

今日水印相机-验真api技术文档

开发者下载版 · 内容与开放平台在线文档保持一致

1 简介

1.1 什么是今日水印相机-验真api

开放平台支持今日水印相机用户自有系统接入，进行验真、查询验真结果等，获得对数据的二次开发的能力，从而满足各类定制化需求。

1.2 今日水印相机-验真api能做什么

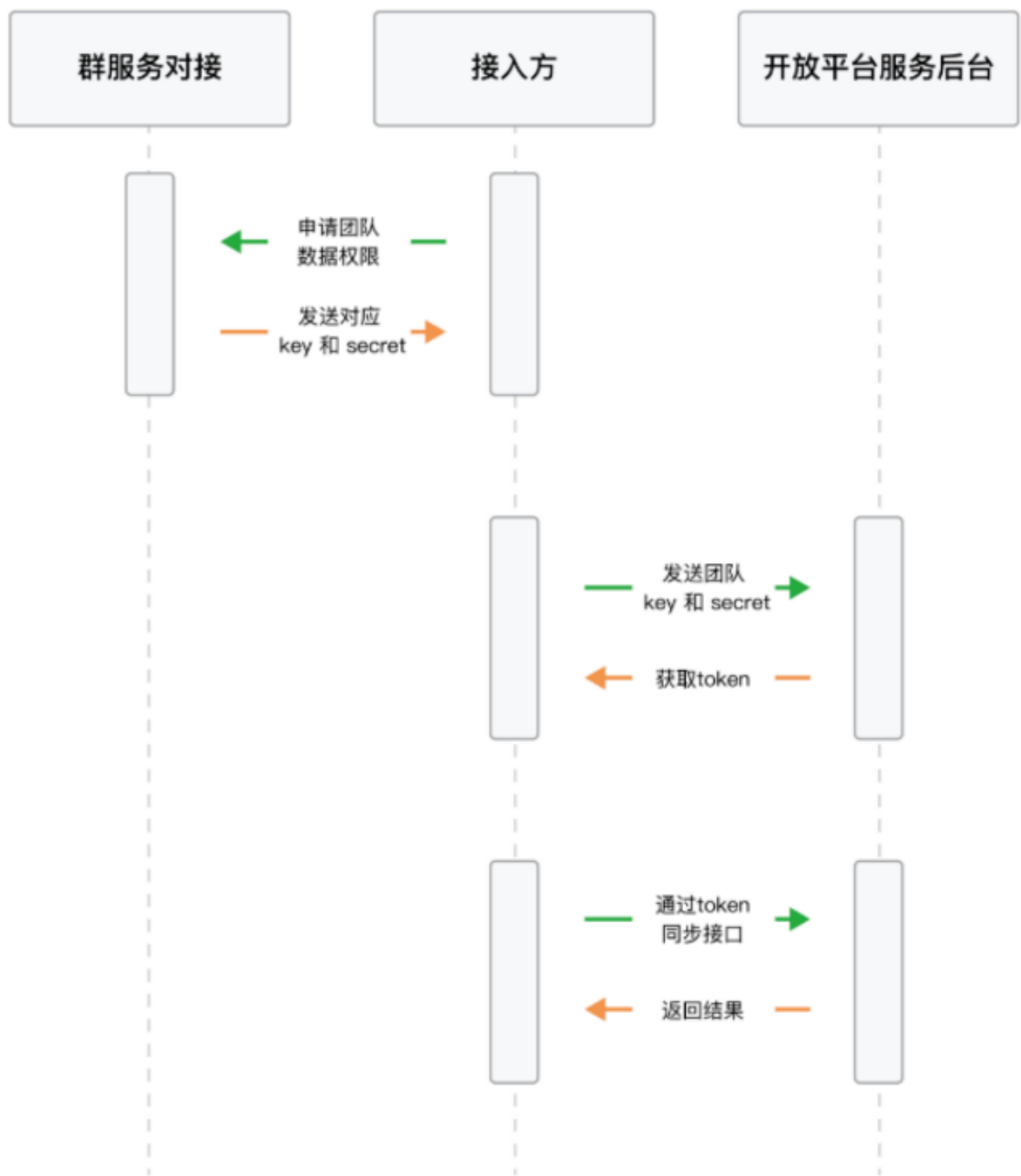
初步开放以下功能，更多功能正在逐步开发中。 - 异步获取结果

1.3 今日水印相机api的安全保障说明

开放平台 API 数据传输全程加密（密钥按月更新，可以支持更高安全标准的密钥动态下发），用户通过开放平台 API 传输的数据全程不存储、不落库，保证用户数据隐私和安全，系统全流程符合国家三级等保标准，系统无被入侵风险，数据无泄漏风险。

