

Id	Field	Type	Constraint	Quality	Default	Conformance
0	Lock- DataType	Lock- DataType- Enum	all			M
1	DataOpera- tionType	DataOpera- tionType- Enum	all			M
2	Opera- tionSource	Opera- tionSourceE num	Unspecified, Keypad, Remote			M
3	UserIndex	uint16	all	X		M
4	FabricIndex	fabric-idx	all	X		M
5	SourceNode	node-id	all	X		M
6	DataIndex	uint16	all	X		M

5.2.8.5.1. LockDataType

The lock data type that was changed.

5.2.8.5.2. DataOperationType

The data operation performed on the lock data type changed.

5.2.8.5.3. OperationSource

The source of the user data change.

5.2.8.5.4. UserIndex

The lock UserIndex associated with the change (if any). This SHALL be null if there is no specific user associated with the data operation. This SHALL be 0xFFFF if all users are affected (e.g. Clear Users).

5.2.8.5.5. FabricIndex

The fabric index of the fabric that performed the change (if any). This SHALL be null if there is no fabric that can be determined to have caused the change. This SHALL NOT be null if the operation source is "Remote".

5.2.8.5.6. SourceNode

The Node ID that that performed the change (if any). The Node ID of the node that performed the change. This SHALL be null if there was no Node involved in the change. This SHALL NOT be null if the operation source is "Remote".

5.2.8.5.7. **DataIndex**

This is the index of the specific item that was changed (e.g. schedule, PIN, RFID, etc.) in the list of items identified by LockDataType. This SHALL be null if the LockDataType does not correspond to a list that can be indexed into (e.g. ProgrammingUser). This SHALL be 0xFFFF if all indices are affected (e.g. Clear PIN Code, Clear RFID Code, Clear Week Day Schedule, Clear Year Day Schedule, etc.).

5.2.9. Data Types

5.2.9.1. **AlarmCodeEnum**

The Alarm Code enum SHALL indicate the alarm type. The data type of the Alarm Code enum is derived from enum8.

Value	Name	Conformance	Summary
0	LockJammed	M	Locking Mechanism Jammed
1	LockFactoryReset	O	Lock Reset to Factory Defaults
3	LockRadioPowerCycled	O	Lock Radio Power Cycled
4	WrongCodeEntryLimit	[USR]	Tamper Alarm - wrong code entry limit
5	FrontEscutcheonRemoved	O	Tamper Alarm - front escutcheon removed from main
6	DoorForcedOpen	[DPS]	Forced Door Open under Door Locked Condition
7	DoorAjar	[DPS]	Door ajar
8	ForcedUser	[USR]	Force User SOS alarm

5.2.9.2. **CredentialRuleEnum**

The CredentialRule enum used in various commands SHALL indicate the credential rule that can be applied to a particular user. The data type of the CredentialRule enum is derived from enum8.

Value	Name	Conformance	Summary
0	Single	USR	Only one credential is required for lock operation
1	Dual	[USR]	Any two credentials are required for lock operation

Value	Name	Conformance	Summary
2	Tri	[USR]	Any three credentials are required for lock operation

5.2.9.3. CredentialStruct

The CredentialStruct is used in LockOperation event and Get User Record Response command and SHALL indicate the credential types and their corresponding indices (if any) for the event or user record.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	Credential-Type	CredentialTypeEnum	all			M
1	CredentialIndex	uint16	all		0	M

5.2.9.3.1. CredentialType

The credential type used to authorize the lock operation.

5.2.9.3.2. CredentialIndex

This is the index of the specific credential used to authorize the lock operation in the list of credentials identified by CredentialType (e.g. schedule, PIN, RFID, etc.). This SHALL be set to 0 if CredentialType is ProgrammingPIN or does not correspond to a list that can be indexed into.

5.2.9.4. CredentialTypeEnum

The Credential Type enum SHALL indicate the credential type. The data type of the Credential Type enum is derived from enum8.

Value	Name	Conformance	Summary
0	ProgrammingPIN	O	Programming PIN code credential type
1	PIN	PIN	PIN code credential type
2	RFID	RID	RFID identifier credential type
3	Fingerprint	FGP	Fingerprint identifier credential type
4	FingerVein	FGP	Finger vein identifier credential type

Value	Name	Conformance	Summary
5	Face	FACE	Face identifier credential type

5.2.9.5. DataOperationTypeEnum

The DataOperationType enum SHALL indicate the data operation performed. The data type of the DataOperationType enum is derived from enum8.

