Bambu Fleet Hub HTTP API v1.0.0

- Document Version: v1.0.0
- Last Updated: 2026-05-20

1. Document Overview

This document provides comprehensive technical specifications for integrating with the Fleet Hub HTTP API. It details available endpoints, request/response formats, authentication mechanisms, and error handling procedures.

1.1 Document Conventions

- Client: The software application consuming this API
- Hub: The physical Print Control Box hardware
- Printer: The 3D printer connected to the Hub

1.2 Hub API Common Format

- Base URL: `https://<hub ip Address>/`
- Authentication: Authorization: Bearer <token>
- Request Format: application/json
- Response Format: application/json
- Response Body
  - code: error code
  - message: error description

2. Security Architecture: Three-Layer Access Control

Fleet Hub implements defense-in-depth security. All three layers must be satisfied to access APIs.

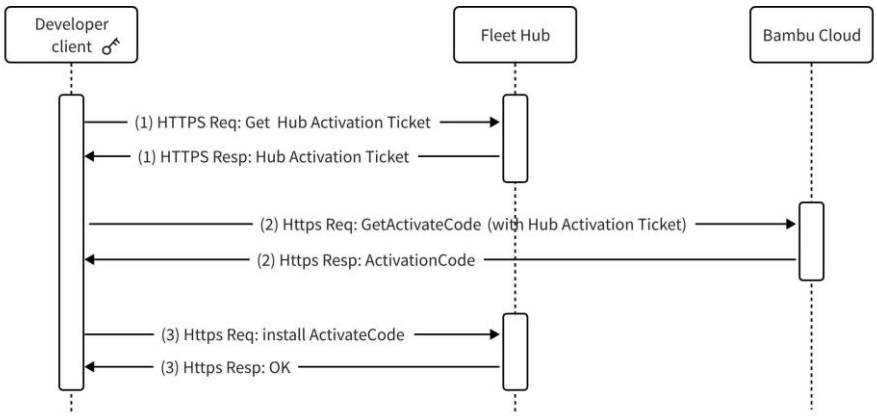
- Layer 1: Transport Security (mTLS)
- Layer 2: Device Activation (Binding)
- Layer 3: Session Access (Token-Ticket)

2.1 Layer 1: Transport Security (mTLS) - Mandatory

All API connections must present a valid client certificate. Connections without certificates are terminated immediately at the TLS handshake layer.

2.2 Layer 2: Device Activation (One-Time Binding)

Activation is required following initial setup or factory reset before subsequent API calls can be made. Once activated, the Hub enforces strict mTLS validation at the TLS layer, accepting only certificates signed under your Bambu Lab account—client certificates from any other party will be rejected.



Step 1: Developer Center Authorization Configuration

After reading and agreeing to the Developer Agreement in the Developer Center, complete the developer authorization. The developer may then proceed to configure the security credentials, including the Activation Key and Client Certificate.

Step 2: Fetch activate_ticket

POST `/v1/hub/activation/tickets`

Response:

```JSON
{
    "message":"",
    "code":0,
    "ticket":"DC2CA5...."
}
```

`ticket`: The activation ticket contains the Hub serial number (SN) and a unique authentication token, digitally signed to enable server-side verification of the Hub's authenticity.

Step 3: Obtain Activation Code

Send the activation ticket to the Bambu cloud to receive a signed activation code for Hub installation.

PUT `https://{bbl_domain}/v1/iot-service2/api/fleet_hub/activate`

HTTP Header:

```JSON
Authorization: Bearer {Activation Key}
```

Request:

```JSON
 {
     "activation_ticket": activation_ticket,
 }
```

Response:

```JSON
{
  "message": "",
  "code": 0,
  "error": null,
  "activation_code": "eyJjb2RlIjoie1wiZXX..."
}
```

- `200 OK`: Success.
- `401 Unauthorized`: Invalid Activation Key (re-fetch via Step 1).

Step 4: Submit activate_code to Hub

PUT `/v1/hub/activation/tickets/current`

Request:

```
JSON
{
     "activation_code": "activation_code",
     "user_name": "api account name",
     "password": "password"
}
```

Parameters: `user_name` and `password` are the local API account used for API access token generation.

- password: Minimum 8 characters, maximum 64 bytes. Must include uppercase letters, lowercase letters, numbers, and special characters.

Response:

- `200 OK`: Success.
- `400 Bad Request` with code 1043: Ticket expired (repeated Steps 1–4).

2.3 Layer 3: API Session (Token-Ticket Exchange)

Step 1: Hub Login Ticket

POST `/v1/hub/login/local/tickets`

Request:

```
JSON
{
    "user_name": "api account name"
}
```

Response:

```
JSON
{
  "message": "reason",
  "code": 0,
  "ticket_id": 123456,
  "ticket":"eyJjaGFsbGVuZ2UiOiJXYkpSKyN..."
}
```

- `ticket_id`: int64. A unique identifier for the ticket.
- `ticket`: A string in base64 format, which contains challenge and salt. The ticket come from the operation: `base64({"challenge": "_4O!1#^>", "salt": "dUf-fIr)*4_>UbnW"})`.

Step 2: Exchange ticket to Hub Login Token

The returned access token is used for all subsequent API authentication.

POST `/v1/hub/login`

Request:

```
JSON
{
    "user_name":"api account name",
    "password":"e3b804aacc4b3a3fb4b2272f1d9ba90893...",
    "ticket_id":2,
}
```

- `user_name`: The same account name used for step 1 operation.
- `ticket_id`: The ID returned in response to step 1 operation.
- `password`: this is password hash string. The challenge and salt come from the unbase64 operation for the "ticket" value and then calculated as `hex(sha256(hex(sha256(hex(sha256(password_plaintext))) + salt) + challenge))`.