2. 接入流程图

2.1 主动拉取数据



3. 开放平台加签流程

对接沟通后，我们会给提供跟团队绑定的唯一key和secret。

3.1 加签流程

以v2/departement/sublist接口为例:

1. 接口请求参数为：

JAVASCRIPT

```
{
  "departmentID": 1
}
```

2. 将该请求参数序列化并进行一次签名，目前仅支持HmacSha256算法，签名函数如下：

```
GOLANG
// data是加密的内容
func sign(data string, groupSecret string) (ret string) {
    hmacObj := hmac.New(sha256.New, []byte(groupSecret))
    hmacObj.Write([]byte(data))
    ret = base64.StdEncoding.EncodeToString(hmacObj.Sum(nil))
    return ret
}
```

3. 把groupKey、参数签名（第二步的结果）、timestamp拼接为一个字符串

```
GOLANG
timestamp := time.Now().Unix()
groupKey := "groupKey"
str := fmt.Sprintf("groupKey=%s&sign=%s&timestamp=%s",
    groupKey,
    sign,
    strconv.FormatInt(timestamp, 10),
)
```

4. 把第三步拼接后的字符串再进行一次签名，并得到最终签名。

```
GOLANG
signature := sign(str, "groupSecret")
```

5. 把对应的信息设置至header头中。

```
GOLANG
req, _ := http.NewRequest(http.MethodGet, "url", nil)
req.Header.Set("GroupKey", groupKey)
req.Header.Set("Timestamp", strconv.FormatInt(timestamp, 10))
req.Header.Set("Signature", signature)
```

完整Http客户端代码如下：

```
GOLANG
func main() {
    data := map[string]interface{}{
        "departmentID": 1,
    }
    buf, _ := json.Marshal(data)

    groupKey := "groupKey"    // 需申请
    groupSecret := "groupSecret" // 需申请
    dataSign := sign(string(buf), groupSecret)
    timestamp := strconv.FormatInt(time.Now().Unix(), 10)
    str := fmt.Sprintf("groupKey=%s&sign=%s&timestamp=%s",
        groupKey,
        dataSign,
        timestamp,
    )
    // sign为加签流程中定义的函数
    signature := sign(str, groupSecret)

    url := "https://openapi.xhey.top/v2/department/sublist"

    client := &http.Client{}
    req, _ := http.NewRequest("POST", url, bytes.NewBuffer(buf))
    req.Header.Set("GroupKey", groupKey)
    req.Header.Set("Timestamp", timestamp)
    req.Header.Set("Signature", signature)

    _, _ = client.Do(req)
}

func sign(source string, groupSecret string) string {
    hmacObj := hmac.New(sha256.New, []byte(groupSecret))
    hmacObj.Write([]byte(source))
    return base64.StdEncoding.EncodeToString(hmacObj.Sum(nil))
}
```

Python版本

PYTHON

```

# -*- coding: utf-8 -*-
import json
import requests
import hashlib
import time
import base64
import hmac

# 计算签名
def generate_signature(data, secret_key):
    hmac_obj = hmac.new(secret_key.encode('utf-8'), data.encode('utf-8'), hashlib.sha256)
    return base64.b64encode(hmac_obj.digest()).decode('utf-8')

# 请求数据
data = {
    "departmentID": 1,
}

json_data = json.dumps(data)

group_secret = "groupSecret"
group_key, timestamp = "groupKey", str(int(time.time()))
data_sign = generate_signature(json_data, group_secret)

str_to_sign = "groupKey={0}&sign={1}&timestamp={2}".format(group_key, data_sign, timestamp)

signature = generate_signature(str_to_sign, group_secret)

url = "https://openapi.xhey.top/v2/department/sublist"

# 构建请求头
headers = {
    "GroupKey": group_key,
    "Timestamp": timestamp,
    "Signature": signature
}

# 发送HTTP POST请求
response = requests.post(url, data=json_data, headers=headers)

