**HomeConnect**

**Interface Specification**

**For**

**Firmware Update Service**

**Author:** GDE-EDS9

**Last Change:** 22.11.2021

**Version:** 5.0

**Eco:** 2A02GP2A02GP

**Index:** D,2

**Document Number:** 5560 0000010585

**Document Type:** Requirement Specification

**Document Name:** Interface Specification for FW Updates

**Status**: Working (~~Released~~)

# I Table of Contents

[I Table of Contents 2](#_Toc88504072)

[II Table of Figures 4](#_Toc88504073)

[III Abbreviations 5](#_Toc88504074)

[IV Revision History 6](#_Toc88504075)

[1 Introduction 7](#_Toc88504076)

[1.1 Goals of the Specification 7](#_Toc88504077)

[1.2 Validity of the Document 7](#_Toc88504078)

[1.3 Reference of Related Documents 7](#_Toc88504079)

[1.4 Requirement Levels 7](#_Toc88504080)

[2 Preconditions 8](#_Toc88504081)

[2.1 Communication Partners 8](#_Toc88504082)

[2.2 Discovery 8](#_Toc88504083)

[2.3 Security 8](#_Toc88504084)

[2.4 Common Data Structures for All Services 8](#_Toc88504085)

[2.5 Data Types 8](#_Toc88504086)

[3 Workflow of a Firmware Update 9](#_Toc88504087)

[3.1 Overview 9](#_Toc88504088)

[3.1.1 Providing a Firmware to the HCA or Local CS 9](#_Toc88504089)

[3.2 Get HA Configuration 10](#_Toc88504090)

[3.2.1 HA Configuration 10](#_Toc88504091)

[3.3 New Update Available 11](#_Toc88504092)

[3.4 Download Update 12](#_Toc88504093)

[3.4.1 HTTPS Connection for Download of Update Packages 13](#_Toc88504094)

[3.4.2 Detailed Download/Validation/Verification Procedure 14](#_Toc88504095)

[3.4.3 Properties File content 15](#_Toc88504096)

[3.4.4 Properties File structure with validation information (Version 5) 16](#_Toc88504097)

[3.4.5 Properties File structure with validation information (Version 6) 18](#_Toc88504098)

[3.5 Install Update 21](#_Toc88504099)

[3.5.1 Recovery / Rollback 21](#_Toc88504100)

[3.6 Finalize Update 22](#_Toc88504101)

[3.7 Finish Update 23](#_Toc88504102)

[4 Service – Firmware Update – v2 24](#_Toc88504103)

[4.1 Overview 24](#_Toc88504104)

[4.2 List of Resources 24](#_Toc88504105)

[4.3 Structures in item <data> 25](#_Toc88504106)

[4.3.1 Content type HACONFIG 25](#_Toc88504107)

[4.3.2 Content type NEWUPDATEAVAIL 27](#_Toc88504108)

[4.3.3 Content type PACKAGEPROPERTIESREQUEST 28](#_Toc88504109)

[4.3.4 Content type PACKAGEPROPERTIES 28](#_Toc88504110)

[4.3.5 Content type FUSTATE 30](#_Toc88504111)

[4.3.6 Content type CONFIGRETRIGGER 32](#_Toc88504112)

[4.3.7 Content type PERMRETRIGGER 32](#_Toc88504113)

[4.3.8 Content type ABORT 32](#_Toc88504114)

[4.4 Functionalities 34](#_Toc88504115)

[4.4.1 Trigger HA Configuration 34](#_Toc88504116)

[4.4.2 Notify HA Configuration 35](#_Toc88504117)

[4.4.3 Post New Update Available 36](#_Toc88504118)

[4.4.4 Notify Package Properties Request 37](#_Toc88504119)

[4.4.5 Post Package Properties 38](#_Toc88504120)

[4.4.6 Get Firmware Update State of HA 39](#_Toc88504121)

[4.4.7 Notify Firmware Update State of HA 40](#_Toc88504122)

[4.4.8 Post Permission Retrigger 41](#_Toc88504123)

[4.4.9 Set Aborted 42](#_Toc88504124)

[4.5 Application Guidelines / Behavior 43](#_Toc88504125)

[5 Service – Firmware Update – v3 44](#_Toc88504126)

[5.1 Overview 44](#_Toc88504127)

[5.2 List of Resources 44](#_Toc88504128)

[5.3 Structures in item <data> 45](#_Toc88504129)

[5.3.1 Content type HACONFIG 45](#_Toc88504130)

[5.3.2 Content type NEWUPDATEAVAIL 47](#_Toc88504131)

[5.3.3 Content type PACKAGEPROPERTIESREQUEST 49](#_Toc88504132)

[5.3.4 Content type PACKAGEPROPERTIES 50](#_Toc88504133)

[5.3.5 Content type FUSTATE 51](#_Toc88504134)

[5.3.6 Content type CONFIGRETRIGGER 52](#_Toc88504135)

[5.3.7 Content type PERMRETRIGGER 53](#_Toc88504136)

[5.3.8 Content type ABORT 53](#_Toc88504137)

[5.3.9 Content type DOWNLOADPROGRESS 54](#_Toc88504138)

[5.4 Functionalities 55](#_Toc88504139)

[5.4.1 Trigger HA Configuration 55](#_Toc88504140)

[5.4.2 Notify HA Configuration 56](#_Toc88504141)

[5.4.3 Post New Update Available 57](#_Toc88504142)

[5.4.4 Notify Package Properties Request 58](#_Toc88504143)

[5.4.5 Post Package Properties 59](#_Toc88504144)

[5.4.6 Get Firmware Update State of HA 60](#_Toc88504145)

[5.4.7 Notify Firmware Update State of HA 61](#_Toc88504146)

[5.4.8 Post Permission Retrigger 62](#_Toc88504147)

[5.4.9 Set Aborted 63](#_Toc88504148)

[5.4.10 Notify Download Progress 64](#_Toc88504149)

[5.5 Application Guidelines / Behavior 64](#_Toc88504150)

[6 Service – Firmware Update – v4 65](#_Toc88504151)

[6.1 Overview 65](#_Toc88504152)

[6.2 List of Resources 65](#_Toc88504153)

[6.3 Structures in item <data> 66](#_Toc88504154)

[6.3.1 Content type HACONFIG 66](#_Toc88504155)

[6.3.2 Content type NEWUPDATEAVAIL 68](#_Toc88504156)

[6.3.3 Content type PACKAGEPROPERTIESREQUEST 70](#_Toc88504157)

[6.3.4 Content type PACKAGEPROPERTIES 71](#_Toc88504158)

[6.3.5 Content type FUSTATE 72](#_Toc88504159)

[6.3.6 Content type CONFIGRETRIGGER 73](#_Toc88504160)

[6.3.7 Content type PERMRETRIGGER 74](#_Toc88504161)

[6.3.8 Content type ABORT 74](#_Toc88504162)

[6.3.9 Content type DOWNLOADPROGRESS 75](#_Toc88504163)

[6.3.10 Content type TRUSTCONFIG 75](#_Toc88504164)

[6.4 Functionalities 77](#_Toc88504165)

[6.4.1 Trigger HA Configuration 77](#_Toc88504166)

[6.4.2 Notify HA Configuration 78](#_Toc88504167)

[6.4.3 Post New Update Available 80](#_Toc88504168)

[6.4.4 Notify Package Properties Request 81](#_Toc88504169)

[6.4.5 Post Package Properties 82](#_Toc88504170)

[6.4.6 Get Firmware Update State of HA 83](#_Toc88504171)

[6.4.7 Notify Firmware Update State of HA 84](#_Toc88504172)

[6.4.8 Post Permission Retrigger 85](#_Toc88504173)

[6.4.9 Set Aborted 86](#_Toc88504174)

[6.4.10 Notify Download Progress 87](#_Toc88504175)

[6.4.11 Notify Trust Config 88](#_Toc88504176)

[6.5 Application Guidelines / Behavior 89](#_Toc88504177)

[7 Service – Firmware Update – v5 90](#_Toc88504178)

[7.1 Overview 90](#_Toc88504179)

[7.2 List of Resources 90](#_Toc88504180)

[7.3 Structures in item <data> 91](#_Toc88504181)

[7.3.1 Content type HACONFIG 91](#_Toc88504182)

[7.3.2 Content type NEWUPDATEAVAIL 93](#_Toc88504183)

[7.3.3 Content type PACKAGEPROPERTIESREQUEST 95](#_Toc88504184)

[7.3.4 Content type PACKAGEPROPERTIES 96](#_Toc88504185)

[7.3.5 Content type FUSTATE 97](#_Toc88504186)

[7.3.6 Content type CONFIGRETRIGGER 98](#_Toc88504187)

[7.3.7 Content type PERMRETRIGGER 99](#_Toc88504188)

[7.3.8 Content type ABORT 99](#_Toc88504189)

[7.3.9 Content type DOWNLOADPROGRESS 100](#_Toc88504190)

[7.3.10 Content type TRUSTCONFIG 100](#_Toc88504191)

[7.4 Functionalities 102](#_Toc88504192)

[7.4.1 Trigger HA Configuration 102](#_Toc88504193)

[7.4.2 Notify HA Configuration 103](#_Toc88504194)

[7.4.3 Post New Update Available 105](#_Toc88504195)

[7.4.4 Notify Package Properties Request 106](#_Toc88504196)

[7.4.5 Post Package Properties 107](#_Toc88504197)

[7.4.6 Get Firmware Update State of HA 108](#_Toc88504198)

[7.4.7 Notify Firmware Update State of HA 109](#_Toc88504199)

[7.4.8 Post Permission Retrigger 110](#_Toc88504200)

[7.4.9 Set Aborted 111](#_Toc88504201)

[7.4.10 Notify Download Progress 112](#_Toc88504202)

[7.4.11 Notify Trust Config 113](#_Toc88504203)

[7.5 Application Guidelines / Behavior 114](#_Toc88504204)

[8 Appendix 115](#_Toc88504205)

[8.1 Return and Error Codes 115](#_Toc88504206)

[8.2 Device Types 115](#_Toc88504207)

# II Table of Figures

[Figure 1 - Flowchart for](#_Toc88504208) *[Get HA Configuration](#_Toc88504208)* [10](#_Toc88504208)

[Figure 2 - Flowchart for](#_Toc88504209) *[New Update Available](#_Toc88504209)* [11](#_Toc88504209)

[Figure 3 - Flowchart for](#_Toc88504210) *[Download Update](#_Toc88504210)* [12](#_Toc88504210)

[Figure 4 - Flowchart for download/validation/verification procedure 14](#_Toc88504211)

[Figure 5 - Flowchart for](#_Toc88504212) *[Install Update](#_Toc88504212)* [21](#_Toc88504212)

[Figure 6 - Flowchart for](#_Toc88504213) *[Finalize Update](#_Toc88504213)* [22](#_Toc88504213)

