MCU Attributes and Update All

This page demonstrates the use of MCU attribute “default values” and how to properly synchronize the correct values of MCU attributes when an attached Afero Secure Radio is rebooted.

Background

In Afero parlance, there are two types of attributes: “MCU attributes” and “ASR Attributes”:

- For MCU attributes, the Afero ecosystem assumes that the MCU “owns” and is the authoritative source on the values of those attributes.
- For all other attributes, the Afero ecosystem assumes that the ASR is the authoritative source for an attribute’s value.

When an ASR reboots and reconnects to the network, it performs an operation called “Update All” to ensure that attribute values stored in the Afero Cloud match the attribute values stored in ASR. ASR will update the Cloud with all of the values for its attributes based on its configuration, connection state, Profile, and any default values assigned to attributes in the Profile.

When an MCU is connected and the ASR Profile contains MCU attributes, MCU attributes are ignored as part of this “Update All” process; ASR and the Cloud recognize the MCU as the authoritative source for its attribute values and it makes no assumptions that ASR knows anything authoritative about those attributes.

In order to ensure the Afero Cloud has correct values for MCU attributes, ASR will perform an “Update All” process with the MCU simultaneously, using its own Update All process for non-MCU attributes. Any MCU

attributes that have default values in the Profile will be sent from ASR to the application on the MCU for verification. The MCU application must then set these attributes to either:

- Validate the default value (if it's correct), or
- Update the default value (if it is incorrect).

The process is not complex, but it **must** be performed by the MCU application to ensure that MCU attributes are in sync between the MCU and the Afero Cloud. ASR will send notifications to the MCU while it is still connecting to the Afero Cloud:

- For simple examples (like the thermometer example used below) in which we don't have a lot of attributes with default values to deal with, it is acceptable to send `set_attribute()` updates to ASR before it's connected. These updates will stay queued until ASR connects, then they will be sent to the Afero Cloud normally.
- For Profiles with larger numbers of default values to be updated, care must be taken to not overrun the afLib queue with updates before ASR can accept them. It's important to check the return code of any `set_attribute()` call and delay and retry if the result != AF_SUCCESS.
- A more robust and complex example would collect the attribute updates that need to be sent, then only send them after receiving `AF_SYSTEM_ASR_STATE` == `AF_MODULE_STATE_LINKED`, **after** ASR has connected.

The afBlink example app in afLib shows one way to defer default attribute updates until after connectivity has been established; the MCU Update All example code shows a simpler implementation to demonstrate where in the event handler you will receive these default value notifications.

Considerations

There are two cases to consider, described below:

1 MCU Attributes with Default Values

In the Afero Profile Editor, you can define a "default value" for an MCU attribute. While uncommon, this feature allows you to provide a starting value for an MCU attribute that can be modified via a simple

Profile update. A default value can be used as a flag to enable certain MCU features, or perhaps as a calibration value for a sensor.

2 MCU Attributes with MCU-Local Values

Your MCU application may keep attributes to store the state of hardware or other physical inputs. The values for these attributes are only known to the MCU and generally would not have a default value defined for the attribute. For example, the MCU may know the state of a connected sensor and then put it in an attribute so that it can be read or modified remotely.

Functional Examples

In order to describe the MCU Update All process, let's take a simple example: an Afero enabled grill thermometer. We need to know the following pieces of information about our thermometer:

- The current temperature reading of the thermometer
- A "target" temperature that alerts the user when a threshold has been crossed
- A sampling rate for the thermometer sensor

The **current temperature** is a physical state; we have no way of estimating what this value could possibly be when ASR reboots. The Profile MCU attribute for this sensor reading would not have a default value, since the correct value is known only to the MCU.

However, for **target temperature**, we might want to use an attribute with a default value. If we define the target as a particular default value, and if the user didn't change the value, the device would still perform properly on reboot. However, if the user should modify that value, then the MCU could save the value and restore it whenever the device is rebooted.

The **sampling rate** is a good example of using a default value to alter the behavior of a device without making any changes to the MCU code. This sampling rate can be an attribute that is never shown to the user, but can be modified with a Profile update to make the device perform more optimally; for example, if it turns out the sampling rate is too high, it might affect battery life of the device, and a Profile update to

MCU Update All

When an ASR attached to an MCU has rebooted, ASR will send to the MCU (via the `attrEventHandler`) a series of `AF_LIB_EVENT_MCU_DEFAULT_NOTIFICATION` notifications, one for each MCU attribute with a default value.

In our thermometer example, ASR would send an `AF_LIB_EVENT_MCU_DEFAULT_NOTIFICATION` event for both the **target temperature** attribute as well as the **sampling rate** attribute. Since the **current temperature** attribute has no default value, the MCU would not receive an `AF_LIB_EVENT_MCU_DEFAULT_NOTIFICATION` event for it.

The MCU would then perform the following functions:

- If the user has set a target temperature different from the default, the MCU would read it (from whatever place the MCU has stored it) and call `af_lib_set_attribute()` on the target temperature attribute. If the user has not set a different value for this setting, the MCU application **MUST** call `af_lib_set_attribute()` and return the default value sent to the MCU by ASR.
- The MCU would read the **sampling rate** from the default value sent by ASR, and then call `af_lib_set_attribute()` to set that default value as the current value.

It is extremely important to remember that the MCU **MUST** call `af_lib_set_attribute()` to set all MCU attributes with default values in this manner, even if it just sets the same default value that is sent from ASR. If the MCU does not validate these values by sending them back to ASR in response to an `AF_LIB_EVENT_MCU_DEFAULT_NOTIFICATION` event, then it is possible that ASR/Cloud would not correctly reflect the values of those MSU attributes.

Next: *Checking Capabilities*

Updated September 25, 2019