Value	Name	Conformance	Summary
0	Add	M	Data is being added or was added
1	Clear	M	Data is being cleared or was cleared
2	Modify	M	Data is being modified or was modified

5.2.9.6. DaysMaskMap

The DaysMask field used in various commands and SHALL indicate the days of the week the Week Day schedule applies for. The data type of the DaysMask field is derived from map8.

Bit	Name	Conformance
0	Sunday	M
1	Monday	M
2	Tuesday	M
3	Wednesday	M
4	Thursday	M
5	Friday	M
6	Saturday	M

5.2.9.7. DoorStateEnum

The DoorState enumeration SHALL indicate the current door state. The data type of the DoorState enum field is derived from enum8.

Value	Name	Conformance	Summary
0	DoorOpen	DPS	Door state is open
1	DoorClosed	DPS	Door state is closed
2	DoorJammed	[DPS]	Door state is jammed
3	DoorForcedOpen	[DPS]	Door state is currently forced open

Value	Name	Conformance	Summary
4	DoorUnspecifiedError	[DPS]	Door state is invalid for unspecified reason
5	DoorAjar	[DPS]	Door state is ajar

5.2.9.8. DoorLockStatus

The cluster-specific status codes for the Door Lock cluster are as follows:

Value	Status Code	Summary
2	DUPLICATE	Entry would cause a duplicate credential/ID.
3	OCCUPIED	Entry would replace an occupied slot.

5.2.9.8.1. DUPLICATE

The provided User ID, PIN or RFID code or other credential is a duplicate of an existing entry.

5.2.9.8.2. OCCUPIED

The provided User ID, Credential ID, or entry location is already occupied. The entry might need to be deleted or a different ID or location chosen.

5.2.9.9. LockDataTypeEnum

The LockDataType enum SHALL indicate the data type that is being or has changed. The data type of the DataType enum is derived from enum8.

Value	Name	Conformance	Summary
0	Unspecified	O	Unspecified or manufacturer specific lock user data added, cleared, or modified.
1	ProgrammingCode	O	Lock programming PIN code was added, cleared, or modified.
2	UserIndex	M	Lock user index was added, cleared, or modified.
3	WeekDaySchedule	WDSCH	Lock user week day schedule was added, cleared, or modified.

Value	Name	Conformance	Summary
4	YearDaySchedule	YDSCH	Lock user year day schedule was added, cleared, or modified.
5	HolidaySchedule	HDSCH	Lock holiday schedule was added, cleared, or modified.
6	PIN	PIN	Lock user PIN code was added, cleared, or modified.
7	RFID	RID	Lock user RFID code was added, cleared, or modified.
8	Fingerprint	FGP	Lock user fingerprint was added, cleared, or modified.
9	FingerVein	FGP	Lock user finger-vein information was added, cleared, or modified.
10	Face	FACE	Lock user face information was added, cleared, or modified.

5.2.9.10. LockOperationTypeEnum

The LockOperationType enumeration SHALL indicate the type of Lock operation performed. The data type of the LockOperationType enum field is derived from enum8.

Value	Name	Conformance	Summary
0	Lock	M	Lock operation
1	Unlock	M	Unlock operation
2	NonAccessUserEvent	O	Triggered by keypad entry for user with User Type set to Non Access User
3	ForcedUserEvent	O	Triggered by using a user with UserType set to Forced User
4	Unlatch	M	Unlatch operation

5.2.9.11. OperationErrorEnum

The OperationError enumeration SHALL indicate the error cause of the Lock/Unlock operation per-

formed. The data type of the OperationError enum field is derived from enum8.

Value	Name	Conformance	Summary
0	Unspecified	O	Lock/unlock error caused by unknown or unspecified source
1	InvalidCredential	USR	Lock/unlock error caused by invalid PIN, RFID, fingerprint or other credential
2	DisabledUserDenied	M	Lock/unlock error caused by disabled USER or credential
3	Restricted	WDSCH YDSCH	Lock/unlock error caused by schedule restriction
4	InsufficientBattery	O	Lock/unlock error caused by insufficient battery power left to safely actuate the lock

5.2.9.12. OperatingModeEnum

The OperatingMode enumeration SHALL indicate the lock operating mode. The data type of the OperatingMode enum field is derived from enum8.

The table below shows the operating mode and which interfaces are enabled, if supported, for each mode.

Value	Name	Conformance	Keypad*	Remote*	RFID*
0	Normal	M	Yes	Yes	Yes
1	Vacation	O	No	Yes	No
2	Privacy	O	No	No	No
3	NoRemote-LockUnlock	M	Yes	No	Yes
4	Passage	O	N/A	N/A	N/A

* Interface Operational: Yes, No or N/A

Note: For modes that disable the remote interface, the door lock SHALL respond to Lock, Unlock, Toggle, and Unlock with Timeout commands with a response status Failure and not take the action requested by those commands. The door lock SHALL NOT disable the radio or otherwise unbind or leave the network. It SHALL still respond to all other commands and requests.

