人脸门禁面板机-Web-API V1.0

接口定义

目录

修改记录

Date	Version	Author	Amendment
2020/08/04	1.0.0	Keta	初始化文档
2020/08/18	1.0.1	Keta	1. 增加 2.10 Websocket 协议详细描述 2. 增加 3.12 实时数据推送 API 接口
2020/08/31	1.0.2	Keta	增加 3.6 设备管理类具体的 API 接口
2020/09/02	1.0.3	Keta	修改 3.7、3.8、3.9 人员、人员组和门禁计划等接口
2020/09/08	1.0.4	Keta	根据 meglink 协议修改如下章节： 1. 修改 3.6.8 网络设置的字段名称 2. 修改 3.6.9 ntp 设置的字段名称 3. 修改 3.6.10 识别设置的字段名称 4. 修改 3.6.11 门禁参数设置的字段名称 5. 修改 3.6.12 设备个性参数的字段名称 6. 修改 3.6.13 报警参数设置的字段名称 7. 修改 3.12.1 推送通行数据的字段名称
2020/09/12	1.0.5	Keta	1. 修改 3.7 人员组管理接口的字段名称 2. 增加 3.6.2 devices resource 资源接口 3. 增加 3.6.15 devices communications 通讯接口
2020/09/15	1.0.6	Keta	3.7 人员组管理、3.8 人员管理、3.9 门禁计划管理接口修正部分参数，增加 HATEOAS 链接

1. 简介

本文档主要介绍人脸门禁面板机 Web API 的接口功能与调用方法；设备提供设备管理、人员管理、照片管理以及通行记录管理等功能的接口，方便第三方平台或者设备进行二次开发对接。

2. 接口架构描述

本文档使用的协议基于 REST 架构，REST 即 REpresentational State Transfer (表现层状态转化) 的缩写，是 WEB 服务的一种架构风格，符合这种风格的架构就可以称为 RESTful 架构。

2.1. RESTFul 架构说明

1. API 与用户的通信协议，可以使用 HTTP 或 HTTPs 协议，不过由于设备端的限制，以 HTTP 协议为主；
2. 使用 HTTP 协议指定的动词 GET、POST、DELETE、PUT 等访问或修改设备资源；
3. 设备服务资源就是可以访问的 API 接口，例如：设备、人员、照片和通行记录等；
4. 设备服务资源使用 URL 描述，例如：GET /api/devices/status 表示获取设备的状态信息；
5. 设备服务资源返回数据格式为 JSON 格式；
6. 设备服务使用 HTTP 协议规定的状态码，如 200 OK、404 NOT FOUND、500 INTERNAL SERVER ERROR 等向用户返回 API 操作的状态；

2.2. 常用 HTTP 动词

1. GET（SELECT）：从服务器取出资源（一项或多项）。
2. POST（CREATE）：在服务器新建一个资源。
3. PUT（UPDATE）：在服务器更新资源（客户端提供改变后的完整资源）。
4. DELETE（DELETE）：从服务器删除资源。

2.3. HTTP 动词安全性和幂等性

1. 安全性：不会改变资源状态，可以理解为只读的；
2. 幂等性：执行 1 次和执行 N 次，对资源状态改变的效果是等价的。

	安全性	幂等性
GET	√	√
POST	×	×
PUT	×	√
DELETE	×	√

安全性和幂等性均不保证反复请求能拿到相同的 response。以 DELETE 为例，第一次 DELETE 返回 200 表示删除成功，第二次返回 404 提示资源不存在，这是允许的。

2.4. 常用 HTTP 状态码

1. 200 OK - [GET]：服务器成功返回用户请求的数据，该操作是幂等的（Idempotent）。
2. 201 CREATED - [POST/PUT/PATCH]：用户新建或修改数据成功。
3. 202 Accepted - [*]：表示一个请求已经进入后台排队（异步任务）
4. 204 NO CONTENT - [DELETE]：用户删除数据成功。
5. 400 INVALID REQUEST - [POST/PUT/PATCH]：用户发出的请求有错误，服务器没有进行新建或修改数据的操作，该操作是幂等的。
6. 401 Unauthorized - [*]：表示用户没有权限（令牌、用户名、密码错误）。
7. 403 Forbidden - [*] 表示用户得到授权（与 401 错误相对），但是访问是被禁止的。
8. 404 NOT FOUND - [*]：用户发出的请求针对的是不存在的记录，服务器没有进行操作，该操作是幂等的。
9. 406 Not Acceptable - [GET]：用户请求的格式不可得（比如用户请求 JSON 格式，但是只有 XML 格式）。
10. 500 INTERNAL SERVER ERROR - [*]：服务器发生错误，用户将无法判断发出的请求

是否成功。

2.5. 接口协议 URL 定义

URL 地址用来定位设备服务资源：

<protocol>://<host>[:port]/api/<services>/[cmd_path[?query]]

protocol：使用 http/https;

host：设备 ip 地址;

port：设备 http 服务端口;

services：服务名称;

cmd_path：请求具体的资源名称;

query：某些请求中，用来指定请求的具体参数;

在请求命令地址中，<>表示变量，[]表示可选项。

2.6. 复杂查询接口参数

部分资源有查询类的接口，如人员、通行记录等，使用时可以在 URL 上带入查询条件，常用的查询参数有：

	示例	备注
过滤条件	?type=1&age=16	允许一定的 uri 冗余，如/persons/1 与 /persons?id=1
排序	?sort=age,desc	
分页	?limit=10&offset=3	

2.7. HTTP 请求头部

HTTP 请求时，头部只支持以下常用 body format:

- Content-Type: application/json;
- Content-Type: application/x-www-form-urlencoded (浏览器 POST 表单用的格式)
- Content-Type: multipart/form-data; boundary=—-RANDOM_jDMUxq4Ot5 (表单有文件上传时的格式)

HTTP 请求的头部可以加入认证字段，认证字段中有获取到的认证信息 SessionID，SessionID 是设备的授权访问令牌，在每个 HTTP 请求的头部 Cookie 字段加入对应的 SessionID，如：Cookie: sessionID=5e0cb625a59ae5482f170ce3da89307e；

