RTX8200 Configuration

Document information

Project .....:	RTX8200 – Bluetooth Low Energy Beacon
Feature .....:	Configuration
Status.....:	Draft
Document revision .....:	8.2
Last updated Date .....:	5/27/2019

This document is intended to outline the DECT Provisioning API for the RTX VoIP bases and RTX8200 Bluetooth Beacon. The target audience is the provisioning server developer.

Contents

History

Revision	Author	Date	Comment
0001	MSA	2018-11-19	Initial revision
0002	MSA	2019-02-09	Updated with BLE interface, and additional details on provisioning.
0003	JDY	2019-05-27	Update provisioning server interface.
0004	JDY	2019-05-28	Add provisioning sequence.

References

[1] RTX A/S, "BTLE Configuration Feature Specification," 11 02 2019. [Online]. Available: https://rtx1-loc.sharepoint.com/:w:/r/sites/rtxbcdoc/_layouts/15/Doc.aspx?sourcedoc=%7B8825A9C1-4D00-4A0E-8104-0E42B204F9F4%7D&file=Feature_Specification_BTLE_Configuration_Interface.docx&action=default&mobileredirect=true. [Accessed 11 02 2019].

[2] Google, "Protocol Buffers," 16 11 2018. [Online]. Available: https://developers.google.com/protocol-buffers/.

Terms & Abbreviations

Term	Description
DECT	Digital Enhanced Cordless Telecommunications
PP	Portable Part (Handset)
FP	Fixed Part (Base station)
BLE	Bluetooth Low Energy
BTLE	Bluetooth Low Energy
GATT	Generic Attributes (GATT) BLE Profile

Introduction

The RTX8200 BLE Location Gateway can be configured via both a BLE GATT interface (which requires the user to be near the device) and via the RTX VoIP DECT system.

Configuring the RTX8200 via the BLE GATT interface requires the user to have a BLE compliant device with either an application or a generic BLE application. This device can be either a PC, a Smart Phone, or and BT-enabled RTX863x devices.

Configuring the RTX8200 via the RTX VoIP DECT system requires the customer to provide a server-backend which complies with the interface defined in this document.

This document will describe

- Configuration via BLE GATT interface
- Configuration via SME VoIP provisioning
    - The API the provisioning server must implement to support RTX VoIP bases
    - The configuration payload supported by RTX8200.

Configuration via BLE GATT interface

This section describes the method for configuring the RTX8200 via the BLE GATT interface.

Overview

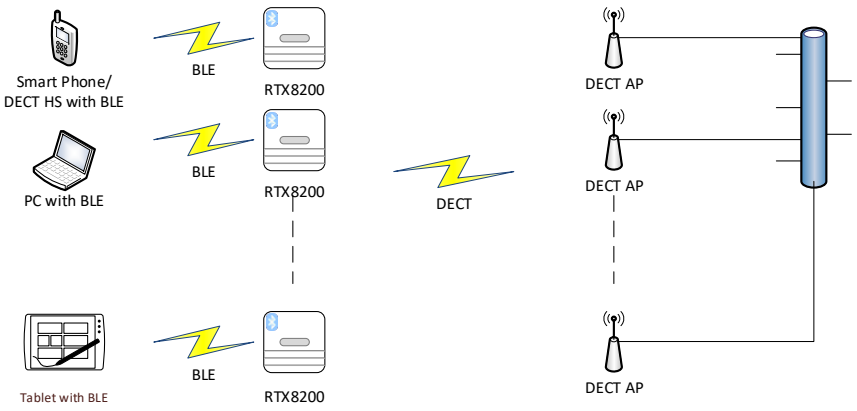


Figure 1 System illustration for configuring via the BLE GATT interface.

Overall functionality

The RTX8200 Bluetooth Low Energy beacon offers several configurable options, available through a BTLE GATT-based interface.

The configuration mode of the beacon is activated by pressing the physical switch located on the beacon. This will start a GATT server and advertisement on the RTX8200 that provides a connectable interface. Any TX or RX mode will be paused during configuration mode. The device will advertise a short friendly name, that is configurable.

The configuration mode will be disabled, on a BLE disconnect or after a timeout of 60 seconds since the last activity on the connection.

The device is configured using standard GATT Service and Characteristics definitions. This enables any user to create an application to configure the device on either Smart phone, tablet, or PC with BLE support. RTX delivers a configuration app on all BLE enabled SME VoIP handsets.

The configurable parameters can be found in the document [1].
The BLE GATT API description is in the document [2]

Configuration via SME VoIP provisioning

This section describes the overall method for configuring the RTX8200 via the SME VoIP DECT system.

Overview

The system is comprised of the elements in Figure 2. The relevant customer part is the HTTP interface on the right.