Response:

```
JSON
{
  "token": "JWT Token"
}
```

3. Scan & bind a printer

3.1 Printer Discovery: Passive Listening

When the Hub and printers are deployed within the same subnet, the Hub can discover printers on the local network via passive SSDP listening. The Hub

continuously monitors SSDP multicast announcements (239.255.255.250:1900) broadcasted by printers, capturing device metadata including IP address, model, and device ID without actively scanning the network.

GET `/v1/hub/local_printers?only_can_bind={boolean}`

Parameter	Type	Default	Description
`only_can_bind`	boolean	`false`	Filter for bindable printers. When `true`, returns only printers that are eligible for binding to this Hub-specifically devices that are: (1) not bound to Bambu Handy mobile app (2) not operating in LAN Only mode (3) not currently bound to another Hub or Farm Manager software. When `false`, returns all discovered printers regardless of binding status.

3.2 Printer Discovery: Active Scanning

When the Hub and printers are not on the same subnet, or when passive discovery is insufficient, the Hub can actively scan specified IP ranges to discover printers. The Hub establishes TCP/TLS connections to port 3002 of target IP addresses and queries device information including model, firmware version, and binding state.

GET
`/v1/hub/scan_printers?prefix={string}&begin={integer}&end={integer}`

Parameter	Required	Description
`prefix`	Yes	IP address prefix. Supports private network segments only: `192.168.x`, `10.x.x`, or `172.x.x` (e.g., `192.168.1`, `10.0.0`).
`begin`	Yes	Starting value for the fourth octet (1-255). Defines the lower bound of the scan range.
`end`	Yes	Ending value for the fourth octet (1-255). Defines the upper bound of the scan range. The maximum range span is 254

		addresses per request.

Response:

```
JSON
{
  "printers": [
    {
      "location": "192.168.1.100",
      "usn": "XXXXA522400071",
      "model": "H2D",
      "name": "NAME",
      "connect": "cloud|lan|farm",
      "bind": "free|occupied",
      "version": "xx.xx.xx.xx",
      "farm": true,
      "server_id": "",
      "server_ip": ""
    }
  ]
}
```

`location`	Printer IP
`usn`	Serial number
`model`	Printer model
`name`	Printer name
`signal`	Wi-Fi strength
`connect`	Connection mode (`cloud`, `lan`, `farm`)
`bind`	Bind state(`free`,`occupied`)
`farm`	if firmware supports connecting to hub

Can Be Bound To A Hub:

connect	bind	farm	Yes/No

`cloud/ farm`	`free`	`true`	yes
`cloud/ farm`	`occupied`	N/A	No (already bound)
`lan`	`free`	N/A	No (LAN mode)
Any	Any	`false`	No (old firmware)

After adding a printer to the hub, the "connect" mode will change to "farm," and the "bind" mode will change to "occupied."

- `server_id`: The unique identifier of the server in the hub to which the printer has been connected. Note: Only valid when the printer has been added to a hub or bambu farm manager (IP-range discovery only).
- `server_ip`: The IP address of the hub or bambu farm manager to which the printer has been connected. This field is only valid when the printer has been added to a hub. (only valid when IP-range discovery)

3.3 Bind a printer

PUT `/v1/hub/bind`

Request:

```JSON
{
  "dev_ip": "",
  "dev_id": ""
}
```

3.4 Unbind a printer

DELETE `/v1/hub/bind`

Request:

```JSON
{
  "dev_ids": ["dev_sn", "dev_sn"]
}
```

Response:

```JSON
{
  "message": "",
  "code": 0,
  "unbind_results": [
   {
      "dev_id": "printer sn",
      "code": 0,
      "message": ""
   }]
}
```

- `unbind_results`: Per-printer result with failure message.

4. Printer Status & Control

4.1 Fetch Status for All Printers

GET `/v1/hub/devices`

Response:

```JSON
"devices": [
    {
      "dev_sn": "01P09A341000035",
      "dev_model": "P1S",
      "dev_ip": "10.20.215.146",
      "name": "3DP-01P-035-p1s--1",
      "report_status": {
        "nozzle_temper": 28.53125,
        "nozzle_target_temper": 0,
        "bed_temper": 24.6875,
        "bed_target_temper": 0,
        "chamber_temper": 5,
        "mc_print_stage": "1",
        "heatbreak_fan_speed": "0",
        "cooling_fan_speed": "0",
        "big_fan1_speed": "0",
        "big_fan2_speed": "0",
        "mc_percent": 100,
        "mc_remaining_time": 0,
```

```
        "ams_status": 0,
        "spd_mag": 100,
        "spd_lvl": 2,
        "print_error": 0,
        "wifi_signal": "-81dBm",
        "gcode_state": "IDLE",
        "gcode_file_prepare_percent": "100",
        "project_id": "0",
        "profile_id": "0",
        "task_id": "2",
        "subtask_id": "2",
        "subtask_name": "Stackable+Storage_w18l24h15",
        "gcode_file": "",
        "layer_num": 750,
        "total_layer_num": 750,
        "s_obj": [],
        "hms": [],
        "ams": {},
        "nozzle_diameter": "0.4",
        "nozzle_type": "stainless_steel",
        "stg": [],
        "stg_cur": 255,
        "home_flag": 6505752,
        "vt_tray": {},
        "vir_slot": [],
        "device": {},
        "upgrade_state": {}
      },
      "tags": [],
      "online": true,
      "mqtt_status": 2,
      "liveview_url": "",
      "liveview_time": "1970-01-01T00:00:00Z",
      "upgrade_info": {},
      "print_cmd_send_retry_cnt": 0,
      "latest_print_cmd_send_fail_reason": ""
    }
  ]
}
```


4.1.1 Printer Basic Information

Located within the `devices` field.