如果客户端 HTTP 请求没有 SessionID 字段，则将返回带有 WWW-Authenticate 标头字段的未授权 HTTP 响应（401）。

2.8. HTTP 响应

1. 响应的 body 直接就是数据，不要过度包装；
2. 各 HTTP 方法成功处理后的数据格式：

	响应格式
GET	单个对象、集合
POST	新增成功的对象
PUT	更新成功的对象
DELETE	空

3. json 格式的约定:

   时间格式采用 ISO 8601 标准，如"2020-01-01T18:25:43[.322]+08:00"，[.322] [表示可选项，为毫秒;

图片采用 urldata 格式，即 base64 编码后的数据加上“”的头部。

4. 复杂查询分页响应的 json 数据格式规定如下：

```
{
"paging":{"limit":10,"offset":0,"total":729},
"data":[ {},{},{}...]
}
```

2.9. 异步 API

基于 HTTP 的访问接口都是同步的，但是用户在调用某些资源时，有时候资源无法马上响应或者是在进行非常耗时的操作，比如升级、照片入库等，就需要进行异步的处理；

通过设计 HTTP 的状态和返回数据，可以达到异步请求的效果，步骤如下：

1. 用户请求异步的操作，如上传文件到升级接口；
2. 升级接口接收到升级文件数据，检查无误后，返回 **HTTP 202 Accepted 状态码**，并在 json 返回值中给出获取升级状态的接口：

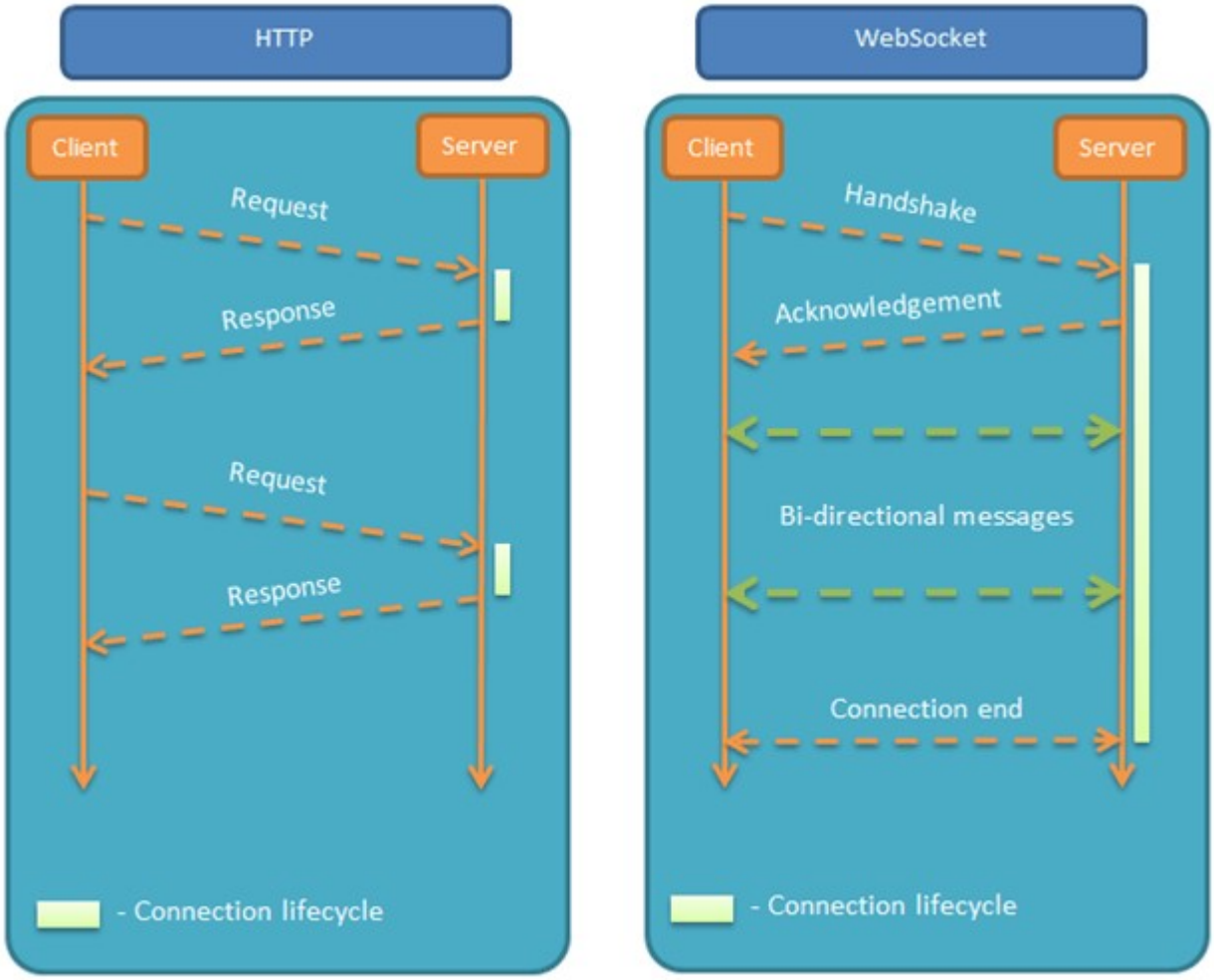
```
HTTP/1.1 202 Accepted
{
   "href": "/api/devices/firmware/status",
   "method": "get"
}
```

3. 用户收到 202 状态码后，解析 json 的 href 字段获取升级状态的 url，每隔一段时间，比如 1 秒去查询升级的状态；
4. 当获取升级状态的 json 返回值中返回 upgrading 状态为 finished 时，升级完成，否则会给出相应的升级错误码。

2.10. WebSocket 协议

HTTP 协议有一个缺陷：通信只能由用户发起，当需要设备主动向用户推送数据，比如实时推送人脸识别数据给用户端，就需要用到 WebSocket 协议。

WebSocket 协议的最大特点就是，服务器可以主动向客户端推送信息，客户端也可以主动向服务器发送信息，是真正的双向平等对话，属于服务器推送技术的一种：



其他特点包括：

- 建立在 TCP 协议之上，服务器端的实现比较容易。
- 与 HTTP 协议有着良好的兼容性。默认端口也是 80 和 443，并且握手阶段采用 HTTP 协议，因此握手时不容易屏蔽，能通过各种 HTTP 代理服务器。
- 数据格式比较轻量，性能开销小，通信高效。

