Provisioning component provides APIs and definitions to facilitate the provisioning service. Provisioning is done over BLE, and the configuration data is stored in the file system.

For more details on functionality overview, data exchange formats and profile definition (with service and characteristics information) refer provisioning application.

# Features and Limitations

This implementation of Provisioning provides the following facilities:

1. Crete Provisioning GATT server.
2. Scan and provide the list of APs in the range.
3. Store configuration data.
4. Check Wi-Fi connectivity using the configured SSID and passphrase.

Limitations:

Provisioning is done by storing the whole configuration data at once. Reading and changing only a specific parameter in the configuration data is not possible.

# Header file/s

Components/prov/inc/prov.h.

# Data Structure Definitions

## prov\_start\_prams\_t

This data structure is used during provisioning module initialization.

|  |  |
| --- | --- |
| ***method*** | Only provisioning over BLE is supported currently |
| ***name*** | Device name. If NULL, PROV\_DFLT\_NAME is set |
| ***appearance*** | Appearance. Default is set to 0 |
| ***manufacturer\_name*** | Manufacturer name. If NULL, PROV\_DFLT\_MANUFCTR\_NAME is set |
| ***cb*** | The user callback to be called when provisioning is done |
| ***cb\_ctx*** | Context pointer to be passed with callback |

Table : prov\_start\_prams\_t – parameters

# API Reference

## prov\_start

### Overview

This API initializes and starts the Provisioning service. It initializes the BLE communication, starts the GATT server and begins advertising the Provisioning service.

All provisioning-related activities are handled internally. If provided through config parameter, a user supplied callback will be called when the new configuration file received and is written into data fs.

### Definition

|  |
| --- |
| int  prov\_start(prov\_start\_prams\_t \*cfg) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *cfg* | Pointer to the data structure of type prov\_start\_prams\_t |

Table : prov\_start - parameters

### Return

Success: 0

Error: -1

## prov\_conf\_param\_str\_get

### Overview

This API gets the configuration parameters of type string. For example: “ssid” or “passphrase”. The value of the key passes is returned.

### Definition

|  |
| --- |
| const char \*  prov\_conf\_param\_str\_get(char \*key) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *key* | Key part of the configuration |

Table : prov\_conf\_param\_str\_get - parameters

### Return

Success: Pointer to the value of the key passed.

Error: NULL.

## prov\_conf\_param\_int\_get

### Overview

This API gets the configuration parameter of type integer. For example: “port”. The value of the key passes is returned.

### Definition

|  |
| --- |
| int  prov\_conf\_param\_int\_get(char \*key) |

### Parameters

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| *key* | Key part of the configuration |

Table : prov\_conf\_param\_int\_get – parameters

### Return

Success: Pointer to the value of the key passed.

No such parameter exists: NULL.

## prov\_is\_provisioned

### Overview

This API is used to check if the device is provisioned atleast once.

### Definition

|  |
| --- |
| bool  prov\_is\_provisioned(void) |

### Parameters

None.

### Return

Not Provisioned: 0

Provisioning: 1

# Application Example

For the example code, refer: *examples/prov application*.