```

4. API总览

- 创建验真任务
- 查询验真结果
- 生成验真报告

5. API列表

5.1 创建验真任务

创建验真任务

基本信息

请求方式：POST

请求地址：https://openapi.xhey.top/v2/truth_build/create

请求参数

名称	类型	示例值	描述
photoUrlList	[]String	["https://net-cloud.xhey.top/group/photo/4efce4e5-806b-4273-9e33-232f8f1a3cb6.jpg"]	照片链接列表，推荐一批次数据少于10条，需要支持通过GET请求直接下载该图片
photoBase64List	[]String	["iVBORw0KGgoAAAANSUhEUgAA... (图片的标准base64，此处省略)"]	照片 base64 列表。要求标准 base64 (StdEncoding)，且不含 data:image/...;base64, 前缀；与 photoUrlList 至少二选一，也可同时传入，合计条数推荐少于10条、最多50条

说明：photoUrlList 与 photoBase64List 至少需要传入一个，可单独使用，也可同时使用（服务端会合并处理）；两个列表的条数合计不得超过 50。photoBase64List 的每个元素必须是图片的标准 base64 编码（如 Go base64.StdEncoding、Java Base64.getEncoder()、Python base64.b64encode、JS btoa 的默认输出），请勿包含 data:image/png;base64, 这类 data URI 前缀，否则将返回参数错误。

返回参数

名称	类型	示例值	描述
code	Integer	200	状态码
msg	String	success	状态码对应的提示内容
data	Object		返回数据
--data	[]Object		任务列表
----taskID	String	2061412837985579008	任务id，用于查询验真的结果
--status	Integer	0	业务状态码（0=成功）
--msg	String		业务提示信息
timestamp	Integer	1780314094	请求时间，Unix时间戳

示例

请求示例（HTTP）

POST https://openapi.xhey.top/v2/truth_build/create

请求示例（Golang API）

```
GOLANG
url := "https://openapi.xhey.top/v2/truth_build/create"
reqStr := `{
    "photoUrlList": ["https://net-cloud.xhey.top/group/photo/4efce4e5-806b-4273-9e33-232f8f1a3cb6.jpg"]
}`
reader := bytes.NewReader([]byte(reqStr))

groupSecret := "groupSecret" // 需申请
groupKey := "groupKey"      // 需申请
dataSign := sign(reqStr, groupSecret)
timestamp := strconv.FormatInt(time.Now().Unix(), 10)
str := fmt.Sprintf("groupKey=%s&sign=%s&timestamp=%s",
    groupKey,
    dataSign,
    timestamp,
)
signature := sign(str, groupSecret)
req, _ := http.NewRequest("POST", url, reader)
req.Header.Add("Content-Type", "application/json")
req.Header.Set("GroupKey", groupKey)
req.Header.Set("Timestamp", timestamp)
req.Header.Set("Signature", signature)

client := &http.Client{}
```

```

defer client.CloseIdleConnections()
resp, err := client.Do(req)

if err != nil {
    fmt.Println(err)
    return
}
defer resp.Body.Close()

var buf bytes.Buffer
_, err = buf.ReadFrom(resp.Body)
if err != nil {
    fmt.Println(err)
    return
}

fmt.Println(buf.String())

```

请求示例 (Golang API , 使用 photoBase64List)

与上面的 photoUrlList 示例相比仅请求体不同：把本地图片编码为标准 base64 后放入 photoBase64List，其余加签、请求头、发送流程完全一致。