- 可以发送文本，也可以发送二进制数据。
- 没有同源限制，客户端可以与任意服务器通信。
- 协议标识符是 ws（如果加密，则为 wss），服务器网址就是 URL，如：

  ws://example.com:80/some/path

3. Web API 接口说明

3.1. 安全机制

- WEB API 接口必须登录后才能使用，登录所用的资源服务接口为 auth；
- auth 接口使用两步登录认证，成功登陆后会返回一个 SessionID；
- 每个 API 请求，需要把 SessionID 加入到 HTTP 请求头部的 Cookie 字段；
- API 请求中如果涉及到密码、个人或者是其它敏感信息，必须使用 AES 加密算法加密原始信息，密钥为当前登录 Session 的 SessionID。

3.2. 错误处理

错误处理分两种情况：

1. 协议错误，是由于协议消息格式错误导致的。当 HTTP 头部错误，或者接收到非法数据，或者遇到套接字超时，接收方将收到协议错误。为了指示和解释协议错误，HTTP 协议定义了一组标准状态代码 [例如 1xx，2xx，3xx，4xx，5xx]，详情可以参考常用 HTTP 状态码；
2. 资源操作返回的错误，比如增删人脸照片、查询通行记录由于操作设备资源原因产生的错误，通常会返回大于 400 的 HTTP 状态码，并且 response body 返回一个以 json 格式描述的错误数据，通用格式如下：

```
{
    errors：[
    {
        "status": 503,
        "source": { "pointer": “/devices/status” },
        "title": "Service Unavailable",
        "detail": "worker is busy",
        "error_code": 0
    }
```

```
    ]
}
```

3.3. 使用限制

WEB API 接口只有在设备使用本地模式和第三方服务器模式时才能使用，云模式、考拉和盘古服务器模式下必须关闭接口的访问功能。

3.4. API 资源列表

资源	说明
auth	认证管理类资源
devices	设备管理类资源，比如说系统信息、网络、升级等
groups	人员组管理类资源
persons	人员管理类资源
schedules	门禁计划管理类资源
passes	通行记录类资源
log	系统日志类资源
subscribe	事件订阅类资源

3.5. 认证管理类 API 接口

序号	URL	说明
1	/api/auth/login/challenge	鉴权挑战
2	/api/auth/login	认证授权
3	/api/auth/logout	认证注销
4	/api/auth/activation	激活查询和启用

5	/api/auth/users	设备用户以及密码管理

3.5.1. /api/auth/login/challenge

URL:	/api/auth/login/challenge?username=用户名			
Method:	GET	POST	PUT	DELETE
	YES	NO	NO	NO
Describe:	返回的 sessioID 要保存作为后续请求的凭证，challenge 和 salt 字段的值用来进行下一步的登录认证，鉴权挑战的超时时间为 30 秒； Challenge 的值: 32 位随机数； Salt 的值: 32、64 或 128 位的 UUID 或者随机数。			
NOTE:				
Json Sample:	成功返回： { "session_id": "7145c267c3d15ea6f039ec6d3166cdd0", "challenge": "zh12auw8g49ny32lfjb2yep00n4bzlj", "salt": "646nsb45urp315itt2r2z77g93q5yos1454x7wjud0o41x5a1g6i207n51y11en" }			

3.5.2. /api/auth/login

URL:	/ISAPI/Security/login?type=web			
Method:	GET	POST	PUT	DELETE
	NO	YES	NO	NO
Describe:	请求的 type 表示平台类型，web 表示网页端，vms 表示平台端，other 表示其它客户端，不带 type 默认 web 登录； 请求 xml 中的 password 字段算法: hash(pwd, salt, challenge)，			

	hash 采用 SHA2-256 算法。 返回失败 xml 锁定字段 lockInfo 信息： locked – 是否处于锁定状态，true 锁定，false 未锁定； retryTime – 剩余重试次数； unlockTime – 锁定后剩余解锁时间，单位为秒。
NOTE:	登录重试 5 次失败将会锁定当前 IP 地址，时间 30 分钟； 登录成功后，如果 5 分钟内用户没有进行 api 的操作，则强制注销本次登录。
Json Sample:	请求 josn: { "session_id":"7145c267c3d15ea6f039ec6d3166cdd0", "username":"admin", "password":"8b0cfab3e1673a40775215d548a07dff9a9cf819c8a4344eca02ea3a98f069df" } 成功返回 json: { "status": 200, "session_id": "7145c267c3d15ea6f039ec6d3166cdd0", } 失败返回 json: { "status": 401, "session_id": "7145c267c3d15ea6f039ec6d3166cdd0", "lock_info": { "locked": false, "retry_time": 5 } }

3.5.3. /api/auth/logout

URL:	/api/auth/logout			
Method:	GET	POST	PUT	DELETE

	YES	NO	NO	NO
Describe:	调用认证注销接口后，将会直接清空 SessionID 的值			
NOTE:				
Json Sample:	无			

3.6. 设备管理类 API 接口

序号	URL	说明
1	/api/devices/status	设备状态信息，如设备、人脸库版本，设备型号以及序列号等
2	/api/devices/resource	设备规格参数，如存储容量、最大人员数等
3	/api/devices/firmware[/ota]	升级设备（ota 升级）
4	/api/devices/reboot	重启设备
5	/api/devices/restore	设备恢复出厂值
6	/api/devices/network	网络设置
7	/api/devices/time	时间设置
8	/api/devices/recognition	识别参数配置
9	/api/devices/door	门禁参数配置
10	/api/devices/profile	个性显示配置
11	/api/devices/alarm	报警接口配置

12	/api/devices/io	IO 触发接口，如远程开门等
13	/api/devices/operation	特殊的操作指令接口，如重置人脸库等

3.6.1. /api/devices/status

URL:	/api/devices/status			
Method:	GET	POST	PUT	DELETE
	YES	NO	NO	NO
Describe:	获取设备状态信息，如版本、型号、序列号等，详见 json 字串。			
NOTE:	model_spec 设备型号 serial_no 设备序列号 mac_str 设备硬件地址 firmware_version 固件版本 firmware_date 固件发布日期 web_version 设备内置 WEB 页面的版本 face_version 设备人脸库版本			
Json Sample:	{ "model_spec": "XXX", "serial_no": "M014200372006000266", "mac_str": "78:ca:83:40:2c:c8", "firmware_version": "V1.1.21", "firmware_date": "build20200831", "web_version": "V1.1.21 build20200831", "face_version": "2.7.3_20200706" }			