# III Abbreviations

API Application Programming Interface

APP Application

COM Communications Module within a Home Appliance

ED End Device

HA Home Appliance

HCA Home Connect Application

LAN Local Area Network

# IV Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Change** | **Name** | **Date** |
| 0.1 | Creation | Clauß | 17.10.2013 |
| 0.2 | Update | Clauß | 31.10.2013 |
| 0.7 | Added description to work flow. Changed Verify to Validation. Added Sequence Diagram.  Added installStarted and aborted command | Clauß | 14.11.2013 |
| 0.9 | Several editorial fixes. Updated content type FWVERSIONS and INTINFO. Updated sequence diagram. Updated /fu/aborted. | Clauß | 04.12.2013 |
| 1.0 | Implemented updates from Groom-in | Clauß | 24.01.2014 |
| 1.1 | Updated validation information to v2 | Clauß | 14.02.2014 |
| 1.2 | Updated validation information to v3 | Clauß | 26.08.2014 |
| 2.0 | Overall update to be compliant with BSH update process. | Tschögl, Clauß | 01.03.2016 |
| 2.1 | Incorporated changes for enhanced low-line concept with multiple update packages | Tschögl, Clauß | 10.05.2016 |
| 3.0 | * Introduced version 3 * Add 3 optional attributes (totalSize, current HAVersion, newHAVersion) to message newUpdateAvailable in version 3 * Add 1 optional attribute (enforceRefresh) to message haConfigRetrigger in version 3 * Add new message downloadProgress to version 3 | Clauß | 27.02.2019 |
| 4.0 | * Introduced version 4 * Completely updated data part of message haConfig in version 4. * Added chapter 2.5 with link to supported/ available data types in JSON * Updated allowed ciphers in chapter 3.4.1 | Clauß | 28.07.2020 |
| 4.1 | * Updated SDD link in inventory parameter description in chapter 6.3.1 * Added new package property file structure in version 6. Chapter 3.4.5 * Added a disclaimer for BTM based devices to chapter 6 | Clauß | 18.03.2021 |
| 4.2 | * Added Trust Config (Security Inventory) in chapter 6 | Arun | 07.07.2021 |
| 5.0 | * Introduced version 5 * Updated chapter 3.4.5 to the final specification | Clauß | 22.11.2021 |

# Introduction

## Goals of the Specification

This document provides a specification which defines the firmware update process for connected devices and the application programming interface (API) of a home appliance (HA) with connectivity features connected to the local customer service (directly at home) or to the BSH internet server also referred as backend.

## Validity of the Document

The document is valid for **local customer service and backend** communication.

This document explicitly contains information about a restricted service. This means, that this service is not available to all interfaces of the HA.

## Reference of Related Documents

[**CBSP13**] *HomeConnect Interface Specification Between Com’Module and Backend*,

PDM Document 5560 0000010292

[**CESP13**] *HomeConnect Interface Specification Between Com’Module and in-home End Devices*, PDM Document 5560 0000010222

[**RFC2119**] *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt.pdf

[4] *Remote Update: HW & SW description, HW\_andSW\_description\_RemoteUpdate.pdf*

## Requirement Levels

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

# Preconditions

The same preconditions as stated in the in-home Wi-Fi specification (see [CESP13]) also apply for this specification. Exceptions are mentioned in the following sub chapters.

## Communication Partners

The communication will be established between the backend and a communications module (COM). For firmware download, an https connection from the COM to another backend server will also be established (concurrent with the standard HCA connection).

Alternatively the customer service may emulate the functionality of the backend servers over a local connection.

Sometimes the wording home appliance (HA) is used, also referring to the communications module.

## Discovery

For the backend connection, discovery is not necessary as the COM initiates the connection establishment. For the local customer service.connection, the same discovery mechanisms as stated in the in-home Wi-Fi specification (see [CESP13]) apply.

## Security

For the backend connection, the same mechanisms as specified in [CBSP13] are used. For the local customer service connection, the same security mechanisms as stated in the in-home Wi-Fi specification (see [CESP13]) apply.

The https connection for firmware download uses a security mechanism similar to that of the backend connection, see *3.4.1 HTTPS Connection for Download of Update Packages*.

## Common Data Structures for All Services

The same common data structures as stated in the in-home Wi-Fi specification (see [CESP13]) also apply for this specification.

## Data Types

The same common and complex data types as stated in the in-home Wi-Fi specification (see [CESP13]) also apply for this specification.

# Workflow of a Firmware Update

## Overview

The following process has been defined for the BSH firmware update supported by COM GEN2:



The process separates the individual steps, which leads to some advantages. The HA can be used while a download is in progress. HA is only not available during install.

### Providing a Firmware to the HCA or Local CS

The product divisions have to provide the HCA with these resources:

* a *Distribution Set* of *Update Packages* containing all packages needed for updating the current HA configuration to a new valid one
* an *Update Manifest* file which contains installation and validation information for the whole *Distribution Set* and is transferred as one of the *Update Package* files
* *Update Package* files containing firmware update files for one or more ECUs
* a *Package Properties* file for each *Update Package* that allows validation of the update package

The HCA is provided with information about the current HA configuration directly from the HA and can decide if there is the need for an update based on the current configuration.

*Note: According to the enhanced lowline update concept, all downloads are controlled by the COM module. However, there are complex modules (e.g. MMB) with firmware sizes that exceed the storage capabilities of the COM module. For these modules the COM has to forward the download data over an internal bus to the module where it is stored. These modules are then also responsible for controlling their own installation (self-update).*

*The need to forward some of the files to other modules during download makes it necessary to split the download in multiple packages.*

## Get HA Configuration



Figure 1 - Flowchart for *Get HA Configuration*

The HCA can actively trigger the HA configuration notification from the COM module.

Additionally, when the COM module boots and establishes the first connection to the HCA, it automatically sends the HA configuration to the HCA. This configuration information comprises all hardware and software versions of the HA electronics including the COM module.

*Note: After the HCA connection is established, the COM module has to wait for the “deviceReady” notification from HCA before sending out the HA Configuration message. This is to make sure that HCA has reached a state where it can handle this message.*

### HA Configuration

Contains information about all updateable hardware/software/content on the HA. It is transferred as an array of entries containing the following information:

* D-Bus node address
* Type of information (e.g. hardware, software)
* ID of the type
* Version of the type
* in case of hardware information type: flags indicating its update capabilities

## New Update Available



Figure 2 - Flowchart for *New Update Available*

When the HCA detects that there are updates available for a HA, it informs the HA by sending a *newUpdateAvailable* messageto the HA.

This message contains a list of package IDs to be downloaded as well as permission flags describing the required permissions for the update procedure. For identification of this concrete update transaction, the HCA also passes an identifier which is the transaction ID. This transaction ID is used for all state notifications and update related messages during the update procedure.

The list of update packages constitutes a *Distribution Set* describing the target configuration of the complete HA. Based on the package ID of a specific package, the COM requests validation information and download links from the HCA for that package.

## Download Update



Figure 3 - Flowchart for *Download Update*

When a new update is available, the COM module asks the user for permission to download (depending on the type of the update and the permission flags received in the *newUpdateAvailable* message). If permission to download is granted, the COM starts downloading all update packages whose package IDs were received in the *newUpdateAvailable* message.

Each package is immediately validated and verified after successful download.

The download does not use the standard HCA <-> COM connection but opens a separate HTTPS connection (see 3.4.1) . The package file download may also be provided by a server different from the HCA. Integrity of the downloaded packages is assured by the validation information stored in the *Package Properties* file provided by the HCA as part of the packageProperties message for each package.

*Note: If a power loss occurs during the state WAIT\_FOR\_DOWNLOAD\_PERMISSION, the HA will remain in this state indefinitely. Therefore the HCA has to retrigger the request for permission by posting a permissionRetrigger message if it detects no progress for a certain time.*

### HTTPS Connection for Download of Update Packages

Update Packages are downloaded through a separate HTTPS connection that does not use the standard HCA-COM connection.

This HTTPS connection is based on TLS 1.2 and uses the same 2-sided certificate check as the standard connection.

For the download connection there are additional requirements:

* The HCA (or whichever server provides the package files) MUST support the HTTP ‘Range’ header [RFC7233] to allow resuming partial downloads.
* It also MUST support the TLS extension ‘*Maximum Fragment Length Negotiation’* [RFC3546] or a similar mechanism to limit the maximum fragment size of SSL records to 2 KiB.
* The only allowed cipher suites for COM2 are:  
  TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CBC\_SHA256 **(mandatory)**TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_GCM\_SHA256  
  TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CBC\_SHA384  
  TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_GCM\_SHA384
* The only allowed cipher suites for System Master (SMM) are:  
  TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_GCM\_SHA256 **(mandatory)**  
  TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_GCM\_SHA384  
  TLS\_ECDHE\_ECDSA\_WITH\_CHACHA20\_POLY1305\_SHA256

### Detailed Download/Validation/Verification Procedure



Figure 4 - Flowchart for download/validation/verification procedure

All packages specified in the *packageIDs* list are downloaded, validated and verified in the following sequence (colored box in the high-level flowchart):

1. The COM identifies the package to download by using the next package ID in the *packageIDs* list.
2. The COM requests from the HCA the *Package Properties* for the next package by passing it the current package ID.
3. The HCA responds with a *Package Properties* message for the package corresponding to the package ID.  
   The *Package Properties* message contains:

* the transaction ID
* the download URL for the package
* the URL of the OCSP server that will be used to check the validity of the certificate provided in the *Package Properties* file
* the dateTime (current date and time of the HCA)
* a *Package* *Properties* file containing validation information for the package (file format: see below)

1. The package file is downloaded from the specified URL.
2. Validation and Verification: After successful download the package is validated against the *Package Properties* fileprovided by the HCA for this package ID. In a further step the package is verified by the HA. A validation or verification error in any package leads to a cancellation of the whole update transaction.
3. If the checks were successful and there are more packages in the *packageIDs* list, continue with (1).

In the validation step the HA checks the authenticity and integrity of the package file using the signature and certificate from the provided *Package Properties* file. In the Verification step, the HA checks if the files contained in the update package are appropriate for the HA and not corrupt (by checking the hash). This should normally be the case as long as the HCA is provided with proper update packages that match the HA.

*Note: The COM resends the request for Package Properties if no Package Properties message from the HCA is received for 30s after the request was issued by the COM module. After additional 30s without a response from HCA the COM switches to state ERROR\_DOWNLOAD.*

*The reason for this mechanism is that there is no acknowledge mechanism for notifications between COM and HCA and there is a possibility of message loss if the connection is instable.*

*Note: The dateTime received from HCA can be used in the http GET request header field “If-Unmodified-Since” when requesting the download file. This way the FDS server can deny the request if the file to download has been modified in the meantime.*

*Note: The URL for the OCSP server is necessary as the URL stored in the certificate itself is not accessible from all regions where HCAs are deployed. Therefore this is a static configuration value of the HCA that is specific for the region where the HCA is located.*

### Properties File content

The *PackageProperties* file is a binary file containing the following information:

* Cryptographic hash of update package
* The destination node for the Update Package
* The unique update package ID
* File size
* Flags
* ECDSA Signature
* Public Signing Certificate

### Properties File structure with validation information (Version 5)

The validation information will have the following structure:

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

+-------------------------------+-------------------------------+

| Length | Version |

+-------------------------------+---------------+---------------+

| DestinationNode | Flags | Reserved |

+-------------------------------+---------------+---------------+

| |

+ PackageID +

| |

+---------------------------------------------------------------+

| ImageLength |

+---------------------------------------------------------------+

| |

+ HashType (8-byte ASCII, unused bytes 0x00) +

| |

+---------------------------------------------------------------+

| |

| PackageHash (Hash of Package File) |

| |

+-------------------------------+-------------------------------+

| SignatureLength | CertificateLength |

+-------------------------------+-------------------------------+

| |

. .