GOLANG

```

// 读取本地图片并编码为标准 base64 ( StdEncoding ) ; 注意不要添加 data:image/...;base64, 前缀
imgBytes, _ := os.ReadFile("/path/to/photo.png")
photoBase64 := base64.StdEncoding.EncodeToString(imgBytes)

reqBytes, _ := json.Marshal(map[string]string{
    "photoBase64List": {photoBase64},
})
reqStr := string(reqBytes) // 使用 reqStr 进行加签，请求体发送 reqBytes

```

返回示例

JAVASCRIPT

```

{
  "code": 200,
  "msg": "success",
  "data": {
    "data": [
      {"taskID": "2061412837985579008"}
    ],
    "status": 0,
    "msg": ""
  },
  "timestamp": 1780314094
}

```

5.2 验真结果查询

验真结果查询

基本信息

请求方式：POST

请求地址：https://openapi.xhey.top/v2/truth_build/query

请求参数

名称	类型	示例值	描述
taskID	String	"2061632364592914432"	由 5.1 创建接口返回的 taskID

返回参数

经纬度坐标系为gcj02

名称	类型	示例值	描述
code	Integer	200	状态码

名称	类型	示例值	描述
msg	String	success	状态码对应的提示内容
data	Object		返回数据
--taskStatus	Integer	5	任务状态，1-未处理，2-处理中，5-处理完成，6-任务撤销
--status	Integer	0	业务状态码（0=成功）
--msg	String		业务提示信息
--list	[]Object		单张照片的验真结果列表
----photoUrl	String	https://net-cloud.xhey.top/authVerify/photos/f9c24b58-b628-4fcb-9484-dfc0e8ad107a.png	照片地址。photoUrlList 创建时返回调用方原始 URL；photoBase64List 创建时返回服务端重新上传后的 OSS URL
----photoBase64Idx	Integer	0	仅当该图片由 photoBase64List 创建时返回；为该图片在创建请求 photoBase64List 中的下标（0 起）。photoUrlList 创建的图片不返回该字段
----status	Integer	0	验真结果：0=真，1=处理中；部分错误/假码 RESOLUTION_TOO_LOW = -1 # 分辨率过低；NETWORK_ISSUES = -1001；# 网络连接出错或照片损坏 ALG_ISSUES = -1002 # 程序出错；UNKNOWN_ERROR = -1003 # 未知错误；OCR_INIT_ERR = -2300 # OCR启动错误；NO_ANTICODE = -2301 # 无防伪码；OCR_ERR = -2302 # OCR识别错误；ANTICODE_LENGTH_FAIL = -2303 # 防伪码长度错误；NO_TODAY_CAMER_OR_NO_WATERMARK = -2305 # 非今拍或者无水印
----msg	String	时间真实、地点真实、防伪码真实	判定描述（真实情况说明或错误原因）
----lat	String	34.384929	纬度
----lng	String	115.839913	经度
----photoTime	String	2026年05月29日 19:30	拍照时间
----photoAddress	String	虞城县·至信一路	拍照地点
----antiFakeCode	String	CXPT4TX469K29W	照片的防伪码
timestamp	Integer	1780366439	请求时间，Unix时间戳

示例

请求示例（HTTP）

POST https://openapi.xhey.top/v2/truth_build/query

请求示例（Golang API）

```
GOLANG
url := "https://openapi.xhey.top/v2/truth_build/query"
reqStr := `{
    "taskID": "2061632364592914432"
}`
reader := bytes.NewReader([]byte(reqStr))

groupSecret := "groupSecret" // 需申请
```

```

groupKey := "groupKey"    // 需申请
dataSign := sign(reqStr, groupSecret)
timestamp := strconv.FormatInt(time.Now().Unix(), 10)
str := fmt.Sprintf("groupKey=%s&sign=%s&timestamp=%s",
    groupKey,
    dataSign,
    timestamp,
)
signature := sign(str, groupSecret)
req, _ := http.NewRequest("POST", url, reader)
req.Header.Add("Content-Type", "application/json")
req.Header.Set("GroupKey", groupKey)
req.Header.Set("Timestamp", timestamp)
req.Header.Set("Signature", signature)

client := &http.Client{}
defer client.CloseIdleConnections()
resp, err := client.Do(req)

if err != nil {
    fmt.Println(err)
    return
}
defer resp.Body.Close()

var buf bytes.Buffer
_, err = buf.ReadFrom(resp.Body)
if err != nil {
    fmt.Println(err)
    return
}

fmt.Println(buf.String())