3.6.2. /api/devcies/resource

URL:	/api/devcies/resource

Method:	GET	POST	PUT	DELETE
	YES	NO	NO	NO
Describe:	获取设备资源信息，如最大人员数、最大通行数，当前人员、通行、卡号记录数，是否支持蓝牙、二维码、id card 通行等，详见 json 字串。			
NOTE:	max_person_number 设备支持的最大人员数； max_access_records 设备支持的最大通讯记录； saved_person_number 设备当前保存的人员信息总数； saved_access_records 设备当前通讯记录的总数； bluetooth_support 设备是否支持蓝牙； qrcode_support 设备是否支持二维码通行； idcard_support 设备是否支持刷卡通行；			
Json Sample:	{ "max_person_number": 50000, "max_access_records": 50000, "saved_person_number": 13, "saved_access_records": 203, "saved_card_number": 2, "bluetooth_support": false, "qrcode_support": false, "idcard_support": true }			

3.6.3. /api/devices/firmware

URL:	/api/devices/firmware			
Method:	GET	POST	PUT	DELETE

	NO	YES	NO	NO
Describe:	上传升级文件并开启设备升级流程，流程参见 2.9 异步 API。			
NOTE:	上传文件使用 HTTP 指定的 multipart/form-data 格式；			
Json Sample:	成功响应 json: { "href": "/api/devices/firmware/status", "method": "get" } 失败响应 json: { "errors": [{ "status": 503, "source": { "pointer": "/api/devices/firmware" }, "title": "Service Unavailable", "detail": "worker is busy/upload file is too large" }] }			

3.6.4. /api/devices/firmware/ota

URL:	/api/devices/firmware/ota			
Method:	GET	POST	PUT	DELETE
	NO	YES	NO	NO
Describe:	提供 ota 升级的参数，与非 ota 升级不同的地方需要设备主动通过接口提供的 uri 获取升级文件，并根据升级类型进行相应的升级操作；			
NOTE:	type 升级文件的类型，rom，apk md5 升级文件的 md5 校验码			

Json Sample:	请求 json: { "ota_uri": "http://xx.xx.xx/upgrade.bin", "type": "rom", "md5": "xxxxxxxx" }

3.6.5. /api/devices/firmware/status

URL:	/api/devices/firmware/status			
Method:	GET	POST	PUT	DELETE
	YES	NO	NO	NO
Describe:	获取设备升级时的状态，如升级的阶段、进度等。			
NOTE:	upgrading 枚举内容: ready – 准备升级 start – 开始升级 process – 升级中 finished – 升级完成 error – 升级错误 error_message 消息内容: tipsUpgradeCheckfile – 文件检查错误 tipsUpgradeCoverfile – 文件替换错误 tipsUpgradeParam – 参数错误 tipsUpgradeRwfile – 读写文件失败 tipsUpgradeOpenfail – 升级模块开启失败			

	tipsUpgradeNetupgrade – 网络升级失败
Json Sample:	{ "upgrading": "start", "percent": 0 } 升级错误 json 字串 { "upgrading": "error", "error_message": "tipsUpgradeCheckfile", "percent": 0 }

3.6.6. /api/devices/reboot

URL:	/api/devices/reboot			
Method:	GET	POST	PUT	DELETE
	YES	NO	NO	NO
Describe:	远程重启设备。			
NOTE:	delay 设备预计重新可用的时间，单位秒			
Json Sample:	{ "href": "/", "delay": "60" }			

3.6.7. /api/devices/restore

URL:	/api/devices/restore			
Method:	GET	POST	PUT	DELETE
	YES	NO	NO	NO
Describe:	远程恢复设备到出厂值。			

NOTE:	delay 设备预计重新可用的时间，单位秒
Json Sample:	{ "href": "/", "delay": "60" }

3.6.8. /api/devices/network

URL:	/api/devices/network			
Method:	GET	POST	PUT	DELETE
	YES	NO	YES	NO
Describe:	获取或设置设备的网络参数，如 IP、子网掩码、网关等； lan 表示有线网络配置的参数； wifi 表示无线网络配置参数； ports 表示当前设备开启的服务端口，如 http、rtsp 和 https 等			
NOTE:	如果设备不支持设置网络参数，PUT 时返回 HTTP 500 错误			
Json Sample:	{ "lan": { "dhcp": false, "ip": "192.168.30.94", "mask": "255.255.255.0", "gateway": "192.168.30.1", "dns1": "10.0.0.1", "dns2": "0.0.0.0" }, "wifi": { "mode": true, "ip": "192.168.30.177", "ssid": "Guest", "password": "xxxxxx" }, "ports": { "http_port": 80,			

	"rtsp_port": 554, "https_port": 443 } }

3.6.9. /api/devices/time

URL:	/api/devices/time			
Method:	GET	POST	PUT	DELETE
	YES	NO	YES	NO
Describe:	获取或设置设备的时间，包括本地时间、时区和 ntp 同步设置等；			
NOTE:	如果设备不支持修改时间，PUT 时返回 HTTP 500 错误； local_time 设备当前时间； ntp 设备 ntp 客户端设置参数： enable - 是否启用 ntp 客户端 time_zone – 时区； server – ntp 服务器的地址，可以是 IP 或域名地址； port – ntp 服务器的端口； sync_period – 自动同步的间隔时间，单位秒			
Json Sample:	{ "local_time": "2020-08-31T15:47:47+08:00", "ntp": { "enable": true, "time_zone": "GMT+08:00:00", "port": 123, "sync_period": 60, "server": "cn.pool.ntp.org" } }			