. Signature .

. (variable length) .

. .

| |

+---------------------------------------------------------------+

| |

. .

. Certificate .

. (variable length) .

. .

| |

+---------------------------------------------------------------+

All data in this structure is transmitted in Big Endian.

The Length field contains the number of bytes for the complete validation information including signature and certificate. The maximum Length allowed is 1200 bytes.

The Version field MUST be set to 0x0005.

The DestinationNode field contains the node address where the package shall be stored.

The 8-bit field Flags is used as a bit field to indicate the type of an update package:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Bit** | **Mask** | **Description** |
| UPDATE\_MANIFEST | 7 | 0x80 | This package contains control information for the installation as well as the complete package list for validation. |
| REPAIR | 6 | 0x40 | Omit customer permission request and ignore HA permission request after timeout. |
| FORCED | 5 | 0x20 | Omit customer permission request but do HA permission request. |
| AUTO\_INSTALL\_CAPABLE | 4 | 0x10 | For this package auto installation is allowed. |

Table 1 Flags

The PackageID field is a 64 bit unsigned integer that contains the unique update package ID. The values 0x0 and 0xFFFFFFFFFFFFFFFF are considered invalid values. PackageID has to match the package ID used in the *packagePropertiesRequest* message.

The ImageLength field contains a 32 bit unsigned integer with the length of the firmware image.

The HashType field is an 8-byte ASCII field where unused bytes MUST be set to 0x00. The following HashType values are allowed:

|  |  |  |
| --- | --- | --- |
| **HashType** | **Description** | **Length (in bytes)** |
| SHA-256 | SHA-2 hash with 256 bits in length | 32 |
| SHA-512 | SHA-2 hash with 512 bits in length | 64 |

The PackageHash field contains the hash of the package file. The hash MUST be created with the HashType and MUST match the length of that HashType.

The SignatureLength field contains the length of the Signature which can vary due to its ASN.1 encoding.

The CertificateLength field contains the length of the Certificate.

The Signature will be created using the Elliptic Curve Digital Signature Algorithm (ECDSA) with usage of SHA-256. The Signature will be created of the validation information including the fields from Version up to PackageHash (Version and PackageHash included). The byte order, as defined in the validation information structure, MUST be ensured. The signature is encoded in Distinguished Encoding Rules (DER).

The Certificate field contains the public signing certificate for the key used to generate the signature. It is used for authenticating the origin of the Signature.

The certificate must be X.509 public key certificates encoded in Distinguished Encoding Rules (DER).

### Properties File structure with validation information (Version 6)

The validation information will have the following structure:

0 1 2 3

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

+-------------------------------+-------------------------------+

| Length | Version |

+-------------------------------+---------------+---------------+

| DestinationNode | Flags | Reserved |

+-------------------------------+---------------+---------------+

| |

+ PackageID +

| |

+---------------------------------------------------------------+

| ImageLength |

+---------------------------------------------------------------+

| |

+ HashType (8-byte ASCII, unused bytes 0x00) +

| |

+---------------------------------------------------------------+

| |

| PackageHash (Hash of Package File) |

| |

+-------------------------------+-------------------------------+

| CRC32 (CRC of Package File) |

+---------------------------------------------------------------+

| DIType | DILength | |

+-------------------------------+ .

. .

. DeviceIdentifier (62 bytes) .

| |

+---------------------------------------------------------------+

| SignatureLength | TotalCertificatesLength |

+-------------------------------+-------------------------------+

| |

. .

. Signature .

. (variable length) .

. .

| |

+---------------------------------------------------------------+

| CertificateLength1 | |

+-------------------------------+ .

. .

| Certificate1 |

+---------------------------------------------------------------+

| CertificateLengthN (opt) | |

+-------------------------------+ .

. .

| CertificateN (opt) |

+---------------------------------------------------------------+

All data in this structure is transmitted in Big Endian.

The Length field contains the number of bytes for the complete validation information including signature and certificate. For BTM based devise the maximum Length allowed is 1200 bytes, otherwise the length is not limited.

The Version field MUST be set to 0x0006.

The DestinationNode field contains the node address where the package shall be stored.

The 8-bit field Flags is used as a bit field to indicate the type of an update package:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Bit** | **Mask** | **Description** |
| UPDATE\_MANIFEST | 7 | 0x80 | This package contains control information for the installation as well as the complete package list for validation. |
| REPAIR | 6 | 0x40 | Omit customer permission request and ignore HA permission request after timeout. |
| FORCED | 5 | 0x20 | Omit customer permission request but do HA permission request. |
| AUTO\_INSTALL\_CAPABLE | 4 | 0x10 | For this package auto installation is allowed. |

Table 1 Flags

The PackageID field is a 64 bit unsigned integer that contains the unique update package ID. The values 0x0 and 0xFFFFFFFFFFFFFFFF are considered invalid values. PackageID has to match the package ID used in the *packagePropertiesRequest* message.

The ImageLength field contains a 32 bit unsigned integer with the length of the firmware image.

The HashType field is an 8-byte ASCII field where unused bytes MUST be set to 0x00. The following HashType values are allowed:

|  |  |  |
| --- | --- | --- |
| **HashType** | **Description** | **Length (in bytes)** |
| SHA-256 | SHA-2 hash with 256 bits in length | 32 |
| SHA-512 | SHA-2 hash with 512 bits in length | 64 |

The PackageHash field contains the hash of the package file. The hash MUST be created with the HashType and MUST match the length of that HashType.

The CRC32 field contains the CRC32 of the package file.

The DeviceIdentifier field contains an identification element of the home appliance. This field is always 62 bytes long. Unused bytes will be filled up with 0x00. Depending on the device identifier type (DIType), the length of the device identifier (DILength) information may vary. The following types, with length and their format are defined:

|  |  |  |  |
| --- | --- | --- | --- |
| **DIType** | **DIType Descr.** | **DILength** | **Format of DeviceIdentifier** |
| 0x45 (“E”) | eNumber | 13-21 byte | <VIB> + “/” + <KI> |
| 0x53 (“S”) | SERNR | 18 byte | <SERNR> |
| 0x54 (“T”) | TracingID | 29 byte | <BUID> |

The “S” and “T” type MUST be supported via remote firmware update. Via a local update (e.g. in SMM using the recovery mode) all types MUST be supported.

The SignatureLength field contains the length of the Signature which can vary due to its ASN.1 encoding.

The TotalCertificatesLength field contains the length of all CertificateLength and Certificate elements.

The Signature will be created using the Elliptic Curve Digital Signature Algorithm (ECDSA) with usage of SHA-256. The Signature will be created of the validation information including the fields from Version up to PackageHash (Version and PackageHash included). The byte order, as defined in the validation information structure, MUST be ensured. The signature is encoded in Distinguished Encoding Rules (DER).

The Certificate1 field contains the public signing certificate for the key used to generate the signature. It is used for authenticating the origin of the Signature.

This MAY be followed by additional certificates (one or more CertificateN), with each subsequent certificate being the one used to certify the previous one. The recipient MUST validate the certificate chain according to [RFC5280] section 6.1 and consider the certificate or certificate chain to be invalid if any validation failure occurs.

All certificates must be X.509 public key certificates encoded in Distinguished Encoding Rules (DER).

BTM based devices are limited to only one certificate which must be issued by the trust anchor.

## Install Update



Figure 5 - Flowchart for *Install Update*

The HCA or local CS will be informed once an install has been started. HCA and local CS MUST give the electronic some time to update itself.

The HCA connection can stay active throughout the whole installation. However, remote operation is not possible. If the connection to HCA is lost in this state, it will have no impact and installation will proceed anyway. Yet, Rollback will not be possible if the connection is lost.

Once the install has finished, all components are reset and the Wi-Fi connection is cut. Then the COM will perform a restart and connect Wi-Fi again.

Note that the COM cannot automatically reconnect to the CS after Wi-Fi has been turned on again. Nevertheless automatic reconnection to the HCA is possible.

*Note: If a power loss occurs during the state WAIT\_FOR\_INSTALL\_PERMISSION, the HA will remain in this state indefinitely. Therefore the HCA has to retrigger the request for permission by posting a permissionRetrigger message if it detects no progress for a certain time.*

### Recovery / Rollback

If an update failure is detected, the COM initiates the Recovery procedure where a Recovery package is downloaded from the backend and installed on the HA. In case this Recovery also fails, HA is marked/blocked by backend for further update and customer service has to be informed by the backend.

The detailed process of the recovery / rollback process is not defined yet.

## Finalize Update



Figure 6 - Flowchart for *Finalize Update*

After the installation has finished successfully, the FINALIZING state is entered. The new HA Configuration is collected from the components and sent to the HCA. The collected HA Configuration is checked against the configuration in the Update Manifest and the HA performs a final self test. If all was successful the update proceeds to finish the update.

*Note: Due to restarts of the COM module the HCA connection will be established only temporarily until the update process has finished.*

## Finish Update



Figure 7 - Flowchart for *Finish Update*

After installation was successful, COM and HA perform a restart.

After this final restart the HA is fully functional and the remote operation service is enabled again. The COM is in FINISHED state. It will proceed from this state only after it has received a request for the update state from HCA (to make sure HCA notices the update success). The HCA issues this request a soon as the COM-HCA connection is reestablished. The COM responds with update state FINISHED to indicate that the update transaction has completed.

Finally the COM switches to IDLE state.

# Service – Firmware Update – v2

This service is defined as:

serviceShortName = fu

versionNumber = 2

The service is superseded by version 3 for SMM and later.

Version 2 is valid for COM GEN2 only.

## Overview

Service Provider: COM

Service Consumer: ED

Always available: No

The ED is in this case the backend or the local customer service.

This service can be also accessed during initialization of the connection.

## List of Resources

The following <detailedResource> are used within this service:

|  |  |  |
| --- | --- | --- |
| **Resource** | **Content Type** | **Valid Actions** |
| haConfigRetrigger | CONFIGRETRIGGER | POST |
| haConfig | HACONFIG | NOTIFY |
| newUpdateAvailable | NEWUPDATEAVAIL | POST |
| packagePropertiesRequest | PACKAGEPROPERTIESREQUEST | NOTIFY |
| packageProperties | PACKAGEPROPERTIES | POST |
| state | FUSTATE | GET, NOTIFY |
| permissionRetrigger | PERMRETRIGGER | POST |
| abort | ABORT | POST |

## Structures in item <data>

### Content type HACONFIG

This content type provides information about all current electronics on the HA.