`dev_sn`	Serial number
`dev_model`	Printer model
`name`	Printer name
`report_status.nozzle_temper`	Current nozzle temps(℃)
`report_status.nozzle_target_temper`	Target nozzle temps(℃) - single-toolhead
`report_status.bed_temper`	Target bed temps(℃)
`report_status.mc_print_stage`	Print main control status • INITING • IDLE • RUNNING
`report_status.mc_percent`	Print progress(%)
`report_status.mc_remaining_time`	Remaining time (seconds)
`report_status.spd_lvl`	Print speed mode • 1: Silent Mode • 2: Normal Mode • 3: Sport Mode • 4: Ludicrous Mode
`report_status.ams_status`	AMS status. In the 16-bit representation, the high 8 bits indicate the AMS status. • 0: Idle; AMS consumables are editable. • else: Not idle; AMS consumables cannot be edited.
`report_status.print_error`	Hex error code (see Bambu Wiki). For example, 83935248 to 0500-C010.

`report_status.gcode_state`	Print job status • IDLE • PREPARE • RUNNING • PAUSE • FINISH • FAILED
`report_status.gcode_file`	Current print file
`report_status.gcode_file_prepare_percent`	Preparation progress (%)
`report_status.layer_num`	Current print layer
`report_status.total_layer_num`	Total print layer
`online`	Printer online status

4.1.2 Extruder and Nozzle Information

Located within the `devices.report_status` field.

```JSON
{
    "device": {
        "extruder": {
            "info": [
                {
                    "id": 0,
                    "temp": 34
                 },
                 {
                     "id": 1,
                     "temp": 35
                  }
            ]
        },
        "nozzle": {
            "info": [
```

```
                {
                    "id": 0,
                    "type": "HS01",
                    "diameter": 0.4,
                },
                {
                    "id": 1,
                    "type": "HS01",
                    "diameter": 0.4,
                }
            ]
        },
        "type": 1,
    },
}
```

`device.extruder.info.id`	Extruder ID
`device.extruder.info.temp`	Packed current/target extruder temps (bits 0–15: current; 16–31: target).
`device.nozzle.info.type`	Nozzle type, format `H[Byte1][Byte2-3]` • Byte1 Hotend type ◦ S: Standard flow ◦ H: High flow ◦ U: TPU high-flow • Bytes2-3 Hotend materials ◦ 00: Stainless Steel ◦ 01: Hardened Steel ◦ 05: Tungsten Carbide (e.g., "HS00": H=Hotend, S=Standard flow, 00=Stainless steel)
`device.nozzle.info.diameter`	Nozzle diameter(mm)
`device.type`	Current Operational mode, bitmask

	• 0x01: FDM • 0x02: Laser

4.1.3 HMS /Error Code Information

Located within the `devices.report_status` field.

```
JSON
"hms":[
     {
          "attr":50336256,
          "code":131073,
          "action":0,
          "timestamp":1732612487
     }
],
"err": "D0300801A",
"err2": {
    "err_code": "D0300801A",
    "img_id": "taskid_errcode_timestamp1",
}
```

- `hms`: Health code. convert `attr`/ `code` to hex: "attr":50336256 convert to hex 03001200. "code":131073 convert to hex 0002 0001. The whole HMS is 0300120000020001 (see Bambu Wiki HMS).
- `err`/`err2`: Error codes. When both the `err` and `err2` fields are present, the `err2` field takes precedence over parsing. For example, if the error code field is "D0300801A", the whole error code is 0300801A (see Bambu Wiki Error Codes).
- For a brief explanation of HMS and Error Code, please refer to the "4.10 **Query HMS/Error Code Descriptions**" documentation and use the Hub API to perform a lookup.

4.1.4 AMS and External Spool Information

Located within the `devices.report_status` field.

```
JSON
{
    "ams":{
          "ams_exist_bits":"1",
```

```
          "tray_exist_bits":"f",
          "ams":[
              {
                  "id": "0",
                  "humidity": "3",
                  "temp": "0.0",
                  "tray": [
                      {
                          "id": "0",
                          "tag_uid": "0000000000000000",
                          "tray_info_idx": "GFL00",
                          "tray_type": "PLA",
                          "tray_sub_brands": "",
                          "tray_color": "F98C36FF"
                      }]
               }]
    },
    "vt_tray":{
        "id":"255",
        "tray_info_idx":"GFA00",
        "tray_type":"PLA",
        "tray_sub_brands":"",
        "tray_color":"F98C36FF"
    },
    "vir_slot": [
        {
            "id": "255",
            "nozzle_temp_max": "290",
            "nozzle_temp_min": "260",
            "tray_color": "F98C36FF",
            "tray_id_name": "",
            "tray_info_idx": "GFA00",
            "tray_type": "PLA"
        }
    ]
}
```


`ams.ams_exist_bits`	Hex mask for AMS presence • bits 0-3: AMS/Pro • bits 4-11: AMS-HT

`ams.tray_exist_bits`	Bitmask for filament slots • bit0-bit15: A1–D4 • bit16-bit23: HT-A~ HT-H
`ams.ams[].tray.tag_uid`	Bambu RFID tag UID. Non-zero values indicate authenticated spools; color and type fields are read-only when present.
`ams.ams[].tray.color`	Spool color
`ams.ams[].tray.type`	Spool types: PLA, PETG, etc.
`vt_tray`/`vir_slot`	External spool info • vt_tray: P1/A1 series • vir_slot: X/H/P2 series

4.1.5 Version and Upgrade Information

Located within the `devices` field.

```JSON
{
    "upgrade_info":{
        "current_ver":"01.06.20.20",
        "upgrade_state":"IDLE",
    }
}
```

- `current_ver`: Printer firmware version.
- `upgrade_state`: OTA status
- *IDLE*
- *DOWNLOADING*
- *UPGRADE_SUCCESS*
- *UPGRADE_FAIL*

5) MQTT Channel Status

Located within the `devices` field.

```JSON
{
    "mqtt_status": 0,
}
```

- `mqtt_status`: Printer MQTT Channel Connection Status
  - *0: Disconnect*
  - *1: Connecting*
  - *2: Connected*

Note: When sending control commands to the printer, it is recommended to check the status of the MQTT connection to prevent message transmission failures caused by issues during link establishment or link instability. A `mqtt_status` value of 2 indicates that the connection is functioning normally.