3.6.10. /api/devices/recognition

URL:	/api/devices/recognition			
Method:	GET	POST	PUT	DELETE
	YES	NO	YES	NO
Describe:	获取或设置设备人脸和识别参数的设置，如人脸模式、识别阈值、宽动态、活体检测参数等；			
NOTE:	recognition_mode 使用模式，single 单人模式，double 双人模式 recognition_threshold 识别阈值，范围 1~100 camera_wdr 宽动态等级，范围 1~3 max_recognition_distance 最大识别距离，单位米 second_recognition_distance 识别间隔，单位秒 face 人脸参数 face_ae FACE AE 功能开关，true 开启，false 关闭 yaw_threshold 水平角度，范围 0~90 pitch_threshold 垂直角度，范围 0~90 roll_threshold 旋转角度，范围 0~90 sharpness_threshold 清晰度，范围 0~100 liveness_mode – 活体检测开关，true 开启，false 关闭 rgb_liveness_threshold – 活体检测阈值，范围 0~100 ir_detect – IR 检测，true 开启，false 关闭 mask_mode – 口罩检测开关，true 开启，false 关闭			

	mask_threshold – 口罩检测阈值，范围 0~100
Json Sample:	{ "recognition_mode": "single", "recognition_threshold": 65, "camera_wdr": 1, "max_recognition_distance": 1.5, "second_recognition_distance": 2, "face": { "face_ae": false, "yaw_threshold": 35, "pitch_threshold": 35, "roll_threshold": 35, "sharpness_threshold": 80 }, "liveness_mode": true, "rgb_liveness_threshold": 84, "ir_detect": true, "mask_mode": true, "mask_threshold": 70 }

3.6.11. /api/devices/door

URL:	/api/devices/door			
Method:	GET	POST	PUT	DELETE
	YES	NO	YES	NO
Describe:	获取或设置设备门禁参数配置，如门磁类型、认证方式、开门延时时间和网络继电器等。			
NOTE:	sensor_type 门磁类型: open – 常开 close – 常闭 none – 不使用			

	verification_mode 认证方式: face – 仅人脸 face_or_card – 人脸或刷卡 face_and_card – 人脸和刷卡 face_and_pass – 人脸和密码 card_and_pass – 刷卡和密码 open_timeout 开门超时时长，单位秒 open_duration 门锁控制时间，单位毫秒 open_delay 延时门锁控制，单位秒 network_relay_enable 网络继电器，true 启用，false 关闭
Json Sample:	{ "sensor_type": "open", "verification_mode": "face", "open_timeout": 10, "open_duration": 1000, "open_delay": 0, "network_relay_enable": false }

3.6.12. /api/devices/profile

URL:	/api/devices/profile			
Method:	GET	POST	PUT	DELETE
	YES	NO	YES	NO
Describe:	获取或设置设备个性参数设置，如公司名称、设备地点、音量和屏幕亮度调节等。			
NOTE:	company_name 公司名称			

	location 设备地点 show_person_name 是否显示人员名称，true 显示，false 不显示 picture_type 显示照片的类型 snapshot – 显示抓拍照片 library – 显示底库照片 none – 不显示照片 voice_volume 音量调节，范围 0~100 touch_volume 触摸音量调节，范围 0~100 stanby_time 进入待机时间，范围 0~30，单位分钟 lcd_backlight 屏幕亮度，范围 0~100 force_pass_enable 胁迫报警开关，true 开启，false 关闭 force_password 胁迫报警密码，Base64 编码
Json Sample:	{ "company_name": "", "location": "", "show_person_name": true, "picture_type": "library", "voice_volume": 80, "touch_volume": 80, "stanby_time": 0, "lcd_backlight": 50, "force_pass_enable": false, "force_password": "NjY2Njg4ODg=" }

3.6.13. /api/devices/alarm

URL:	/api/devices/alarm			
Method:	GET	POST	PUT	DELETE

	YES	NO	YES	NO
Describe:	获取或设置设备报警参数，如报警输入 1、输入 2 和报警输出的用法等;			
NOTE:	input_1 报警输入 1 触发配置: alert – 报警信号 fire – 消防信号 none – 不启用 input_2 报警输入 2 触发配置，同报警输入 1 output 报警输出类型配置: input_2 – 报警输入 2 input_1 – 报警输出 1 doorcontact_timeout – 门磁超时 blacklist – 黑名单 bell – 门铃 none – 不启用			
Json Sample:	{ "input_1": "none", "input_2": "none", "output": "bell" }			

3.6.14. /api/devices/io

URL:	/api/devices/io			
Method:	GET	POST	PUT	DELETE

	NO	YES	NO	NO
Describe:	触发设备 IO 操作，如远程开门等；			
NOTE:				
Json Sample:				

3.6.15. /api/devices/communications

URL:	/api/devices/communications			
Method:	GET	POST	PUT	DELETE
	YES	NO	YES	NO
Describe:	获取或设置设备周边通讯接口，如韦根等；			
NOTE:	wiegand 韦根通讯接口参数: enabled – 是否启用 mode – 韦根使用模式，input 输入模式，output 输出模式 type – 韦根格式，支持 wiegand26 和 wiegand34 output_info – 韦根输出模式时发送的内容 output_person_no 发送人员编码 output_card_no 发送人员卡号			
Json Sample:	{ "wiegand": { "enabled": false, "mode": "input", "type": "wiegand26", "output_info": "output_person_no" }			

	}

3.6.16. /api/devices/operation

URL:	/api/devices/operation[?type=face_db_reset]			
Method:	GET	POST	PUT	DELETE
	NO	YES	NO	NO
Describe:	操作接口通过 type 类型传递操作参数，type 类型定义如下： face_db_reset – 把人员数据库等恢复默认值			
NOTE:				
Json Sample:				

3.7. 人员组管理 API 接口

序号	URL	说明
1	/api/groups/item{/id}	人员组新增、更新和删除管理
2	/api/groups/query	人员组信息查询

3.7.1. /api/groups/item

URL:	/api/groups/item{/id}			
Method:	GET	POST	PUT	DELETE
	YES	YES	YES	YES
Describe:	人员组获取、新增、删除和修改接口；删除人员组(DELETE)不需要 json 字串；			