Valid for the following <action> items: NOTIFY

Note: the format for id and version is specified in detail in [4]

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu2:haconfig",

"properties" :

{

"node":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The D-Bus node.",

"required" : true

}

"type":

{

"type" : "integer",

"minimum" : 1,

"maximum" : 65535,

"description" : "The type of the ID and version information.",

"required" : true

},

"id":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 18446744073709551615,

"description" : "The unique ID of the type.",

"required" : false

},

"version":

{

"type" : "object",

"required" : false,

"description" : "The version of the type.",

"properties":

{

"major":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The Major version number.",

"required" : true

},

"minor":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The Minor version number.",

"required" : true

},

"revision":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The Revision (Patch) number.",

"required" : true

},

"build":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 4294967295,

"description" : "The Build number.",

"required" : true

}

}

required" : true

},

"capabilities":

{

"type" : "array",

"minItems" : 0,

"items" :

{

"type" : "string",

"enum" : ["UPDATEABLE","SELF\_UPDATE"]

},

"description" : "Capabilities provided by the hardware.",

"required" : false

}

}

}

### Content type NEWUPDATEAVAIL

This content type provides information about an available update. It specifies all packages contained in an update *Distribution Set*.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu2:newupdateavail",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

},

"packageIDs":

{

"type" : "array",

"minItems" : 2,

"items" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614

},

"description" : "Unique IDs that identify downloadable packages.",

"required" : true

},

"flags":

{

"type" : "array",

"minItems" : 0,

"items" :

{

"type" : "string",

"enum" : ["SKIP\_DOWNLOAD\_PERMISSION","SKIP\_INSTALL\_PERMISSION","REPAIR",

"FORCED"]

},

"description" : "Flags to controll handling during the update.",

"required" : true

}

}

}

### Content type PACKAGEPROPERTIESREQUEST

This content type provides information about transaction ID and requested package ID during the download process.

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu2:packagepropertiesrequest",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

},

"packageID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies an update package.

The values 0x0 and 0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

}

}

}

### Content type PACKAGEPROPERTIES

This content type provides download and validation information for an update package.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu2:packageproperties",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

}, "dateTime" :

{

"type" : "string",

"minLength" : 19,

"maxLength" : 19,

"description" : "Complex Type: DateTime. The HCA’s current http DateTime.",

"required" : true

},

"link" :

{

"type" : "string",

"minLength" : 1,

"maxLength" : 127,

"description" : "Link for downloading the update package.",

"required" : true

},

"ocspURL" :

{

"type" : "string",

"minLength" : 1,

"maxLength" : 127,

"description" : "URL specifying the location of a valid OCSP server.",

"required" : true

},

"packageProperties" :

{

"type" : "string",

"maxLength" : 1600,

"description" : "Package Properties file containing validation information

for the update package.",

"required" : true

} }

}

### Content type FUSTATE

This content type provides information of the firmware update state on the HA.

Valid for the following <action> items: NOTIFY, RESPONSE

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu2:fustate",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : false

},

"state" :

{

"type" : "string",

"enum" : ["INIT","IDLE","WAIT\_FOR\_DOWNLOAD\_PERMISSION",

"DOWNLOADING",

"WAIT\_FOR\_INSTALL\_PERMISSION","INSTALLING",

"FINALIZING","FINISHED",

"ERROR","ERROR\_DO\_INVENTORY",

"ERROR\_DOWNLOAD","ERROR\_VALIDATION",

"ERROR\_VERIFICATION","ERROR\_INSTALL",

"ERROR\_FINALIZE"],

"description" : "The current state of the firmware update.",

"required" : true

},

"packageID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies an update package.

The values 0x0 and 0xFFFFFFFFFFFFFFFF are invalid.",

"required" : false

},

"errorID" :

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "Generic error code.",

"required" : false

},

"haErrorID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 65535,

"description" : "Device specific error code.",

"required" : false

},

"reason" :

{

"type" : "string",

"maxLength" : 127,

"description" : "Optionally an error reason in case of error state.",

"required" : false

},

"errorLevel" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 2,

"description" : "Specifies the severity of an error.",

"required" : false

}

}

}

### Content type CONFIGRETRIGGER

This content type provides information about transaction ID.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu2:configretrigger",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": false

}

}

}

### Content type PERMRETRIGGER

This content type provides information about transaction ID and permissions to request.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu2:permretrigger",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": true

},

"permission" :

{

"type" : "string",

"enum" : ["DOWNLOAD","INSTALL"],

"description" : "The current state of the firmware update.",

"required" : true

},

}

}

### Content type ABORT

This content type provides information about transaction ID.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu2:abort",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": true

}

}

}

## Functionalities

### Trigger HA Configuration

The backend can request the current version from the HA.

The backend sends a POST to resource “haConfigRetrigger” using content type CONFIGRETRIGGER. If the message is sent during an active transaction, the transactionID has to be included.

The response will not contain any data.

*Note: As the process of doing the inventory of the whole HA configuration can take several seconds, the HA configuration is returned asynchronously with a separate NOTIFY message. The response is therefore only signaling that the trigger was accepted.*

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 2,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 2,

"action" : "RESPONSE",

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 2,

"action" : "RESPONSE",

"code" : 400

}

### Notify HA Configuration

The COM sends a notification about its configuration once the connection HCA<->COM is established for the first time after restart or after it receives a haConfigRetrigger message.

The notification from the resource “haConfig” will contain a list of elements of the content type HACONFIG.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042940,

"resource" : "/fu/haConfig",

"version" : 2,

"action" : "NOTIFY",

"data" :

[

{

"node" : 177,

"type" : 0,

"id" : 660604,

"version" :

{

"major" : 1,

"minor" : 9,

"revision" : 0,

"build" : 289322

},

"capabilities" : ["UPDATEABLE","SELF\_UPDATE"]

},

{

"node" : 177,

"type" : 1,

"id" : 160603,

"version" :

{

"major" : 2,

"minor" : 0,

"revision" : 1,

"build" : 180

}

}

]

}

### Post New Update Available

The backend sends a POST request to the resource “newUpdateAvailable” using the content type NEWUPDATEAVAIL. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 2,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"packageIDs" : [123456789012,210469270401,220604080568],

"flags" : ["SKIP\_DOWNLOAD\_PERMISSION"]

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 2,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 2,

"action" : "RESPONSE",

"code" : 400

}

### Notify Package Properties Request

With the notification *packagePropertiesRequest* the COM informs the HCA that it needs the Package Properties information for a download package with a specific package ID.

The notification from the resource “packagePropertiesRequest” contains the content type PACKAGEPROPERTIESREQUEST.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packagePropertiesRequest",

"version" : 2,

"action" : "NOTIFY",

"data" :

[

{

"transactionID": 105230,

"packageID" : 123456789012

}

]

}

### Post Package Properties

The backend sends a POST request to the resource “packageProperties” using the content type PACKAGEPROPERTIES. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

*Note: An error response may indicate an invalid Package Properties file.*

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 2,

"action" : "POST",

"data" :

[

{

"transactionID": 105230,

"dateTime" : "2016-02-29T22:15:01",

"link" : "https://fu-server-domain/path/specific-update-package",

"ocspURL" : "http://ocsp-server-domain:8080",

"packageProperties" : " base64url encoded binary data "

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 2,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 2,

"action" : "RESPONSE",

"code" : 400

}

### Get Firmware Update State of HA

The backend can request the current state of the firmware update on the HA.

The backend sends a GET request to “state”. The response will contain a single element of the content type FUSTATE.

**Example:**

# Request

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 2,

"action" : "GET"

}

# Response

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 2,

"action" : "RESPONSE",

"data" :

[

{

"transactionID" : 105230

"state" : "FINALIZING",

}

]

}

### Notify Firmware Update State of HA

The COM sends a notification about the firmware update state to the HCA.

The notification from the resource “state” will contain a list of elements of the content type FUSTATE.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 2,

"action" : "NOTIFY",

"data" :

[

{

"transactionID" : 105230

"state" : "ERROR\_VALIDATION\_FAILED",

"packageID" : 123456789012,

"errorID" : 61005,

"haErrorID" : 301,

"reason" : "Invalid signature in package.",

"errorLevel" : 1

}

]

}

### Post Permission Retrigger

The backend sends a POST request to the resource “permissionRetrigger” using the content type PERMRETRIGGER. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 2,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"permission" : "INSTALL"

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 2,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 2,

"action" : "RESPONSE",

"code" : 400

}

### Set Aborted

If an error occurs during the firmware update, the HCA can send an abort message specifying the current transaction ID.

The backend sends a POST request to the resource “abort” using content type ABORT. The response MUST NOT contain a <data> item. The <code> item MAY display any errors that occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 2,

"action" : "POST"

"data" :

[

{

"transactionID" : 105230

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 2,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 2,

"action" : "RESPONSE",

"code" : 400

}

## Application Guidelines / Behavior

This service does not have any specific application guidelines / behavior beyond those already specified.

# Service – Firmware Update – v3

This service is defined as:

serviceShortName = fu

versionNumber = 3

## Overview

Service Provider: COM

Service Consumer: ED

Always available: No

The ED is in this case the backend.

## List of Resources

The following <detailedResource> are used within this service:

|  |  |  |
| --- | --- | --- |
| **Resource** | **Content Type** | **Valid Actions** |
| haConfigRetrigger | CONFIGRETRIGGER | POST |
| haConfig | HACONFIG | NOTIFY |
| newUpdateAvailable | NEWUPDATEAVAIL | POST |
| packagePropertiesRequest | PACKAGEPROPERTIESREQUEST | NOTIFY |
| packageProperties | PACKAGEPROPERTIES | POST |
| state | FUSTATE | GET, NOTIFY |
| permissionRetrigger | PERMRETRIGGER | POST |
| abort | ABORT | POST |
| downloadProgress | DOWNLOADPROGRESS | NOTIFY |

## Structures in item <data>

### Content type HACONFIG

This content type provides information about all current electronics on the HA.

Valid for the following <action> items: NOTIFY

Note: the format for id and version is specified in detail in [4]

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:haconfig",

"properties" :

{

"node":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The D-Bus node.",

"required" : true

}

"type":

{

"type" : "integer",

"minimum" : 1,

"maximum" : 65535,

"description" : "The type of the ID and version information.",

"required" : true

},

"id":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 18446744073709551615,

"description" : "The unique ID of the type.",

"required" : false

},

"version":

{

"type" : "object",

"required" : false,

"description" : "The version of the type.",

"properties":

{

"major":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The Major version number.",

"required" : true

},

"minor":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The Minor version number.",

"required" : true

},

"revision":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The Revision (Patch) number.",

"required" : true

},

"build":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 4294967295,

"description" : "The Build number.",

"required" : true

}

}

required" : true

},

"capabilities":

{

"type" : "array",

"minItems" : 0,

"items" :

{

"type" : "string",

"enum" : ["UPDATEABLE","SELF\_UPDATE"]

},

"description" : "Capabilities provided by the hardware.",

"required" : false

}

}

}

### Content type NEWUPDATEAVAIL

This content type provides information about an available update. It specifies all packages contained in an update *Distribution Set*.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:newupdateavail",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

},

"packageIDs":

{

"type" : "array",

"minItems" : 2,

"items" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614

},

"description" : "Unique IDs that identify downloadable packages.",

"required" : true

},

"flags":

{

"type" : "array",

"minItems" : 0,

"items" :

{

"type" : "string",

"enum" : ["SKIP\_DOWNLOAD\_PERMISSION","SKIP\_INSTALL\_PERMISSION","REPAIR",

"FORCED"]

},

"description" : "Flags to controll handling during the update.",

"required" : true

},

"totalSize" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The total size in bytes of all packages for this update.