4.2 Fetch Status for Single Printer

GET `/v1/hub/devices/{dev_sn}`

Response: Single printer status (see Section "Fetch Status for All Printers" for details).

4.3 Fetch Camera Snapshot

Printer uploads JPEGs to the server periodically. Check `liveview_time` for updates.

GET `/v1/hub/file/info?file_path=liveview/{dev_sn}/liveview.jpeg`

Response:

- `200 OK`
  - Headers:
    - `Content-Type`: application/octet-stream
    - `Content-Length`: File size in bytes (if available)
  - Body: Binary image data

4.4 Stop/Pause/Resume/Bed_clean

Send commands to the printer (validate gcode_state first via "Fetch Status").

POST`/v1/hub/devices/{dev_sn}/opt`

```
JSON
{
    "opt": "stop" | "pause" | "resume" | "bed_clean"
}
```

Request (one per request):

- `stop`: Stop printing, requires: `gcode_state`: `RUNNING`/ `PAUSED`.
- `pause`: Pause print, requires: `gcode_state`: `RUNNING`.
- `resume`: Resume print, requires `gcode_state`: `PAUSED`.
- `bed_clean`: Confirm print removal, requires `gcode_state`: `FINISHED`/ `FAILED`.

Response:

`200 OK`: Command acknowledged (state updates may delay ~10s).

4.5 Edit Printer Filament

4.5.1 External Spool Filament

Pre-checks:

- Verify printer/nozzle compatibility (see Filament Guide).

POST `/v1/hub/devices/{dev_sn}/opt`

Request:

```
JSON
{
    "opt": "external_filament_setting",
    "external_filament_setting": {
        "tray_type": "PLA",
        "tray_info_idx": "GFA00",
        "tray_color": "FFFFFFFF",
        "nozzle_temp_min": 190,
        "nozzle_temp_max": 240,
        "ams_id": 255,
        "slot_id": 0
    }
}
```

- "ams_id", "slot"_id", "tray_id": for external spool use the fixed values shown below:

Printer Model	ams_id	slot_id	tray_id
A1/A1mini/P1P/P1S	255	0	254
X1C/X1E	255	0	0
H2D	255(Right), 254(Left)	0	0

- `tray_type`: e.g., `PLA`, `PETG`.
- `tray_info_idx`: Bambu filament ID (e.g., `GFA00` for PLA).
- `tray_color`: ARGB hex value.

tray_type	tray_info_idx
PLA	GFA00
PETG	GFG00
PETG-CF	GFG50
TPU	GFU00
ABS	GFB00
ABS-GF	GFB50
ASA	GFB01
ASA-AERO	GFB02

4.5.2 AMS Slot Filament

Pre-checks:

- Verify printer/nozzle compatibility (see Filament Guide).
- Only spools without RFID tags (tag_uid: 0) are editable. Modifying RFID-tagged spool data is forbidden as it may cause communication errors between the printer and AMS.

POST `/v1/hub/devices/{dev_sn}/opt`

Request:

```JSON
{
    "opt": "ams_filament_setting",
    "ams_filament_setting": {
        "tray_type": "PLA",
        "tray_info_idx": "GFA00",
        "tray_color": "161616FF",
        "nozzle_temp_min": 190,
        "nozzle_temp_max": 240,
        "ams_id": 0,
        "slot_id": 0
    }
}
```

- `ams_id`: AMS unit ID (0–3 for AMS/Pro; 128–135 for AMS-HT).
- `slot_id`: Slot index (0–3 for AMS/Pro; 0 for AMS-HT).
- `tray_type`/ `tray_info_idx`: See Section "External Spool Filament".

4.6 Initiate Print Job

Hub provides 40 GB EMMC cache storage for model files. When initiating a print job, files are uploaded to the Hub prior to printing. The Hub calculates the MD5 hash of the gcode.3mf file for indexing and storage, then downloaded by the printer. Storage space is automatically reclaimed at regular intervals based on availability thresholds.

Hub supports initiating a print job based on a previously uploaded gcode.3mf file; simply specify the MD5 hash of the corresponding file in the print command. There is no need to re-upload the complete gcode.3mf file.

Available Space	Cleanup Frequency	Target Files
> 20 GB	No Cleanup	
≤ 20 GB	Every 10 minutes	Prioritize deleting files that have not been used for more than 30 days; then clean up files based on upload time.

PUT `/v1/hub/devices/print`

HTTP Request:

```
JSON
Authorization: Bearer {token}
Content-Type: multipart/form-data
```

4.6.1 Upload gcode.3mf Files and Start Printing

Request Body (Multipart) - Method for Uploading Complete Files:

Suitable for printing new model files.

Field	Type	Description
file	binary	3MF model file (application/octet-stream)
print_cmd	string	JSON-encoded print parameters
dev_sns	array	List of target device serial numbers

4.6.2 Initiating a Print Based on a Cached gcode.3mf File

Request Body (Multipart) - Method to Select File By Hash:

Suitable for repeatedly printing model files that have been cached in the Hub.