	{/id}: 可选的资源访问索引
NOTE:	cmd 固定为 group_manage request_id 客户端请求的 id id 当前组 id，创建时不需要指定 group_name 人员组的名称 person_count 人员组拥有人员的总数 create_timestamp 创建人员组时的时间戳 schedule_id 人员组绑定的门禁计划 id links: 当前资源和相关联资源的 uri 访问地址
Json Sample:	{ "cmd": "group_manage", "id": "1", "type": "groups", "group_name": "default_1", "person_count": 14, "create_timestamp": "2034-01-01T00:00:00+08:00", "schedule_id": 1, "links": [{ "rel": "self", "href": "/api/groups/item/1" }, { "rel": "schedules", "href": "/api/schedules/item/1" }] }

3.7.2. /api/groups/query

URL:	/api/groups/query

Method:	GET	POST	PUT	DELETE
	NO	YES	NO	NO
Describe:	人员组查询，可以通过人员组名称模糊匹配；			
NOTE:	request_id 客户端请求的 uuid limit 本次查询的最大数量 offset 本次查询的偏移量 sort 排序方式 query_string 模糊查询人员组的字符串 links: 当前资源和相关联资源的 uri 访问地址			
Json Sample:	请求 json: { "cmd": "group_list_query", "request_id": "11111", "limit": 100, "offset": 0, "sort": "asc", "query_string": "group" } 响应 json: { "request_id": "xxxxx", "paging": { "limit": 1, "offset": 0, "total": 1 }, "data": [{ "id": "1", "type": "groups", "group_name": "default_1", "person_count": 14, "create_timestamp": "2034-01-01T00:00:00+08:00",			

	"schedule_id": 1, "links": [{ "rel": "self", "href": "/api/groups/item/1" }, { "rel": "schedules", "href": "/api/schedules/item/1" }] }] }

3.8. 人员管理 API 接口

序号	URL	说明
1	/api/persons/item{/id}	人员注册、更新和删除管理
2	/api/persons/query	人员信息查询
3	/api/persons/group	人员与人员组关系管理

3.8.1. /api/persons/item

URL:	/api/persons/item{/id}			
Method:	GET	POST	PUT	DELETE
	YES	YES	YES	YES
Describe:	人员获取、新增、删除和修改接口，当新增人员(POST)时，[id]可以为空，否则要指定对应组的 id；删除人员(DELETE)不需要 json 字串； {/id}: 可选的资源访问索引			
NOTE:	cmd 固定为 person_manage request_id 客户端请求 id			

	group_list 绑定人员组的列表 person_id 人员编号 person_name 人员名称 card_number 卡号 password 密码，AES 编码(openssl 格式) recognition_type 人员类型 staff – 普通人员 visitor – 访客 blacklist – 黑名单 is_admin 是否启用管理员权限，true 开启，false 关闭 enabled 人员是否启用，true 启用，false 不启用 schedule_list 人员门禁计划列表，与人员组的相同 face_list 人员照片，支持一个人员绑定多张照片 idx – 照片索引 data – Base64 编码的照片数据 links: 当前资源和相关联资源的 uri 访问地址
Json Sample:	{ "cmd": "person_manage", "request_id": "12352", "group_list": ["1", "2", "3"], "person_id": "1", "person_name": "person1", "card_number": "12345", "password": "12345",

```
    "type": "visitor",
    "enabled": true,
    "schedule_list": [
        {
            "idx": 0,
            "type": "workday",
            "rule": "9:00-18:00 * * 1-5 *"
        },
        {
            "idx": 1,
            "type": "holiday",
            "rule": "9:00-10:00 1-7 10 * 2020"
        }
    ],
    "face_list": [
        {
            "idx": "0",
            "data": "xxxx"
        },
        {
            "idx": "1",
            "data": "xxxx"
        }
    ],
    "links": [
        {
            "rel": "groups",
            "href": "/api/groups/item/1"
        },
        {
            "rel": "self",
            "href": "/api/persons/item/3"
        }
    ]
}
```


3.8.2. /api/persons/query

URL:	/api/persons/query			
Method:	GET	POST	PUT	DELETE

	NO	YES	NO	NO
Describe:	人员信息和照片查询接口，可以通过名字模糊查询。			
NOTE:	request_id 客户端请求的 uuid limit 本次查询的最大数量 offset 本次查询的偏移量 sort 排序方式 query_string 模糊查询人员名称或者编号的字符串 links: 当前资源和相关联资源的 uri 访问地址			
Json Sample:	请求 json: { "cmd": "person_list_query", "request_id": "11111", "limit": 10, "offset": 20, "sort": "asc", "query_string": "" } 响应 json: { "query_id": "xxxxx", "paging": { "limit": 5, "offset": 0, "total": 1 }, "data": [{ "recognition_type": "staff", "id": 1, "type": "persons", "is_admin": true, "enabled": true, "person_name": "gy", "person_number": "123",			

	"password": "", "card_number": "11133456", "group_list": [1], "links": [{ "rel": "groups", "href": "/api/groups/item/1" }] }] }

3.8.3. /api/persons/item/{id}/group

URL:	/api/persons/item/{id}/group			
Method:	GET	POST	PUT	DELETE
	YES	NO	YES	NO
Describe:	人员与人员组关系管理接口，比如修改某一人员到其它人员组等; {/id}: 可选的资源访问索引			
NOTE:	id 人员索引值 group_list 关联的人员组列表 links: 当前资源和相关联资源的 uri 访问地址			
Json Sample:	{ "cmd": "person_group_manage", "request_id": "12352", "id": "123", "group_list": [1], "links": [{ "rel": "groups",			

	"href": "/api/groups/item/1" }] }

3.9. 门禁计划管理 API 接口

序号	URL	说明
1	/api/schedules/item{/id}	门禁计划新增和删除管理
2	/api/schedules/query	门禁计划查询

3.9.1. /api/schedules/item

URL:	/api/schedules/item{/id}			
Method:	GET	POST	PUT	DELETE
	YES	YES	YES	YES
Describe:	门禁计划获取、新增、删除和修改接口；删除人员组(DELETE)不需要 json 字串； {/id}: 可选的资源访问索引			
NOTE:	cmd 固定为 schedule_manage request_id 客户端请求的 id id 当前组 id，创建时不需要指定 type 资源类型，固定为 schedules schedule_name 人员组的名称 schedule_list 门禁计划列表 idx – 索引值			

	type – 计划类型，workday 工作日，holiday 假日 rule – 计划规则，参考附录 A links: 当前资源和相关联资源的 uri 访问地址
Json Sample:	<pre>{ "cmd": "schedule_manage", "id": "1", "type": "schedules", "schedule_name": "rule1", "schedule_list": [{ "id": "0", "type": "workday", "rule": "9:00-18:00 * * 1-5 *" }, { "id": "1", "type": "holiday", "rule": "9:00-10:00 1-7 10 * 2020" }], "links": [{ "rel": "self", "href": "/api/schedules/item/1" }] }</pre>