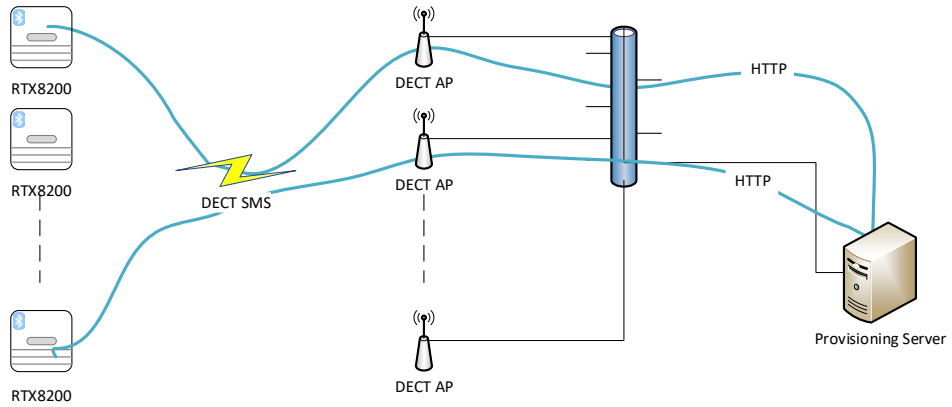


Figure 2 System illustration for configuring via DECT provisioning.

API Description

The API consists of

- An XML-based provisioning server interface, which is compliant with the RTX VoIP bases.
- A configuration payload to the RTX8200, which is Protocol Buffers-based [1]
- An ACTION URL on the RTX VoIP base for triggering provisioning to an RTX8200 device.

Provisioning server XML-interface

The provisioning server must implement the XML-based API described in this section for the RTX VoIP bases to be able to download a configuration for the RTX8200.

The following sections will describe the replies expected by the RTX VoIP bases. All HTTP requests (GET, POST, etc.) are initiated from the RTX VoIP base on the provisioning server.

XML Element description

Element	Attributes	Description	Required
<configs>		Root-node.	Yes
<devices>		Parent for device list. Required.	Yes
<device>		Specifies a single RTX8200 device. Optional	No
	type	Specifies the device type	Yes
	ipei	Unique identifier for the RTX8200. Required. HEX formatted om the form XXXXXXXXXX	Yes
	revision	Revision of configuration. RTX8200 is only provisioned if this differs from the internal revision of the RTX8200.	Yes

		RTX8200 will update local revision upon successful provisioning.	
<data>		Wrapper for encoded configuration payload. Optional.	No
	encoding	Specifies the encoding method of the payload. Supported methods: base64.	Yes

RTX will provide an XML schema for validating the XML.

The provisioning process is described in two parts: Initiating provisioning and requesting configuration data.

Initiating provisioning of RTX8200 devices

The RTX VoIP base can be setup to periodically poll the provisioning server and request whether a given RTX8200 device should start provisioning.

REQUEST FROM BASE:

The RTX VoIP base will request using a list of RTX8200 device IPEI's that is currently DECT connected to the base. The request is `GET /config`, passing the IPEI's as a list of values named `ipei`. The `ipei` list is never empty.

EXAMPLE HTTP REQUEST:

```
GET /config/device?ipei=IPEI_1&ipei=IPEI_2&ipei=IPEI_3
```

EXPECTED RESPONSE FROM PROVISIONING SERVER:

The base has no state of RTX8200 configuration, and thus the provisioning server is responsible for deciding if any of the device <IPEI_X> should start provisioning.

EXAMPLE RESPONSE XML:

```
<?xml version="1.0" encoding="utf-8"?>
<configs>
  <devices>
    <device type="rtx8200" ipei="IPEI_1" revision="3" />
    <device type="rtx8200" ipei="IPEI_3" revision="3" />
  </devices>
</configs>
```

POST-RESPONSE ACTIONS:

Base will prompt the RTX8200 devices included in the response XML to initiate provisioning. Once the provisioning process begins, the RTX8200 device will request configuration data from the provisioning server as described below. If no device is to be provisioned, the base expects an empty `<devices>` list.

Requesting configuration data for an RTX8200 device

REQUEST FROM BASE:

```
GET /config/device/data?ipei=<ipei>&revision=<device revision>
```

EXPECTED RESPONSE FROM PROVISIONING SERVER:

The configuration for a single device with `<ipei>`.
The `<device revision>` is the internal configuration revision reported by the RTX8200 – the provisioning server will use this to determine what configuration version (if any) to send to the RTX8200.
The **`-- Base64 encoded payload --`** is the one described in the section Protocol Buffer Configuration Payload.