This parameter is not supported by COM GEN2 and shall only

be sent for System Master generation or newer.",

"required" : optional

},

"currentHAVersion" :

{

"type" : "string",

"minLength" : 5,

"maxLength" : 127,

"description" : "The current HA version in form of a string containing the

triplet - hardware version, RAC version and firmware

version. This parameter is not supported by COM GEN2 and

shall only be sent for System Master generation or newer.",

"required" : optional

},

"newHAVersion" :

{

"type" : "string",

"minLength" : 5,

"maxLength" : 127,

"description" : "The new HA version in form of a string containing the

triplet - hardware version, RAC version and firmware

version. This parameter is not supported by COM GEN2 and

shall only be sent for System Master generation or newer.",

"required" : optional

}

}

}

### Content type PACKAGEPROPERTIESREQUEST

This content type provides information about transaction ID and requested package ID during the download process.

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:packagepropertiesrequest",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

},

"packageID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies an update package.

The values 0x0 and 0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

}

}

}

### Content type PACKAGEPROPERTIES

This content type provides download and validation information for an update package.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:packageproperties",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

}, "dateTime" :

{

"type" : "string",

"minLength" : 19,

"maxLength" : 19,

"description" : "Complex Type: DateTime. The HCA’s current http DateTime.",

"required" : true

},

"link" :

{

"type" : "string",

"minLength" : 1,

"maxLength" : 127,

"description" : "Link for downloading the update package.",

"required" : true

},

"ocspURL" :

{

"type" : "string",

"minLength" : 1,

"maxLength" : 127,

"description" : "URL specifying the location of a valid OCSP server.",

"required" : true

},

"packageProperties" :

{

"type" : "string",

"maxLength" : 1600,

"description" : "Package Properties file containing validation information

for the update package.",

"required" : true

} }

}

### Content type FUSTATE

This content type provides information of the firmware update state on the HA.

Valid for the following <action> items: NOTIFY, RESPONSE

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:fustate",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : false

},

"state" :

{

"type" : "string",

"enum" : ["INIT","IDLE","WAIT\_FOR\_DOWNLOAD\_PERMISSION",

"DOWNLOADING",

"WAIT\_FOR\_INSTALL\_PERMISSION","INSTALLING",

"FINALIZING","FINISHED",

"ERROR","ERROR\_DO\_INVENTORY",

"ERROR\_DOWNLOAD","ERROR\_VALIDATION",

"ERROR\_VERIFICATION","ERROR\_INSTALL",

"ERROR\_FINALIZE"],

"description" : "The current state of the firmware update.",

"required" : true

},

"packageID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies an update package.

The values 0x0 and 0xFFFFFFFFFFFFFFFF are invalid.",

"required" : false

},

"errorID" :

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "Generic error code.",

"required" : false

},

"haErrorID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 65535,

"description" : "Device specific error code.",

"required" : false

},

"reason" :

{

"type" : "string",

"maxLength" : 127,

"description" : "Optionally an error reason in case of error state.",

"required" : false

},

"errorLevel" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 2,

"description" : "Specifies the severity of an error.",

"required" : false

}

}

}

### Content type CONFIGRETRIGGER

This content type provides information about transaction ID.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:configretrigger",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": false

},

"enforceRefresh" :

{

"type" : "boolean",

"description" : "Indicator whether the HA configuration shall be gathered

internally in the HA anew or not. If false or not present,

the HA will respond with a previously saved configuration.",

"required" : false

}

}

}

### Content type PERMRETRIGGER

This content type provides information about transaction ID and permissions to request.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:permretrigger",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": true

},

"permission" :

{

"type" : "string",

"enum" : ["DOWNLOAD","INSTALL"],

"description" : "The current state of the firmware update.",

"required" : true

},

}

}

### Content type ABORT

This content type provides information about transaction ID.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:abort",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": true

}

}

}

### Content type DOWNLOADPROGRESS

This content type provides information about all current electronics on the HA.

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu3:downloadprogress",

"properties" :

{

"progress":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 100,

"description" : "The progress of the download phase in percent. All packages

of a firmware update are covered in the progress indication.

Resolution and time restrictions have to be applied on this

value to prevent a flodding with messages.",

"required" : true

}

## Functionalities

### Trigger HA Configuration

The backend can request the current home appliance configuration from the HA.

The backend sends a POST to resource “haConfigRetrigger” using content type CONFIGRETRIGGER. If the message is sent during an active transaction, the transactionID has to be included.

The response will not contain any data.

*Note: As the process of doing the inventory of the whole HA configuration can take several seconds, the HA configuration is returned asynchronously with a separate NOTIFY message. The response is therefore only signaling that the trigger was accepted.*

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 3,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"enforceRefresh" : true

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 3,

"action" : "RESPONSE",

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 3,

"action" : "RESPONSE",

"code" : 400

}

### Notify HA Configuration

The COM sends a notification about its configuration once the connection HCA<->COM is established for the first time after restart or after it receives a haConfigRetrigger message.

The notification from the resource “haConfig” will contain a list of elements of the content type HACONFIG.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042940,

"resource" : "/fu/haConfig",

"version" : 3,

"action" : "NOTIFY",

"data" :

[

{

"node" : 177,

"type" : 0,

"id" : 660604,

"version" :

{

"major" : 1,

"minor" : 9,

"revision" : 0,

"build" : 289322

},

"capabilities" : ["UPDATEABLE","SELF\_UPDATE"]

},

{

"node" : 177,

"type" : 1,

"id" : 160603,

"version" :

{

"major" : 2,

"minor" : 0,

"revision" : 1,

"build" : 180

}

}

]

}

### Post New Update Available

The backend sends a POST request to the resource “newUpdateAvailable” using the content type NEWUPDATEAVAIL. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 3,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"packageIDs" : [123456789012,210469270401,220604080568],

"flags" : ["SKIP\_DOWNLOAD\_PERMISSION"],

"totalSize" : 1572864,

"currentHAVersion" : "9.0.396",

"newHAVersion" : "9.0.448"

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 3,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 3,

"action" : "RESPONSE",

"code" : 400

}

### Notify Package Properties Request

With the notification *packagePropertiesRequest* the COM informs the HCA that it needs the Package Properties information for a download package with a specific package ID.

The notification from the resource “packagePropertiesRequest” contains the content type PACKAGEPROPERTIESREQUEST.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packagePropertiesRequest",

"version" : 3,

"action" : "NOTIFY",

"data" :

[

{

"transactionID": 105230,

"packageID" : 123456789012

}

]

}

### Post Package Properties

The backend sends a POST request to the resource “packageProperties” using the content type PACKAGEPROPERTIES. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

*Note: An error response may indicate an invalid Package Properties file.*

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 3,

"action" : "POST",

"data" :

[

{

"transactionID": 105230,

"dateTime" : "2016-02-29T22:15:01",

"link" : "https://fu-server-domain/path/specific-update-package",

"ocspURL" : "http://ocsp-server-domain:8080",

"packageProperties" : " base64url encoded binary data "

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 3,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 3,

"action" : "RESPONSE",

"code" : 400

}

### Get Firmware Update State of HA

The backend can request the current state of the firmware update on the HA.

The backend sends a GET request to “state”. The response will contain a single element of the content type FUSTATE.

**Example:**

# Request

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 3,

"action" : "GET"

}

# Response

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 3,

"action" : "RESPONSE",

"data" :

[

{

"transactionID" : 105230

"state" : "FINALIZING",

}

]

}

### Notify Firmware Update State of HA

The COM sends a notification about the firmware update state to the HCA.

The notification from the resource “state” will contain a list of elements of the content type FUSTATE.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 3,

"action" : "NOTIFY",

"data" :

[

{

"transactionID" : 105230

"state" : "ERROR\_VALIDATION\_FAILED",

"packageID" : 123456789012,

"errorID" : 61005,

"haErrorID" : 301,

"reason" : "Invalid signature in package.",

"errorLevel" : 1

}

]

}

### Post Permission Retrigger

The backend sends a POST request to the resource “permissionRetrigger” using the content type PERMRETRIGGER. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 3,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"permission" : "INSTALL"

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 3,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 3,

"action" : "RESPONSE",

"code" : 400

}

### Set Aborted

If an error occurs during the firmware update, the HCA can send an abort message specifying the current transaction ID.

The backend sends a POST request to the resource “abort” using content type ABORT. The response MUST NOT contain a <data> item. The <code> item MAY display any errors that occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 3,

"action" : "POST"

"data" :

[

{

"transactionID" : 105230

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 3,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 3,

"action" : "RESPONSE",

"code" : 400

}

### Notify Download Progress

The COM sends a notification about the download progress to the HCA. This notify is only sent during the downloading phase of the firmware update. The download progress is comprising the download of all packages to a certain firmware update. The progress has a range from 0 to 100 and is indicating the percentage of the progress.

To restrict the amount of messages for this notification, the HA will two criteria’s which must be met before a new notification is allowed to be sent:

* The progress must have a resolution of 5% (e.g. valid values are 0, 5, 10, 15, ...)
* The last notification of the progress is more than 1 second in the past

Example:

9:24:54 🡺 "progress" : 0

9:24:55 🡺 "progress" : 5

9:24:59 🡺 "progress" : 10

9:25:00 🡺 "progress" : 20

…

For implementation keep in mind, that a round down rule must be applied. E.g. 99MB/100MB must result in 95% instead of 100% progress. Furthermore make the resolution and time restriction easily changeable, e.g. via a constant.

The notification from the resource “downloadProgress” will contain a single element of the content type DOWNLOADPROGRESS.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/downloadProgress",

"version" : 3,

"action" : "NOTIFY",

"data" :

[

{

"progress" : 50

}

]

}

## Application Guidelines / Behavior

This service does not have any specific application guidelines / behavior beyond those already specified.

# Service – Firmware Update – v4

This service is defined as:

serviceShortName = fu

versionNumber = 4

DISCLAIMER:

BTM is also using this service. For better distinction of BTM based devices it uses the special

versionNumber = 4 + 100 = 104.

## Overview

Service Provider: COM

Service Consumer: ED

Always available: No

The ED is in this case the backend.

## List of Resources

The following <detailedResource> are used within this service:

|  |  |  |
| --- | --- | --- |
| **Resource** | **Content Type** | **Valid Actions** |
| haConfigRetrigger | CONFIGRETRIGGER | POST |
| haConfig | HACONFIG | NOTIFY |
| newUpdateAvailable | NEWUPDATEAVAIL | POST |
| packagePropertiesRequest | PACKAGEPROPERTIESREQUEST | NOTIFY |
| packageProperties | PACKAGEPROPERTIES | POST |
| state | FUSTATE | GET, NOTIFY |
| permissionRetrigger | PERMRETRIGGER | POST |
| abort | ABORT | POST |
| downloadProgress | DOWNLOADPROGRESS | NOTIFY |
| trustConfig | TRUSTCONFIG | NOTIFY |

## Structures in item <data>

### Content type HACONFIG

This content type provides information about all current electronics on the HA.

