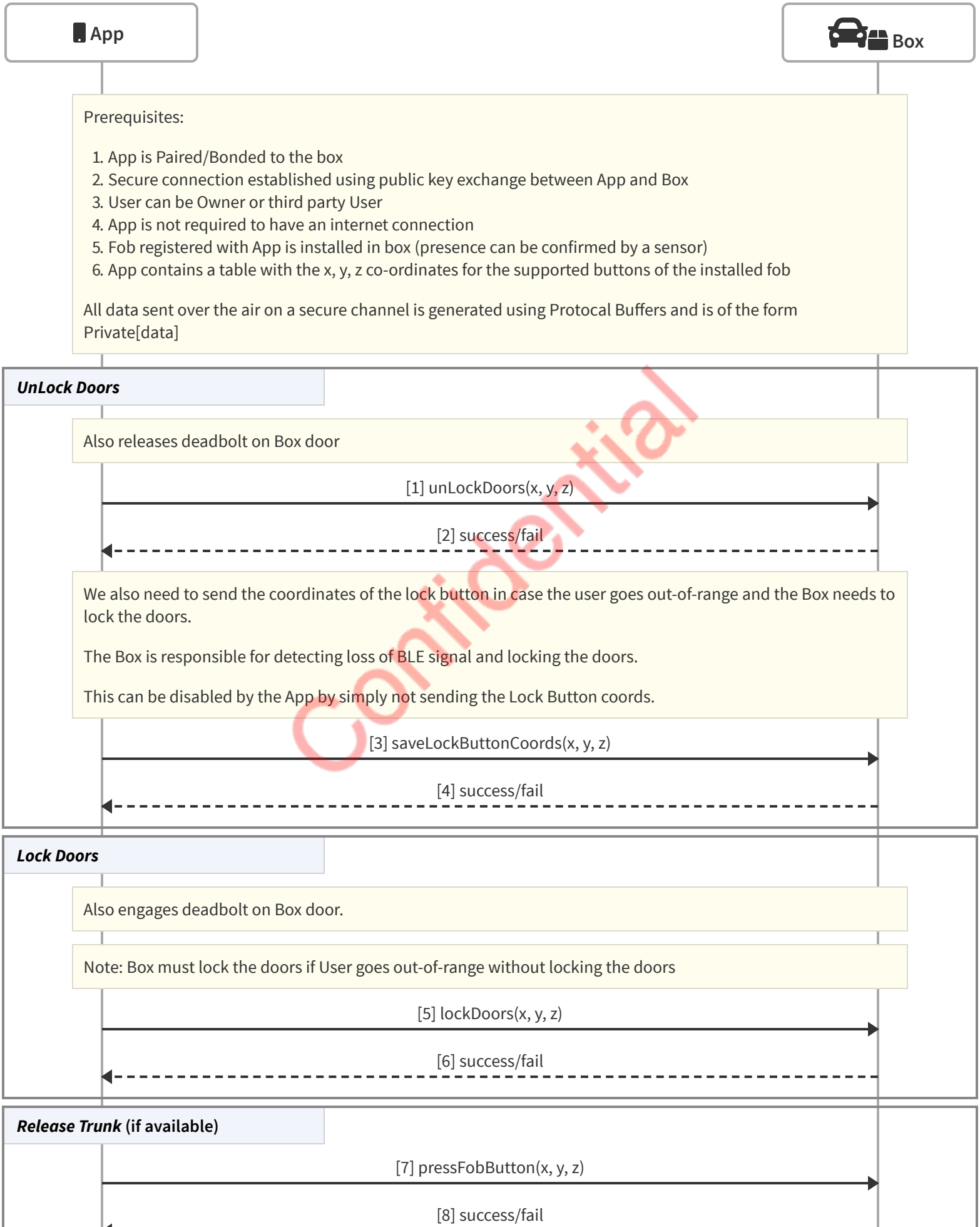
Primary Commands Use Case 0.0.5 (Eject/Reset Button update)



***Panic Button* (if available)**

[9] pressFobButton(x, y, z)

[10] success/fail

Eject Button

There is only one button on the exterior of the enclosure.

Prerequisites for Eject Button mode enabled.

1. Before the Box is paired, the button acts as a 'Pair' button
   (see Box Commissioning Use Case for details)
2. After pairing is complete the button acts as an 'Eject' button
3. A fob must be present in the Box (Fob Sensor: Present)
4. The Box door must be closed (Door Sensor: Closed)
5. When first enabled the Eject Button state is Eject

The Eject Button has two states: Eject and PostEjected.

1. Eject State: When first pressed, if successful, it ejects the fob and changes to PostEjected State
   This uses the same command as called from the App.
   The Door Sensor state changes from Closed to Open
   The Fob Sensor state changes from Present to Absent
2. PostEjected State: When pressed a second time, if successful, it closes the door and changes to Eject State
   The Door Sensor state changes from Open to Closed
   The Fob Sensor state changes from Absent to Present
   (if a fob was inserted, otherwise no change in Fob Sensor state)

These commands can also be issued from the App

Eject Button in Eject State

[11] ejectButtonPressed()

[12] success/fail

Eject Button in PostEjected State

[13] postEjectedButtonPressed()

[14] success/fail

Subscribe/Unsubscribe to Battery Level

receive notifications at App when battery level changes

[15] subscribeBatteryLevel()

[16] success/fail

[17] recieve battery level notifications

[18] unSubscribeBatteryLevel()

[19] success/fail

Locate Box

Sounds the buzzer for duration seconds, with on/off pulses(seconds). These are floats (decimal numbers) (for new Users who don't know where the box is)

[20] locateBox(duration, on, off)

[21] success/fail

Sensor Events

Fob Present Sensor

For some classes of fobs, after the user unlocks the doors, the user needs to eject the fob every time in order to start the engine. We would like to be able to handle the case where the user forgets to re-insert the fob and walks away expecting the auto-lock feature to kick-in. (or after pressing the lock button on the app, thinking the doors have been locked)

It looks like we can get away with implementing this feature with just a sensor.

1. With a sensor the App always knows if the fob is present or not (present/not present). So when the User goes out-of-range while the fob is not present, the App can notify the user.
2. As an enhancement, since the the mobile device hardware already can sense the signal strength, it can notify the user, as the signal weakens, and before the User goes out-of-range.

[22] subscribeFobPresentSensor()

[23] success/fail

[24] recieve fob present status notifications

[25] unSubscribeFobPresentSensor()

[26] success/fail

Box Door Open Sensor

This is used to detect the status of the Open door sensor (open/closed)

[27] subscribeBoxDoorOpenSensor()

[28] success/fail

[29] recieve box door status notifications

[30] unSubscribeBoxDoorOpenSensor()

[31] success/fail

Reset Button

There is a reset button on the exterior of the enclosure. This button is recessed and can only be pressed by a pen or a paperclip. This is a cold reset of the SoC. It is implemented in hardware and does not require any software. The embedded software loses current state and reboots to a known default state.

App

Box