App Note: LAN Mode

02/27/2017

Audience
This document gives an outline of the communication between a mobile app and an Ayla device over a wireless LAN (referred to as LAN mode in Ayla mobile SDK), and is intended for app developers to understand and troubleshoot apps using LAN mode.

Introduction
LAN mode communication is the direct communication between an Ayla module and a mobile phone which are in the same LAN. LAN mode reduces the latency in communication between the app and the device, and can be used to communicate with the device even if Ayla Device Service is not reachable. Currently Ayla devices can support up to 2 simultaneous LAN mode sessions.

LAN mode in Ayla mobile SDK
By default, LAN mode communication is enabled in the Ayla mobile SDK, and the SDK starts LAN mode whenever possible. Each AylaDevice object creates its own AylaLanModule instance to start communicating in LAN mode with the corresponding Ayla device. This AylaLanModule object is responsible for maintaining the LAN session with that Ayla device, and handling the LAN commands sent and received from the Ayla device. For each AylaDevice, the AylaLanModule managing its LAN mode can be accessed using the method `AylaDevice.getLanModule()`.

Starting LAN session: local registration and key exchange.
LAN mode for each device will be started when the SDK method `AylaNetworks.onResume()` is called by the app, and stopped when `AylaNetworks.onPause()` is called by the app.
To check whether the device is currently in LAN mode, use the method `AylaDevice.isLanModeActive()`.

The SDK initiates LAN mode session with the device by sending a POST /local_reg.json HTTP request to the device. If accepted, the device starts a key exchange process with the mobile by sending POST /local_lan/key_exchange.json. If the key exchange was successful, mobile sends a response code of 200 to the device. For any response code other than 200, the Ayla device will delete this LAN client, and the mobile app needs to send another POST /local_reg.json to start a new key exchange. Successful completion of the key exchange means that we now have a secure LAN session, and can start sending encrypted data between the mobile and the Ayla device.

The following steps are done internally in the SDK when a device switches to LAN mode -

- Fetch and update values of all managed properties from the device in LAN mode.
- Stop polling the service for property changes on that device.

Once these steps are completed, the SDK updates the app on the LAN mode state change through the device change listeners registered by the app. (Refer to `DeviceChangeListener.deviceLanStateChanged()`). At this point, the app should update any UI changes required for LAN mode (Sample code is available in `DeviceListFragment` in Aura app, which updates the UI on lan mode state change).

LAN session extension
To keep a LAN session active, the mobile SDK sends a periodic keep-alive request to the Ayla device (PUT /local_reg.json). The default keep-alive time interval in Ayla Mobile SDK is 10 seconds. If the AylaDevice returned from Ayla Device Service has a keep-alive value in its LAN config, then the keep-alive value in SDK is set to (config.keep-alive/3). This value can be changed by app developers for their devices by calling the method `AylaLanModule.setKeepAliveInterval(int interval)` after `AylaDevice.startLanSession().`

Property changes in LAN mode
When a device has an active LAN session, all property changes will be communicated between the device and the app through LAN commands. These LAN commands are queued up in the SDK, and are fetched by the device one by one. The SDK first notifies the module that it has a pending command by setting ‘notify’ field in /local_reg.json to 1. The module then follows with a GET request to fetch the command from the mobile.

For example, consider a property change initiated from the app UI. The mobile SDK sends a PUT /local_reg.json request with notify = 1. The module follows with a GET /local_lan/commands.json request to the SDK, and the mobile app sends an encrypted LAN command in the format { “enc”: “...”, “sign”: “...” } in response to the module’s request.
Here, enc is encrypted (sequence number + clear text JSON object), and sign is the signature. The clear text JSON object is in the format {“seq_no”: “...”, “data”:<...>”} where seq_no is the sequence number of the message and data is the clear text JSON object in the following format.
```
"cmds" :  [{
  "cmd": {
      "cmd_id": "<cmd-cmd_id>", (unsigned 32-bit int)
      "method": "<cmd-method>",
      "resource": "<cmd-resource>",
      "data": "<data-for-cmd>",
      "uri": "<uri-for-response>"
   }
}]
```

A sample value for the data field in this command JSON for a datapoint update on the property BLUE_LED is -
```
{"properties":[{"property":{"base_type":"boolean","name":"Blue_LED","value":1}}]}
```

If a subsequent module request is not expected for a command, it is cleared from the queue after the module fetches the command.

Similarly, if a property changes on the device, the device sends an encrypted JSON with the updated value to the url /local_lan/property/datapoint.json. The SDK decrypts this JSON and sends an update to the app through the device change listeners registered by the app. (Refer `DeviceChangeListener.deviceChanged()`).

Note: SDK methods `AylaDevice.fetchProperties()` and *`AylaProperty.createDatapoint()`* internally use LAN communication if the device is in LAN mode.

App developers can use the following methods in the SDK if they want to specify usage of LAN or cloud mode for fetching properties or creating datapoints. `AylaDevice.fetchPropertiesLAN()` fetches properties from the device using LAN mode, and `AylaDevice.fetchPropertiesCloud()` fetches properties from the Ayla Device Service. Similarly, `AylaProperty.createDatapointLAN()` creates the datapoint via a LAN connection with the device, and `AylaProperty.createDatapointCloud()` creates the datapoint using the cloud service.

Disabling LAN mode
For each AylaDevice, LAN mode communication can be enabled or disabled using the method `AylaDevice.setLanModePermitted(boolean permitted)`.

LAN mode communication in offline mode
Ayla Mobile SDK allows users to sign in to the app using cached authorization if Ayla cloud service is unavailable. This is referred to as offline mode. In offline mode, LAN mode is the only method for the app to communicate with the device. Offline mode can be enabled in the Ayla mobile SDK by setting app config parameter `allowOfflineUse` to true.

When the Ayla cloud service is available, the LAN IP address of the Ayla device is obtained from the service. In case of LAN mode communication in offline mode, the cached LAN IP address is used to start LAN mode. It is possible that the LAN IP address of the device might have changed after the device information was last fetched from the Ayla Device Service. In this case, LAN mode session for that device will fail, and the SDK will attempt to find the correct LAN IP address using multicast DNS. Default time interval for MDNS queries in the SDK is 1s. The app developer can change the query interval using the method `AylaLanModule.setMdnsQueryInterval(int interval)`.

Troubleshooting LAN mode communication
1. Device returns a 503 error code for LAN connection requests from the app.
Ayla devices currently supports only 2 simultaneous LAN sessions. So LAN mode will fail if the device has 2 existing LAN sessions from other phones.

2. SDK returns an error response to key exchange from the device.

These are the error codes that will be returned from the SDK to the device if key exchange failed.

400: Bad Request - The key exchange json object could not be parsed

403: Forbidden - A commands request was received without an established session

404: Not Found: No device or LAN module was found to handle this request

412: Precondition Failed - the key_id did not match, or the AylaDevice object in the SDK does not have a LAN key.

426: Upgrade Required - The prototype number for ciphers/hashes is not supported. This is an unsupported crypto version

495: Cert Error - The SDK could not generate session keys