Field	Type	Description
file_hash	string	3MF Model File MD5 Hash
print_cmd	string	JSON-encoded print parameters
dev_sns	array	List of target device serial numbers

print_cmd:

Compatible with both of the above printing methods.

```
JSON
{
    "task_name": "test123",
    "plate": "Metadata/plate_1.gcode",
    "print_option": {
            "auto_bed_leveling": True,
            "flow_dynamic_calibration": True,
            "timelapse": False,
            "bed_leveling_mode":2,
            "flow_dynamic_cali_mode":2,
```

```
            "nozzle_offset_cali_mode":2
    },
    "filament_slot": [
         {"ams_id":255,"slot_id":0}, //filament id =1
         {"ams_id":255,"slot_id":255},
         {"ams_id":255,"slot_id":255},
         {"ams_id":254,"slot_id": 0} //filament id =4
    ]
}
```

- `task_name`: Job name (<128 bytes).
- `plate`: G-code path (e.g., `Metadata/plate_1.gcode`). Relative path to the G-code file within the uploaded archive. The printer extracts the file from this location before execution; the print job will fail if the file is not found.
- `auto_bed_leveling`, `flow_dynamic_calibration`: Booleans (only P1/A1/X1 series).
- `bed_leveling_mode`, `flow_dynamic_cali_mode`, `nozzle_offset_cali_mode`: 0=Disable, 1=Enable, 2=Auto (H/P2 series).
- `filament_slot`: Target filament slot for printing. See "Filament Mapping".

`task_name`	Job name (<128 bytes)
`plate`	G-code path (e.g., `Metadata/plate_1.gcode`). Relative path to the G-code file within the uploaded archive. The printer extracts the file from this location before execution **** The print job will fail if the file is not found***
`print_option.auto_bed_leveling`	Perform automatic bed leveling (only P1/A1/X1 series)
`print_option.flow_dynamic_calibration`	Perform dynamic flow calibration (only P1/A1/X1 series)
`print_option.bed_leveling_mode`	Bed leveling mode (only H/P2 series) • 0: Off • 1: Open • 2: Automatic

`print_option.flow_dynamic_cali_mode`	Dynamic flow calibration mode (only H/P2 series) • 0: Off • 1: Open • 2: Automatic
`print_option.nozzle_offset_cali_mode`	Nozzle offset calibration mode (only H/P2 series) • 0: Off • 1: Open • 2: Automatic
`filament_slot`	Target filament slot for printing. See "Filament Mapping" **** The print job will fail if the config is incorrect***

Response Codes:

```
JSON
{
    "code":0,
    "message":"error description"
}
```

Code 0 is Success, other is failure

- `1051`: Printer busy (printing/upgrading/offline/bed cleanup required).
- `1052`: P/A series missing SD card.
- `1053`: Invalid filament mapping (empty slot, invalid AMS ID, etc.).

4.6.3 Fetch List of Cached gcode.3mf Files

GET `/v1/hub/file/info?is_encrypted=False&num=100`

Response:

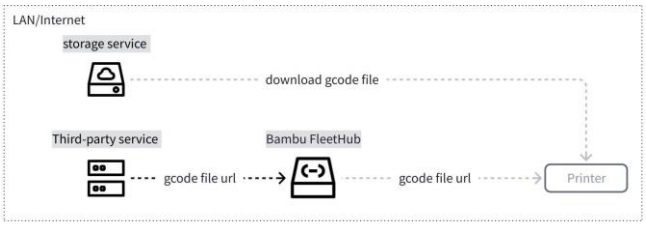
```
JSON
{
    "message": "",
```

```
    "code": 0,
    "file_infos": [
        {
            "file_hash": "d71f00e94e8f62d78273bfaa19828fbe",
            "last_use_time": "2026-05-09 18:56:07"
        }
    ]
}
```

- `file_infos[].file_hash`: Cached gcode.3mf file MD5 Hash

4.7 Print By Specified Model URL

In Hub scenarios, users can directly specify a model URL when initiating a print job to a printer, bypassing the Hub's cache and forwarding mechanism. Direct downloading requires the use of the HTTPS protocol; if the server utilizes a self-signed certificate, the corresponding CA certificate must be injected into each printer via the Fleet Hub API. Only the PEM certificate format is supported.



4.7.1 Injecting a Self-Signed CA Certificate

When the storage server utilizes a self-signed CA certificate, the certificate must be injected via the Hub. The Hub supports the injection of only one certificate; repeated calls to this interface will result in the storage of only the most recently provided certificate.

PUT `/v1/hub/external_ca`

Request:

```
JSON
{
  "ca_cert": "-----BEGIN CERTIFICATE-----\nxxxxxxxxx\n-----END
CERTIFICATE-----\n"
}
```

Note: When "ca_cert" contains empty data, it indicates that the injected certificate is being cleared.

4.7.2 Retrieving the Injected CA Certificate

GET `/v1/hub/external_ca`

Response:

```
JSON
{
    "message": "",
    "code": 0,
    "ca_cert": "-----BEGIN CERTIFICATE-----\nxxxxxxxxx\n-----END
CERTIFICATE-----\n"
}
```

- `200 OK`: Success.
- If the "ca_cert" data is empty, it indicates that no certificate has been injected.

4.7.3 Initiate Printing via Specified Model URL

PUT `/v1/hub/devices/external_url_print`

Request:

```
JSON
{
    "dev_sns": ["dev_sn"],
    "print_cmd": {
        "task_name": "X2D-dan-you_plate_2-5",
        "plate": "Metadata/plate_2.gcode",
        "print_option": {
                "auto_bed_leveling": False,
                "flow_dynamic_calibration": False,
                "timelapse": False,
                "bed_leveling_mode": 0,
                "flow_dynamic_cali_mode": 0,
                "nozzle_offset_cali_mode": 0
        },
        "filament_slot": [{"ams_id": 255, "slot_id": 0}]
    },
    "resolve_result": "10.20.0.01",
    "url": "url",
    "is_enc_print": False
```

```
}
```

- `url`: Specify the download link for the model file; the server can be either a domain name or an IP address.
- `resolve_result`: Specifies the default server resolution result; if the URL is inaccessible, access will be attempted via this IP address.
- `is_enc_print`: False
- `print_cmd`: refer to 4.6 **Initiate Print Job**

4.8 Setting the Printer Name

PUT `/v1/hub/devices/{dev_sn}/name`

Request:

```JSON
{
  "name": "new_name"
}
```

4.9 Update Printer Firmware

Before triggering the firmware update, uploading the firmware to the Hub is needed (see Section "Upload Printer Firmware").
This API commands the printer to download the firmware from the Hub and execute the upgrade.