3.9.2. /api/schedules/query

URL:	/api/schedules/query			
Method:	GET	POST	PUT	DELETE
	NO	YES	NO	NO
Describe:	门禁计划列表查询接口。			

NOTE:	request_id 客户端请求的 uuid limit 本次查询的最大数量 offset 本次查询的偏移量 sort 排序方式 links: 当前资源和相关联资源的 uri 访问地址
Json Sample:	请求 json: { "cmd": "person_list_query", "request_id": "11111", "limit": 10, "offset": 20, "sort": "asc", } 响应 json: { "request_id": "11111", "paging": { "limit": 1, "offset": 0, "total": 1 }, "data": [{ "schedule_id": "1", "schedule_name": "rule1", "schedule_list": [{ "id": "0", "type": "workday", "rule": "9:00-18:00 * * 1-5 *" }, { "id": "1", "type": "holiday", "rule": "9:00-10:00 1-7 10 * 2020" }] }

	] }

3.10. 通行记录管理 API 接口

序号	URL	说明
1	/api/passes/query	通行、识别记录查询
2	/api/passes/download	通行记录下载

3.10.1. /api/passes/query

URL:	/api/passes/query			
Method:	GET	POST	PUT	DELETE
	NO	YES	NO	NO
Describe:	通行记录查询接口，可以根据事件时间、人员索引、名字或编号进行匹配搜索。			
NOTE:				
Json Sample:	请求 json: { "request_id": "11111", "limit": 10, "offset": 20, "sort": "asc", "begin_time": "2020-09-01T00:00:00+08:00", "end_time": "2020-09-02T23:59:59+08:00", "query_string": "", "query_person_idx": 0 } 响应 json: { "request_id": "11111", "paging": { "limit": 1,			

```
            "offset": 20,
            "total": 27
        },
        "data": [
            {
                "face_idx": 1,
                "score": 73.500526,
                "living": false,
                "living_score": 0.000000,
                "face_number": "1",
                "face_name": "gy",
                "card_number": "",
                "timestamp": "2020-09-01T18:31:41+08:00"
            }
        ]
}
```


3.11. 运维日志 API 接口

序号	URL	说明
1	/api/log/query	运维日志查询
2	/api/log/download	运维日志下载

3.12. 实时数据推送 API 接口

推送接口采用 Websocket 协议，由设备主动向客户端发送实时数据；Websocket 推送最大连接数为 5 个连接。

序号	URL	说明
1	/api/subscribe	实时通行或事件数据推送订阅
2	/api/subscribe/push	第三方服务器设置接口，启用后可以给第三方服务器推送报警信息

3.12.1. /api/subscribe

URL:	/api/subscribe?session_id=xxxxxx			
Protocol	Websocket			
Method:	GET	POST	PUT	DELETE
	YES	NO	NO	NO
Describe:	xxxxxx 为 3.5.2 登录认证后返回的 sessioID；成功连接后，每隔 5 秒服务器会发 PING 心跳，客户端直接回复 PONG 即可。 本接口需要使用 Websocket 协议连接。			
NOTE:	普通报警（alarm_info）消息格式解析 type 报警类型： tamper – 防拆报警 doorTimeout – 开门超时 alarm1Fire – 报警输入 1 消防报警 alarm1Alert – 报警输入 1 普通报警 alarm2Fire – 报警输入 2 消防报警 alarm2Alert – 报警输入 2 普通报警 doorbell – 门铃 addFace – 新增人脸 authCrack – 认证超限? storageFull – 存储空间满 forcePass – 胁迫通行 forceOpen – 门磁强开报警			

	doorOpenSuccess – 开门成功 doorOpenFail – 开门失败 通行报警消息格式解析 recognition_type 消息类型: staff – 普通人员 visitor – 访问者 blacklist – 黑名单人员 stranger – 陌生人 none – 未定义 passed 是否成功通行，true 已通行，false 未通行 pass_mode 通行模式: face – 人脸 card – 卡号 face_and_card – 人脸和卡号 face_and_password – 人脸和密码 card_and_password – 卡号和密码 gate – 开门 qrcode – 二维码 force – 胁迫通行 none – 未定义 verification_mode 认证模式: face – 人脸

	face_or_card – 人脸或卡号 face_and_card – 人脸和卡号 face_and_pass – 人脸和密码 card_and_pass – 卡号和密码 none – 未定义 recognition_score 比对分数 liveness 是否为活体 liveness_score 活体分数 mask 是否戴口罩 therm 测温结果是否通过 temperature 人员体温 person_id 人员 id person_name 人员姓名 card_number 卡号 timestamp – 时间戳
Json Sample	普通报警 (pass_info) 消息: { "alarm_info": { "channel": 1, "type": "doorOpenSuccess", "timestamp": "2020-08-18T15:57:24+08:00" } } 通行报警消息: {