5.2.9.12.1. Normal

The lock operates normally. All interfaces are enabled.

5.2.9.12.2. Vacation

Only remote interaction is enabled. The keypad SHALL only be operable by the master user.

5.2.9.12.3. Privacy

This mode is only possible if the door is locked. Manual unlocking changes the mode to Normal operating mode. All external interaction with the door lock is disabled. This mode is intended to be used so that users, presumably inside the property, will have control over the entrance.

5.2.9.12.4. NoRemoteLockUnlock

This mode only disables remote interaction with the lock. This does not apply to any remote proprietary means of communication. It specifically applies to the Lock, Unlock, Toggle, and Unlock with Timeout Commands.

5.2.9.12.5. Passage

The lock is open or can be opened or closed at will without the use of a Keypad or other means of user validation (e.g. a lock for a business during work hours).

5.2.9.13. OperationSourceEnum

The OperationSource enumeration SHALL indicate the source of the Lock/Unlock operation performed. The data type of the OperationSource enum field is derived from enum8.

Value	Name	Conformance	Summary
0	Unspecified	O	Lock/unlock operation came from unspecified source
1	Manual	O	Lock/unlock operation came from manual operation (key, thumb-turn, handle, etc).
2	ProprietaryRemote	O	Lock/unlock operation came from proprietary remote source (e.g. vendor app/cloud)
3	Keypad	O	Lock/unlock operation came from keypad
4	Auto	O	Lock/unlock operation came from lock automatically (e.g. relock timer)

Value	Name	Conformance	Summary
5	Button	O	Lock/unlock operation came from lock button (e.g. one touch or button)
6	Schedule	HDSCH	Lock/unlock operation came from lock due to a schedule
7	Remote	M	Lock/unlock operation came from remote node
8	RFID	RID	Lock/unlock operation came from RFID card
9	Biometric	[USR]	Lock/unlock operation came from biometric source (e.g. face, finger-print/fingervein)

5.2.9.14. PIN/RFID Code Format

The PIN/RFID codes defined in this specification are all octet strings.

All value in the PIN/RFID code SHALL be ASCII encoded regardless if the PIN/RFID codes are number or characters. For example, code of “1, 2, 3, 4” SHALL be represented as 0x31, 0x32, 0x33, 0x34.

5.2.9.15. UserStatusEnum

The UserStatus enum used in various commands SHALL indicate what the status is for a specific user ID.

Value	Name	Conformance
0	Available	M
1	OccupiedEnabled	M
3	OccupiedDisabled	O

5.2.9.16. UserTypeEnum

The UserType enum used in various commands SHALL indicate what the type is for a specific user ID.

Value	Name	Conformance
0	UnrestrictedUser	M
1	YearDayScheduleUser	O
2	WeekDayScheduleUser	O

Value	Name	Conformance
3	ProgrammingUser	O
4	NonAccessUser	O
5	ForcedUser	[USR]
6	DisposableUser	[USR]
7	ExpiringUser	[USR]
8	ScheduleRestrictedUser	WDSCH YDSCH
9	RemoteOnlyUser	USR & COTA & PIN

5.2.9.16.1. UnrestrictedUser

User has access 24/7 provided proper PIN or RFID is supplied (e.g., owner).

5.2.9.16.2. YearDayScheduleUser

User has ability to open lock within a specific time period (e.g., guest).

5.2.9.16.3. WeekDayScheduleUser

User has ability to open lock based on specific time period within a reoccurring weekly schedule (e.g., cleaning worker).

5.2.9.16.4. ProgrammingUser

User has ability to both program and operate the door lock. This user can manage the users and user schedules. In all other respects this user matches the unrestricted (default) user. ProgrammingUser is the only user that can disable the user interface (keypad, remote, etc...).

5.2.9.16.5. NonAccessUser

User is recognized by the lock but does not have the ability to open the lock. This user will only cause the lock to generate the appropriate event notification to any bound devices.

5.2.9.16.6. ForcedUser

User has ability to open lock but a ForcedUser LockOperationType and ForcedUser silent alarm will be emitted to allow a notified Node to alert emergency services or contacts on the user account when used.

5.2.9.16.7. DisposableUser

User has ability to open lock once after which the lock SHALL change the corresponding user record UserStatus value to OccupiedDisabled automatically.

5.2.9.16.8. ExpiringUser

User has ability to open lock for ExpiringUserTimeout attribute minutes after the first use of the PIN code, RFID code, Fingerprint, or other credential. After ExpiringUserTimeout minutes the corre-