Valid for the following <action> items: NOTIFY

Note: the format for id and version is specified in detail in [4]

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:haconfig",

"properties" :

{

"deviceData":

{

"type" : "object",

"required" : true,

"description" : "The basic device data. Most of this data is also transmitted

with the /iz/info message.",

"properties":

{

"haID":

{

"type" : "string",

"minLength" : 18,

"maxLength" : 18,

"description" : "The home appliance device ID (HAID).",

"required" : true

},

"deviceType" :

{

"type" : "string",

"description" : "A device type according to the device description file

specification.",

"required" : true

},

"brand":

{

"type" : "string",

"maxLength" : 32,

"description" : "The brand of the home appliance according to the device

description file specification.",

"required" : true

},

"vib":

{

"type" : "string",

"maxLength" : 32,

"description" : "The vib of the home appliance.",

"required" : true

},

"customerIndex":

{

"type" : "string",

"minLength" : 2,

"maxLength" : 2,

"description" : "An alpha numeric indexer for the customer service.",

"required" : true

},

"mac":

{

"type" : "string",

"minLength" : 17,

"maxLength" : 17,

"description" : "Complex Type: EuiAddress. The MAC address of the

registered device. If the device posess multiple MAC

addresses, then the Wi-Fi MAC address will be

returned.",

"required" : true

},

"ddfMajorVersion":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The major version part of the DDF (Device Description

File) version.",

"required" : true

},

"fdString":

{

"type" : "string",

"minLength" : 4,

"maxLength" : 4,

"description" : "FD information about the home appliance. This has the

format 'YYMM'. But is YY = (YYYY-20)%100.",

"required" : true

},

"manufacturingTS":

{

"type" : "string",

"minLength" : 19,

"maxLength" : 19,

"description" : "Complex Type: DateTime. The manufacturing timestamp of

the home appliance in ISO8601 format.",

"required" : true

},

"hub":

{

"type" : "string",

"maxLength" : 31,

"description" : "The short name of the HUB.",

"required" : true

}

}

},

"inventory":

{

"description" : "The inventory of the home appliance. The definition of this

object can be found in: https://scr.bsh-sdd.com/projects/HC/repos/architecture/browse/schemas/firmware-module/smm-inventory/field/schema.smm-field-inventory.0.1.0.json",

"$ref" : "bshg.com/smm-inventory.json",

"required" : true

}

}

}

### Content type NEWUPDATEAVAIL

This content type provides information about an available update. It specifies all packages contained in an update *Distribution Set*.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:newupdateavail",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

},

"packageIDs":

{

"type" : "array",

"minItems" : 2,

"items" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614

},

"description" : "Unique IDs that identify downloadable packages.",

"required" : true

},

"flags":

{

"type" : "array",

"minItems" : 0,

"items" :

{

"type" : "string",

"enum" : ["SKIP\_DOWNLOAD\_PERMISSION","SKIP\_INSTALL\_PERMISSION","REPAIR",

"FORCED"]

},

"description" : "Flags to controll handling during the update.",

"required" : true

},

"totalSize" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The total size in bytes of all packages for this update.

This parameter is not supported by COM GEN2 and shall only

be sent for System Master generation or newer.",

"required" : optional

},

"currentHAVersion" :

{

"type" : "string",

"minLength" : 5,

"maxLength" : 127,

"description" : "The current HA version in form of a string containing the

triplet - hardware version, RAC version and firmware

version. This parameter is not supported by COM GEN2 and

shall only be sent for System Master generation or newer.",

"required" : optional

},

"newHAVersion" :

{

"type" : "string",

"minLength" : 5,

"maxLength" : 127,

"description" : "The new HA version in form of a string containing the

triplet - hardware version, RAC version and firmware

version. This parameter is not supported by COM GEN2 and

shall only be sent for System Master generation or newer.",

"required" : optional

}

}

}

### Content type PACKAGEPROPERTIESREQUEST

This content type provides information about transaction ID and requested package ID during the download process.

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:packagepropertiesrequest",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

},

"packageID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies an update package.

The values 0x0 and 0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

}

}

}

### Content type PACKAGEPROPERTIES

This content type provides download and validation information for an update package.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:packageproperties",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

}, "dateTime" :

{

"type" : "string",

"minLength" : 19,

"maxLength" : 19,

"description" : "Complex Type: DateTime. The HCA’s current http DateTime.",

"required" : true

},

"link" :

{

"type" : "string",

"minLength" : 1,

"maxLength" : 127,

"description" : "Link for downloading the update package.",

"required" : true

},

"ocspURL" :

{

"type" : "string",

"minLength" : 1,

"maxLength" : 127,

"description" : "URL specifying the location of a valid OCSP server.",

"required" : true

},

"packageProperties" :

{

"type" : "string",

"maxLength" : 1600,

"description" : "Package Properties file containing validation information

for the update package.",

"required" : true

} }

}

### Content type FUSTATE

This content type provides information of the firmware update state on the HA.

Valid for the following <action> items: NOTIFY, RESPONSE

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:fustate",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : false

},

"state" :

{

"type" : "string",

"enum" : ["INIT","IDLE","WAIT\_FOR\_DOWNLOAD\_PERMISSION",

"DOWNLOADING",

"WAIT\_FOR\_INSTALL\_PERMISSION","INSTALLING",

"FINALIZING","FINISHED",

"ERROR","ERROR\_DO\_INVENTORY",

"ERROR\_DOWNLOAD","ERROR\_VALIDATION",

"ERROR\_VERIFICATION","ERROR\_INSTALL",

"ERROR\_FINALIZE"],

"description" : "The current state of the firmware update.",

"required" : true

},

"packageID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies an update package.

The values 0x0 and 0xFFFFFFFFFFFFFFFF are invalid.",

"required" : false

},

"errorID" :

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "Generic error code.",

"required" : false

},

"haErrorID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 65535,

"description" : "Device specific error code.",

"required" : false

},

"reason" :

{

"type" : "string",

"maxLength" : 127,

"description" : "Optionally an error reason in case of error state.",

"required" : false

},

"errorLevel" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 2,

"description" : "Specifies the severity of an error.",

"required" : false

}

}

}

### Content type CONFIGRETRIGGER

This content type provides information about transaction ID.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:configretrigger",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": false

},

"enforceRefresh" :

{

"type" : "boolean",

"description" : "Indicator whether the HA configuration shall be gathered

internally in the HA anew or not. If false or not present,

the HA will respond with a previously saved configuration.",

"required" : false

}

}

}

### Content type PERMRETRIGGER

This content type provides information about transaction ID and permissions to request.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:permretrigger",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": true

},

"permission" :

{

"type" : "string",

"enum" : ["DOWNLOAD","INSTALL"],

"description" : "The current state of the firmware update.",

"required" : true

},

}

}

### Content type ABORT

This content type provides information about transaction ID.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:abort",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": true

}

}

}

### Content type DOWNLOADPROGRESS

This content type provides information about all current electronics on the HA.

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu4:downloadprogress",

"properties" :

{

"progress":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 100,

"description" : "The progress of the download phase in percent. All packages

of a firmware update are covered in the progress indication.

Resolution and time restrictions have to be applied on this

value to prevent a flodding with messages.",

"required" : true

}

### Content type TRUSTCONFIG

This content type provides information about the security related configuration on the HA

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id": “urn:schemas-bshg-com:js:data:fu4:trustconfig”

"properties":

{

"hab":

{

"type": "object",

"properties":

{

"keySetId":

{

"type": "integer",

"minimum": 0,

"maximum": 255

"required": true

},

"keyNumber":

{

"type": "array",

"minItems": 1,

"items":

{

"type": "integer",

"minimum": 1,

"maximum": 4

"required": true

}

},

"required": true

}

},

"dm-verity":

{

"type": "object",

"properties":

{

"trustAnchorFingerprints":

{

"type": "array",

"minItems": 1,

"items":

{

"type": "string",

"pattern": "^([a-zA-Z0-9]{2}:){15}[a-zA-Z0-9]{2}$",

"minLength": 47,

"maxLength": 47

"required": true

}

}

}

"required": true

}

}

}

## Functionalities

### Trigger HA Configuration

The backend can request the current home appliance configuration from the HA.

The backend sends a POST to resource “haConfigRetrigger” using content type CONFIGRETRIGGER. If the message is sent during an active transaction, the transactionID has to be included.

The response will not contain any data.

*Note: As the process of doing the inventory of the whole HA configuration can take several seconds, the HA configuration is returned asynchronously with a separate NOTIFY message. The response is therefore only signaling that the trigger was accepted.*

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 4,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"enforceRefresh" : true

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 4,

"action" : "RESPONSE",

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 4,

"action" : "RESPONSE",

"code" : 400

}

### Notify HA Configuration

The COM sends a notification about its configuration once the connection HCA<->COM is established for the first time after restart or after it receives a haConfigRetrigger message.

The notification from the resource “haConfig” will contain a list of elements of the content type HACONFIG.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042940,

"resource" : "/fu/haConfig",

"version" : 4,

"action" : "NOTIFY",

"data" :

[

{

"deviceData" :

{

"haID" : "ABCD12345ABCD12345",

"deviceType" : "Dishwasher",

"brand" : "BOSCH",

"vib" : "SMI69T45EU",

"customerIndex" : "H9",

"mac" : "11-22-33-44-55-66",

"ddfMajorVersion" : 1,

"fdString" : "0005",

"manufacturingTS" : "2020-05-14T16:10:59",

"hub" : "EU-PRODUCTION"

},

"inventory" :

{

"ECUsSchemaVersion" : 2,

"ECUs" :

[

{

"tracingId" : "80011370220000440335000000123",

"name" : "SMM",

"hardwareId" : 216739047866238720,

"version" :

{

"major" : 1,

"minor" : 9,

"revision" : 0,

"build" : 289322

},

"firmware" :

[

{

"firmwareId": 283686952306183,

"version":

{

"major": 4660,

"minor": 39030,

"revision": 65244,

"build": 11259375

}

},

{

"firmwareId": 843793843548540,

"version":

{

"major": 3,

"minor": 3,

"revision": 0,

"build": 0

}

}

],

"content":

[

{

"contentId": 1234,

"version":

{

"major": 1,

"minor": 4,

"revision": 0,

"build": 0

}

}

]

}

]

}

}

]

}

### Post New Update Available

The backend sends a POST request to the resource “newUpdateAvailable” using the content type NEWUPDATEAVAIL. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 4,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"packageIDs" : [123456789012,210469270401,220604080568],

"flags" : ["SKIP\_DOWNLOAD\_PERMISSION"],

"totalSize" : 1572864,

"currentHAVersion" : "9.0.396",

"newHAVersion" : "9.0.448"

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 4,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 4,

"action" : "RESPONSE",

"code" : 400

}

### Notify Package Properties Request

With the notification *packagePropertiesRequest* the COM informs the HCA that it needs the Package Properties information for a download package with a specific package ID.