```

返回示例（验真为真，由 photoBase64List 创建，含 photoBase64Idx 字段）

```

JAVASCRIPT
{
  "code": 200,
  "msg": "success",
  "data": {
    "list": [
      {
        "status": 0,
        "msg": "时间真实、地点真实、防伪码真实",
        "photoUrl": "https://net-cloud.xhey.top/authVerify/photos/f9c24b58-b628-4fcb-9484-dfc0e8ad107a.png",
        "photoBase64Idx": 0,
        "lat": "34.384929",
        "lng": "115.839913",
        "photoTime": "2026年05月29日 19:30",
        "photoAddress": "虞城县·至信一路",
        "antiFakeCode": "CXPT4TX469K29W"
      }
    ],
    "taskStatus": 5,
    "status": 0,
    "msg": ""
  },
  "timestamp": 1780366439
}

```

说明：- photoBase64Idx 字段仅在该图片由 photoBase64List 创建时返回；由 photoUrlList 创建的图片不会返回该字段。- 当 photoBase64List 创建时，photoUrl 为服务端重新上传后的 OSS 地址（如示例中的 .../authVerify/photos/f9c24b58-....png）；当 photoUrlList 创建时，photoUrl 为调用方提交时的原始 URL。- status=0 表示验真为真，status<0 表示验真为假或处理异常（详见请求/响应参数表的状态码说明）。

5.3 生成验真报告

根据验真任务 ID，为任务中的每张照片生成验真报告图片并上传至 OSS。一个任务可以只包含 1 张照片，也可以包含多张照片；接口统一通过 reports 返回每张照片对应的报告结果。

建议在验真结果查询的 taskStatus=5（处理完成）后调用。本接口的单项 status 表示报告生成/上传结果，与验真结果中的照片业务状态不是同一含义；即使照片验真结果为非 0，仍可能成功生成报告。

基本信息

请求方式：POST

请求地址：https://openapi.xhey.top/v2/truth_build/verify_report

请求参数

名称	类型	示例值	描述
taskID	String	"2061632364592914432"	验真任务 ID

返回参数

名称	类型	示例值	描述
code	Integer	200	状态码
msg	String	success	状态码对应的提示内容
data	Object		返回数据
--taskID	String	2061632364592914432	验真任务 ID
--reports	[]Object		报告结果列表；每张任务照片对应一项结果。单项失败不会阻断其他照片报告生成
----photoUrl	String	https://example.com/photos/original-1.jpg	仅该照片由 photoUrlList 创建时返回，值为调用方创建任务时提交的原始 URL
----photoBase64Idx	Integer	0	仅该照片由 photoBase64List 创建时返回，为其在创建请求 photoBase64List 中的下标（从 0 开始）；该场景不返回 photoUrl
----timestamp	Integer	1780367810074	本项报告生成时间戳（毫秒）
----ossUrl	String	https://net-cloud.xhey.top/authVerify/reports/2061632364592914432-1780367810074.png	报告图片 OSS 地址
----elapsed	Float	2.4	本项报告生成耗时（秒）
----status	Integer	0	本项报告生成结果状态码，0 表示成功
----msg	String		本项生成失败原因或补充说明
timestamp	Integer	1780367812	请求时间，Unix 时间戳