PUT `/v1/hub/devices/firmware`

Request:

```JSON
{
  "dev_ids": ["printer1_sn", "printer2_sn"],
  "firmware_name": "offline-ota-p003_v01.09.01.00-xxx.zip"
}
```

Response:

```JSON
{
  "init_upgrade_results": [
```

```
    {
      "dev_id": "printer sn",
      "message": "",
      "code": 0
    }
  ]
}
```

Query upgrade status and results via the `upgrade_info`field in the `GET /v1/hub/devices` endpoint. See Fetch Status for All Printers section for details.

4.10 Query HMS / Error Code Descriptions

1) HMS

GET `/v1/hub/hms?dev_model={dev_model}&error_code={error_code}`

2) Error Code

GET
`/v1/hub/device_error?dev_model={dev_model}&error_code={error_code}`

dev_model: Printer model

error_code: The whole HMS/Error code, please refer to the HMS descriptions in the "4.1 Fetch Status for All Printers" section.

Response:

```
JSON
{
    {
      "message":"",
      "code":0,
      "description":"AMS D slot 2 the tube inside the AMS is
broken, or feed-out hall sensor is faulty and cannot detect the
filament."}
    }
}
```

5. Filament Mapping

5.1 3mf File Format

3MF is a ZIP archive containing:

- **G-code Files**: Print instructions (e.g., `Metadata/plate_1.gcode`).
- **Slice Information Configuration File**: Filament requirements (parsed for mapping).

```
Shell
代码块
E:\test.gcode\Metadata
drwxr-xr-x 1 user 1049089       0 Nov  7 15:50 _rels/
-rw-r--r-- 1 user 1049089     139 Nov 11 21:18 cut_information.xml
-rw-r--r-- 1 user 1049089     720 Nov 11 21:18
model_settings.config
-rw-r--r-- 1 user 1049089    1323 Nov 11 21:18 pick_1.png
-rw-r--r-- 1 user 1049089 3048153 Nov 11 21:18 plate_1.gcode
-rw-r--r-- 1 user 1049089      32 Nov 11 21:18 plate_1.gcode.md5
-rw-r--r-- 1 user 1049089     394 Nov 11 21:18 plate_1.json
-rw-r--r-- 1 user 1049089   19423 Nov 11 21:18 plate_1.png
-rw-r--r-- 1 user 1049089    3740 Nov 11 21:18 plate_1_small.png
-rw-r--r-- 1 user 1049089    2343 Nov 11 21:18
plate_no_light_1.png
-rw-r--r-- 1 user 1049089   53019 Nov 11 21:18
project_settings.config
-rw-r--r-- 1 user 1049089    1292 Nov 11 21:18 slice_info.config
-rw-r--r-- 1 user 1049089    4935 Nov 11 21:18 top_1.png
```

5.2 Filament Requirements for 3MF

The `slice_info.config` file:

```
XML
<?xml version="1.0" encoding="UTF-8"?>
<config>
  <header>
    <header_item key="X-BBL-Client-Type" value="slicer"/>
    <header_item key="X-BBL-Client-Version" value="02.05.03.61"/>
  </header>
  <plate>
    <metadata key="index" value="1"/>
    <metadata key="printer_model_id" value="O1C2"/>
    <metadata key="enable_filament_dynamic_map" value="false"/>
```

```
    <metadata key="filament_maps" value="1 2 2"/>
    <object identify_id="139" name="Bambu box" skipped="false" />
    <object identify_id="69" name="box" skipped="false" />
    <object identify_id="122" name="box" skipped="false" />
    <filament id="1" tray_info_idx="GFA18" type="PLA"
              color="#ECC3B2" used_m="5.00" used_g="15.89"
group_id="0"
              nozzle_diameter="0.40" volume_type="High Flow"
              used_for_object="true" used_for_support="true"/>
    <filament id="2" tray_info_idx="GFA01" type="PLA"
              color="#F7D959" used_m="8.23" used_g="26.14"
group_id="1"
              nozzle_diameter="0.40" volume_type="High Flow"
              used_for_object="true" used_for_support="true"/>
    <filament id="3" tray_info_idx="GFA00" type="PLA"
              color="#F4EE2A" used_m="6.03" used_g="18.28"
group_id="2"
              nozzle_diameter="0.40" volume_type="High Flow"
              used_for_object="true" used_for_support="true"/>
    <warning msg="bed_temperature_too_high_than_filament"
             level="3" error_code ="1000C001"  />
    <nozzle id="0" extruder_id="1" nozzle_diameter="0.4"
volume_type="High Flow"/>
    <nozzle id="1" extruder_id="2" nozzle_diameter="0.4"
volume_type="High Flow"/>
    <layer_filament_lists>
      <layer_filament_list filament_list="0 1 2" layer_ranges="0
124" />
      <layer_filament_list filament_list="1 2" layer_ranges="125
199" />
    </layer_filament_lists>
  </plate>
</config>
```

`filament.id`	The filament ID used for slicing, starting from 1; a value of 255 indicates that the filament was not used.
`filament.tray_info_idx`	Filament model
`filament.type`	Filament type, such as PLA, etc.