The notification from the resource “packagePropertiesRequest” contains the content type PACKAGEPROPERTIESREQUEST.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packagePropertiesRequest",

"version" : 4,

"action" : "NOTIFY",

"data" :

[

{

"transactionID": 105230,

"packageID" : 123456789012

}

]

}

### Post Package Properties

The backend sends a POST request to the resource “packageProperties” using the content type PACKAGEPROPERTIES. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

*Note: An error response may indicate an invalid Package Properties file.*

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 4,

"action" : "POST",

"data" :

[

{

"transactionID": 105230,

"dateTime" : "2016-02-29T22:15:01",

"link" : "https://fu-server-domain/path/specific-update-package",

"ocspURL" : "http://ocsp-server-domain:8080",

"packageProperties" : " base64url encoded binary data "

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 4,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 4,

"action" : "RESPONSE",

"code" : 400

}

### Get Firmware Update State of HA

The backend can request the current state of the firmware update on the HA.

The backend sends a GET request to “state”. The response will contain a single element of the content type FUSTATE.

**Example:**

# Request

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 4,

"action" : "GET"

}

# Response

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 4,

"action" : "RESPONSE",

"data" :

[

{

"transactionID" : 105230

"state" : "FINALIZING",

}

]

}

### Notify Firmware Update State of HA

The COM sends a notification about the firmware update state to the HCA.

The notification from the resource “state” will contain a list of elements of the content type FUSTATE.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 4,

"action" : "NOTIFY",

"data" :

[

{

"transactionID" : 105230

"state" : "ERROR\_VALIDATION\_FAILED",

"packageID" : 123456789012,

"errorID" : 61005,

"haErrorID" : 301,

"reason" : "Invalid signature in package.",

"errorLevel" : 1

}

]

}

### Post Permission Retrigger

The backend sends a POST request to the resource “permissionRetrigger” using the content type PERMRETRIGGER. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 4,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"permission" : "INSTALL"

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 4,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 4,

"action" : "RESPONSE",

"code" : 400

}

### Set Aborted

If an error occurs during the firmware update, the HCA can send an abort message specifying the current transaction ID.

The backend sends a POST request to the resource “abort” using content type ABORT. The response MUST NOT contain a <data> item. The <code> item MAY display any errors that occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 4,

"action" : "POST"

"data" :

[

{

"transactionID" : 105230

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 4,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 4,

"action" : "RESPONSE",

"code" : 400

}

### Notify Download Progress

The COM sends a notification about the download progress to the HCA. This notify is only sent during the downloading phase of the firmware update. The download progress is comprising the download of all packages to a certain firmware update. The progress has a range from 0 to 100 and is indicating the percentage of the progress.

To restrict the amount of messages for this notification, the HA will two criteria’s which must be met before a new notification is allowed to be sent:

* The progress must have a resolution of 5% (e.g. valid values are 0, 5, 10, 15, ...)
* The last notification of the progress is more than 1 second in the past

Example:

9:24:54 🡺 "progress" : 0

9:24:55 🡺 "progress" : 5

9:24:59 🡺 "progress" : 10

9:25:00 🡺 "progress" : 20

…

For implementation keep in mind, that a round down rule must be applied. E.g. 99MB/100MB must result in 95% instead of 100% progress. Furthermore make the resolution and time restriction easily changeable, e.g. via a constant.

The notification from the resource “downloadProgress” will contain a single element of the content type DOWNLOADPROGRESS.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/downloadProgress",

"version" : 4,

"action" : "NOTIFY",

"data" :

[

{

"progress" : 50

}

]

}

### Notify Trust Config

The COM sends a notification about its security configuration every time it sends it HA configuration (HACONFIG).

The notification from the resource “trustConfig” will contain a list of elements of the content type TRUSTCONFIG.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042940,

"resource" : "/fu/trustConfig",

"version" : 4,

"action" : "NOTIFY",

"data" :

[

{

"hab":

{

"keySetId": 1,

"keyNumber":

[

1,

2

]

},

"dm-verity":

{

"trustAnchorFingerprints":

[

"DO:xx:NO:Tx:xx:US:Ex:TH:IS:xx:ON:LY:xx:SA:MP:LE"

]

}

}

]

}

## Application Guidelines / Behavior

This service does not have any specific application guidelines / behavior beyond those already specified.

# Service – Firmware Update – v5

This service is defined as:

serviceShortName = fu

versionNumber = 5

DISCLAIMER:

BTM is also using this service. For better distinction of BTM based devices it uses the special

versionNumber = 5 + 100 = 105.

Furthermore, this service in version 5.0, has no changes in any message format. Only the support for the newer PPF format changed. So all older firmware update versions supported PPF v5 (with Worldline certificates), while this service supports PFF v6 (with new BSH-PKI certificates).

## Overview

Service Provider: COM

Service Consumer: ED

Always available: No

The ED is in this case the backend.

## List of Resources

The following <detailedResource> are used within this service:

|  |  |  |
| --- | --- | --- |
| **Resource** | **Content Type** | **Valid Actions** |
| haConfigRetrigger | CONFIGRETRIGGER | POST |
| haConfig | HACONFIG | NOTIFY |
| newUpdateAvailable | NEWUPDATEAVAIL | POST |
| packagePropertiesRequest | PACKAGEPROPERTIESREQUEST | NOTIFY |
| packageProperties | PACKAGEPROPERTIES | POST |
| state | FUSTATE | GET, NOTIFY |
| permissionRetrigger | PERMRETRIGGER | POST |
| abort | ABORT | POST |
| downloadProgress | DOWNLOADPROGRESS | NOTIFY |
| trustConfig | TRUSTCONFIG | NOTIFY |

## Structures in item <data>

### Content type HACONFIG

This content type provides information about all current electronics on the HA.

Valid for the following <action> items: NOTIFY

Note: the format for id and version is specified in detail in [4]

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:haconfig",

"properties" :

{

"deviceData":

{

"type" : "object",

"required" : true,

"description" : "The basic device data. Most of this data is also transmitted

with the /iz/info message.",

"properties":

{

"haID":

{

"type" : "string",

"minLength" : 18,

"maxLength" : 18,

"description" : "The home appliance device ID (HAID).",

"required" : true

},

"deviceType" :

{

"type" : "string",

"description" : "A device type according to the device description file

specification.",

"required" : true

},

"brand":

{

"type" : "string",

"maxLength" : 32,

"description" : "The brand of the home appliance according to the device

description file specification.",

"required" : true

},

"vib":

{

"type" : "string",

"maxLength" : 32,

"description" : "The vib of the home appliance.",

"required" : true

},

"customerIndex":

{

"type" : "string",

"minLength" : 2,

"maxLength" : 2,

"description" : "An alpha numeric indexer for the customer service.",

"required" : true

},

"mac":

{

"type" : "string",

"minLength" : 17,

"maxLength" : 17,

"description" : "Complex Type: EuiAddress. The MAC address of the

registered device. If the device posess multiple MAC

addresses, then the Wi-Fi MAC address will be

returned.",

"required" : true

},

"ddfMajorVersion":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "The major version part of the DDF (Device Description

File) version.",

"required" : true

},

"fdString":

{

"type" : "string",

"minLength" : 4,

"maxLength" : 4,

"description" : "FD information about the home appliance. This has the

format 'YYMM'. But is YY = (YYYY-20)%100.",

"required" : true

},

"manufacturingTS":

{

"type" : "string",

"minLength" : 19,

"maxLength" : 19,

"description" : "Complex Type: DateTime. The manufacturing timestamp of

the home appliance in ISO8601 format.",

"required" : true

},

"hub":

{

"type" : "string",

"maxLength" : 31,

"description" : "The short name of the HUB.",

"required" : true

}

}

},

"inventory":

{

"description" : "The inventory of the home appliance. The definition of this

object can be found in: https://scr.bsh-sdd.com/projects/HC/repos/architecture/browse/schemas/firmware-module/smm-inventory/field/schema.smm-field-inventory.0.1.0.json",

"$ref" : "bshg.com/smm-inventory.json",

"required" : true

}

}

}

### Content type NEWUPDATEAVAIL

This content type provides information about an available update. It specifies all packages contained in an update *Distribution Set*.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:newupdateavail",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

},

"packageIDs":

{

"type" : "array",

"minItems" : 2,

"items" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614

},

"description" : "Unique IDs that identify downloadable packages.",

"required" : true

},

"flags":

{

"type" : "array",

"minItems" : 0,

"items" :

{

"type" : "string",

"enum" : ["SKIP\_DOWNLOAD\_PERMISSION","SKIP\_INSTALL\_PERMISSION","REPAIR",

"FORCED"]

},

"description" : "Flags to controll handling during the update.",

"required" : true

},

"totalSize" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The total size in bytes of all packages for this update.

This parameter is not supported by COM GEN2 and shall only

be sent for System Master generation or newer.",

"required" : optional

},

"currentHAVersion" :

{

"type" : "string",

"minLength" : 5,

"maxLength" : 127,

"description" : "The current HA version in form of a string containing the

triplet - hardware version, RAC version and firmware

version. This parameter is not supported by COM GEN2 and

shall only be sent for System Master generation or newer.",

"required" : optional

},

"newHAVersion" :

{

"type" : "string",

"minLength" : 5,

"maxLength" : 127,

"description" : "The new HA version in form of a string containing the

triplet - hardware version, RAC version and firmware

version. This parameter is not supported by COM GEN2 and

shall only be sent for System Master generation or newer.",

"required" : optional

}

}

}

### Content type PACKAGEPROPERTIESREQUEST

This content type provides information about transaction ID and requested package ID during the download process.

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:packagepropertiesrequest",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

},

"packageID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies an update package.

The values 0x0 and 0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

}

}

}

### Content type PACKAGEPROPERTIES

This content type provides download and validation information for an update package.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:packageproperties",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : true

}, "dateTime" :

{

"type" : "string",

"minLength" : 19,

"maxLength" : 19,

"description" : "Complex Type: DateTime. The HCA’s current http DateTime.",

"required" : true

},

"link" :

{

"type" : "string",

"minLength" : 1,

"maxLength" : 127,

"description" : "Link for downloading the update package.",

"required" : true

},

"ocspURL" :

{

"type" : "string",

"minLength" : 1,

"maxLength" : 127,

"description" : "URL specifying the location of a valid OCSP server.",

"required" : true

},

"packageProperties" :

{

"type" : "string",

"maxLength" : 1600,

"description" : "Package Properties file, in version 6, containing validation

information for the update package.",

"required" : true

} }

}

### Content type FUSTATE

This content type provides information of the firmware update state on the HA.

Valid for the following <action> items: NOTIFY, RESPONSE

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:fustate",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required" : false

},

"state" :

{

"type" : "string",

"enum" : ["INIT","IDLE","WAIT\_FOR\_DOWNLOAD\_PERMISSION",

"DOWNLOADING",

"WAIT\_FOR\_INSTALL\_PERMISSION","INSTALLING",

"FINALIZING","FINISHED",

"ERROR","ERROR\_DO\_INVENTORY",

"ERROR\_DOWNLOAD","ERROR\_VALIDATION",

"ERROR\_VERIFICATION","ERROR\_INSTALL",

"ERROR\_FINALIZE"],

"description" : "The current state of the firmware update.",

"required" : true

},

"packageID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies an update package.