请通过 photoUrl 或 photoBase64Idx 关联创建任务时的原始输入，不要依赖 reports 中的数组位置。

示例

请求示例（HTTP）

POST https://openapi.xhey.top/v2/truth_build/verify_report

请求示例（Golang API）

```
GOLANG
url := "https://openapi.xhey.top/v2/truth_build/verify_report"
reqStr := `{
  "taskID": "2061632364592914432"
}`
reader := bytes.NewReader([]byte(reqStr))

groupSecret := "groupSecret" // 需申请
groupKey := "groupKey"      // 需申请
dataSign := sign(reqStr, groupSecret)
```

```

timestamp := strconv.FormatInt(time.Now().Unix(), 10)
str := fmt.Sprintf("groupKey=%s&sign=%s&timestamp=%s",
    groupKey,
    dataSign,
    timestamp,
)
signature := sign(str, groupSecret)
req, _ := http.NewRequest("POST", url, reader)
req.Header.Add("Content-Type", "application/json")
req.Header.Set("GroupKey", groupKey)
req.Header.Set("Timestamp", timestamp)
req.Header.Set("Signature", signature)

client := &http.Client{}
defer client.CloseIdleConnections()
resp, err := client.Do(req)

if err != nil {
    fmt.Println(err)
    return
}
defer resp.Body.Close()

var buf bytes.Buffer
_, err = buf.ReadFrom(resp.Body)
if err != nil {
    fmt.Println(err)
    return
}

fmt.Println(buf.String())

```

返回示例

场景一：通过 URL 创建验真任务

photoUrlList 可传入 1 张或多张照片。报告结果中使用 photoUrl 返回每张照片创建任务时提交的原始地址。

创建任务时传入：

JSON

```

{
  "photoUrlList": [
    "https://example.com/photos/original-1.jpg",
    "https://example.com/photos/original-2.jpg"
  ]
}

```

JAVASCRIPT

```

{
  "code": 200,
  "msg": "success",
  "data": {
    "taskId": "2061632364592914432",
    "reports": [
      {
        "photoUrl": "https://example.com/photos/original-1.jpg",
        "timestamp": 1780367810074,
        "ossUrl": "https://net-cloud.xhey.top/authVerify/reports/report-1.png",
        "elapsed": 2.4,
        "status": 0,
        "msg": ""
      },
      {
        "photoUrl": "https://example.com/photos/original-2.jpg",
        "timestamp": 1780367810075,
        "ossUrl": "https://net-cloud.xhey.top/authVerify/reports/report-2.png",
        "elapsed": 2.1,
        "status": 0,
        "msg": ""
      }
    ]
  }
},
"timestamp": 1780367812
}

```

场景二：通过 Base64 创建验真任务

photoBase64List 可传入 1 张或多张照片。报告结果中使用 photoBase64Idx 返回该照片在创建请求 photoBase64List 中的下标；从 0 开始计数。

创建任务时传入：

```
JSON
{
  "photoBase64List": [
    "iVBORw0KGgoAAAANSUhEUgAA...第 1 张...",
    "iVBORw0KGgoAAAANSUhEUgAA...第 2 张..."
  ]
}

JAVASCRIPT
{
  "code": 200,
  "msg": "success",
  "data": {
    "taskId": "2061632364592914432",
    "reports": [
      {
        "photoBase64Idx": 0,
        "timestamp": 1780367810074,
        "ossUrl": "https://net-cloud.xhey.top/authVerify/reports/report-1.png",
        "elapsed": 2.4,
        "status": 0,
        "msg": ""
      },
      {
        "photoBase64Idx": 1,
        "timestamp": 1780367810075,
        "ossUrl": "https://net-cloud.xhey.top/authVerify/reports/report-2.png",
        "elapsed": 2.1,
        "status": 0,
        "msg": ""
      }
    ]
  },
  "timestamp": 1780367812
}
```

错误返回示例

```
JAVASCRIPT
{
  "code": 504,
  "msg": "报告生成超时(60s)",
  "data": null,
  "timestamp": 1776062479
}
```

6. 状态码列表

当code值不等于200时，返回数据的数据为null

code值	类型	错误内容
200	Integer	success
2001	Integer	you can query a maximum of 1000 records, please be precise with your query conditions
400	Integer	param error
401	Integer	verify error
403	Integer	forbidden
404	Integer	not found
500	Integer	service error
4002	Integer	invalid token

code值	类型	错误内容
4007	Integer	group key or secret not right
4008	Integer	you can query a maximum of 1000 personnel information at a time
4009	Integer	time param error
4010	Integer	you can query longest of 180 days data
4011	Integer	you can use a maximum of three keywords to search
4012	Integer	one keyword contain a maximum of 15 characters
4013	Integer	not support watermark ids query