"recognition_type": "staff",
"passed": true,
"pass_mode": "face",
"verification_mode": "face",
"recognition_score": 82.85865,
"liveness": true,
"liveness_score": 99.995979,
"mask": false,
"therm": false,
"temperature": 0,
"person_id": "123",
"person_name": "gy",
"card_number": "11133456",
"timestamp": "2020-09-08T18:44:31+08:00",
"picture_data":"
AB4AAD/2wBDAAMCAgMCAgMDAwMEAwMEBQgFBQQEBQoHBwYID
AoMDAsKCwsNDhIQDQ4RDgsLEBYQERMUFRUVDA8XGBYUGBIUFRT/2
wBDAQMEBAUEBQkFBQkUDQsNFBQUFBQUFBQUFBQUFBQUFBQUFB
QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBT/wAARCAAs
ADQDASIAAhEBAxEB/8QAHwAAAQUBAQEBAQEAAAAAAAAAAAECAw
QFBgcICQoL/8QAtRAAAgEDAwIEAwUFBAQAAAF9AQIDAAQRBRIhMUE
GE1FhByJxFDKBkaEII0KxwRVS0fAkM2JyggkKFhcYGRolJicoKSo0NTY3O
Dk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqDhIWGh
4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NX
W19jZ2uHi4+Tl5ufo6erx8vP09fb3+Pn6/8QAHwEAAwEBAQEBAQEBAQ
AAAAAAAAECAwQFBgcICQoL/8QAtREAAgECBAQDBAcFBAQAAQJ3AA
ECAxEEBSExBhJBUQdhcRMiMoEIFEKRobHBCSMzUvAVYnLRChYkNOEl8
RcYGRomJygpKjU2Nzg5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN
0dXZ3eHl6goOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uL
m6wsPExcbHyMnK0tPU1dbX2Nna4uPk5ebn6Onq8vP09fb3+Pn6/9oA
DAMBAAIRAxEAPwDD8f25bwnqI5HyDt/tCvn/AFTwjqF3ai/gglNqFbMg
hLLx6nHuOvrXs2sfEiy8TaLPZJZyQXNxCzLllYAAbskj6H8q9h+Bl9p3hP8A
Z98VeItRtLW9tNOiuJJLacnMxGwovUdxj8a8p81N6rUu6ktD4JvkGnxyme
eIfKD8yY6kccCqEMyXp/0ZUk24+Q5HHr1q94tttV8Z3OqeI/siQx3d1JPKt
smETJztA5+UZwPYCua0tzZzZY7efmU9/cV304KUdGNqUd0b8lrKqkMib
gcEq44x2zmo7OLF9bADC5JznPcj+lXrEHVIWRcDbufcSc4xkj9Kn0+wea7
tH2cInLfVzWOsZWZme3/D2Dd4bjIz/rG/nEB4Vts8Es5/8AHjRW
dzU99+Lmt/C7Xvhq1l4JFWOKX7DHCVXow3KTgEV8/an8RYNI

+DXiT4fTaaw1G8Xz2uoZORHwwTGMYLIB1zXf3UYXOQBXhnxk8Y/8ACP
3kFu1lJLFlXlYNhWjIIK57HjNcKqTm77lw5VJKWx0XhWG08PaRaafJYi6jkV
fk8qVsgjOSURgvOetZ/wARvgP4cmj029tJrjR5NSOV+R5kBJxwiIzHn0xX
ofgDxrps/ge3vVtg5kixEZcfLzjBPY5BH4V6ZpvjfSZPC+lTNZxTSWkoEizM
mSGY/d5znNYwqSjLmWh9X9XhUpJP3l+R8peAfA9l8LPiBq1h43tBqOm
XGkSyWF5ErLE7NgK+OCMZIIPSsB9PtY9ST7LdQzR4K7I5Mn7xI4+hr2z9
rD4hQa7fJ4a0mzUXbLEby4lcKkcf3/Lz6ngn2wB1r5u8M6O1n4ksyzJI+9u
I/uqNpH416cZyqe9M+WxEYU58kWfVPw7sV/4RCwO3qpP6miui8B2Gz
wjpgJwfKBopGN0Zs10Q20HODj6VwnxU8FL4j+HPiPUVtofPsmicXDMf
M7jaB0Od1dpJ95vqB+te2fCP4b6HrmlLNqNsb5LmfEkE+GjwpGPlx7d6y
ow5mkiW0uh8mfC/wfrPgvw1Z/20JbePVo/t1rZuvKwk4Dn/AHiM4x0APc
1714FvdOk0mZXiYhPmYyZ2jHfJ6dK9D/aq8N2NrB4I1aKLy7vdPaHbgKY
8K2MY7Efqa4jwRpdvrGv6HpNwv+hXd7HFNGnG9SRkZ96yrRca/KurPq
cJUbwt3sjm/wBon9mm4s/CunePLKCV4tWtEuLm3QEtFKVyOOSAVwcdj
u6DAr5Z0HR2g1mzmI+8H/DG4V+5Gt2kNvpaxpEojjwFXHAAB4xXgfxe
/Zh+H3xM02e41HRzYXy/Ot7pMn2WXPU5K8HqeoPWvep0FJcx8lOpzS
ufNvhVBH4b01NnIgToPaivO9chu/hn4k1nw1p2q317YafdGOCTUJRLKF2
qdpYAZAye1Fc8o8rsUf/Z"
    }
}

3.12.2. /api/subscribe/push

URL:	/api/subscribe/push			
Protocol	HTTP			
Method:	GET	POST	PUT	DELETE
	YES	NO	YES	NO
Describe:	第三方服务器设置接口。			
NOTE:				
Json	{ "server_uri": "http://xx.xx.xx.xx/push",			

Sample:	"enabled": false, }

4. 附录 A

门禁通行规则语法说明，时间参考 crontab 格式，但是时分不做循环安排（否则像 9:20~11:20，这么简单的时间区间要设置 3 条规则），仅对日、月、周、年支持循环.

一条规则组成:

time	分隔符	date
见以下说明	空格	见以下说明

time 格式说明:

采用 24 进制表达。

符号	符号说明	含义
:	冒号	分割，前为小时，后为分，没有秒
-	减号	表示区间
,	逗号	表示并列关系

示例如下:

表达式	含义
9:00-10:00	9 点到 10 点
9:20-23:00	9 点 20 到 23 点
0:00-2:00,9:30-18:00,23:00-23:59	特殊地，如果时间跨越凌晨，应该分为多个时间段， 早班 9 点半到 18 点，晚班 23 点到凌晨 2 点

date 格式说明（参考 crontab）:

总共四列，每列按空格分割，分别表示日（按月）、月、日（按周）、年

符号	符号说明	含义
*	星号	取值范围内的所有数字
/	斜杠	每过多少数字
-	减号	区间
,	逗号	并列关系

示例如下:

表达式	含义
* * 1-5 *	每个工作日(Mon - Fri)
1 * * *	每月 1 日
*/2 * * *	隔天
1,15 9 * *	9 月 1 号以及 9 月 15 号
15 10 * 2018	2018 年 10 月 15 日
2-20 9 * 2018	2018 年 9 月 2 日至 2018 年 9 月 20 日
*/14 * * *	隔周（大小周）