The values 0x0 and 0xFFFFFFFFFFFFFFFF are invalid.",

"required" : false

},

"errorID" :

{

"type" : "integer",

"minimum" : 0,

"maximum" : 65535,

"description" : "Generic error code.",

"required" : false

},

"haErrorID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 65535,

"description" : "Device specific error code.",

"required" : false

},

"reason" :

{

"type" : "string",

"maxLength" : 127,

"description" : "Optionally an error reason in case of error state.",

"required" : false

},

"errorLevel" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 2,

"description" : "Specifies the severity of an error.",

"required" : false

}

}

}

### Content type CONFIGRETRIGGER

This content type provides information about transaction ID.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:configretrigger",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": false

},

"enforceRefresh" :

{

"type" : "boolean",

"description" : "Indicator whether the HA configuration shall be gathered

internally in the HA anew or not. If false or not present,

the HA will respond with a previously saved configuration.",

"required" : false

}

}

}

### Content type PERMRETRIGGER

This content type provides information about transaction ID and permissions to request.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:permretrigger",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": true

},

"permission" :

{

"type" : "string",

"enum" : ["DOWNLOAD","INSTALL"],

"description" : "The current state of the firmware update.",

"required" : true

},

}

}

### Content type ABORT

This content type provides information about transaction ID.

Valid for the following <action> items: POST

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:abort",

"properties" :

{

"transactionID" :

{

"type" : "integer",

"minimum" : 1,

"maximum" : 18446744073709551614,

"description" : "The unique ID that identifies this concrete update

transaction on the appliance. The values 0x0 and

0xFFFFFFFFFFFFFFFF are invalid.",

"required": true

}

}

}

### Content type DOWNLOADPROGRESS

This content type provides information about all current electronics on the HA.

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id" : "urn:schemas-bshg-com:js:data:fu5:downloadprogress",

"properties" :

{

"progress":

{

"type" : "integer",

"minimum" : 0,

"maximum" : 100,

"description" : "The progress of the download phase in percent. All packages

of a firmware update are covered in the progress indication.

Resolution and time restrictions have to be applied on this

value to prevent a flodding with messages.",

"required" : true

}

### Content type TRUSTCONFIG

This content type provides information about the security related configuration on the HA

Valid for the following <action> items: NOTIFY

The following schema describes the specific object used within the <data> array:

{

"id": “urn:schemas-bshg-com:js:data:fu5:trustconfig”

"properties":

{

"hab":

{

"type": "object",

"properties":

{

"keySetId":

{

"type": "integer",

"minimum": 0,

"maximum": 255

"required": true

},

"keyNumber":

{

"type": "array",

"minItems": 1,

"items":

{

"type": "integer",

"minimum": 1,

"maximum": 4

"required": true

}

},

"required": true

}

},

"dm-verity":

{

"type": "object",

"properties":

{

"trustAnchorFingerprints":

{

"type": "array",

"minItems": 1,

"items":

{

"type": "string",

"pattern": "^([a-zA-Z0-9]{2}:){15}[a-zA-Z0-9]{2}$",

"minLength": 47,

"maxLength": 47

"required": true

}

}

}

"required": true

}

}

}

## Functionalities

### Trigger HA Configuration

The backend can request the current home appliance configuration from the HA.

The backend sends a POST to resource “haConfigRetrigger” using content type CONFIGRETRIGGER. If the message is sent during an active transaction, the transactionID has to be included.

The response will not contain any data.

*Note: As the process of doing the inventory of the whole HA configuration can take several seconds, the HA configuration is returned asynchronously with a separate NOTIFY message. The response is therefore only signaling that the trigger was accepted.*

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 5,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"enforceRefresh" : true

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 5,

"action" : "RESPONSE",

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82041120,

"resource" : "/fu/haConfigRetrigger",

"version" : 5,

"action" : "RESPONSE",

"code" : 400

}

### Notify HA Configuration

The COM sends a notification about its configuration once the connection HCA<->COM is established for the first time after restart or after it receives a haConfigRetrigger message.

The notification from the resource “haConfig” will contain a list of elements of the content type HACONFIG.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042940,

"resource" : "/fu/haConfig",

"version" : 5,

"action" : "NOTIFY",

"data" :

[

{

"deviceData" :

{

"haID" : "ABCD12345ABCD12345",

"deviceType" : "Dishwasher",

"brand" : "BOSCH",

"vib" : "SMI69T45EU",

"customerIndex" : "H9",

"mac" : "11-22-33-44-55-66",

"ddfMajorVersion" : 1,

"fdString" : "0005",

"manufacturingTS" : "2020-05-14T16:10:59",

"hub" : "EU-PRODUCTION"

},

"inventory" :

{

"ECUsSchemaVersion" : 2,

"ECUs" :

[

{

"tracingId" : "80011370220000440335000000123",

"name" : "SMM",

"hardwareId" : 216739047866238720,

"version" :

{

"major" : 1,

"minor" : 9,

"revision" : 0,

"build" : 289322

},

"firmware" :

[

{

"firmwareId": 283686952306183,

"version":

{

"major": 4660,

"minor": 39030,

"revision": 65244,

"build": 11259375

}

},

{

"firmwareId": 843793843548540,

"version":

{

"major": 3,

"minor": 3,

"revision": 0,

"build": 0

}

}

],

"content":

[

{

"contentId": 1234,

"version":

{

"major": 1,

"minor": 4,

"revision": 0,

"build": 0

}

}

]

}

]

}

}

]

}

### Post New Update Available

The backend sends a POST request to the resource “newUpdateAvailable” using the content type NEWUPDATEAVAIL. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 5,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"packageIDs" : [123456789012,210469270401,220604080568],

"flags" : ["SKIP\_DOWNLOAD\_PERMISSION"],

"totalSize" : 1572864,

"currentHAVersion" : "9.0.396",

"newHAVersion" : "9.0.448"

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 5,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/newUpdateAvailable",

"version" : 5,

"action" : "RESPONSE",

"code" : 400

}

### Notify Package Properties Request

With the notification *packagePropertiesRequest* the COM informs the HCA that it needs the Package Properties information for a download package with a specific package ID.

The notification from the resource “packagePropertiesRequest” contains the content type PACKAGEPROPERTIESREQUEST.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packagePropertiesRequest",

"version" : 5,

"action" : "NOTIFY",

"data" :

[

{

"transactionID": 105230,

"packageID" : 123456789012

}

]

}

### Post Package Properties

The backend sends a POST request to the resource “packageProperties” using the content type PACKAGEPROPERTIES. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

*Note: An error response may indicate an invalid Package Properties file.*

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 5,

"action" : "POST",

"data" :

[

{

"transactionID": 105230,

"dateTime" : "2016-02-29T22:15:01",

"link" : "https://fu-server-domain/path/specific-update-package",

"ocspURL" : "http://ocsp-server-domain:8080",

"packageProperties" : " base64url encoded binary data "

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 5,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/packageProperties",

"version" : 5,

"action" : "RESPONSE",

"code" : 400

}

### Get Firmware Update State of HA

The backend can request the current state of the firmware update on the HA.

The backend sends a GET request to “state”. The response will contain a single element of the content type FUSTATE.

**Example:**

# Request

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 5,

"action" : "GET"

}

# Response

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 5,

"action" : "RESPONSE",

"data" :

[

{

"transactionID" : 105230

"state" : "FINALIZING",

}

]

}

### Notify Firmware Update State of HA

The COM sends a notification about the firmware update state to the HCA.

The notification from the resource “state” will contain a list of elements of the content type FUSTATE.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/state",

"version" : 5,

"action" : "NOTIFY",

"data" :

[

{

"transactionID" : 105230

"state" : "ERROR\_VALIDATION\_FAILED",

"packageID" : 123456789012,

"errorID" : 61005,

"haErrorID" : 301,

"reason" : "Invalid signature in package.",

"errorLevel" : 1

}

]

}

### Post Permission Retrigger

The backend sends a POST request to the resource “permissionRetrigger” using the content type PERMRETRIGGER. The response MUST NOT contain a <data> item. The <code> item MAY display any errors occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 5,

"action" : "POST",

"data" :

[

{

"transactionID" : 105230,

"permission" : "INSTALL"

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 5,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82042942,

"resource" : "/fu/permissionRetrigger",

"version" : 5,

"action" : "RESPONSE",

"code" : 400

}

### Set Aborted

If an error occurs during the firmware update, the HCA can send an abort message specifying the current transaction ID.

The backend sends a POST request to the resource “abort” using content type ABORT. The response MUST NOT contain a <data> item. The <code> item MAY display any errors that occurred while processing the request.

**Example:**

# Post

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 5,

"action" : "POST"

"data" :

[

{

"transactionID" : 105230

}

]

}

# Response OK

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 5,

"action" : "RESPONSE"

}

# Response ERROR

{

"sID" : 1234,

"msgID" : 82040022,

"resource" : "/fu/abort",

"version" : 5,

"action" : "RESPONSE",

"code" : 400

}

### Notify Download Progress

The COM sends a notification about the download progress to the HCA. This notify is only sent during the downloading phase of the firmware update. The download progress is comprising the download of all packages to a certain firmware update. The progress has a range from 0 to 100 and is indicating the percentage of the progress.

To restrict the amount of messages for this notification, the HA will two criteria’s which must be met before a new notification is allowed to be sent:

* The progress must have a resolution of 5% (e.g. valid values are 0, 5, 10, 15, ...)
* The last notification of the progress is more than 1 second in the past

Example:

9:24:54 🡺 "progress" : 0

9:24:55 🡺 "progress" : 5

9:24:59 🡺 "progress" : 10

9:25:00 🡺 "progress" : 20

…

For implementation keep in mind, that a round down rule must be applied. E.g. 99MB/100MB must result in 95% instead of 100% progress. Furthermore make the resolution and time restriction easily changeable, e.g. via a constant.

The notification from the resource “downloadProgress” will contain a single element of the content type DOWNLOADPROGRESS.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042954,

"resource" : "/fu/downloadProgress",

"version" : 5,

"action" : "NOTIFY",

"data" :

[

{

"progress" : 50

}

]

}

### Notify Trust Config

The COM sends a notification about its security configuration every time it sends it HA configuration (HACONFIG).

The notification from the resource “trustConfig” will contain a list of elements of the content type TRUSTCONFIG.

**Example:**

# Notification

{

"sID" : 1234,

"msgID" : 82042940,

"resource" : "/fu/trustConfig",

"version" : 5,

"action" : "NOTIFY",

"data" :

[

{

"hab":

{

"keySetId": 1,

"keyNumber":

[

1,

2

]

},

"dm-verity":

{

"trustAnchorFingerprints":

[

"DO:xx:NO:Tx:xx:US:Ex:TH:IS:xx:ON:LY:xx:SA:MP:LE"

]

}

}

]

}

## Application Guidelines / Behavior

This service does not have any specific application guidelines / behavior beyond those already specified.

# Appendix

## Return and Error Codes

The same return and error codes as stated in the in-home Wi-Fi specification (see [CESP13]) also apply for this specification.

## Device Types

The same device types as stated in the in-home Wi-Fi specification (see [CESP13]) also apply for this specification.