`filament.used_m`	Used filament length, in meters
`filament.used_g`	Used filament weight, in grams
`filament.group_id`	The nozzle number corresponding to the filament.`nozzle.id`
`nozzle.id`	Nozzle number, starting from 0 ** The field may be absent*
`nozzle.extruder_id`	The extruder corresponding to the nozzle • 1: Left extruder • 2: Right extruder ** The field may be absent*
`nozzle.nozzle_diameter`	The nozzle diameter ** The field may be absent*
`filament_maps`	List of extruders corresponding to each filament, index in field`value`with `filament.id` - 1 • 1: Left extruder • 2: Right extruder ** Used when the `nozzle` field is absent*

5.3 Filament Slot Configuration for Print Start

When initiating a print job, developer must refer to the consumable and nozzle information specified in this file and configure the commands accurately.

Map logical `filament id` s to physical slots (AMS trays/external spools):

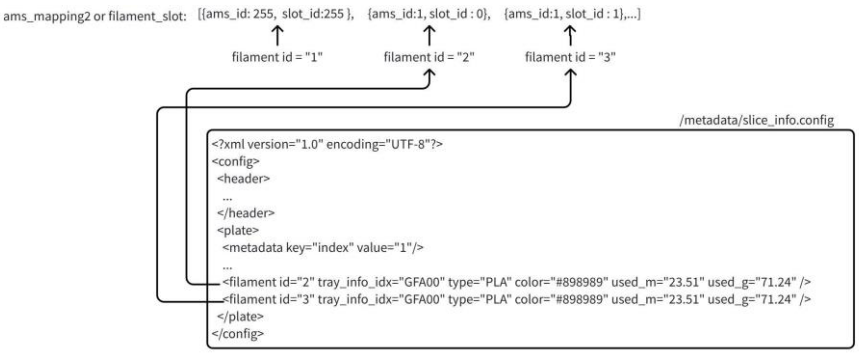
Corresponds to the `print_cmd.filament_slot` field:

```JSON
[
  { "ams_id": <value>, "slot_id": <value> }, //filament id = "1"
  { "ams_id": <value>, "slot_id": <value> }, //filament id = "2"
  ...
]
```

- Array Index: Zero-based position corresponding to <filament_id - 1>. Supports up to 16 filaments (indices 0-15)
- Select a physical slot by specifying the `ams_id` and `slot_id`

ams_id	slot_id	Spool Position	Name
255	255	Not used	N/A
255	0	External Spool	Ext for X1/A1/P1/P2S Ext Right for H2D
254	0	External Spool	Ext Left for H2D
128~135	0	AMS HT	HT-A (ams_id: 128) ~ HT-H (amd_id: 135)
0~3	0~3	AMS, AMS Lite or AMS Pro	A1–D4

- *Example*: XML mapping of `filament_id` to `ams_id`/ `slot_id`.



6. HUB Control API

6.1 Hub Management

6.1.1 Get Hub Information

GET `/v1/hub/captain`

Response:

```
JSON
{
```

```
    "name": "Hub",
    "hub_sn": "34K202601290000005",
    "server_id": "34K202601290000005",
    "activate_state": "Activate",
    "activate_uid": "bambu account uid",
    "max_bind_num": 50,
    "version": "01.00.02.06",
    "build_tag": "20260429144857_ee26bd",
    "internal": {
        "available": 40,
        "used": 0
    },
    "udisk": {
        "conn_state": 0,
        "available": 0,
        "used": 0
    },
    "sd": {
        "conn_state": 0,
        "available": 0,
        "used": 0
    }
}
```


- `name`: Display name of the Hub device.
- `activate_state`: Activation status of the Hub.
- `activate_uid`: Bambu Account UID that activated this Hub. Empty if not yet activated.
- `version`: Current firmware version running on the Hub.
- `internal`: Internal eMMC storage information.
    - available/used: Total available free/used space of internal storage, in GB.
- `udisk`: USB flash drive (U-disk) status and storage details.
    - conn_state: Connection status of the U-disk.
    - Available/used: Total available/used storage space on the U-disk, in GB.

`name`	Display name of the Hub device
`hub_sn`	Hub Serial Number
`activate_state`	Hub activation status

	• Activate • NoActivate
`activate_uid`	Bambu Account UID that activated this Hub. Empty if not yet activated
`version`	Current firmware version running on the Hub
`internal.available`	Total available space of internal eMMC storage, in GB
`internal.used`	Total used space of internal eMMC storage, in GB
`udisk.conn_state`	USB flash drive (U-disk) status, 0 means Not inserted
`udisk.available`	Total available space on the U-disk, in GB
`udisk.used`	Total used space on the U-disk, in GB
`sd`	Reserved field, currently unused

Note:
Inserting a U-disk enables IP address configuration file export, log collection, and factory reset operations. The Hub never exports model or G-code files to the U-disk.

The SD card slot does not currently support any functionality.

6.1.2 Restore Factory Settings

Performs a complete factory reset of the Hub. Warning: This action irreversibly deletes all printer bindings, cached model files, stored firmware images, configuration data, and operational logs. After reset, the Hub returns to its initial unactivated state and must be reactivated before further use.

POST `/factory/reset`

6.1.3 NTP and Date Setting

The Hub supports setting the device time independently, as well as enabling the NTP service to automatically synchronize the time. The NTP service is disabled by default.

PUT `/v1/hub/captain/ntp`

Request:

```
JSON
{
  "ntp_switch": "open",
  "time_zone": "UTC+08:00",
  "date": "2026-03-30 08:00:00",
  "ntp_url": ""
}
```

- ntp_switch: (Optional) "open" to enable NTP service; "close" to disable NTP service.
- time_zone: (Optional) Sets the time zone. Only supports the UTC time standard.
- date: (Optional) Sets the system time.
- ntp_url: (Optional) Currently unused.