EXAMPLE RESPONSE XML:

```
<?xml version="1.0" encoding="utf-8"?>
<configs>
  <devices>
    <device type="rtx8200" ipei="IPEI" revision="3">
```

```
      <data encoding="base64">
        -- Base64 encoded payload --
      </data>
    </device>
  </devices>
</configs>
```

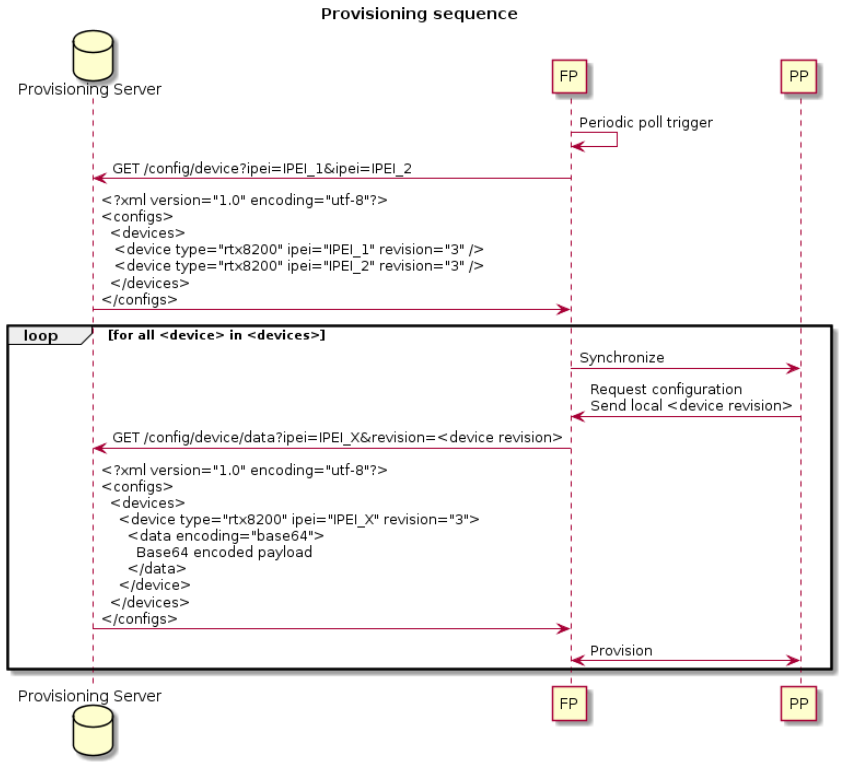
POST-RESPONSE ACTIONS:
Base will provision the device included in the response XML. On server error, the base will expect an empty `<devices>` list.

Commented [A1]: Each base shall only require the configs for the RTX8200 connected to that particular base.

So in the case where multiple devices are in the same xml, all bases will get all devices and sort through the relevant ipei's themselves.

Provisioning sequence

The provisioning sequence is started by the RTX VoIP base on a periodic trigger. The base requests the provisioning server for a list of devices that should be provisioned, based on what is DECT locked to the base. The devices in the server response is then iterated to start provisioning on each of them.



ACTION URL triggering of provisioning

The RTX VoIP bases will provide an URL for triggering provisioning of the RTX8200 devices. The URL can be called on any bases in the Multi-Cell system. The system will ensure that the base handling the RTX8200 will initiate the HTTP calls described in the section Provisioning server XML-interface.

The base action URL can be found at:

http://<base hostname>/admin/config.html?ipei=<device ipei>

The base will initiate download of relevant XML provisioning data from the provisioning server as described in the section Provisioning server XML-interface, essentially skipping the initiation process.

Protocol Buffer Configuration Payload

The configuration server must implement a method for creating the Protocol Buffer-based configuration payload for the RTX8200. Additionally, the binary Protocol Buffer serialized data must be base64 encoded before it is delivered to the RTX VoIP bases.

RTX will provide a `.proto` file, which defines the data structure to be serialized. From this, there are a number of packages that will be able to create API's for a number of popular programming languages (C, C++, C#, Python, PHP, and so on).

The `.proto` file is of the format:

```
syntax = "proto3";

package rtx8200;

message ConfigSettings {
    required uint32 revision;
}

message RFSettings {
    enum AntennaArray {
      LEFT=0;
      RIGHT=1;
      ACROSS=2;
    }
    AntennaArray antennaArraySelection = 1;

    enum ExternalAntenna {
      OFF=0;
      MONOPOL1=1;
      MONOPOL2=2;
      BOTH=3;
    }
    AntennaArray externalAntennaSelection = 2;
}
```

This is just for illustrative purposes.

Protocol Buffers supports partial configurations, so only the changed parts can be encoded. This saves bandwidth on the DECT-side.