`ntp_switch`	(Optional) "open" to enable NTP service; "close" to disable NTP service
`time_zone`	(Optional) Sets the time zone. Only supports the UTC time standard
`date`	(Optional) Sets the system time
`ntp_url`	(Optional) Reserved field, currently unused

The Hub supports retrieving NTP and time configuration information.

GET `/v1/hub/captain/ntp`

Response:

```
JSON
{
  "ntp_switch": "open",
  "time_zone": "UTC+08:00",
  "date": "2026-03-30 08:00:00",
  "ntp_url": ""
}
```

6.2 Local Account Management

The Hub provides a local web interface for printer adding/removing, printer firmware upgrading, hub log retrieval, and hub firmware updates. Access to this web interface is secured via a dedicated local web account, separate from API authentication credentials.

API user credentials are established during initial Hub activation. The web interface account can be enabled or disabled programmatically via API.

All account credentials and password hashes are stored exclusively within the Hub's hardware secure enclave and are never transmitted to or stored on Bambu Lab cloud servers.

6.2.1 Change the Password of the Local API Account

Minimum 8 characters, maximum 64 bytes. Must include uppercase letters, lowercase letters, numbers, and special characters.

PATCH `/v1/hub/users/current_user/passwd`

Request:

```JSON
{
  "old_password": "xxxx",
  "new_password": "xxxx"
}
```

6.2.2 Add a Web Account

Minimum 8 characters. Must include uppercase letters, lowercase letters, numbers, and special characters. Only one web account is permitted per Hub.

POST `/v1/hub/web/user`

Request:

```JSON
{
  "username": "xxx",
  "password": "xxx"
}
```

6.2.3 Delete a Web Account

DELETE `/v1/hub/web/user`

6.3 Printer frimware Management

Download offline firmware from the Bambu Lab official website, upload it to the Hub via API, and trigger the update when the printer is idle (not actively printing).

6.3.1 Upload Printer Firmware

POST `/v1/hub/file/firmwares`

Request:

```JSON
{
  "name": "offline-ota-p003_v01.09.01.00-xxx.zip",
  "md5": "hash md5 value"
}
```

- `name`: Use the original firmware filename as the parameter without modification. The Hub validates the filename to identify compatible printer models.

6.3.2 Get the List of Printer Firmwares

GET `/v1/hub/file/firmwares`

Response:

```JSON
{
  "firmwares": [
    {
      "name": "",
      "model": "",
      "version": "",
      "upload_time": ""
    }
  ]
}
```

6.3.3 Delete Firmware

DELETE `/v1/hub/file/firmwares`

Request:

```
JSON
{
  "firmwares": ["firmname1", "firmwarename2"]
}
```

---

7. API Error Code

Error Code	HTTP Status Code	Description
1	500	internal server error
2	500	internal db error
3	403	not activated
4	400	invalid request
5	400	bad request
6	401	invalid token
7	401	token expired
8	426	client upgrade required
9	426	server upgrade required
10	400	server and client mismatch
11	400	parameter out of range
12	400	3mf not exist in from folder
13	408	support log timeout
14	500	file system error
15	400	upgrade device not ready
16	400	upgrade device in sch
17	400	upgrade failed in downloading
18	408	upgrade downloading timeout
19	408	upgrade timeout
20	500	upgrade failed
21	400	upgrade unexpected state
22	400	security failed
23	400	parameter error
1000	400	folder delete not empty
1001	400	folder name exist
1002	400	folder not exist
1003	400	is not gcode 3mf
1004	400	is not one plate gcode 3mf
1005	408	mqttx request failed
1006	400	unbinded device
1007	400	no tag found
1008	400	no firmware

1009	400	task ongoing
1010	400	firmware not exist
1012	400	firmware invalid
1013	409	firmware exist
1014	400	bad 3mf
1015	400	unsupported option
1016	400	bulk actions ongoing
1017	404	bulk actions not found
1018	400	invalid license
1018	409	invalid activate uid
1019	409	opt outdated ongoing task
1020	409	launch task err finished or terminated
1021	409	launch task err no subtask
1022	409	launch task err device is suspended
1022	400	launch task err for task is launched
1023	400	wrong user name or password
1024	400	user name duplicated
1025	403	user not authorized
1026	429	login too frequent
1027	403	bambu login not supported
1028	400	bambu login failed
1029	400	printer bind failed
1030	400	update not ongoing task
1031	400	update task bad task cnt
1032	409	too much support logs
1033	404	support log not exist
1034	409	too much ongoing logs
1035	400	device offline
1036	400	bind device info invalid
1037	400	device restriction limit
1038	400	bind network error
1039	400	bind no device response
1040	400	statistics out of range
1041	400	sdk request with invalid parameter
1042	500	sdk request request failed
1043	400	sdk invalid login ticket
1043	429	sdk too many requests
1044	401	sdk not logged
1045	400	sdk activation failed
1046	400	sdk dev login failed
1047	404	notification config not found
1048	400	notification config exist
1049	400	operation not execute
1050	400	operation not support
1051	400	device busy
1052	400	device internal error
1053	400	invalid filament setting
1054	400	operation hardware not support
1055	416	file out of range

1056	400	username must be greater than 1 character and less than 32 characters
1057	400	password must contain: numbers, letters, special characters, at least 8 characters long
1058	400	date and timestamp are both empty
1059	400	date must format '2025-01-01 15:00:00'
1060	400	timestamp out of range, is seconds level
1061	400	time zone invalid, need like: 'UTC+8:00'
1062	400	user exist
1063	400	user not exist
1063	400	too many devices
1064	400	region need CN | COM
1065	400	Insufficient available space
10001	403	not allowed
10002	500	no response